

GoTools

Generated by Doxygen 1.8.11

Contents

- 1 GoTools Library 1**
 - 1.1 Introduction 1
 - 1.2 Building GoTools 2
 - 1.2.1 Compilers 2

- 2 GoTools CompositeModel 3**

- 3 GoTools example files 5**

- 4 GoTools Core 7**
 - 4.1 Geometric Data Structures 7
 - 4.1.1 Parametric Curves 7
 - 4.1.2 Parametric Surfaces 8
 - 4.2 B-spline Curves 10
 - 4.2.1 B-spline Basis Functions 10
 - 4.2.1.1 Linear B-splines 11
 - 4.2.1.2 Quadratic B-splines 11
 - 4.2.1.3 Higher-order B-splines 11
 - 4.2.1.4 Basic properties 11
 - 4.2.2 The Control Polygon 12
 - 4.2.3 The Knot Vector 12
 - 4.2.4 NURBS Curves 12
 - 4.2.5 Spline Curve Functionality 13
 - 4.3 B-spline Surfaces 13
 - 4.3.1 The Basis Functions 14

4.3.2	NURBS Surfaces	14
4.3.3	Spline Surface Functionality	14
4.4	Other Curve Types	15
4.4.1	Curve On Surface	15
4.4.2	Elementary Curve	15
4.5	Other Surface Types	15
4.5.1	Bounded Surface	16
4.5.2	Elementary Surface	16
4.6	Geometry Construction	16
4.6.1	Curve Construction	17
4.6.2	Surface Construction	18
4.7	The utils module	18
4.7.1	Data structures	18
4.7.2	Geometrical objects	18
4.7.3	Misc.	18
4.8	Tesselation	19
5	The g2-format, GoTools file format for geometry entities	21
5.1	The object header	22
5.2	SplineCurve	23
5.3	CurveOnSurface	23
5.4	Line	23
5.5	Circle	24
5.6	Ellipse	24
5.7	Bounded Curve	25
5.8	SplineSurface	25
5.9	BoundedSurface	26
5.10	Plane	27
5.11	Cylinder	27
5.12	Sphere	28
5.13	Cone	28
5.14	Torus	29
5.15	SurfaceOfRevolution	29
5.16	Disc	30
5.17	PointCloud	30
5.18	LineCloud	30
5.19	SplineVolume	31
5.20	Parallelepiped	31
5.21	SphereVolume	32
5.22	CylinderVolume	32
5.23	ConeVolume	32
5.24	TorusVolume	33

6	GoTools Igeslib	35
7	GoTools Implicitization	37
8	GoTools Intersections	39
9	GoTools IsogeometricModel	41
10	GoTools LRsplines2D	43
11	GoTools Parametrization	45
11.1	Purpose	45
11.2	How it can be used	45
11.2.1	Choice of data structure	45
11.2.2	Choice of boundary parametrization	46
11.2.3	Choice of interior parametrization	46
11.2.4	Putting it all together	46
11.3	Example program	47
12	GoTools QualityModule	49
12.1	Tolerances	49
12.2	Tests	49
13	The filter routines main page	51
13.1	Introduction	51
14	GoTools Topology	53
15	GoTools Trivariate	55
15.1	Data Structures	55
15.2	B-spline Volumes	56
15.2.1	The Basis Functions	56
15.2.2	NURBS Volumes	57
15.2.3	Spline Volume Functionality	57
15.2.4	Construction Methods for SplineVolume	58
15.3	Surface on Volume	58
15.4	Curves Related to a Parametric Volume	58

16 GoTools Trivariatemodel	59
16.1 Volume Topology	59
16.2 Volume Model	60
16.3 Topological Volume Entity	60
16.4 Extensions to Face	61
16.5 Radial Edge	61
17 Application Programming Interface to TTL (API)	63
17.1 DartType	63
17.2 TraitsType	64
18 Example using TTL and the half-edge data structure	67
19 The Half-Edge Data Structure and Adaption to TTL	69
19.1 The Class Diagram	69
19.2 Example	69
20 GoTools Viewlib	71
20.1 Dependencies	71
21 Bug List	73
22 Module Index	75
22.1 Modules	75
23 Namespace Index	77
23.1 Namespace List	77
24 Hierarchical Index	79
24.1 Class Hierarchy	79
25 Class Index	91
25.1 Class List	91
26 File Index	105
26.1 File List	105

27 Module Documentation	157
27.1 RBD common library	157
27.1.1 Detailed Description	157
27.1.2 Macro Definition Documentation	157
27.1.2.1 ATandT	157
27.1.2.2 bool_LIB	158
27.1.2.3 DEFAULT_HEADER	158
27.1.2.4 TypeDefException	158
27.1.2.5 use_namespace	158
27.1.2.6 UseExceptions	158
27.1.2.7 USING_DOUBLE	158
28 Namespace Documentation	159
28.1 Go Namespace Reference	159
28.1.1 Detailed Description	179
28.1.2 Typedef Documentation	179
28.1.2.1 BaryCoordSystem2D	179
28.1.2.2 BaryCoordSystem3D	179
28.1.2.3 BoundaryPiece	179
28.1.2.4 BsplineIndexMap	179
28.1.2.5 GraphIter	179
28.1.2.6 PointCloud3D	179
28.1.2.7 PointCloud4D	180
28.1.2.8 PointIter	180
28.1.2.9 PointList	180
28.1.2.10 sideConstraint	180
28.1.2.11 sideConstraintSet	180
28.1.2.12 simpleElement	180
28.1.2.13 Vector2D	180
28.1.2.14 Vector3D	180
28.1.2.15 Vector4D	180

28.1.3	Enumeration Type Documentation	181
28.1.3.1	anonymous enum	181
28.1.3.2	AlgorithmChoice	181
28.1.3.3	BdConditionType	181
28.1.3.4	ClassType	182
28.1.3.5	closestPointLevel	183
28.1.3.6	CoefStatus	183
28.1.3.7	Direction2D	183
28.1.3.8	EstimateDirection	184
28.1.3.9	EvalKind	184
28.1.3.10	FileFormat	184
28.1.3.11	ftCurveType	184
28.1.3.12	ftmessages	185
28.1.3.13	ftTangPriority	185
28.1.3.14	IGESSection	186
28.1.3.15	IntersectionType	186
28.1.3.16	IntPtClassification	186
28.1.3.17	IntPtLocation	187
28.1.3.18	IteratorType	187
28.1.3.19	LinkType	187
28.1.3.20	PointSequenceType	188
28.1.3.21	RegistrationReturnType	188
28.1.3.22	SingularityClassification	189
28.1.3.23	SingularityType	189
28.1.3.24	SubdivisionClassification	189
28.1.3.25	TangentDomain	190
28.1.3.26	testSuite	190
28.1.3.27	tpJointType	191
28.1.4	Function Documentation	191
28.1.4.1	add_reachables_from(const IntersectionPoint *pt, const IntersectionPoint *last←→ _pt, std::set< IntersectionPoint * > &result)	191

28.1.4.2	<code>addToLinearSystem(int pt_idx, const std::vector< Point > &points_fixed, const std::vector< Point > &points_transform, bool allow_rescaling, const std::vector< std::vector< double > > &id, const Point &fine_R, const Point &fine_T, double fine_s, const std::vector< std::vector< double > > &m_rot_R, double s2, double R2, bool zero_R, const std::vector< std::vector< std::vector< double > > > &lhs_matrix, const std::vector< std::vector< double > > &rhs_matrix)</code>	191
28.1.4.3	<code>area(const Array< T, 2 > *c)</code>	191
28.1.4.4	<code>area(const Array< T, 3 > *c)</code>	191
28.1.4.5	<code>areaTriangle(const Vector2D &corner1, const Vector2D &corner2, const Vector2D &corner3)</code>	191
28.1.4.6	<code>backwardSubstitution(const SquareMatrix &U, T *x, int num_unknows)</code>	191
28.1.4.7	<code>backwardSubstitution(const SquareMatrix &U, std::vector< double > *x, int num_unknows)</code>	192
28.1.4.8	<code>benchmarkSfRefinement(LRSplineSurface &lr_sf, const std::vector< LRSplineSurface::Refinement2D > &refs, bool single_insertions=false)</code>	192
28.1.4.9	<code>bernsteinToSpline(const Array< BernsteinPoly, Ndim > &curve_bp, bool rational, SplineCurve &segment)</code>	192
28.1.4.10	<code>bernsteinToSpline(const Array< BernsteinMulti, Ndim > &surface_bm, bool rational, SplineSurface &patch)</code>	192
28.1.4.11	<code>binom(int n, int i)</code>	193
28.1.4.12	<code>brent_minimize(const Functor &f, double a, double b, double c, double &parmin, const double rel_tolerance=std::sqrt(std::numeric_limits< double > ::epsilon()))</code>	193
28.1.4.13	<code>cart_to_bary(const SplineCurve &cv, const BaryCoordSystem2D &bc, SplineCurve &cv_bc)</code>	193
28.1.4.14	<code>cart_to_bary(const SplineCurve &cv, const BaryCoordSystem3D &bc, SplineCurve &cv_bc)</code>	193
28.1.4.15	<code>cart_to_bary(const SplineSurface &sf, const BaryCoordSystem3D &bc, SplineSurface &sf_bc)</code>	193
28.1.4.16	<code>cart_to_bary(const PointCloud3D &cloud, const BaryCoordSystem3D &bc, PointCloud4D &cloud_bc)</code>	193
28.1.4.17	<code>checkCoincide(ParamCurveInt *curve, double start, double end, shared_ptr< GeoTol > tol, ParamCurveInt *other, double other_start, double other_end)</code>	193
28.1.4.18	<code>checkCoincide(ParamCurveInt *curve, double start, double end, ParamSurfaceInt *surf, Point su_start, Point su_end, shared_ptr< GeoTol > tol)</code>	194
28.1.4.19	<code>checkCoincide(ParamCurveInt *curve, double start, double end, SplineSurfaceInt *surf, const Point &su_start, const Point &su_end, shared_ptr< GeoTol > tol)</code>	194
28.1.4.20	<code>checkCoincide(Param1FunctionInt *func1, double start, double end, Param0FunctionInt *C, shared_ptr< GeoTol > tol)</code>	194
28.1.4.21	<code>checkCoincide(ParamSurfaceInt *surf1, ParamSurfaceInt *surf2, std::vector< double > &par_loop, const shared_ptr< GeoTol > tol)</code>	194

28.1.4.22	<code>closestDistances(const std::vector< float > &pts, const shared_ptr< boxStructuring::BoundingBoxStructure > &structure, const std::vector< std::vector< double > > &rotationMatrix, const Point &translation)</code>	194
28.1.4.23	<code>closestPointCalculations(const std::vector< float > &pts, const shared_ptr< boxStructuring::BoundingBoxStructure > &structure, const std::vector< std::vector< double > > &rotationMatrix, const Point &translation, int return_type, int start_idx, int skip, int max_idx, int search_extend=3, bool m_core=true)</code>	194
28.1.4.24	<code>closestPointCalculations(const std::vector< float > &pts, const shared_ptr< boxStructuring::BoundingBoxStructure > &structure, const std::vector< std::vector< double > > &rotationMatrix, const Point &translation, int return_type)</code>	195
28.1.4.25	<code>closestPoints(const std::vector< float > &pts, const shared_ptr< boxStructuring::BoundingBoxStructure > &structure, const std::vector< std::vector< double > > &rotationMatrix, const Point &translation)</code>	195
28.1.4.26	<code>closestPointSingleCalculation(int pt_idx, int start_idx, int skip, const std::vector< float > &inPoints, const std::vector< std::vector< double > > &rotationMatrix, const Point &translation, const shared_ptr< boxStructuring::BoundingBoxStructure > &boxStructure, std::vector< float > &result, std::vector< std::vector< int > > &lastBoxCall, int return_type, int search_extend)</code>	195
28.1.4.27	<code>closestPtCurves(SplineCurve *cv1, SplineCurve *cv2, double epsge, double &par1, double &par2, double &dist)</code>	195
28.1.4.28	<code>closestSignedDistances(const std::vector< float > &pts, const shared_ptr< boxStructuring::BoundingBoxStructure > &structure, const std::vector< std::vector< double > > &rotationMatrix, const Point &translation)</code>	196
28.1.4.29	<code>closestVectorsOld(const std::vector< float > &inPoints, const shared_ptr< boxStructuring::BoundingBoxStructure > &boxStructure, const std::vector< std::vector< double > > &rotationMatrix, const Point &translation, int return_type, int start_idx, int skip, int max_idx, int search_extend=3)</code>	196
28.1.4.30	<code>compare_first(const std::pair< T1, T2 > &x, const std::pair< T1, T2 > &y)</code>	196
28.1.4.31	<code>compare_seq(iterator begin_1, iterator end_1, iterator begin_2, iterator end_2)</code>	196
28.1.4.32	<code>computeLoopGap(const std::vector< PtrToCurveType > &curves)</code>	196
28.1.4.33	<code>constructIntersectionCurve(const iterator begin, const iterator end)</code>	197
28.1.4.34	<code>create_bary_coord_system2D(const SplineCurve &curve, BaryCoordSystem2D &bc)</code>	197
28.1.4.35	<code>create_bary_coord_system3D(const SplineCurve &curve, BaryCoordSystem3D &bc)</code>	197
28.1.4.36	<code>create_bary_coord_system3D(const SplineSurface &surface, BaryCoordSystem3D &bc)</code>	197
28.1.4.37	<code>create_bary_coord_system3D(const PointCloud3D &cloud, BaryCoordSystem3D &bc)</code>	197
28.1.4.38	<code>create_bary_coord_system3D(const BoundingBox &box, BaryCoordSystem3D &bc)</code>	197
28.1.4.39	<code>curvatureRadius(const std::vector< Point > &der, std::vector< Point > &unitder)</code>	197

28.1.4.40 Curve2SISL(const SplineCurve &cv, bool copy=true)	197
28.1.4.41 Curve2SISL_rat(const SplineCurve &cv)	198
28.1.4.42 debug_write_line(const Point &p1, const Point &p2, const char *fname)	198
28.1.4.43 debug_write_point(const Point &p1, const char *fname)	198
28.1.4.44 degenerate_triangle(const Vector2D &c1, const Vector2D &c2, const Vector2D &c3)	198
28.1.4.45 determinantOf(const Array< T, 2 > *a)	198
28.1.4.46 determinantOf(const Array< T, 3 > *a)	198
28.1.4.47 determine_seed(double *par, int par_start, int par_end, const Point &pt, const IntersectionPoint *const ip1, const IntersectionPoint *const ip2)	198
28.1.4.48 determineCoincidenceRegion(const ParamObjectInt *obj1, const ParamObject↵ Int *obj2, shared_ptr< const GeoTol > tol, const double *current_params, int dir, bool forward, double &last_param_val_inside, double &first_param_val_outside)	199
28.1.4.49 determineCoincidenceRegion(const ParamFunctionInt *obj_1d, double C, shared_ptr< const GeoTol > tol, const double *current_params, int dir, bool forward, double &last_param_val_inside, double &first_param_val_outside)	199
28.1.4.50 estimate_seed_by_interpolation(const IntersectionPoint *p1, const Intersection↵ Point *p2, int fixed_dir, double fixed_value, double *result)	199
28.1.4.51 evalDistCurve(const std::vector< Point > &marching_curve, const std::vector< Point > &other_curve, std::vector< Point > &dist_curve)	199
28.1.4.52 evalProjectedCurve(const std::vector< Point > &ft, const std::vector< Point > &gs, std::vector< Point > &res)	199
28.1.4.53 extract_chains(const std::vector< shared_ptr< IntersectionPoint > > pts, std↵ ::vector< std::vector< shared_ptr< IntersectionPoint > > > &chains)	199
28.1.4.54 extremalPtSurfSurf(ParamSurface *psurf1, ParamSurface *psurf2, int con- straints[2], double constraints_par[2], double limit[], double enext[], double gpos[], double angle_tol)	199
28.1.4.55 factorial(int n)	200
28.1.4.56 find_point_in(IntersectionPoint *p, const std::vector< shared_ptr< Intersection↵ Point > > &vec)	200
28.1.4.57 fineRegistration(const std::vector< Point > &points_fixed, const std::vector< Point > &points_transform, bool allow_rescaling, RegistrationInput params)	200
28.1.4.58 flip(Direction2D d)	201
28.1.4.59 flip_intersecting_objects(IntersectionPoint *p)	201
28.1.4.60 forwardSubstitution(const SquareMatrix &L, T *x, int num_unknowns)	201
28.1.4.61 forwardSubstitution(const SquareMatrix &L, std::vector< double > *x, int num↵ _unknowns)	201
28.1.4.62 gaussian_quadrature(Funcor &f, double a, double b)	201

28.1.4.63 gaussian_quadrature2D(Functor2D &f, double ax, double bx, double ay, double by)	202
28.1.4.64 GaussQuadInner(const BsplineBasis &basis, int ider, double lim1, double lim2, double ***integral)	202
28.1.4.65 GaussQuadInner2(const BsplineBasis &basis, int ider, double lim1, double lim2, double **integral)	202
28.1.4.66 GaussQuadInnerFlat(const BsplineBasis &basis, int derivs, int start_der, int gap, double lim1, double lim2, std::vector< double > &integral)	202
28.1.4.67 GaussQuadInnerRational(const BsplineBasis &basis, int ider, double lim1, double lim2, shared_ptr< SplineCurve > bspline_curve, double ***integral)	203
28.1.4.68 GaussQuadValues(const BsplineBasis &basis, std::vector< double > ¶meters, std::vector< double > &par_weights)	203
28.1.4.69 getCurrentTime()	203
28.1.4.70 getHermiteData(const std::vector< Point > &der1, const std::vector< Point > &der2, double &parint, double &len1, double &len2)	204
28.1.4.71 GoSurf2SISL(const SplineSurface &sf, bool copy=true)	204
28.1.4.72 intersect2Dcurves(const ParamCurve *cv1, const ParamCurve *cv2, double epsge, std::vector< std::pair< double, double > > &intersections, std::vector< int > &pretopology, std::vector< std::pair< std::pair< double, double >, std::pair< double, double > > > &int_crvs)	204
28.1.4.73 intersectCurvePoint(const ParamCurve *crv, Point pnt, double epsge, std::vector< double > &intersections, std::vector< std::pair< double, double > > &int_crvs)	204
28.1.4.74 intersectCurves(shared_ptr< ParamCurve > crv1, shared_ptr< ParamCurve > crv2, double tol, std::vector< std::pair< double, double > > &intersection_points)	205
28.1.4.75 intersectcurves(SplineCurve *cv1, SplineCurve *cv2, double epsge, std::vector< std::pair< double, double > > &intersections)	205
28.1.4.76 intersectCurveSurf(const SplineCurve *cv, const SplineSurface *sf, double epsge, std::vector< std::pair< double, Point > > &int_pts, std::vector< int > &pretopology, std::vector< std::pair< std::pair< double, Point >, std::pair< double, Point > > > &int_crvs)	205
28.1.4.77 interval_overlap(ValueType front1, ValueType back1, ValueType front2, ValueType back2)	205
28.1.4.78 is_inside(const std::vector< Go::Vector3D > &trim_curve_p, const std::vector< int > &contour, const double u, const double v)	205
28.1.4.79 is_on_contour(const std::vector< Go::Vector3D > &trim_curve_p, const std::vector< int > &contour, const double u, const double v)	205
28.1.4.80 is_on_corner(const std::vector< Go::Vector3D > &trim_curve_p, const std::vector< int > &contour, const double u, const double v)	205
28.1.4.81 link_is_iso(shared_ptr< IntersectionLink > link)	205
28.1.4.82 link_is_iso_in(shared_ptr< IntersectionLink > link, int dir)	206

28.1.4.83	<code>link_is_iso_in_other_than(shared_ptr< IntersectionLink > link, int dir)</code>	206
28.1.4.84	<code>LUDecomp(SquareMatrix &mat, int num_rows, int *perm, bool &parity)</code>	206
28.1.4.85	<code>LUsolveSystem(SquareMatrix &A, int num_unknowns, T *vec)</code>	206
28.1.4.86	<code>make_curve_on_surface(shared_ptr< const ParamSurface > surf, double u_↔ start, double v_start, double u_end, double v_end, double p_start, double p_end)</code> 206	206
28.1.4.87	<code>make_implicit_gauss(std::vector< std::vector< double > > &mat, std::vector< double > &b)</code>	207
28.1.4.88	<code>make_implicit_svd(std::vector< std::vector< double > > &mat, std::vector< dou- ble > &b, double &sigma_min)</code>	207
28.1.4.89	<code>make_matrix(const SplineCurve &curve, int deg, std::vector< std::vector< double > > &mat)</code>	207
28.1.4.90	<code>make_matrix(const SplineSurface &surf, int deg, std::vector< std::vector< double > > &mat)</code>	207
28.1.4.91	<code>make_matrix(const PointCloud4D &cloud, int deg, std::vector< std::vector< dou- ble > > &mat)</code>	207
28.1.4.92	<code>make_trimmed_mesh(shared_ptr< ParamSurface > srf, std::vector< shared_ptr< ParamCurve > > &crv_set, std::vector< Vector3D > &vert, std::vector< Vector2D > &vert_p, std::vector< int > &bd, std::vector< Vector3D > &norm, std::vector< int > &mesh, std::vector< Vector3D > &trim_curve, std::vector< Vector3D > &trim_curve_p, const int dn, const int dm, double bd_res_ratio)</code>	207
28.1.4.93	<code>minimise_conjugated_gradient(FunctionMinimizer< Functor > &dfmin)</code>	207
28.1.4.94	<code>no_parent(const shared_ptr< IntersectionPoint > &p)</code>	207
28.1.4.95	<code>nondecreasing(ValueType a, ValueType b, ValueType c)</code>	207
28.1.4.96	<code>nonincreasing(ValueType a, ValueType b, ValueType c)</code>	207
28.1.4.97	<code>normalNoise(double *res, double mean_err, int num_samples)</code>	207
28.1.4.98	<code>operator*(const Rational &r1, const Rational r2)</code>	208
28.1.4.99	<code>operator*(const Array< double, Dim > &a, const T b)</code>	208
28.1.4.100	<code>operator*(const BernsteinPoly &p1, const BernsteinPoly &p2)</code>	208
28.1.4.101	<code>operator*(const BernsteinPoly &p, double c)</code>	208
28.1.4.102	<code>operator*(double c, const BernsteinPoly &p)</code>	208
28.1.4.103	<code>operator*(const BernsteinTriangularPoly &p1, const BernsteinTriangularPoly &p2)</code> 208	208
28.1.4.104	<code>operator*(const BernsteinTriangularPoly &p, double c)</code>	208
28.1.4.105	<code>operator*(double c, const BernsteinTriangularPoly &p)</code>	209
28.1.4.106	<code>operator*(const BernsteinTetrahedralPoly &p1, const BernsteinTetrahedralPoly &p2)</code>	209

28.1.4.107operator*(const BernsteinTetrahedralPoly &p, double c)	209
28.1.4.108operator*(double c, const BernsteinTetrahedralPoly &p)	209
28.1.4.109operator*(const BernsteinMulti &m1, const BernsteinMulti &m2)	209
28.1.4.110operator*(const BernsteinMulti &m1, double c)	209
28.1.4.111operator*(double c, const BernsteinMulti &m1)	209
28.1.4.112operator*(T d, const Go::Array< T, Dim > &v)	210
28.1.4.113operator*(double d, const Point &p)	210
28.1.4.114operator+(const Rational &r1, const Rational r2)	210
28.1.4.115operator+(const BernsteinPoly &p1, const BernsteinPoly &p2)	210
28.1.4.116operator+(double c, const BernsteinPoly &p)	210
28.1.4.117operator+(const BernsteinPoly &p, double c)	210
28.1.4.118operator+(const BernsteinTriangularPoly &p1, const BernsteinTriangularPoly &p2)	210
28.1.4.119operator+(double c, const BernsteinTriangularPoly &p)	210
28.1.4.120operator+(const BernsteinTriangularPoly &p, double c)	211
28.1.4.121operator+(const BernsteinTetrahedralPoly &p1, const BernsteinTetrahedralPoly &p2)	211
28.1.4.122operator+(double c, const BernsteinTetrahedralPoly &p)	211
28.1.4.123operator+(const BernsteinTetrahedralPoly &p, double c)	211
28.1.4.124operator+(const BernsteinMulti &m1, const BernsteinMulti &m2)	211
28.1.4.125operator+(const BernsteinMulti &m, double c)	211
28.1.4.126operator+(double c, const BernsteinMulti &m)	211
28.1.4.127operator-(const Rational &r1, const Rational r2)	212
28.1.4.128operator-(const BernsteinPoly &p1, const BernsteinPoly &p2)	212
28.1.4.129operator-(double c, const BernsteinPoly &p)	212
28.1.4.130operator-(const BernsteinPoly &p, double c)	212
28.1.4.131operator-(const BernsteinTriangularPoly &p1, const BernsteinTriangularPoly &p2)	212
28.1.4.132operator-(double c, const BernsteinTriangularPoly &p)	212
28.1.4.133operator-(const BernsteinTriangularPoly &p, double c)	212
28.1.4.134operator-(const BernsteinTetrahedralPoly &p1, const BernsteinTetrahedralPoly &p2)	212
28.1.4.135operator-(double c, const BernsteinTetrahedralPoly &p)	213
28.1.4.136operator-(const BernsteinTetrahedralPoly &p, double c)	213

28.1.4.137	operator-(const BernsteinMulti &m1, const BernsteinMulti &m2)	213
28.1.4.138	operator-(const BernsteinMulti &m, double c)	213
28.1.4.139	operator-(double c, const BernsteinMulti &m)	213
28.1.4.140	operator/(const Rational &r1, const Rational r2)	213
28.1.4.141	operator/(const BernsteinPoly &p, double c)	213
28.1.4.142	operator/(const BernsteinTriangularPoly &p, double c)	213
28.1.4.143	operator/(const BernsteinTetrahedralPoly &p, double c)	214
28.1.4.144	operator/(const BernsteinMulti &m, double c)	214
28.1.4.145	operator<(const Point &p1, const Point &p2)	214
28.1.4.146	operator<<(std::ostream &os, const Go::Streamable &obj)	214
28.1.4.147	operator<<(std::ostream &os, const Rational &p)	214
28.1.4.148	operator<<(std::ostream &os, const Go::BernsteinTriangularPoly &p)	214
28.1.4.149	operator<<(std::ostream &os, const Go::BernsteinTetrahedralPoly &p)	214
28.1.4.150	operator<<(std::ostream &os, const LRBSpline2D &b)	214
28.1.4.151	operator<<(std::ostream &os, const GPos &g)	215
28.1.4.152	operator<<(std::ostream &os, const Mesh2D &m)	215
28.1.4.153	operator<<(std::ostream &os, const Go::Array< T, Dim > &v)	215
28.1.4.154	operator<<(std::ostream &os, const MatrixXD< T, Dim > &m)	215
28.1.4.155	operator<<(std::ostream &os, const Go::Point &v)	215
28.1.4.156	operator==(const Point &p1, const Point &p2)	215
28.1.4.157	operator>>(std::istream &is, Go::Streamable &obj)	215
28.1.4.158	operator>>(std::istream &is, Go::BernsteinTriangularPoly &p)	215
28.1.4.159	operator>>(std::istream &is, Go::BernsteinTetrahedralPoly &p)	216
28.1.4.160	operator>>(std::istream &is, LRBSpline2D &b)	216
28.1.4.161	operator>>(std::istream &is, GPos &g)	216
28.1.4.162	operator>>(std::istream &is, Mesh2D &m)	216
28.1.4.163	operator>>(std::istream &is, Go::Array< T, Dim > &v)	216
28.1.4.164	operator>>(std::istream &is, Go::Point &v)	216
28.1.4.165	point_inside_contour(const double x0, const double y0, const double *const vertices, const std::vector< int > &contour)	216

28.1.4.166	preProcessClosestVectors(const std::vector< shared_ptr< GeomObject > > &surfaces, double par_len_el)	216
28.1.4.167	quadrinomial(int n, int i, int j, int k)	216
28.1.4.168	rawRegistration(const std::vector< Point > &points_fixed, const std::vector< Point > &points_transform, bool allow_rescaling, RegistrationInput params) . . .	217
28.1.4.169	Register()	217
28.1.4.170	registration(const std::vector< Point > &points_fixed, const std::vector< Point > &points_transform, bool allow_rescaling, RegistrationInput params)	217
28.1.4.171	segment_contour_intersection_for_s2m(const double x0, const double y0, const double x1, const double y1, const double *const vertices, const std::vector< int > &contour, double &x, double &y, double &s, const bool snap_ends=false) . . .	217
28.1.4.172	signed_area(const Array< T, 3 > *c, const Array< T, 3 > &normal)	217
28.1.4.173	simplex_volume(const Array< T, Dim > *a)	217
28.1.4.174	simpsons_rule(Funcor &f, double a, double b, const double eps=1.0e-6, const int max_iter=20)	218
28.1.4.175	simpsons_rule2D(Funcor2D &f, double ax, double bx, double ay, double by, const double eps=1.0e-6, const int max_iter=20)	218
28.1.4.176	SISLCurve2Go(const SISLCurve *const cv)	218
28.1.4.177	SISLSurf2Go(SISLSurf *sf)	218
28.1.4.178	sort_2dpoly_segments(const double *const vertices, const std::vector< int > &contour, const bool transposed=false)	218
28.1.4.179	spline_to_bernstein(const SplineCurve &seg, int dd, BernsteinPoly &bp)	218
28.1.4.180	spline_to_bernstein(const SplineSurface &pat, int dd, BernsteinMulti &bm)	219
28.1.4.181	spline_to_bernstein(const SplineCurve &seg, std::vector< BernsteinPoly > &seg_bp)	219
28.1.4.182	spline_to_bernstein(const SplineSurface &pat, std::vector< BernsteinMulti > &pat_bm)	219
28.1.4.183	splineToBernstein(const SplineCurve &segment, Array< BernsteinPoly, Ndim > &curve_bp)	219
28.1.4.184	splineToBernstein(const SplineSurface &patch, Array< BernsteinMulti, Ndim > &surface_bm)	219
28.1.4.185	stepLenFromRadius(double radius, double aepsge)	220
28.1.4.186	stepLength(const std::vector< Point > &ft, const std::vector< Point > &gs, double delta, bool forward, double &delta_t)	220
28.1.4.187	stepLength(const std::vector< Point > &ft, const std::vector< Point > &gs, bool forward, std::vector< Point > &cvder, double min_step, double max_step, double aepsge)	220

28.1.4.188	<code>strictly_decreasing(Iterator begin, Iterator end)</code>	220
28.1.4.189	<code>strictly_decreasing(const Array &A)</code>	220
28.1.4.190	<code>strictly_decreasing(ValueType a, ValueType b, ValueType c)</code>	220
28.1.4.191	<code>strictly_increasing(Iterator begin, Iterator end)</code>	220
28.1.4.192	<code>strictly_increasing(const Array &A)</code>	220
28.1.4.193	<code>strictly_increasing(ValueType a, ValueType b, ValueType c)</code>	220
28.1.4.194	<code>systemSleep(double sleep_time)</code>	220
28.1.4.195	<code>tanLenFromRadius(double radius, double angle)</code>	220
28.1.4.196	<code>trapezoidal(Func f, double a, double b, double &s, int n)</code>	221
28.1.4.197	<code>trinomial(int n, int i, int j)</code>	221
28.1.4.198	<code>uniformNoise(double *res, double lval, double uval, int num_samples)</code>	221
28.1.4.199	<code>volume(const Array< T, 3 > *c)</code>	221
28.1.4.200	<code>weakly_decreasing(Iterator begin, Iterator end)</code>	221
28.1.4.201	<code>weakly_decreasing(const Array &A)</code>	221
28.1.4.202	<code>weakly_increasing(Iterator begin, Iterator end)</code>	221
28.1.4.203	<code>weakly_increasing(const Array &A)</code>	222
28.1.4.204	<code>writePostscriptMesh(Go::LRSplineSurface &lr_spline_sf, std::ostream &out, const bool close=true)</code>	222
28.1.5	Variable Documentation	222
28.1.5.1	MAJOR_VERSION	222
28.1.5.2	MINOR_VERSION	222
28.2	Go::AdaptSurface Namespace Reference	222
28.2.1	Detailed Description	223
28.2.2	Function Documentation	223
28.2.2.1	<code>adaptSurface(shared_ptr< ParamSurface > surf, shared_ptr< SplineSurface > init_surf, double tol)</code>	223
28.2.2.2	<code>approxInSplineSpace(shared_ptr< ParamSurface > surf, shared_ptr< SplineSurface > surf2, double tol)</code>	223
28.2.2.3	<code>createTriangulation(shared_ptr< ParamSurface > surf, const RectDomain &dom, shared_ptr< ftPointSet > &points, vector< int > &corner, bool consider_joints=true, int nmb=-1)</code>	223
28.2.2.4	<code>curveApprox(shared_ptr< ParamCurve > cvs[], int nmb_cvs, const BsplineBasis &init_basis, double tol)</code>	223

28.2.2.5	<code>curveApprox(shared_ptr< ParamCurve > cvs[], int nmb_cvs, double tol, double degree=3)</code>	223
28.2.2.6	<code>doApprox(shared_ptr< SplineSurface > init_surf, int max_iter, shared_ptr< ftPointSet > points, double tol, double &max_error, double &mean_error)</code>	223
28.2.2.7	<code>expressInSameSplineSpace(shared_ptr< ParamSurface > surf1, shared_ptr< ParamSurface > surf2, double tol)</code>	223
28.2.2.8	<code>getBoundaryData(shared_ptr< ParamSurface > surf, const RectDomain &dom, int nmb_sample, shared_ptr< ftPointSet > points, std::vector< int > &corner)</code>	224
28.2.2.9	<code>getCornerCorrespondance(shared_ptr< ParamSurface > surf1, shared_ptr< ParamSurface > surf2, int &idx, bool &turned)</code>	224
28.2.2.10	<code>getInnerData(shared_ptr< ParamSurface > surf, const RectDomain &dom, int nmb_sample, shared_ptr< ftPointSet > points, bool consider_joint=true)</code>	224
28.2.2.11	<code>parameterizePoints(shared_ptr< SplineSurface > init_surf, shared_ptr< ftPointSet > points, std::vector< int > corner)</code>	224
28.2.2.12	<code>projectPoints(shared_ptr< SplineSurface > surf, shared_ptr< ftPointSet > points)</code>	224
28.2.2.13	<code>updatePointTopology(shared_ptr< ParamSurface > surf, ftPointSet &points)</code>	224
28.3	<code>Go::BoundedUtils</code> Namespace Reference	224
28.3.1	Detailed Description	226
28.3.2	Function Documentation	226
28.3.2.1	<code>checkAndFixLoopOrientation(shared_ptr< BoundedSurface > surf)</code>	226
28.3.2.2	<code>checkCurveCoincidence(shared_ptr< CurveOnSurface > cv1, shared_ptr< CurveOnSurface > cv2, double tol, bool same_orient)</code>	226
28.3.2.3	<code>convertToBoundedSurface(const SplineSurface &surf, double space_epsilon)</code>	226
28.3.2.4	<code>createMissingParCvs(Go::BoundedSurface &bd_sf)</code>	226
28.3.2.5	<code>createMissingParCvs(std::vector< Go::CurveLoop > &bd_loops)</code>	226
28.3.2.6	<code>createTrimmedSurfs(std::vector< std::vector< shared_ptr< CurveOnSurface > > > &loops, shared_ptr< ParamSurface > under_sf, double epsgeo)</code>	227
28.3.2.7	<code>fixInvalidBoundedSurface(shared_ptr< Go::BoundedSurface > &bd_sf, double max_tol_mult=1.0)</code>	227
28.3.2.8	<code>getBoundaryLoops(const BoundedSurface &sf, std::vector< shared_ptr< CurveOnSurface > > &part_bnd_cvs, double eps, int last_split=-1)</code>	227
28.3.2.9	<code>getCylinderIntersections(const shared_ptr< ParamSurface > &surf, Point point, Point axis, double radius, double epsge, shared_ptr< BoundedSurface > &bounded_sf)</code>	228
28.3.2.10	<code>getIntersectionCurve(shared_ptr< ParamSurface > &sf1, shared_ptr< ParamSurface > &sf2, std::vector< shared_ptr< CurveOnSurface > > &int_segments1, std::vector< shared_ptr< CurveOnSurface > > &int_segments2, double epsge)</code>	228

28.3.2.11	<code>getPlaneIntersections(const shared_ptr< ParamSurface > &surf, Point point, Point normal, double epsge, shared_ptr< BoundedSurface > &bounded_sf)</code> . . .	228
28.3.2.12	<code>getSurfaceIntersections(const shared_ptr< ParamSurface > &surf1, const shared_ptr< ParamSurface > &surf2, double epsge, std::vector< shared_ptr< CurveOnSurface > > &int_cv1, shared_ptr< BoundedSurface > &bounded_sf1, std::vector< shared_ptr< CurveOnSurface > > &int_cv2, shared_ptr< BoundedSurface > &bounded_sf2)</code>	228
28.3.2.13	<code>getTrimCrvsParam(const shared_ptr< ParamSurface > &surf, Point parval1, Point parval2, double epsge, shared_ptr< BoundedSurface > &bounded_sf)</code> . . .	228
28.3.2.14	<code>getTrimCrvsPcrv(const shared_ptr< ParamSurface > &surf, shared_ptr< ParamCurve > &pcurve, double epsge, shared_ptr< BoundedSurface > &bounded_sf)</code>	229
28.3.2.15	<code>intersectWithCylinder(shared_ptr< ParamSurface > &surf, Point pnt, Point vec, double radius, double geom_tol)</code>	229
28.3.2.16	<code>intersectWithPlane(shared_ptr< ParamSurface > &surf, Point pnt, Point normal, double geom_tol)</code>	229
28.3.2.17	<code>intersectWithSurface(CurveOnSurface &curve, BoundedSurface &bounded_surf, double epsge)</code>	229
28.3.2.18	<code>intersectWithSurfaces(std::vector< shared_ptr< CurveOnSurface > > &curves1, shared_ptr< BoundedSurface > &bd_sf1, std::vector< shared_ptr< CurveOnSurface > > &curves2, shared_ptr< BoundedSurface > &bd_sf2, double epsge)</code>	230
28.3.2.19	<code>loopsDegenerate(std::vector< shared_ptr< CurveOnSurface > > &loop, double epsgeo)</code>	230
28.3.2.20	<code>makeTrimmedPlane(shared_ptr< Go::Plane > &plane, std::vector< shared_ptr< Go::ParamCurve > > &space_crvs)</code>	230
28.3.2.21	<code>rotateBoundedSurf(Point rot_axis, double alpha, BoundedSurface &bf_sf, double deg_eps)</code>	231
28.3.2.22	<code>splitBetweenParams(const shared_ptr< ParamSurface > &surf, Point parval1, Point parval2, double epsge)</code>	231
28.3.2.23	<code>splitBetweenParPairs(const shared_ptr< ParamSurface > &surf, std::vector< std::pair< Point, Point > > parvals, double epsge)</code>	231
28.3.2.24	<code>splitWithPlane(const shared_ptr< ParamSurface > &surf, Point point, Point normal, double epsge)</code>	231
28.3.2.25	<code>splitWithTrimSegments(shared_ptr< BoundedSurface > surf, std::vector< shared_ptr< CurveOnSurface > > &bnd_cvs, double eps)</code>	231
28.3.2.26	<code>subtractSfPart(shared_ptr< BoundedSurface > surf, std::vector< shared_ptr< CurveOnSurface > > &bnd_cvs, double eps)</code>	231
28.3.2.27	<code>translateBoundedSurf(Point trans_vec, BoundedSurface &bd_sf, double deg_eps)</code>	231
28.3.2.28	<code>translatePlaneToCurves(shared_ptr< Go::Plane > &plane, std::vector< shared_ptr< Go::ParamCurve > > &space_crvs)</code>	232

28.3.2.29	trimSurfaceKinks(const BoundedSurface &sf, double max_normal_angle, std::vector< double > &g1_disc_u, std::vector< double > &g1_disc_v, bool compute_g1_disc=true)	232
28.3.2.30	trimSurfsWithSurfs(const std::vector< shared_ptr< ParamSurface > > &sfs1, const std::vector< shared_ptr< ParamSurface > > &sfs2, double epsge)	232
28.3.2.31	trimSurfWithSurf(const shared_ptr< ParamSurface > &sf1, const shared_ptr< ParamSurface > &sf2, double epsge)	232
28.3.2.32	trimWithPlane(const shared_ptr< ParamSurface > &surf, Point point, Point normal, double epsge)	232
28.4	Go::boxStructuring Namespace Reference	233
28.4.1	Detailed Description	233
28.5	Go::ClosestPoint Namespace Reference	233
28.5.1	Detailed Description	234
28.5.2	Function Documentation	234
28.5.2.1	closestPtCurves(const ParamCurve *cv1, const ParamCurve *cv2, double &par1, double &par2, double &dist, Point &ptc1, Point &ptc2)	234
28.5.2.2	closestPtCurves(const ParamCurve *cv1, const ParamCurve *cv2, double tmin1, double tmax1, double tmin2, double tmax2, double seed1, double seed2, double &par1, double &par2, double &dist, Point &ptc1, Point &ptc2)	234
28.5.2.3	closestPtCurves2D(ParamCurve *cv1, ParamCurve *cv2, double aepsge, double tmin1, double tmax1, double tmin2, double tmax2, double seed1, double seed2, int method, bool quick, double &par1, double &par2, double &dist, Point &ptc1, Point &ptc2, int &istat)	235
28.5.2.4	closestPtCurveSurf(ParamCurve *pcurve, ParamSurface *psurf, double aepsge, double astart1, double aend1, RectDomain *domain, double anext1, double enext2[], double &cpos1, double gpos2[], double &dist, Point &pt_cv, Point &pt_su, bool second_order=false)	235
28.5.2.5	closestPtCurveSurf(ParamCurve *pcurve, ParamSurface *psurf, double aepsge, double astart1, double estart2[], double aend1, double eend2[], double anext1, double enext2[], double &cpos1, double gpos2[], double &dist, Point &pt_cv, Point &pt_su, int &istat, bool second_order=false)	236
28.5.2.6	closestPtSurfSurfPlane(const std::vector< Point > &epoint, const std::vector< Point > &epnt1, const std::vector< Point > &epnt2, const Point &epar1, const Point &epar2, const ParamSurface *psurf1, const ParamSurface *psurf2, double aepsge, std::vector< Point > &gpnt1, std::vector< Point > &gpnt2, Point &gpar1, Point &gpar2, int &jstat, AlgorithmChoice algo=FUNCTIONAL)	237
28.5.2.7	closestPtSurfSurfPlaneFunctional(const std::vector< Point > &epoint, const std::vector< Point > &epnt1, const std::vector< Point > &epnt2, const Point &epar1, const Point &epar2, const ParamSurface *psurf1, const ParamSurface *psurf2, double aepsge, std::vector< Point > &gpnt1, std::vector< Point > &gpnt2, Point &gpar1, Point &gpar2, int &jstat)	237

28.5.2.8	<code>closestPtSurfSurfPlaneGeometrical(const std::vector< Point > &epoint, const std::vector< Point > &epnt1, const std::vector< Point > &epnt2, const Point &epar1, const Point &epar2, const ParamSurface *psurf1, const ParamSurface *psurf2, double aepsge, std::vector< Point > &gpnt1, std::vector< Point > &gpnt2, Point &gpar1, Point &gpar2, int &jstat)</code>	237
28.6	Go::cmUtils Namespace Reference	238
28.6.1	Detailed Description	238
28.6.2	Function Documentation	238
28.6.2.1	<code>abovePlane(Go::Point pt, Go::Point plane_pt, Go::Point normal)</code>	238
28.6.2.2	<code>ccwAngle(Go::Point first_leg, Go::Point second_leg, Go::Point *normal=NULL)</code>	239
28.6.2.3	<code>cwOrientation(std::vector< Go::ftEdgeBase * > &meeting_edges, std::vector< bool > &start, double angle_tol=1e-05)</code>	239
28.6.2.4	<code>cwOrientation2(std::vector< Go::ftEdgeBase * > &meeting_edges, std::vector< bool > &start)</code>	239
28.6.2.5	<code>estimatedCurveLength(ftEdgeBase *edge, int nmb_samples=4)</code>	239
28.6.2.6	<code>extendWithDegBd(std::vector< int > &corner, std::vector< shared_ptr< Go::ParamCurve > > &bd_curves, std::vector< shared_ptr< Go::ParamCurve > > &cross_curves, int idxmin)</code>	239
28.6.2.7	<code>geometricParamDomain(ParamSurface *sf)</code>	239
28.6.2.8	<code>getG1FaceCurves(Go::ftCurve &bd_curve)</code>	239
28.6.2.9	<code>insideBdTrain(const Go::Vector2D &sample_pt, const std::vector< Go::Vector2D > &bd_train)</code>	239
28.6.2.10	<code>removeInnerCorners(const std::vector< Go::ftEdgeBase * > &outer_loop, std::vector< int > &corners)</code>	240
28.6.2.11	<code>reparametrizeBdCvs(const std::vector< shared_ptr< Go::SplineCurve > > &bd_cvs, double appr_tol, std::vector< shared_ptr< Go::SplineCurve > > &new_bd_cvs)</code>	240
28.6.2.12	<code>reparametrizeBdCvs2(const std::vector< shared_ptr< Go::SplineCurve > > &bd_cvs, double appr_tol, std::vector< shared_ptr< Go::SplineCurve > > &new_bd_cvs)</code>	240
28.6.2.13	<code>splitEdgesInCorners(std::vector< shared_ptr< Go::ftEdgeBase > > &edges, const std::vector< Go::Point > &corner_pts, double gap)</code>	240
28.6.2.14	<code>updateWithNewCorners(const std::vector< Go::ftEdgeBase * > &outer_loop, std::vector< int > &corners, const std::vector< Go::Point > &add_corner_pts, double gap)</code>	240
28.7	Go::CoonsPatchGen Namespace Reference	240
28.7.1	Detailed Description	241
28.7.2	Function Documentation	241

28.7.2.1	<code>addMissingCrossCurves(const std::vector< shared_ptr< SplineCurve > > &bd_curves, std::vector< shared_ptr< SplineCurve > > &cross_crvs)</code>	241
28.7.2.2	<code>blendcoef(double evecu[], double evecv[], double etang[], int idim, int isign, double *coef1, double *coef2)</code>	241
28.7.2.3	<code>createCoonsPatch(const CurveLoop &boundary)</code>	242
28.7.2.4	<code>createCoonsPatch(std::vector< shared_ptr< ParamCurve > > &bd_curves, std::vector< shared_ptr< ParamCurve > > &cross_curves, double epsge, double kink_tol)</code>	242
28.7.2.5	<code>createCoonsPatch(std::vector< shared_ptr< SplineCurve > > &bd_curves, std::vector< shared_ptr< SplineCurve > > &cross_curves)</code>	242
28.7.2.6	<code>createGordonSurface(std::vector< shared_ptr< SplineCurve > > &mesh_← curves, std::vector< double > &params, int &nmb_u_crvs, bool use_param_← values)</code>	243
28.7.2.7	<code>createGordonSurface(std::vector< shared_ptr< SplineCurve > > &mesh_← curves, std::vector< double > &params, int &nmb_u_crvs, std::vector< shared_← _ptr< SplineCurve > > &cross_curves, std::vector< int > &cross_index, bool use_param_values=true)</code>	243
28.7.2.8	<code>doCreateSurface(std::vector< shared_ptr< SplineCurve > > &mesh_curves, std::vector< double > &params, int &nmb_u_crvs, std::vector< shared_ptr< SplineCurve > > &cross_curves, std::vector< int > &cross_index)</code>	243
28.7.2.9	<code>fixCrossEndPts(const std::vector< shared_ptr< SplineCurve > > &bd_curves, const std::vector< shared_ptr< SplineCurve > > &cross_curves)</code>	244
28.7.2.10	<code>getCrossTangs(const std::vector< shared_ptr< SplineCurve > > &curves, std_← ::vector< shared_ptr< SplineCurve > > &mod_cross_curves, double tol1, double tol2)</code>	244
28.7.2.11	<code>getTangBlends(std::vector< shared_ptr< SplineCurve > > &curves, int icedge, std::vector< shared_ptr< SplineCurve > > &blend_functions)</code>	244
28.7.2.12	<code>hermit(double econd[], int icond, bool hasder1, double astart, double aend, int idim)</code>	244
28.7.2.13	<code>loftSurface(std::vector< shared_ptr< SplineCurve > >::iterator first_curve, int nmb_crvs)</code>	245
28.7.2.14	<code>loftSurface(std::vector< shared_ptr< SplineCurve > >::iterator first_curve, std_← ::vector< double >::iterator first_param, int nmb_crvs)</code>	245
28.7.2.15	<code>loftSurface(std::vector< shared_ptr< SplineCurve > >::iterator first_curve, std_← ::vector< double >::iterator first_param, int nmb_crvs, std::vector< shared_ptr< SplineCurve > >::iterator first_cross_curve, std::vector< int > &cross_index)</code>	245
28.7.2.16	<code>makeLoftParams(std::vector< shared_ptr< SplineCurve > >::const_iterator first_curve, int nmb_crvs, double param_length, std::vector< double > &params)</code>	246
28.7.2.17	<code>reparamBoundaryCurve(std::vector< shared_ptr< SplineCurve > > &curves, double aconst)</code>	246
28.7.2.18	<code>sortMeshCurves(std::vector< shared_ptr< SplineCurve > > &mesh_curves, std::vector< double > &params, int nmb_u_crvs, std::vector< int > &cross_← index)</code>	246

28.7.2.19	splitMeshCurves(std::vector< shared_ptr< SplineCurve > > &mesh_curves, std::vector< double > ¶ms, int &nmb_u_crvs, std::vector< int > &cross_index, double epsgeo)	247
28.7.2.20	tpSurface(const std::vector< shared_ptr< SplineCurve > > &mesh_curves, std::vector< double > params, int nmb_u_crvs, const std::vector< shared_ptr< SplineCurve > > &cross_curves, std::vector< int > &cross_index)	247
28.8	Go::CoonsPatchVolumeGen Namespace Reference	247
28.8.1	Detailed Description	248
28.8.2	Function Documentation	248
28.8.2.1	createCoonsPatch(const Go::SplineSurface *surf_u_min, const Go::SplineSurface *surf_u_max, const Go::SplineSurface *surf_v_min, const Go::SplineSurface *surf_v_max, const Go::SplineSurface *surf_w_min, const Go::SplineSurface *surf_w_max, double tol=1.0e-05)	248
28.8.2.2	createCoonsPatchDirectly(const Go::SplineSurface *surf_u_min, const Go::SplineSurface *surf_u_max, const Go::SplineSurface *surf_v_min, const Go::SplineSurface *surf_v_max, const Go::SplineSurface *surf_w_min, const Go::SplineSurface *surf_w_max)	248
28.8.2.3	edge_curves_equal(shared_ptr< Go::SplineSurface > sf1, double par1, bool is_u_dir1, shared_ptr< Go::SplineSurface > sf2, double par2, bool is_u_dir2, double tol)	249
28.8.2.4	get_corners(shared_ptr< Go::SplineSurface > surf, std::vector< Go::Point > &pts)	249
28.8.2.5	push_corners(std::vector< Go::Point > &pts_to, const std::vector< Go::Point > pts_from, int pos0, int pos1, int pos2, int pos3)	249
28.9	Go::CreatorsUtils Namespace Reference	249
28.9.1	Detailed Description	250
28.9.2	Function Documentation	250
28.9.2.1	createCrossTangent(const Go::CurveOnSurface &cv)	250
28.9.2.2	createCrossTangent(const Go::CurveOnSurface &cv, shared_ptr< Go::SplineCurve > basis_space_cv, const Go::SplineCurve *cross_cv_ref, bool appr_offset_cv=true)	250
28.9.2.3	fixSeemCurves(shared_ptr< BoundedSurface > bd_sf, std::vector< shared_ptr< CurveOnSurface > > &loop_cvs, bool closed_dir_u, bool closed_dir_v, double tol)	250
28.9.2.4	fixTrimCurves(shared_ptr< Go::BoundedSurface > bd_sf, double epsgeo, frac=1.0, double tol=1.0e-3, double tol2=1.0e-2, double ang_tol=1.0e-2)	251
28.9.2.5	getParametricCurve(const std::vector< shared_ptr< const CurveOnSurface > > &cv)	251
28.9.2.6	projectCurvePoint(const ParamSurface *sf, bool closed_dir_u, bool closed_dir_v, const Go::ParamCurve *space_cv, double cv_par, double epsgeo=1e-04)	251

28.9.2.7	<code>projectCurvePoint(const SplineSurface &sf, bool closed_dir_u, bool closed_dir_v, const Go::ParamCurve *space_cv, double cv_par, double epsgeo=1e-04)</code>	251
28.9.2.8	<code>projectPoint(const Go::ParamSurface *sf, bool closed_dir_u, bool closed_dir_v, const Go::Point &space_pt, double epsgeo=1e-04)</code>	252
28.10	<code>Go::Curvature</code> Namespace Reference	252
28.10.1	Detailed Description	252
28.10.2	Function Documentation	252
28.10.2.1	<code>curvatureRadiusPoints(const SplineCurve &curve, double curveRad, std::vector< double > &pos)</code>	252
28.10.2.2	<code>minimalCurvatureRadius(const SplineCurve &curve, double &mincurv, double &pos)</code>	252
28.11	<code>Go::CurvatureAnalysis</code> Namespace Reference	253
28.11.1	Detailed Description	253
28.11.2	Function Documentation	253
28.11.2.1	<code>computeFirstFundamentalForm(const ParamSurface &sf, double u, double v, int derivs, std::vector< double > &form)</code>	253
28.11.2.2	<code>computeSecondFundamentalForm(const ParamSurface &sf, double u, double v, double form1[3], double form2[3])</code>	253
28.11.2.3	<code>curvatures(const ParamSurface &sf, double u, double v, double &K, double &H)</code> .	254
28.11.2.4	<code>evaluateMinCurvatureRadius(const ParamSurface &sf, double start_u, double end_u, double start_v, double end_v, double tolerance, std::vector< double > &param_u, std::vector< double > &param_v, std::vector< std::vector< double > > &curvs, double &mincurv, double &minpos_u, double &minpos_v, bool initialize)</code> .	254
28.11.2.5	<code>minimalCurvatureRadius(const ParamSurface &sf, double tolerance, double &mincurv, double &pos_u, double &pos_v, double degenerate_eps, double curv_tol=1.0e-3)</code>	254
28.11.2.6	<code>principalCurvatures(const ParamSurface &sf, double u, double v, double &k1, Point &d1, double &k2, Point &d2)</code>	255
28.12	<code>Go::CurveCreators</code> Namespace Reference	255
28.12.1	Detailed Description	255
28.12.2	Function Documentation	255
28.12.2.1	<code>approxCurves(shared_ptr< ParamCurve > *first_crv, shared_ptr< ParamCurve > *last_crv, const std::vector< Point > &start_pt, const std::vector< Point > &end_pt, double approxtol, double &maxdist, int max_iter=5, int degree=3)</code>	255
28.12.2.2	<code>blend(const SplineCurve &alpha_1, const SplineCurve &f_1, const SplineCurve &alpha_2, const SplineCurve &f_2)</code>	256
28.12.2.3	<code>createCircle(Point center, Point axis, Point normal, double radius)</code>	256

28.12.2.4	<code>curveApprox(shared_ptr< ParamCurve > cvs[], int nmb_cvs, const BsplineBasis &init_basis, double tol)</code>	257
28.12.2.5	<code>curveApprox(shared_ptr< ParamCurve > cvs[], int nmb_cvs, double tol, double degree=3)</code>	257
28.12.2.6	<code>insertParamDomain(const Go::SplineCurve &cv_1d, double knot_tol=1e-08)</code>	257
28.12.2.7	<code>liftParameterCurve(shared_ptr< ParamCurve > &parameter_cv, shared_ptr< ParamSurface > &surf, double epsge)</code>	257
28.12.2.8	<code>multCurveWithFunction(const SplineCurve &alpha, const SplineCurve &f)</code>	258
28.12.2.9	<code>offsetCurve(const SplineCurve &cv, Point offset_val)</code>	258
28.12.2.10	<code>projectCurve(shared_ptr< ParamCurve > &space_cv, shared_ptr< ParamSurface > &surf, double epsge, shared_ptr< SplineCurve > &proj_cv, shared_ptr< SplineCurve > &par_cv)</code>	258
28.12.2.11	<code>projectSpaceCurve(shared_ptr< ParamCurve > &space_cv, shared_ptr< ParamSurface > &surf, shared_ptr< Point > &start_par_pt, shared_ptr< Point > &end_par_pt, double epsge, const RectDomain *domain_of_interest=NULL)</code>	258
28.13	<code>Go::CurveInterpolator</code> Namespace Reference	259
28.13.1	Detailed Description	259
28.13.2	Function Documentation	259
28.13.2.1	<code>regularInterpolation(const BsplineBasis &basis, std::vector< double > &par, std::vector< double > &points, int dimension, bool rational, std::vector< double > &weights)</code>	259
28.14	<code>Go::FaceUtilities</code> Namespace Reference	259
28.14.1	Detailed Description	259
28.14.2	Function Documentation	260
28.14.2.1	<code>enforceCoLinearity(ftSurface *face1, ftEdge *edge1, ftSurface *face2, double tol, double ang_tol)</code>	260
28.14.2.2	<code>enforceVxCoLinearity(shared_ptr< Vertex > vx, double tol, double ang_tol)</code>	260
28.14.2.3	<code>getBoundaryData(ftSurface *face, int nmb_sample, std::vector< SamplePointData > &sample_points)</code>	260
28.14.2.4	<code>getInnerData(ftSurface *face, int nmb_sample_u, int nmb_sample_v, std::vector< SamplePointData > &sample_points)</code>	260
28.15	<code>Go::ftVolumeTools</code> Namespace Reference	260
28.15.1	Detailed Description	260
28.15.2	Function Documentation	260
28.15.2.1	<code>boundaryStatus(ftVolume *vol, shared_ptr< ftSurface > &bd_face, double tol)</code>	260

28.15.2.2	<code>splitVolumes(shared_ptr< ftVolume > &vol1, shared_ptr< ftVolume > &vol2, double eps, std::vector< int > &config)</code>	261
28.15.2.3	<code>splitVolumes(shared_ptr< ftVolume > &vol, shared_ptr< ftSurface > &face, double eps)</code>	261
28.15.2.4	<code>updateWithSplitFaces(shared_ptr< SurfaceModel > shell, shared_ptr< ftSurface > &face1, shared_ptr< ftSurface > &face2, std::vector< std::pair< ftEdge *, ftEdge * > > &replaced_wires)</code>	261
28.16	<code>Go::GapRemoval Namespace Reference</code>	261
28.16.1	Detailed Description	262
28.16.2	Function Documentation	262
28.16.2.1	<code>checkBoundaryDist(shared_ptr< CurveOnSurface > bd1, shared_ptr< CurveOnSurface > bd2, double start1, double end1, double start2, double end2, int nmb_sample, double &mdist1, double &mdist2)</code>	262
28.16.2.2	<code>getBoundarySamples(std::vector< shared_ptr< CurveOnSurface > > &all_bd, std::vector< shared_ptr< CurveOnSurface > > &bd_cv, std::vector< double > &pts, std::vector< double > &pars, std::vector< int > &bd_idx, double epsge)</code>	262
28.16.2.3	<code>getCoefConstraints(shared_ptr< SplineCurve > &crv1, int idx1, shared_ptr< SplineCurve > &crv2, int idx2, double tol)</code>	263
28.16.2.4	<code>getSplineAndBd(shared_ptr< ParamSurface > psurf, std::vector< shared_ptr< CurveOnSurface > > &bd_crvs)</code>	263
28.16.2.5	<code>modifyAtVertex(shared_ptr< SplineSurface > srf, Point face_param, Point vertex, double epsge)</code>	263
28.16.2.6	<code>modifySplines(shared_ptr< ParamSurface > &srf1, std::vector< shared_ptr< CurveOnSurface > > &bd_cv1, std::vector< double > &start1, std::vector< double > &end1, shared_ptr< ParamSurface > &srf2, std::vector< shared_ptr< CurveOnSurface > > &bd_cv2, std::vector< double > &start2, std::vector< double > &end2, std::vector< Point > &vertex, double epsge)</code>	263
28.16.2.7	<code>modifySplineSf(shared_ptr< ParamSurface > &srf1, std::vector< shared_ptr< CurveOnSurface > > &bd_cv1, std::vector< double > start1, std::vector< double > end1, shared_ptr< ParamSurface > &srf2, std::vector< shared_ptr< CurveOnSurface > > &bd_cv2, std::vector< double > start2, std::vector< double > end2, double epsge)</code>	263
28.16.2.8	<code>removeGapSpline(shared_ptr< SplineVolume > &vol1, shared_ptr< SurfaceOnVolume > &bd_sf1, double sf1_start1, double sf1_end1, double sf1_start2, double sf1_end2, shared_ptr< SplineVolume > &vol2, shared_ptr< SurfaceOnVolume > &bd_sf2, double sf2_start1, double sf2_end1, double sf2_start2, double sf2_end2, Point vertex_ll, Point vertex_ur, double epsge, int orientation)</code>	263
28.16.2.9	<code>removeGapSpline(shared_ptr< SplineSurface > &srf1, shared_ptr< CurveOnSurface > &bd_cv1, double start1, double end1, shared_ptr< SplineSurface > &srf2, shared_ptr< CurveOnSurface > &bd_cv2, double start2, double end2, Point vertex1, Point vertex2, double epsge, bool *same_orientation=NULL)</code>	263
28.16.2.10	<code>removeGapSpline2(std::vector< shared_ptr< CurveOnSurface > > &bd_cv1, std::vector< double > &start1, std::vector< double > &end1, std::vector< shared_ptr< CurveOnSurface > > &bd_cv2, std::vector< double > &start2, std::vector< double > &end2, std::vector< Point > &vertex, double epsge)</code>	264

28.16.2.11	removeGapSplineTrim(shared_ptr< SplineSurface > &srf1, std::vector< shared_ptr< CurveOnSurface > > &bd_cv1, std::vector< double > start1, std::vector< double > end1, std::vector< shared_ptr< CurveOnSurface > > &bd_cv2, std::vector< double > start2, std::vector< double > end2, Point vertex1, Point vertex2, double epsge)	264
28.16.2.12	removeGapTrim(shared_ptr< CurveOnSurface > &bd_cv1, double start1, double end1, shared_ptr< CurveOnSurface > &bd_cv2, double start2, double end2, Point vertex1, Point vertex2, double epsge)	264
28.16.2.13	replaceCurvePiece(shared_ptr< SplineCurve > crv, shared_ptr< SplineCurve > sub_crv, double par1, int cont1, double par2, int cont2)	264
28.17	Go::GeometryTools Namespace Reference	264
28.17.1	Function Documentation	266
28.17.1.1	analyzePeriodicity(const SplineCurve &cv, double knot_tol=1e-12)	266
28.17.1.2	analyzePeriodicity(const SplineSurface &sf, int direction, double knot_tol=1e-12)	266
28.17.1.3	analyzePeriodicity(const BsplineBasis &basis, double knot_tol=1e-12)	267
28.17.1.4	analyzePeriodicityDerivs(const ParamCurve &cv, int max_derivs, double tol=1e-14)	267
28.17.1.5	analyzePeriodicityDerivs(const SplineSurface &sf, int direction, int max_derivs, double tol=1e-14)	267
28.17.1.6	averageBoundaryCoefs(shared_ptr< SplineSurface > &srf1, int bd1, bool keep_first, shared_ptr< SplineSurface > &srf2, int bd2, bool keep_second, bool found_corner1, Point corner1, bool found_corner2, Point corner2, bool opposite)	268
28.17.1.7	checkConstantCoef(SplineCurve &cv, int idx, double val, double max_dist, double tol)	268
28.17.1.8	curveKinks(const SplineCurve &cv, double tol, double ang_tol, std::vector< double > &c1_disconts, std::vector< double > &g1_disconts)	268
28.17.1.9	curveSum(const SplineCurve &crv1, double fac1, const SplineCurve &crv2, double fac2, double num_tol=1e-05)	268
28.17.1.10	degenerateToCurve(const SplineSurface &srf, bool dir_u, double tol)	268
28.17.1.11	estimateIsoCurveLength(const SplineSurface &srf, bool dir_u, double par, double &length)	268
28.17.1.12	estimateSurfaceSize(const ParamSurface &srf, double &length_u, double &length_v, double *area=NULL)	269
28.17.1.13	findDominant(const SplineSurface &surface, Vector3D &dominant_u, Vector3D &dominant_v)	269
28.17.1.14	getGnJoints(const ParamCurve &curve, const std::vector< double > &cont, std::vector< double > &gn_joints)	269
28.17.1.15	getGnJoints(const CurveLoop &loop, const std::vector< double > &cont, std::vector< std::vector< double > > &gn_joints)	270
28.17.1.16	getKnotAtLargestInterval(const BsplineBasis &basis)	270

28.17.1.17	<code>getLargestParameterInterval(const BsplineBasis &basis)</code>	270
28.17.1.18	<code>getRotationMatrix(const Point &unit_axis_dir, double alpha)</code>	270
28.17.1.19	<code>insertKnotsEvenly(BsplineBasis &basis, int num_knots)</code>	271
28.17.1.20	<code>insertKnotsEvenly(BsplineBasis &basis, double tmin, double tmax, int num_knots, double knot_diff_tol=1e-05)</code>	271
28.17.1.21	<code>isCoincident(const ParamCurve &cv1, const ParamCurve &cv2, double epsge)</code>	271
28.17.1.22	<code>joinPatches(const std::vector< shared_ptr< SplineSurface > > &patches, const SplineSurface &spline_space)</code>	271
28.17.1.23	<code>makeBdDegenerate(SplineSurface &srf, int bd_idx)</code>	271
28.17.1.24	<code>makeUnionKnots(std::vector< BsplineBasis > &bbasis, double tol, std::vector< double > &union_knots)</code>	272
28.17.1.25	<code>makeUnionKnots(std::vector< std::vector< double > > &knots, double tol, std::vector< double > &union_knots)</code>	272
28.17.1.26	<code>negativeProj(const SplineSurface &surface, const Array< Vector3D, 2 > &refvector, const double eps=0.0)</code>	272
28.17.1.27	<code>projectCurve(shared_ptr< ParamCurve > incurve, const Point &normal, bool planar)</code>	272
28.17.1.28	<code>projectCurve(const SplineCurve &incurve, const Point &normal, bool planar)</code>	272
28.17.1.29	<code>representCurveAsSurface(const SplineCurve &curve, int cv_dir, const BsplineBasis &other_bas, bool rational)</code>	273
28.17.1.30	<code>representSurfaceAsCurve(const SplineSurface &surface, int cv_dir)</code>	273
28.17.1.31	<code>rotateLineCloud(Point rot_axis, double alpha, LineCloud &lc)</code>	274
28.17.1.32	<code>rotatePoint(Point rot_axis, double alpha, double *space_pt)</code>	274
28.17.1.33	<code>rotatePoint(Point rot_axis, double alpha, Point &space_pt)</code>	274
28.17.1.34	<code>rotateSplineCurve(Point rot_axis, double alpha, SplineCurve &cv)</code>	274
28.17.1.35	<code>rotateSplineSurf(Point rot_axis, double alpha, SplineSurface &sf)</code>	275
28.17.1.36	<code>setParameterDomain(std::vector< shared_ptr< BoundedSurface > > &sfs, double u1, double u2, double v1, double v2)</code>	275
28.17.1.37	<code>setSfBdCoefToConst(SplineSurface &srf, int bd_idx, int idx_d, double val, double deg_tol)</code>	275
28.17.1.38	<code>splitCurveIntoSegments(const SplineCurve &cv, std::vector< SplineCurve > &seg)</code>	275
28.17.1.39	<code>splitInKinks(const SplineSurface &sf, const std::vector< double > &u_kinks, const std::vector< double > &v_kinks)</code>	275
28.17.1.40	<code>splitSurfaceIntoPatches(const SplineSurface &sf, std::vector< SplineSurface > &pat)</code>	275

28.17.1.41	surfaceKinks(const SplineSurface &sf, double max_normal_angle, std::vector< double > &g1_disc_u, std::vector< double > &g1_disc_v, bool compute_g1_↔disc=true)	275
28.17.1.42	surfaceSum(const SplineSurface &sf1, double fac1, const SplineSurface &sf2, double fac2, double num_tol=1e-05)	276
28.17.1.43	translateLineCloud(const Point &trans_vec, LineCloud &lc)	276
28.17.1.44	translateSplineCurve(const Point &trans_vec, SplineCurve &cv)	276
28.17.1.45	translateSplineSurf(const Point &trans_vec, SplineSurface &sf)	276
28.17.1.46	unifyCurveSplineSpace(std::vector< shared_ptr< SplineCurve > > &curves, double tol)	276
28.17.1.47	unifySurfaceSplineSpace(std::vector< shared_ptr< SplineSurface > > &surfaces, double tol, int dir=0)	276
28.17.1.48	unifySurfaceSplineSpaceOneDir(std::vector< shared_ptr< SplineSurface > > &surfaces, double tol, bool unify_u_dir)	276
28.18	Go::HahnsSurfaceGen Namespace Reference	276
28.18.1	Detailed Description	277
28.18.2	Function Documentation	277
28.18.2.1	constructHahnsSurface(std::vector< shared_ptr< Go::SplineCurve > > &bnd_↔_curves, std::vector< shared_ptr< Go::SplineCurve > > &mod_cross_curves, double neighbour_tol, double kink_tol, double knot_diff_tol)	277
28.18.2.2	constructPolygonalSurface(std::vector< shared_ptr< Go::ParamCurve > > &bnd_↔_curves, std::vector< shared_ptr< Go::ParamCurve > > &cross_curves, double neighbour_tol, double kink_tol, double knot_diff_tol)	277
28.19	Go::IntersectionUtils Namespace Reference	278
28.19.1	Detailed Description	278
28.19.2	Function Documentation	278
28.19.2.1	create1DSplineCurve(const SplineCurve &cv, int dim_id)	278
28.19.2.2	create1DSplineSurface(const SplineSurface &sf, int dim_id)	278
28.19.2.3	distImplRepresentationCompFunction(const SplineSurface &spline_sf, const BernsteinTetrahedralPoly &impl, const BaryCoordSystem3D &bc, const Spline_↔Surface &comp_1d_sf, double upar, double vpar)	278
28.19.2.4	insertCvInAlgcv(const SplineCurve &cv, AlgObj2DInt *alg_obj2d_int)	279
28.19.2.5	insertSfnInAlgsf(const SplineSurface &sf, AlgObj3DInt *alg_obj3d_int)	279
28.19.2.6	insertSfnInAlgsf2(const SplineSurface &sf, AlgObj3DInt *alg_obj3d_int)	279
28.19.2.7	insertSfnImplObj(const SplineSurface &spline_sf, const BernsteinTetrahedral_↔Poly &impl, const BaryCoordSystem3D &bc)	279

28.19.2.8 splineCurveProduct(std::vector< shared_ptr< SplineCurve > > &cv, Alg2DElem term)	279
28.19.2.9 splineSurfaceProduct(std::vector< shared_ptr< SplineSurface > > &sf, Alg3DElem term)	279
28.20Go::LinDepUtils Namespace Reference	279
28.20.1 Function Documentation	279
28.20.1.1 isPeelable(const LRSplineSurface &)	279
28.20.1.2 unpeelableBasisFunctions(const LRSplineSurface &)	279
28.21Go::LoftSurfaceCreator Namespace Reference	279
28.21.1 Detailed Description	280
28.21.2 Function Documentation	280
28.21.2.1 loftNonrationalSurface(std::vector< shared_ptr< SplineCurve > >::iterator first_curve, std::vector< double >::iterator first_param, int nmb_crvs)	280
28.21.2.2 loftRationalSurface(std::vector< shared_ptr< SplineCurve > >::iterator first_curve, std::vector< double >::iterator first_param, int nmb_crvs)	280
28.21.2.3 loftSurface(std::vector< shared_ptr< SplineCurve > >::iterator first_curve, int nmb_crvs)	280
28.21.2.4 loftSurface(std::vector< shared_ptr< SplineCurve > >::iterator first_curve, std::vector< double >::iterator first_param, int nmb_crvs)	281
28.21.2.5 loftSurface(std::vector< shared_ptr< SplineCurve > > &loft_curves, int nmb_crvs)	281
28.21.2.6 loftSurfaceFromUnifiedCurves(std::vector< shared_ptr< SplineCurve > >::iterator first_curve, std::vector< double >::iterator first_param, int nmb_crvs)	281
28.21.2.7 makeLoftParams(std::vector< shared_ptr< SplineCurve > >::const_iterator first_curve, int nmb_crvs, double param_length, std::vector< double > ¶ms)	282
28.21.2.8 unifiedCurvesCopy(std::vector< shared_ptr< SplineCurve > >::iterator first_curve, int nmb_crvs)	282
28.22Go::LoftVolumeCreator Namespace Reference	282
28.22.1 Detailed Description	283
28.22.2 Function Documentation	283
28.22.2.1 loftNonrationalVolume(std::vector< shared_ptr< SplineSurface > >::iterator first_surface, std::vector< double >::iterator first_param, int nmb_srf)	283
28.22.2.2 loftRationalVolume(std::vector< shared_ptr< SplineSurface > >::iterator first_surface, std::vector< double >::iterator first_param, int nmb_srf)	283
28.22.2.3 loftVolume(std::vector< shared_ptr< SplineSurface > >::iterator first_surface, int nmb_srf)	284

28.22.2.4	loftVolume(std::vector< shared_ptr< SplineSurface > >::iterator first_surface, std::vector< double >::iterator first_param, int nmb_srfs)	284
28.22.2.5	loftVolumeFromUnifiedSurfaces(std::vector< shared_ptr< SplineSurface > >::iterator first_surface, std::vector< double >::iterator first_param, int nmb_srfs)	284
28.22.2.6	makeLoftParams(std::vector< shared_ptr< SplineSurface > >::const_iterator first_surface, int nmb_crvs, double param_length, std::vector< double > ¶ms)	285
28.22.2.7	unifiedSurfacesCopy(std::vector< shared_ptr< SplineSurface > >::iterator first_surface, int nmb_srfs)	285
28.23	Go::LoopUtils Namespace Reference	285
28.23.1	Detailed Description	286
28.23.2	Function Documentation	286
28.23.2.1	firstLoopInsideSecond(const std::vector< shared_ptr< Go::CurveOnSurface > > &first_loop, const std::vector< shared_ptr< Go::CurveOnSurface > > &second_loop, double loop_tol, double int_tol)	286
28.23.2.2	loopsCCW(const std::vector< shared_ptr< Go::SplineCurve > > &simple_←par_loop, double space_epsilon, double int_tol)	286
28.23.2.3	loopsCCW(const CurveLoop &loop, double int_tol)	287
28.23.2.4	makeLoopCCW(std::vector< shared_ptr< ParamCurve > > &loop_cvcs, double tol)	287
28.23.2.5	paramsCCW(const std::vector< shared_ptr< Go::CurveOnSurface > > &loop, double space_epsilon, double int_tol)	287
28.23.2.6	representAsSurfaceCurves(std::vector< shared_ptr< ParamCurve > > &curves, shared_ptr< BoundedSurface > surf, std::vector< shared_ptr< CurveOnSurface > > &cvcs_on_sf)	287
28.24	Go::LRApproxApp Namespace Reference	287
28.24.1	Function Documentation	288
28.24.1.1	categorizeCloudFromDist(std::vector< double > &points, shared_ptr< LR←SplineSurface > &surf, std::vector< double > &limits, double &max_above, double &max_below, double &avdist, int &nmb_points, std::vector< int > &classification, std::vector< int > &nmb_group)	288
28.24.1.2	categorizeCloudFromDist_omp(std::vector< double > &points, shared_ptr< LR←RSplineSurface > &surf, std::vector< double > &limits, double &max_above, double &max_below, double &avdist, int &nmb_points, std::vector< int > &classification, std::vector< int > &nmb_group)	288
28.24.1.3	classifyCloudFromDist(std::vector< double > &points, shared_ptr< LRSpline←Surface > &surf, std::vector< double > &limits, double &max_above, double &max_below, double &avdist, int &nmb_points, std::vector< std::vector< double > > &level_points, std::vector< int > &nmb_group)	288
28.24.1.4	classifyCloudFromDist_omp(std::vector< double > &points, shared_ptr< LR←SplineSurface > &surf, std::vector< double > &limits, double &max_above, double &max_below, double &avdist, int &nmb_points, std::vector< std::vector< double > > &level_points, std::vector< int > &nmb_group)	288

28.24.1.5	computeDistPointSpline(std::vector< double > &points, shared_ptr< LRSplineSurface > &surf, double &max_above, double &max_below, double &avdist, int &nmb_points, std::vector< double > &pointsdist)	288
28.24.1.6	computeDistPointSpline_omp(std::vector< double > &points, shared_ptr< LRSplineSurface > &surf, double &max_above, double &max_below, double &avdist, int &nmb_points, std::vector< double > &pointsdist)	288
28.24.1.7	pointCloud2Spline(std::vector< double > &points, int dim, double domain[], double reduced_domain[], double eps, int max_iter, shared_ptr< LRSplineSurface > &surf, double &maxdist, double &avdist, double &avdist_out, int &nmb_out)	288
28.25	Go::LRBSpline2DUtils Namespace Reference	288
28.25.1	Function Documentation	289
28.25.1.1	derive_knots(const Mesh2D &m, Direction2D d, int beg, int end, int orto_min, int orto_max)	289
28.25.1.2	split_function(const LRBSpline2D &orig, const Mesh2D &mesh, Direction2D d, const double *const knotvalues, int new_knot_ix, LRBSpline2D *&new_1, LRBSpline2D *&new_2)	289
28.25.1.3	split_several(const double *knotvals, const std::vector< int > &k_vec_in, const std::vector< int > &new_knots, std::vector< int > &k_vec_out, std::vector< double > &b_spline_weighths)	289
28.25.1.4	try_split_once(const LRBSpline2D &b, const Mesh2D &mesh, LRBSpline2D *&b1, LRBSpline2D *&b2)	289
28.26	Go::LRSplineMBA Namespace Reference	289
28.26.1	Function Documentation	289
28.26.1.1	add_contribution(int dim, std::map< const LRBSpline2D *, Array< double, 2 > > &target, const LRBSpline2D *bspline, double nom[], double denom)	289
28.26.1.2	add_contribution2(int dim, std::map< const LRBSpline2D *, Array< double, 4 > > &target, const LRBSpline2D *bspline, double nom[], double denom)	289
28.26.1.3	MBADistAndUpdate(LRSplineSurface *srf)	289
28.26.1.4	MBADistAndUpdate_omp(LRSplineSurface *srf)	289
28.26.1.5	MBAUpdate(LRSplineSurface *srf)	290
28.26.1.6	MBAUpdate(LRSplineSurface *srf, std::vector< Element2D * > &elems, std::vector< Element2D * > &elems2)	290
28.26.1.7	MBAUpdate_omp(LRSplineSurface *srf)	290
28.27	Go::LRSplineUtils Namespace Reference	290
28.27.1	Function Documentation	291
28.27.1.1	all_meshlines_uniform(Direction2D d, const Mesh2D &m)	291
28.27.1.2	compute_alpha(int degree, const int *const oldvec_ix, const int *const newvec_ix, const double *const kval)	291

28.27.1.3	<code>compute_greville(const std::vector< int > &v_ixs, const double *const vals)</code> . . .	291
28.27.1.4	<code>compute_greville(int deg, const std::vector< int > &k_vec_in, const double *const knotvals)</code>	291
28.27.1.5	<code>distributeDataPoints(LRSplineSurface *srf, std::vector< double > &points, bool add_distance_field=false, bool primary_points=true)</code>	291
28.27.1.6	<code>elementLineClouds(const LRSplineSurface &lr_spline_sf)</code>	291
28.27.1.7	<code>elementOK(const Element2D *elem, const Mesh2D &m)</code>	291
28.27.1.8	<code>fullTensorProductSurface(const LRSplineSurface &lr_spline_sf)</code>	291
28.27.1.9	<code>identify_elements_from_mesh(const Mesh2D &m)</code>	291
28.27.1.10	<code>increment_knotvec_indices(LRSplineSurface::BSplineMap &bmap, Direction2D d, int from_ix)</code>	291
28.27.1.11	<code>insert_basis_function(std::unique_ptr< LRBSpline2D > &b, const Mesh2D &mesh, LRSplineSurface::BSplineMap &bmap)</code>	291
28.27.1.12	<code>insert_knots(const std::vector< int > &new_knots, std::unique_ptr< LRBSpline2D > &bfun, const Direction2D d, const double *const kval)</code>	291
28.27.1.13	<code>insertParameterFunctions(LRSplineSurface *lr_spline_sf)</code>	291
28.27.1.14	<code>iteratively_split(std::vector< std::unique_ptr< LRBSpline2D > > &bfuncs, const Mesh2D &mesh)</code>	291
28.27.1.15	<code>iteratively_split2(std::vector< LRBSpline2D * > &bsplines, const Mesh2D &mesh, LRSplineSurface::BSplineMap &bmap, double domain[])</code>	291
28.27.1.16	<code>knots_to_insert(const std::vector< int > &ref, const std::vector< int > &mults)</code>	291
28.27.1.17	<code>locate_interval(const Mesh2D &m, Direction2D d, double value, double other_value, bool at_end)</code>	291
28.27.1.18	<code>mostComparableBspline(LRSplineSurface *lr_spline_sf, Point pos)</code>	292
28.27.1.19	<code>refine_mesh(Direction2D d, double fixed_val, double start, double end, int mult, bool absolute, int spline_degree, double knot_tol, Mesh2D &mesh, LRSplineSurface::BSplineMap &bmap)</code>	292
28.27.1.20	<code>set_uniform_meshlines(Direction2D d, Mesh2D &mesh)</code>	292
28.27.1.21	<code>support_equal(const LRBSpline2D *b1, const LRBSpline2D *b2)</code>	292
28.27.1.22	<code>tensor_split(std::unique_ptr< LRBSpline2D > &bfun, const std::vector< int > &x_mults, const std::vector< int > &y_mults, const Mesh2D &tensor_mesh, LRSplineSurface::BSplineMap &bmap)</code>	292
28.27.1.23	<code>update_elements_with_single_bspline(LRBSpline2D *b, LRSplineSurface::ElementMap &emap, const Mesh2D &mesh, bool remove)</code>	292
28.28	Go::LRSurfStitch Namespace Reference	292
28.28.1	Function Documentation	292

28.28.1.1	averageCorner(std::vector< std::pair< shared_ptr< ParamSurface >, int > > &sfs, double tol)	292
28.28.1.2	averageEdge(shared_ptr< ParamSurface > surf1, int edge1, shared_ptr< ParamSurface > surf2, int edge2, double tol)	292
28.28.1.3	averageEdge(shared_ptr< LRSplineSurface > surf1, int edge1, shared_ptr< LRSplineSurface > surf2, int edge2, double tol)	292
28.28.1.4	defineRefinements(const Mesh2D &mesh, Direction2D dir, int edge, int ix, std::vector< double > &knot_vals, int element_width, std::vector< LRSplineSurface::Refinement2D > &refs)	292
28.28.1.5	extractBoundaryBsplines(shared_ptr< LRSplineSurface > surf, int edge, std::vector< LRBSpline2D * > &bsplines)	293
28.28.1.6	extractMissingKnots(std::vector< double > &union_vec, std::vector< double > &vec, double tol, int order, std::vector< double > &resvec)	293
28.28.1.7	fetchEdgeCorners(shared_ptr< LRSplineSurface > surf, int edge, double &u1, double &v1, double &u2, double &v2)	293
28.29Go::	Mesh2DUtills Namespace Reference	293
28.29.1	Function Documentation	293
28.29.1.1	first_larger_knotvalue_ix(const Mesh2D &m, Direction2D d, double par)	293
28.29.1.2	identify_patch_lower_left(const Mesh2D &m, double u, double v, int &x_ix, int &y_ix)	293
28.29.1.3	identify_patch_upper_right(const Mesh2D &m, double u, double v, int &x_ix, int &y_ix)	293
28.29.1.4	last_nonlarger_knotvalue_ix(const Mesh2D &m, Direction2D d, double par)	293
28.29.1.5	search_downwards_for_nonzero_multiplicity(const Mesh2D &m, Direction2D d, int start_ix, int other_ix)	293
28.29.1.6	search_downwards_for_nonzero_multiplicity(const Mesh2D &m, Direction2D d, int start_ix, int ix1, int ix2)	293
28.29.1.7	search_upwards_for_nonzero_multiplicity(const Mesh2D &m, Direction2D d, int start_ix, int other_ix)	293
28.29.1.8	search_upwards_for_nonzero_multiplicity(const Mesh2D &m, Direction2D d, int start_ix, int ix1, int ix2)	293
28.30Go::	ModifySurf Namespace Reference	293
28.30.1	Detailed Description	294
28.30.2	Function Documentation	294
28.30.2.1	enforceCoefCoLinearity(shared_ptr< SplineSurface > sf1, int bd1, shared_ptr< SplineSurface > sf2, int bd2, double tol, std::vector< std::vector< int > > &enumeration)	294

28.30.2.2	<code>enforceVxCoefCoLinearity(std::vector< shared_ptr< SplineSurface > > &sfs, std::vector< int > &vx_enum, std::vector< std::pair< std::vector< int >, std::pair< int, int > > > &coef_cond, double tol)</code>	294
28.30.2.3	<code>replaceBoundary(shared_ptr< SplineSurface > surf, shared_ptr< SplineCurve > curve, int bd_idx, double tol)</code>	294
28.31	Go::OffsetSurfaceUtils Namespace Reference	294
28.31.1	Function Documentation	294
28.31.1.1	<code>offsetSurfaceSet(const std::vector< shared_ptr< ParamSurface > > &param_sfs, double offset_dist, double epsgeo, shared_ptr< SplineSurface > &offset_sf)</code>	294
28.32	Go::OffsetUtils Namespace Reference	295
28.32.1	Detailed Description	295
28.32.2	Function Documentation	295
28.32.2.1	<code>blend_s1421(const SplineSurface *ps, double aoffset, int ider, const Point &epar, int &ilfs, int &ilft, std::vector< Point > &eoffpnt, std::vector< Point > &epnt, int *jstat)</code>	295
28.33	Go::orientCurves Namespace Reference	295
28.33.1	Detailed Description	295
28.33.2	Function Documentation	295
28.33.2.1	<code>orientCurves(const std::vector< PtrToCurveType > &curves, std::vector< int > &permutation, std::vector< bool > &reversed, double neighbour_tol, bool assume_manifold=true)</code>	295
28.34	Go::Path Namespace Reference	296
28.34.1	Detailed Description	296
28.34.2	Function Documentation	296
28.34.2.1	<code>classifyCorners(const std::vector< ftEdge * > &edges, double tol, std::vector< shared_ptr< Vertex > > &corner, std::vector< shared_ptr< Vertex > > &non_corner)</code>	296
28.34.2.2	<code>closestPoint(std::vector< ftEdge * > edges, const Point &pt, int &clo_ind, double &clo_par, Point &clo_pt, double &clo_dist)</code>	297
28.34.2.3	<code>edgeChain(ftEdge *edg, double angtol, shared_ptr< Vertex > &v1, shared_ptr< Vertex > &v2)</code>	297
28.34.2.4	<code>estimateHoleInfo(const std::vector< ftEdge * > &edges, Point &centre, Point &axis, double &radius, double &angle)</code>	297
28.34.2.5	<code>getEdgeCurves(std::vector< ftEdge * > &loop, std::vector< shared_ptr< ParamCurve > > &space_cvs, std::vector< Point > &joint_points, double eps, double tol, bool corner_in_Tjoint=true)</code>	297
28.34.2.6	<code>identifyLoop(std::vector< ftEdge * > edges, shared_ptr< Vertex > vx)</code>	297

28.35Go::PointSetApp Namespace Reference	297
28.35.1 Detailed Description	297
28.35.2 Function Documentation	298
28.35.2.1 parameterizeTriang(const double *xyz_points, int nmbp, const int *triangles, int nmbt, std::vector< double > &uv_pars)	298
28.35.2.2 recognizeBoundary(shared_ptr< ftPointSet > &triang, std::vector< int > &corner_ix)	298
28.35.2.3 recognizeCornerNodes(const shared_ptr< ftPointSet > &triang, const std::vector< int > &bd_nodes, std::vector< int > &corner_ix)	298
28.36Go::qualityUtils Namespace Reference	298
28.36.1 Function Documentation	298
28.36.1.1 estimateArea(shared_ptr< ParamSurface > surf)	298
28.36.1.2 estimateLoopArea(shared_ptr< CurveLoop > loop)	298
28.36.1.3 hasIndistinctKnots(shared_ptr< ParamSurface > surf, double tol, std::vector< shared_ptr< ParamCurve > > &trim_cv_knots)	298
28.36.1.4 isSliverFace(shared_ptr< ParamSurface >, double thickness, double factor=2.0)	298
28.36.1.5 isSliverFace(const SplineSurface &sf, double thickness, double factor=2.0)	298
28.36.1.6 isSliverFace(const BoundedSurface &sf, double thickness, double factor=2.0)	298
28.36.1.7 isSliverFace2(const BoundedSurface &sf, double thickness, double factor=2.0)	298
28.37Go::RegularizeUtils Namespace Reference	298
28.37.1 Detailed Description	300
28.37.2 Function Documentation	300
28.37.2.1 adjustTrimSeg(std::vector< shared_ptr< CurveOnSurface > > &trim_segments, Point *parval1, Point *parval2, shared_ptr< ftSurface > face, shared_ptr< BoundedSurface > &bd_sf, std::vector< shared_ptr< Vertex > > &non_corner, double tol, double epsge)	300
28.37.2.2 angleInEndpoints(shared_ptr< CurveOnSurface > seg, shared_ptr< Vertex > vx1, shared_ptr< Vertex > vx2, shared_ptr< ftSurface > face, double &min_ang1, double &min_ang2)	300
28.37.2.3 checkPath(shared_ptr< Vertex > vx1, shared_ptr< Vertex > vx2, shared_ptr< Vertex > vx, shared_ptr< ftSurface > face, double angtol)	300
28.37.2.4 checkRegularity(std::vector< shared_ptr< Vertex > > &cand_vx, shared_ptr< ftSurface > face, bool checkConvex=true)	300
28.37.2.5 checkStrightParCv(shared_ptr< ftSurface > face, const Point &pos1, const Point &pos2, double epsge)	300

28.37.2.6	checkStrightParCv(shared_ptr< ftSurface > face, shared_ptr< Vertex > vx1, shared_ptr< Vertex > vx2, double epsge)	300
28.37.2.7	checkStrightParCv(shared_ptr< ftSurface > face, shared_ptr< Vertex > vx1, const Point &mid, double epsge)	300
28.37.2.8	checkTrimConfig(shared_ptr< ftSurface > face, std::vector< shared_ptr< CurveOnSurface > > &trim_segments, shared_ptr< Vertex > vx, std::vector< shared_ptr< Vertex > > &corners, double epsge)	300
28.37.2.9	checkTrimSeg(std::vector< shared_ptr< CurveOnSurface > > &trim_segments, std::vector< shared_ptr< Vertex > > &next_vxs, const Point &vx_point, const Point &other_pt, double epsge)	300
28.37.2.10	checkTrimSeg2(std::vector< shared_ptr< CurveOnSurface > > &trim_← segments, const Point &vx_par1, const Point &vx_par2, double epsge)	300
28.37.2.11	checkTrimSeg3(std::vector< shared_ptr< CurveOnSurface > > &trim_← segments, const Point &vx_par1, const Point &vx_par2, double epsge)	301
28.37.2.12	cornerInShortestPath(shared_ptr< Vertex > vx1, shared_ptr< Vertex > vx2, shared_ptr< ftSurface > face, double angtol)	301
28.37.2.13	createFaces(std::vector< shared_ptr< BoundedSurface > > &sub_sfs, shared_ptr< ftSurface > face, double epsge, double tol2, double angtol, std_← :vector< shared_ptr< Vertex > > non_corner)	301
28.37.2.14	divideVertex(shared_ptr< ftSurface > face, shared_ptr< Vertex > vx, std_← :vector< shared_ptr< Vertex > > &cand_vx, ftEdge *cand_edge, std::vector< shared_ptr< Vertex > > &prio_vx, double epsge, double tol2, double angtol, double bend, std::vector< shared_ptr< Vertex > > &non_corner, const Point ¢re, const Point &axis, bool strong=false)	301
28.37.2.15	endVxInChain(shared_ptr< ftSurface > face, ftSurface *face1, ftSurface *face2, shared_ptr< Vertex > vx, shared_ptr< Vertex > prev, shared_ptr< Vertex > vx0, std::vector< shared_ptr< Vertex > > &met_already)	301
28.37.2.16	findVertexSplit(shared_ptr< ftSurface > face, shared_ptr< Vertex > vx, std_← :vector< shared_ptr< Vertex > > &cand_vx, ftEdge *cand_edge, std::vector< shared_ptr< Vertex > > &prio_vx, double epsge, double tol2, double angtol, double bend, std::vector< shared_ptr< Vertex > > &non_corner, const Point ¢re, const Point &axis, shared_ptr< BoundedSurface > &bd_sf, bool strong=false) 301	301
28.37.2.17	getClosestBoundaryPar(shared_ptr< ftSurface > face, shared_ptr< Vertex > vx, std::vector< shared_ptr< ParamCurve > > &vx_cvs, const Point &pnt, double epsge, int &close_idx, double &close_dist, Point &close_par, int loop_idx=-1) . . .	301
28.37.2.18	getDivisionPlane(shared_ptr< ftSurface > face, shared_ptr< Vertex > vx, double epsge, Point &pnt, Point &normal)	301
28.37.2.19	getInVec(shared_ptr< Vertex > vx, shared_ptr< ftSurface > face)	301
28.37.2.20	getMaxParFrac(shared_ptr< ftSurface > face)	301
28.37.2.21	getOppositeBoundaryPar(shared_ptr< ftSurface > face, shared_ptr< Vertex > vx, std::vector< shared_ptr< Vertex > > &corners, double epsge, Point &point, double &par, double &dist)	301

28.37.2.22	getPath(ftEdge *edg, shared_ptr< Vertex > vx, shared_ptr< Vertex > last, shared_ptr< ftSurface > face, std::vector< ftEdge * > &path)	301
28.37.2.23	getSourceCvs(std::vector< shared_ptr< ftEdge > > &all_edg, std::vector< shared_ptr< ParamCurve > > &all_cvs)	301
28.37.2.24	mergeSituationContinuation(ftSurface *init_face, shared_ptr< Vertex > vx, ftEdge *edge, double angtol)	301
28.37.2.25	noExtension(shared_ptr< Vertex > vx, ftSurface *face, shared_ptr< Vertex > &vx2, std::pair< Point, Point > &co_par1, std::pair< Point, Point > &co_par2, int &dir1, int &dir2, double &val1, double &val2, double angtol, bool check_constant_curve)	302
28.37.2.26	selectCandVx(shared_ptr< ftSurface > face, shared_ptr< Vertex > vx, const Point &in_vec, vector< shared_ptr< Vertex > > cand_vx, RectDomain &dom, double epsge, double angtol, const Point ¢re, const Point &normal, std::vector< shared_ptr< ParamCurve > > &vx_cvs, double close_dist, const Point &close_pt, double &cyl_rad, bool strong=false)	302
28.37.2.27	traverseUntilTJoint(std::vector< ftSurface * > vx_faces, shared_ptr< Vertex > vx, shared_ptr< Vertex > &vx2, std::vector< ftSurface * > &vx_faces2)	302
28.38	Go::Singular Namespace Reference	302
28.38.1	Function Documentation	302
28.38.1.1	vanishingNormal(shared_ptr< ParamSurface > srf, double tol, std::vector< Point > &singular_pts, std::vector< std::vector< Point > > &singular_sequences)	302
28.38.1.2	vanishingTangent(shared_ptr< ParamCurve > crv, double start, double end, double tol, std::vector< double > &singular_pts, std::vector< std::vector< double > > &singular_sequences)	302
28.39	Go::SplineDebugUtils Namespace Reference	302
28.39.1	Detailed Description	302
28.39.2	Function Documentation	302
28.39.2.1	objsToFile(std::vector< shared_ptr< GeomObject > > &geom_objs, char *to_file)	302
28.39.2.2	objToFile(GeomObject *geom_obj, char *to_file)	303
28.39.2.3	writeCvsOnSf(const std::vector< shared_ptr< Go::ParamCurve > > &loop_cvs, double epsgeo, std::ofstream &fileout)	303
28.39.2.4	writeCvsOnSf(const std::vector< shared_ptr< Go::CurveOnSurface > > &loop_cvs, double epsgeo, std::ofstream &fileout)	303
28.39.2.5	writeOuterBoundaryLoop(ParamSurface &srf, std::ostream &os)	303
28.39.2.6	writeSISLFormat(const SplineCurve &spline_cv, std::ostream &os)	303
28.39.2.7	writeSpaceParamCurve(const SplineCurve &pcurve, std::ostream &os, double z=0.0)	303
28.39.2.8	writeSpaceParamCurve(const Line &pline, std::ostream &os, double z=0.0)	304

28.39.2.9 writeTrimmedInfo(BoundedSurface &bd_sf, std::ostream &os, double z=0.0)	304
28.40 Go::SplineUtils Namespace Reference	304
28.40.1 Detailed Description	304
28.40.2 Function Documentation	304
28.40.2.1 closest_in_array(const double *pt, const double *array, int n, int dim)	304
28.40.2.2 closest_on_line_segment(const Vector3D &pt, const Vector3D &beg, const Vector3D &end)	305
28.40.2.3 closest_on_rectgrid(const double *pt, const double *array, int m, int n, double &clo_u, double &clo_v)	305
28.40.2.4 closest_on_rectgrid(const double *pt, const double *array, int u_min, int u_max, int v_min, int v_max, int nmb_coefs_u, double &clo_u, double &clo_v)	305
28.40.2.5 closest_on_triangle(const Vector3D &pt, const Vector3D tri[3], double &clo_dist2)	306
28.40.2.6 curve_ratder(double const eder[], int idim, int ider, double gder[])	306
28.40.2.7 extractBezierCoefs(const double *coefs, const int num_coefs_u, const int num_coefs_v, const int ind_u_min, const int ind_v_min, const std::vector< double > &transf_mat_u, const std::vector< double > &transf_mat_v, std::vector< double > &bezier_coefs)	307
28.40.2.8 insertKnots(const Go::SplineSurface &spline_sf, const std::vector< double > new_knots_u, const std::vector< double > new_knots_v)	307
28.40.2.9 make_coef_array_from_rational_coefs(const double *rationals, double *coefs, int num_coefs, int dim)	307
28.40.2.10 sloadg(int ij, int imy, int ik, int in, int *jpl, int *jfi, int *jla, const double *et, const double *etau, double *galfa)	307
28.40.2.11 refinedBezierCoefsCubic(Go::SplineSurface &spline_sf, int ind_u_min, int ind_v_min, std::vector< double > &bez_coefs)	308
28.40.2.12 refineToBezier(const Go::SplineSurface &spline_sf)	308
28.40.2.13 refmatrix(const double *et, int im, int ik, const double *etau, int in, double *ea, int *nfirst, int *nlast)	308
28.40.2.14 splineToBezierTransfMat(const double *knots, std::vector< double > &transf_mat)	308
28.40.2.15 surface_ratder(double const eder[], int idim, int ider, double gder[])	308
28.40.2.16 transpose_array(int dim, int m, int n, double *array_start)	309
28.41 Go::SplitModelUtils Namespace Reference	309
28.41.1 Detailed Description	309
28.41.2 Function Documentation	309
28.41.2.1 splitInFreeCorners(shared_ptr< SurfaceModel > sfmodel, const Point &pnt, const Point &axis)	309

28.41.2.2	<code>splitInNonCorners(shared_ptr< SurfaceModel > sfmodel, const Point &pnt, const Point &axis)</code>	310
28.41.2.3	<code>splitInOuterVertices(shared_ptr< SurfaceModel > sfmodel, shared_ptr< face Surface > face, const Point &pnt, const Point &axis)</code>	310
28.42	Go::SurfaceCreators Namespace Reference	310
28.42.1	Detailed Description	310
28.42.2	Function Documentation	310
28.42.2.1	<code>createSmoothTransition(const std::vector< shared_ptr< const ParamSurface > > &surfs, const std::vector< shared_ptr< const CurveOnSurface > > &int_cvs, double dist_0, double dist_1, double epsge, std::vector< shared_ptr< SplineCurve > > &trim_crvs)</code>	310
28.42.2.2	<code>insertParamDomain(const Go::SplineSurface &sf_1d)</code>	311
28.42.2.3	<code>mergeRationalParts(const Go::SplineSurface &nom_sf, const Go::SplineSurface &den_sf, bool weights_in_first=false)</code>	311
28.42.2.4	<code>mult1DBezierPatches(const SplineSurface &patch1, const SplineSurface &patch2)</code>	311
28.42.2.5	<code>mult1DSurfaces(const SplineSurface &sf1, const SplineSurface &sf2)</code>	311
28.42.2.6	<code>separateRationalParts(const Go::SplineSurface &sf)</code>	312
28.43	Go::SurfaceInterpolator Namespace Reference	312
28.43.1	Detailed Description	312
28.43.2	Function Documentation	312
28.43.2.1	<code>regularInterpolation(const BsplineBasis &basis_u, const BsplineBasis &basis_v, std::vector< double > &par_u, std::vector< double > &par_v, std::vector< double > &points, int dimension, bool rational, std::vector< double > &weights)</code>	312
28.44	Go::SurfaceModelUtils Namespace Reference	313
28.44.1	Detailed Description	313
28.44.2	Function Documentation	313
28.44.2.1	<code>checkClosedFaces(shared_ptr< ParamSurface > surface, double tol)</code>	313
28.45	Go::SurfaceOnVolumeTools Namespace Reference	313
28.45.1	Detailed Description	313
28.45.2	Function Documentation	313
28.45.2.1	<code>getOuterBoundaryLoop(shared_ptr< SurfaceOnVolume > sf, double eps)</code>	313
28.46	Go::SurfaceTools Namespace Reference	313
28.46.1	Detailed Description	314
28.46.2	Function Documentation	314

28.46.2.1	<code>absolutelyAllBoundarySfLoops(shared_ptr< ParamSurface > surf, double degenerate_epsilon)</code>	314
28.46.2.2	<code>allBoundarySfLoops(shared_ptr< ParamSurface > surf, double degenerate_epsilon)</code>	314
28.46.2.3	<code>checkCoefCoLinearity(shared_ptr< SplineSurface > sf1, shared_ptr< SplineSurface > sf2, int bd1, int bd2, bool same_orient, double tol, double ang_tol, std::vector< std::vector< int > > &enumeration)</code>	315
28.46.2.4	<code>checkSurfaceClosed(const Go::ParamSurface &sf, bool &closed_dir_u, bool &closed_dir_v, double closed_tol=1e-06)</code>	315
28.46.2.5	<code>cornerToCornerSfs(shared_ptr< ParamSurface > sf1, shared_ptr< CurveOnSurface > sf_cv1, shared_ptr< ParamSurface > sf2, shared_ptr< CurveOnSurface > sf_cv2, double tol)</code>	315
28.46.2.6	<code>estimateTangentLength(SplineSurface *surf, int pardir, bool at_start)</code>	315
28.46.2.7	<code>getCoefEnumeration(shared_ptr< SplineSurface > sf, int bd, std::vector< int > &enumeration)</code>	315
28.46.2.8	<code>getCoefEnumeration(shared_ptr< SplineSurface > sf, int bd, std::vector< int > &enumeration_bd, std::vector< int > &enumeration_bd2)</code>	316
28.46.2.9	<code>getCornerCoefEnum(shared_ptr< SplineSurface > sf, int bd1, int bd2, int &enumeration)</code>	316
28.46.2.10	<code>getCorrCoefEnum(shared_ptr< SplineSurface > sf1, shared_ptr< SplineSurface > sf2, int bd1, int bd2, bool same_orient, std::vector< std::pair< int, int > > &enumeration)</code>	316
28.46.2.11	<code>getParEpsilon(const ParamSurface &sf, double epsgeo)</code>	316
28.46.2.12	<code>getSfAdjacencyInfo(shared_ptr< ParamSurface > sf1, shared_ptr< CurveOnSurface > sf_cv1, shared_ptr< ParamSurface > sf2, shared_ptr< CurveOnSurface > sf_cv2, double tol, int &bd1, int &bd2, bool &same_orient)</code>	316
28.46.2.13	<code>iterateCornerPos(Point &vertex, std::vector< std::pair< shared_ptr< ParamSurface >, Point > > sfs, double tol)</code>	317
28.46.2.14	<code>outerBoundarySfLoop(shared_ptr< ParamSurface > surf, double degenerate_epsilon)</code>	317
28.46.2.15	<code>parameterizeByBaseSurf(const ParamSurface &sf, const std::vector< double > &points, std::vector< double > &parvals)</code>	317
28.46.2.16	<code>surface_seedfind(const Point &pt, const ParamSurface &sf, const RectDomain *rd, double &u, double &v)</code>	317
28.46.2.17	<code>surfaceClosed(const Go::SplineSurface &sf, bool &closed_dir_u, bool &closed_dir_v, double closed_tol=1e-06)</code>	317
28.47	<code>Go::TesselatorUtils Namespace Reference</code>	318
28.47.1	Detailed Description	318
28.47.2	Function Documentation	318

28.47.2.1	<code>getCtrPol(GeomObject *obj)</code>	318
28.47.2.2	<code>getResolution(const ParamSurface *surf, int &u_nmb, int &v_nmb, int uv_nmb=400)</code>	318
28.48	Go::tpUtils Namespace Reference	318
28.48.1	Function Documentation	318
28.48.1.1	<code>adjacentEdges(edgeType *edge, bool at_start_of_edge, std::vector< edgeType * > &adjacent, std::vector< bool > &at_start)</code>	318
28.48.1.2	<code>checkContinuity(const edgeType *edge1, const edgeType *edge2, double neighbour, double gap, double bend, double kink)</code>	319
28.49	Go::Utils Namespace Reference	319
28.49.1	Detailed Description	319
28.49.2	Function Documentation	319
28.49.2.1	<code>distance_squared(ForwardIterator first1, ForwardIterator last1, ForwardIterator first2)</code>	319
28.49.2.2	<code>eatwhite(InputStream &is)</code>	320
28.49.2.3	<code>inner(ForwardIterator first, ForwardIterator last, ForwardIterator second)</code>	320
28.49.2.4	<code>normalize(ForwardIterator first, ForwardIterator last)</code>	320
28.49.2.5	<code>sum(ForwardIterator first, ForwardIterator last)</code>	320
28.49.2.6	<code>sum_squared(ForwardIterator first, ForwardIterator last)</code>	320
28.50	Go::VolumeInterpolator Namespace Reference	320
28.50.1	Detailed Description	321
28.50.2	Function Documentation	321
28.50.2.1	<code>regularInterpolation(const BsplineBasis &basis_u, const BsplineBasis &basis_v, const BsplineBasis &basis_w, std::vector< double > &par_u, std::vector< double > &par_v, std::vector< double > &par_w, std::vector< double > &points, int dimension, bool rational, std::vector< double > &weights)</code>	321
28.51	Go::VolumeModelCreator Namespace Reference	321
28.51.1	Function Documentation	321
28.51.1.1	<code>createRotationalModel(shared_ptr< SurfaceModel > &sfmodel, shared_ptr< VolumeModel > &volmodel)</code>	321
28.51.1.2	<code>linearSweptModel(shared_ptr< SurfaceModel > &sfmodel, shared_ptr< VolumeModel > &volmodel)</code>	321
28.52	Go::VolumeTools Namespace Reference	321
28.52.1	Detailed Description	322
28.52.2	Function Documentation	322

28.52.2.1	<code>analyzePeriodicity(const SplineVolume &sf, int direction, double knot_tol=1e-12)</code>	322
28.52.2.2	<code>approxVolParamCurve(shared_ptr< ParamCurve > spacecurve, shared_ptr< ParamVolume > vol, double tol, int max_iter, double &maxdist)</code>	323
28.52.2.3	<code>cornerToCornerVols(shared_ptr< ParamVolume > vol1, shared_ptr< SurfaceOnVolume > vol_sf1, shared_ptr< ParamVolume > vol2, shared_ptr< SurfaceOnVolume > vol_sf2, double tol)</code>	323
28.52.2.4	<code>getBoundarySurface(shared_ptr< SplineVolume > vol, int idx)</code>	323
28.52.2.5	<code>getBoundarySurfaces(shared_ptr< ParamVolume > vol)</code>	323
28.52.2.6	<code>getCorrCoefVolEnum(shared_ptr< SplineVolume > vol1, shared_ptr< SplineVolume > vol2, int bd1, int bd2, int orientation, bool same_seq, std::vector< std::pair< int, int > > &enumeration)</code>	323
28.52.2.7	<code>getOrientedBoundarySurface(shared_ptr< SplineVolume > vol, int idx)</code>	324
28.52.2.8	<code>getOrientedBoundarySurfaces(shared_ptr< ParamVolume > vol)</code>	324
28.52.2.9	<code>getVolAdjacencyInfo(shared_ptr< ParamVolume > vol1, shared_ptr< SurfaceOnVolume > vol_sf1, shared_ptr< ParamVolume > vol2, shared_ptr< SurfaceOnVolume > vol_sf2, double tol, int &bd1, int &bd2, int &orientation, bool &same_seq)</code>	324
28.52.2.10	<code>getVolBdCoefEnumeration(shared_ptr< SplineVolume > vol, int bd, int bd_cv, std::vector< int > &enumeration)</code>	324
28.52.2.11	<code>getVolCoefEnumeration(shared_ptr< SplineVolume > vol, int bd, std::vector< int > &enumeration)</code>	325
28.52.2.12	<code>getVolCoefEnumeration(shared_ptr< SplineVolume > vol, int bd, std::vector< int > &enumeration_bd, std::vector< int > &enumeration_bd2)</code>	325
28.52.2.13	<code>liftVolParamCurve(shared_ptr< ParamCurve > pcurve, shared_ptr< ParamVolume > vol, double tol)</code>	325
28.52.2.14	<code>projectVolParamCurve(shared_ptr< ParamCurve > spacecurve, shared_ptr< ParamVolume > vol, double tol)</code>	325
28.52.2.15	<code>representCurveAsVolume(const SplineCurve &curve, int cv_dir, const BsplineBasis &other_bas1, const BsplineBasis &other_bas2, bool rational)</code>	325
28.52.2.16	<code>representSurfaceAsVolume(const SplineSurface &surface, int sf_dir1, int sf_dir2, const BsplineBasis &other_bas, bool rational)</code>	325
28.52.2.17	<code>representVolumeAsCurve(const SplineVolume &volume, int cv_dir)</code>	326
28.52.2.18	<code>representVolumeAsSurface(const SplineVolume &volume, int sf_dir1, int sf_dir2)</code>	326
28.52.2.19	<code>volCommonSplineSpace(shared_ptr< SplineVolume > vol1, int bd1, shared_ptr< SplineVolume > vol2, int bd2, int orientation, bool same_seq)</code>	327
28.53	hed Namespace Reference	327
28.54	hetriang Namespace Reference	327
28.55	NEWMAT Namespace Reference	328

28.55.1 Function Documentation	330
28.55.1.1 Cholesky(const SymmetricMatrix &)	330
28.55.1.2 Cholesky(const SymmetricBandMatrix &)	330
28.55.1.3 Compare(const MatrixType &, MatrixType &)	330
28.55.1.4 DCT(const ColumnVector &, ColumnVector &)	330
28.55.1.5 DCT_II(const ColumnVector &, ColumnVector &)	330
28.55.1.6 DCT_II_inverse(const ColumnVector &, ColumnVector &)	330
28.55.1.7 DCT_inverse(const ColumnVector &, ColumnVector &)	330
28.55.1.8 Determinant(const BaseMatrix &B)	330
28.55.1.9 DotProduct(const Matrix &A, const Matrix &B)	331
28.55.1.10DST(const ColumnVector &, ColumnVector &)	331
28.55.1.11DST_II(const ColumnVector &, ColumnVector &)	331
28.55.1.12DST_II_inverse(const ColumnVector &, ColumnVector &)	331
28.55.1.13DST_inverse(const ColumnVector &, ColumnVector &)	331
28.55.1.14EigenValues(const SymmetricMatrix &, DiagonalMatrix &)	331
28.55.1.15EigenValues(const SymmetricMatrix &, DiagonalMatrix &, SymmetricMatrix &)	331
28.55.1.16EigenValues(const SymmetricMatrix &, DiagonalMatrix &, Matrix &)	331
28.55.1.17FFT(const ColumnVector &, const ColumnVector &, ColumnVector &, ColumnVector &)	331
28.55.1.18FFTI(const ColumnVector &, const ColumnVector &, ColumnVector &, ColumnVector &)	331
28.55.1.19HHDecompose(Matrix &X, LowerTriangularMatrix &L)	331
28.55.1.20HHDecompose(const Matrix &X, Matrix &Y, Matrix &M)	331
28.55.1.21IsZero(const BaseMatrix &A)	331
28.55.1.22IsZero(const GeneralMatrix &A)	331
28.55.1.23Jacobi(const SymmetricMatrix &, DiagonalMatrix &)	331
28.55.1.24Jacobi(const SymmetricMatrix &, DiagonalMatrix &, SymmetricMatrix &)	331
28.55.1.25Jacobi(const SymmetricMatrix &, DiagonalMatrix &, Matrix &)	331
28.55.1.26Jacobi(const SymmetricMatrix &, DiagonalMatrix &, SymmetricMatrix &, Matrix &, bool=true)	331
28.55.1.27KP(const BaseMatrix &, const BaseMatrix &)	331
28.55.1.28LogDeterminant(const BaseMatrix &B)	331

28.55.1.29	MatrixErrorNoSpace(void *)	332
28.55.1.30	Maximum(const BaseMatrix &B)	332
28.55.1.31	MaximumAbsoluteValue(const BaseMatrix &B)	332
28.55.1.32	Minimum(const BaseMatrix &B)	332
28.55.1.33	MinimumAbsoluteValue(const BaseMatrix &B)	332
28.55.1.34	Norm1(const BaseMatrix &B)	332
28.55.1.35	Norm1(RowVector &RV)	332
28.55.1.36	NormFrobenius(const BaseMatrix &B)	332
28.55.1.37	NormInfinity(const BaseMatrix &B)	332
28.55.1.38	NormInfinity(ColumnVector &CV)	332
28.55.1.39	operator!=(const GeneralMatrix &A, const GeneralMatrix &B)	333
28.55.1.40	operator!=(const BaseMatrix &A, const BaseMatrix &B)	333
28.55.1.41	operator*(Real f, const BaseMatrix &BM)	333
28.55.1.42	operator+(Real f, const BaseMatrix &BM)	333
28.55.1.43	operator-(Real, const BaseMatrix &)	333
28.55.1.44	operator<(const BaseMatrix &A, const BaseMatrix &)	333
28.55.1.45	operator<<(std::ostream &, const BaseMatrix &)	333
28.55.1.46	operator<<(std::ostream &, const GeneralMatrix &)	333
28.55.1.47	operator<=(const BaseMatrix &A, const BaseMatrix &)	333
28.55.1.48	operator==(const GeneralMatrix &A, const GeneralMatrix &B)	333
28.55.1.49	operator==(const BaseMatrix &A, const BaseMatrix &B)	333
28.55.1.50	operator>(const BaseMatrix &A, const BaseMatrix &)	333
28.55.1.51	operator>=(const BaseMatrix &A, const BaseMatrix &)	333
28.55.1.52	QRZ(Matrix &, UpperTriangularMatrix &)	334
28.55.1.53	QRZ(const Matrix &, Matrix &, Matrix &)	334
28.55.1.54	QRZT(Matrix &, LowerTriangularMatrix &)	334
28.55.1.55	QRZT(const Matrix &, Matrix &, Matrix &)	334
28.55.1.56	RealFFT(const ColumnVector &, ColumnVector &, ColumnVector &)	334
28.55.1.57	RealFFTI(const ColumnVector &, const ColumnVector &, ColumnVector &)	334
28.55.1.58	Rectangular(MatrixType a, MatrixType b, MatrixType c)	334

28.55.1.59	SortAscending(GeneralMatrix &)	334
28.55.1.60	SortDescending(GeneralMatrix &)	334
28.55.1.61	SortSV(DiagonalMatrix &D, Matrix &U, bool ascending=false)	334
28.55.1.62	SortSV(DiagonalMatrix &D, Matrix &U, Matrix &V, bool ascending=false)	334
28.55.1.63	SP(const BaseMatrix &, const BaseMatrix &)	334
28.55.1.64	Sum(const BaseMatrix &B)	334
28.55.1.65	SumAbsoluteValue(const BaseMatrix &B)	334
28.55.1.66	SumSquare(const BaseMatrix &B)	334
28.55.1.67	SVD(const Matrix &, DiagonalMatrix &, Matrix &, Matrix &, bool=true, bool=true)	334
28.55.1.68	SVD(const Matrix &, DiagonalMatrix &)	334
28.55.1.69	SVD(const Matrix &A, DiagonalMatrix &D, Matrix &U, bool withU=true)	334
28.55.1.70	Trace(const BaseMatrix &B)	335
28.56	RBD_COMMON Namespace Reference	335
28.56.1	Typedef Documentation	335
28.56.1.1	Exception	335
28.56.1.2	long_Real	335
28.56.1.3	Real	336
28.56.2	Function Documentation	336
28.56.2.1	Terminate()	336
28.57	RBD_LIBRARIES Namespace Reference	336
28.58	std Namespace Reference	336
28.58.1	Detailed Description	337
28.58.2	Function Documentation	337
28.58.2.1	__power(_Tp __x, _Integer __n, _MonoidOperation __oper)	337
28.58.2.2	__power(_Tp __x, _Integer __n)	337
28.58.2.3	identity_element(plus<_Tp >)	337
28.58.2.4	identity_element(multiplies<_Tp >)	337
28.58.2.5	operator<<(std::ostream &os, const Go::BaryCoordSystem< Dim > &bc)	337
28.58.2.6	operator<<(std::ostream &os, const Go::BoundingBox &bbox)	337
28.58.2.7	operator<<(std::ostream &os, const Go::BernsteinPoly &p)	337

28.58.2.8	operator<<(std::ostream &os, const Go::BernsteinMulti &m)	338
28.58.2.9	operator>>(std::istream &is, Go::BaryCoordSystem< Dim > &bc)	338
28.58.2.10	operator>>(std::istream &is, Go::BoundingBox &bbox)	338
28.58.2.11	operator>>(std::istream &is, Go::BernsteinPoly &p)	338
28.58.2.12	operator>>(std::istream &is, Go::BernsteinMulti &m)	338
28.58.2.13	power(_Tp __x, _Integer __n, _MonoidOperation __oper)	338
28.58.2.14	power(_Tp __x, _Integer __n)	338
28.59	tl Namespace Reference	338
28.59.1	Detailed Description	340
28.59.2	Function Documentation	341
28.59.2.1	convexBoundary(const DartType &dart)	341
28.59.2.2	degenerateTriangle(const DartType &dart)	341
28.59.2.3	get_0_orbit_boundary(const DartType &dart, DartListType &orbit)	341
28.59.2.4	get_0_orbit_interior(const DartType &dart, DartListType &orbit)	342
28.59.2.5	getAdjacentTriangles(const DartType &dart, DartType &t1, DartType &t2, DartType &t3)	342
28.59.2.6	getBoundary(const DartType &dart, DartListType &boundary)	343
28.59.2.7	getDegreeOfNode(const DartType &dart)	343
28.59.2.8	getNeighborNodes(const DartType &dart, std::list< DartType > &node_list, bool &boundary)	343
28.59.2.9	insertConstraint(DartType &dstart, DartType &dend, bool optimize_delaunay)	343
28.59.2.10	insertNode(DartType &dart, PointType &point)	344
28.59.2.11	insertNodes(ForwardIterator first, ForwardIterator last, DartType &dart)	345
28.59.2.12	nTriangle(const PointType &point, const DartType &dart)	345
28.59.2.13	nTriangleSimplest(const PointType &point, const DartType &dart)	345
28.59.2.14	sBoundaryEdge(const DartType &dart)	346
28.59.2.15	sBoundaryFace(const DartType &dart)	346
28.59.2.16	sBoundaryNode(const DartType &dart)	346
28.59.2.17	sMemberOfFace(const TopologyElementType &topologyElement, const DartType &dart)	346
28.59.2.18	ocateFaceSimplest(const PointType &point, DartType &dart)	346
28.59.2.19	ocateFaceWithNode(const NodeType &node, DartType &dart_iter)	347

28.59.2.20	locateTriangle(const PointType &point, DartType &dart)	347
28.59.2.21	optimizeDelaunay(DartListType &elist)	347
28.59.2.22	optimizeDelaunay(DartListType &elist, const typename DartListType::iterator end)	348
28.59.2.23	positionAtNextBoundaryEdge(DartType &dart)	348
28.59.2.24	recSwapDelaunay(DartType &diagonal)	348
28.59.2.25	removeBoundaryNode(DartType &dart)	349
28.59.2.26	removeInteriorNode(DartType &dart)	349
28.59.2.27	removeNode(DartType &dart)	349
28.59.2.28	removeRectangularBoundary(DartType &dart)	349
28.59.2.29	same_0_orbit(const DartType &d1, const DartType &d2)	350
28.59.2.30	same_1_orbit(const DartType &d1, const DartType &d2)	350
28.59.2.31	same_2_orbit(const DartType &d1, const DartType &d2)	350
28.59.2.32	swapEdgeInList(const typename DartListType::iterator &it, DartListType &elist)	350
28.59.2.33	swapEdgesAwayFromBoundaryNode(DartType &dart, ListType &swapped_edges)	350
28.59.2.34	swapEdgesAwayFromInteriorNode(DartType &dart, ListType &swapped_edges)	351
28.59.2.35	swappableEdge(const DartType &dart, bool allowDegeneracy=false)	351
28.59.2.36	swapTestDelaunay(const DartType &dart, bool cycling_check=false)	352
28.60	ttl_constr Namespace Reference	352
28.60.1	Detailed Description	353
28.60.2	Function Documentation	353
28.60.2.1	crossesConstraint(DartType &dstart, DartType &dend, DartType &d1, DartType &d2)	353
28.60.2.2	findCrossingEdges(const DartType &dstart, const DartType &dend, ListType &elist)	353
28.60.2.3	getAtSmallestAngle(const DartType &dstart, const DartType &dend)	353
28.60.2.4	isTheConstraint(const DartType &dart, const DartType &dstart, const DartType &dend)	353
28.60.2.5	transformToConstraint(DartType &dstart, DartType &dend, std::list< DartType > &elist)	353
28.61	ttl_util Namespace Reference	353
28.61.1	Detailed Description	354
28.61.2	Function Documentation	354
28.61.2.1	createRandomData(int noPoints, int seed=1)	354
28.61.2.2	crossProduct2d(real_type dx1, real_type dy1, real_type dx2, real_type dy2)	355
28.61.2.3	orient2dfast(real_type pa[2], real_type pb[2], real_type pc[2])	355
28.61.2.4	scalarProduct2d(real_type dx1, real_type dy1, real_type dx2, real_type dy2)	355

29 Class Documentation	357
29.1 Go::AdaptCurve Class Reference	357
29.1.1 Detailed Description	357
29.1.2 Constructor & Destructor Documentation	357
29.1.2.1 AdaptCurve(const EvalCurve *evalcrv, double aepsge)	357
29.1.2.2 AdaptCurve(const EvalCurve *evalcrv, double aepsge, int in, int ik)	358
29.1.2.3 AdaptCurve(const EvalCurve *evalcrv, double aepsge, int in, int ik, std::vector< double > &knots)	358
29.1.2.4 AdaptCurve(const EvalCurve *evalcrv, double aepsge, shared_ptr< SplineCurve > curve)	358
29.1.2.5 ~AdaptCurve()	359
29.1.2.6 AdaptCurve()	359
29.1.3 Member Function Documentation	359
29.1.3.1 approximate(int max_iter=5)	359
29.1.3.2 getAdaptCurve(double &maxdist, double &avdist, int max_iter=5)	359
29.1.3.3 setEndPoints(const std::vector< Point > &start_point, const std::vector< Point > &end_point)	359
29.1.3.4 setSmooth(double w)	360
29.1.3.5 unsetSmooth()	360
29.2 NEWMAT::AddedMatrix Class Reference	360
29.2.1 Detailed Description	361
29.2.2 Constructor & Destructor Documentation	362
29.2.2.1 AddedMatrix(const BaseMatrix *bm1x, const BaseMatrix *bm2x)	362
29.2.2.2 ~AddedMatrix()	362
29.2.3 Member Function Documentation	362
29.2.3.1 BandWidth() const	362
29.2.3.2 Evaluate(MatrixType mt=MatrixTypeUnSp)	362
29.2.4 Friends And Related Function Documentation	362
29.2.4.1 BaseMatrix	362
29.2.4.2 GeneralMatrix	362
29.2.4.3 GenericMatrix	362
29.3 Go::AdjacencyInfo Struct Reference	363

29.3.1	Detailed Description	363
29.3.2	Constructor & Destructor Documentation	364
29.3.2.1	AdjacencyInfo()	364
29.3.3	Member Data Documentation	364
29.3.3.1	adjacency_found_	364
29.3.3.2	bd_idx_1_	364
29.3.3.3	bd_idx_2_	364
29.3.3.4	corner_failed_	364
29.3.3.5	same_orient_	364
29.4	Go::Alg2DElem Struct Reference	365
29.4.1	Detailed Description	365
29.4.2	Constructor & Destructor Documentation	365
29.4.2.1	Alg2DElem(double factor, int degree_x, int degree_y)	365
29.4.3	Member Data Documentation	366
29.4.3.1	degrees_	366
29.4.3.2	factor_	366
29.5	Go::Alg3DElem Struct Reference	366
29.5.1	Detailed Description	367
29.5.2	Constructor & Destructor Documentation	367
29.5.2.1	Alg3DElem(double factor, int degree_x, int degree_y, int degree_z)	367
29.5.3	Member Data Documentation	367
29.5.3.1	degrees_	367
29.5.3.2	factor_	367
29.6	Go::AlgObj2DInt Class Reference	368
29.6.1	Detailed Description	369
29.6.2	Constructor & Destructor Documentation	369
29.6.2.1	AlgObj2DInt(int degree)	369
29.6.2.2	AlgObj2DInt(const std::vector< Alg2DElem > &terms)	369
29.6.2.3	~AlgObj2DInt()	369
29.6.3	Member Function Documentation	369

29.6.3.1	numTerms()	369
29.6.3.2	term(int index)	370
29.6.4	Member Data Documentation	370
29.6.4.1	degree_	370
29.6.4.2	power_basis_	370
29.6.4.3	terms_	370
29.7	Go::AlgObj3DInt Class Reference	370
29.7.1	Detailed Description	371
29.7.2	Constructor & Destructor Documentation	371
29.7.2.1	AlgObj3DInt(int degree)	371
29.7.2.2	AlgObj3DInt(const std::vector< Alg3DElem > &terms)	372
29.7.2.3	AlgObj3DInt(const BernsteinTetrahedralPoly &implicit, const BaryCoord↔ System3D &bc)	372
29.7.2.4	~AlgObj3DInt()	372
29.7.3	Member Function Documentation	372
29.7.3.1	degree()	372
29.7.3.2	getImplicit(BernsteinTetrahedralPoly &impl, BaryCoordSystem3D &bc)	372
29.7.3.3	numTerms()	373
29.7.3.4	term(int index)	373
29.7.3.5	usingPowerBasis()	373
29.7.4	Member Data Documentation	373
29.7.4.1	bc_	373
29.7.4.2	degree_	373
29.7.4.3	implicit_	373
29.7.4.4	power_basis_	374
29.7.4.5	terms_	374
29.8	Go::AlgObjectInt Class Reference	374
29.8.1	Detailed Description	374
29.8.2	Constructor & Destructor Documentation	375
29.8.2.1	AlgObjectInt()	375
29.8.2.2	~AlgObjectInt()	375

29.9	Go::ApproxCrvToSeqs Class Reference	375
29.9.1	Detailed Description	376
29.9.2	Constructor & Destructor Documentation	376
29.9.2.1	ApproxCrvToSeqs(const std::vector< std::vector< double > > &points, const std::vector< std::vector< double > > &parvals, int dim, double aepsge)	376
29.9.2.2	ApproxCrvToSeqs(const std::vector< std::vector< double > > &points, const std::vector< std::vector< double > > &parvals, int dim, double aepsge, int in, int ik)	376
29.9.2.3	ApproxCrvToSeqs(const std::vector< std::vector< double > > &points, const std::vector< std::vector< double > > &parvals, int dim, double aepsge, int in, int ik, std::vector< double > &knots)	377
29.9.2.4	~ApproxCrvToSeqs()	377
29.9.2.5	ApproxCrvToSeqs()	377
29.9.3	Member Function Documentation	377
29.9.3.1	getApproxCurves(double &maxdist, double &avdist, int max_iter=5)	377
29.9.3.2	setC1Approx(double fac)	378
29.9.3.3	setEndPoints(const std::vector< std::vector< Point > > &start_point, const std::vector< std::vector< Point > > &end_point)	378
29.9.3.4	setSmooth(double w)	378
29.9.3.5	unsetSmooth()	378
29.10	Go::ApproxCurve Class Reference	378
29.10.1	Detailed Description	379
29.10.2	Constructor & Destructor Documentation	379
29.10.2.1	ApproxCurve(const std::vector< double > &points, const std::vector< double > &parvals, int dim, double aepsge)	379
29.10.2.2	ApproxCurve(const std::vector< double > &points, const std::vector< double > &parvals, int dim, double aepsge, int in, int ik)	380
29.10.2.3	ApproxCurve(const std::vector< double > &points, const std::vector< double > &parvals, int dim, double aepsge, int in, int ik, const std::vector< double > &knots)	380
29.10.2.4	~ApproxCurve()	380
29.10.2.5	ApproxCurve()	380
29.10.3	Member Function Documentation	380
29.10.3.1	getApproxCurve(double &maxdist, double &avdist, int max_iter=5)	380
29.10.3.2	setC1Approx(double fac)	381

29.10.3.3	setEndPoints(const std::vector< Point > &start_point, const std::vector< Point > &end_point)	381
29.10.3.4	setSmooth(double w)	381
29.10.3.5	unsetSmooth()	381
29.11	Go::ApproxSurf Class Reference	382
29.11.1	Detailed Description	382
29.11.2	Constructor & Destructor Documentation	382
29.11.2.1	ApproxSurf(std::vector< shared_ptr< SplineCurve > > &crvs, const std::vector< double > &points, const std::vector< double > &parvals, double domain[], int dim, double aepsge, int constdir=0, bool repar=true)	382
29.11.2.2	ApproxSurf(shared_ptr< SplineSurface > &srf, const std::vector< double > &points, const std::vector< double > &parvals, int dim, double aepsge, int constdir=0, bool approx_orig=false, bool close_belt=false, int nmb_stabil=0, bool repar=true)	383
29.11.2.3	~ApproxSurf()	383
29.11.2.4	ApproxSurf()	383
29.11.3	Member Function Documentation	384
29.11.3.1	edgeFix(int edge_fix[])	384
29.11.3.2	getApproxSurf(double &maxdist, double &avdist, int &nmb_out_eps, int max_iter=4, int keep_init=0)	384
29.11.3.3	getDoRefine()	384
29.11.3.4	reParam()	384
29.11.3.5	setC1Approx(double fac1, double fac2)	385
29.11.3.6	setDoRefine(bool refine)	385
29.11.3.7	setFixBoundary(bool fix_boundary)	385
29.11.3.8	setNormalConditions(const std::vector< double > &points, const std::vector< double > &parvals, int nmb_stabil=0)	385
29.11.3.9	setSmoothingWeight(double smooth)	385
29.12	Go::Array< T, Dim > Class Template Reference	386
29.12.1	Detailed Description	387
29.12.2	Constructor & Destructor Documentation	387
29.12.2.1	Array()	387
29.12.2.2	Array(T x)	388
29.12.2.3	Array(T x, T y)	388

29.12.2.4	Array(T x, T y, T z)	388
29.12.2.5	Array(T x, T y, T z, T w)	388
29.12.2.6	Array(const Array &v)	388
29.12.2.7	Array(RandomAccessIterator start)	388
29.12.2.8	Array(const Array< U, Dim > &v)	388
29.12.3	Member Function Documentation	389
29.12.3.1	angle(const Array &v) const	389
29.12.3.2	begin() const	389
29.12.3.3	begin()	389
29.12.3.4	cosAngle(const Array &v) const	389
29.12.3.5	cross(const Array &v) const	389
29.12.3.6	dist(const Array &v) const	389
29.12.3.7	dist2(const Array &v) const	389
29.12.3.8	distInf(const Array &v) const	390
29.12.3.9	end() const	390
29.12.3.10	end()	390
29.12.3.11	length() const	390
29.12.3.12	length2() const	390
29.12.3.13	lengthInf() const	390
29.12.3.14	normalize()	390
29.12.3.15	operator%(const Array &v) const	391
29.12.3.16	operator*(T d) const	391
29.12.3.17	operator*(const Array &v) const	391
29.12.3.18	operator*=(T d)	391
29.12.3.19	operator+(const Array &v) const	391
29.12.3.20	operator+=(const Array &v)	391
29.12.3.21	operator-(const Array &v) const	391
29.12.3.22	operator-() const	392
29.12.3.23	operator-=(const Array &v)	392
29.12.3.24	operator/(double d) const	392

29.12.3.25	operator/=(double d)	392
29.12.3.26	operator=(const Array &v)	392
29.12.3.27	operator==(const Array &v) const	392
29.12.3.28	operator[](int i) const	392
29.12.3.29	operator[](int i)	393
29.12.3.30	read(std::istream &is)	393
29.12.3.31	setValue(RandomAccessIterator from)	393
29.12.3.32	setValueConvert(RandomAccessIterator from)	393
29.12.3.33	size() const	393
29.12.3.34	write(std::ostream &os) const	393
29.12.3.35	x() const	393
29.12.3.36	x()	394
29.12.3.37	y() const	394
29.12.3.38	y()	394
29.12.3.39	z() const	394
29.12.3.40	z()	394
29.12.3.41	zero()	394
29.13	NEWMAT::ArrayLengthSpecifier Class Reference	394
29.13.1	Detailed Description	395
29.13.2	Constructor & Destructor Documentation	395
29.13.2.1	ArrayLengthSpecifier(int l)	395
29.13.3	Member Function Documentation	395
29.13.3.1	Value() const	395
29.14	RBD_COMMON::Bad_alloc Class Reference	395
29.14.1	Detailed Description	396
29.14.2	Constructor & Destructor Documentation	396
29.14.2.1	Bad_alloc(const char *a_what=0)	396
29.14.3	Member Data Documentation	396
29.14.3.1	Select	396
29.15	NEWMAT::BandLUMatrix Class Reference	397

29.15.1 Detailed Description	398
29.15.2 Constructor & Destructor Documentation	398
29.15.2.1 BandLUMatrix(const BaseMatrix &)	398
29.15.2.2 ~BandLUMatrix()	398
29.15.3 Member Function Documentation	398
29.15.3.1 CleanUp()	398
29.15.3.2 GetCol(MatrixRowCol &)	398
29.15.3.3 GetCol(MatrixColX &c)	399
29.15.3.4 GetRow(MatrixRowCol &)	399
29.15.3.5 IsEqual(const GeneralMatrix &) const	399
29.15.3.6 IsSingular() const	399
29.15.3.7 LogDeterminant() const	399
29.15.3.8 lubksb(Real *, int=0)	399
29.15.3.9 MakeSolver()	399
29.15.3.10 operator=(const BaseMatrix &)	399
29.15.3.11 operator=(const BandLUMatrix &m)	400
29.15.3.12 Solver(MatrixColX &, const MatrixColX &)	400
29.15.3.13 Type() const	400
29.16 NEWMAT::BandMatrix Class Reference	400
29.16.1 Detailed Description	402
29.16.2 Constructor & Destructor Documentation	402
29.16.2.1 BandMatrix()	402
29.16.2.2 ~BandMatrix()	402
29.16.2.3 BandMatrix(int n, int lb, int ub)	402
29.16.2.4 BandMatrix(const BaseMatrix &)	403
29.16.2.5 BandMatrix(const BandMatrix &gm)	403
29.16.3 Member Function Documentation	403
29.16.3.1 BandWidth() const	403
29.16.3.2 CornerClear() const	403
29.16.3.3 element(int, int)	403

29.16.3.4	element(int, int) const	403
29.16.3.5	GetCol(MatrixRowCol &)	403
29.16.3.6	GetCol(MatrixColX &)	403
29.16.3.7	GetRow(MatrixRowCol &)	403
29.16.3.8	LogDeterminant() const	403
29.16.3.9	MakeSolver()	404
29.16.3.10	Maximum() const	404
29.16.3.11	MaximumAbsoluteValue() const	404
29.16.3.12	Minimum() const	404
29.16.3.13	MinimumAbsoluteValue() const	404
29.16.3.14	NextRow(MatrixRowCol &)	404
29.16.3.15	operator()(int, int)	404
29.16.3.16	operator()(int, int) const	404
29.16.3.17	operator<<(Real)	404
29.16.3.18	operator<<(int f)	405
29.16.3.19	operator<<(const Real *r)	405
29.16.3.20	operator<<(const BaseMatrix &X)	405
29.16.3.21	operator=(const BaseMatrix &)	405
29.16.3.22	operator=(Real f)	405
29.16.3.23	operator=(const BandMatrix &m)	405
29.16.3.24	ReSize(int, int, int)	405
29.16.3.25	ReSize(const GeneralMatrix &A)	405
29.16.3.26	ReSizeForAdd(const GeneralMatrix &A, const GeneralMatrix &B)	405
29.16.3.27	ReSizeForSP(const GeneralMatrix &A, const GeneralMatrix &B)	405
29.16.3.28	RestoreCol(MatrixRowCol &)	406
29.16.3.29	RestoreCol(MatrixColX &c)	406
29.16.3.30	SameStorageType(const GeneralMatrix &A) const	406
29.16.3.31	SetParameters(const GeneralMatrix *)	406
29.16.3.32	SimpleAddOK(const GeneralMatrix *gm)	406
29.16.3.33	Sum() const	406

29.16.3.34	SumAbsoluteValue() const	406
29.16.3.35	SumSquare() const	406
29.16.3.36	Trace() const	407
29.16.3.37	Type() const	407
29.16.4	Member Data Documentation	407
29.16.4.1	lower	407
29.16.4.2	upper	407
29.17	Go::BaryCoordSystem< Dim > Class Template Reference	407
29.17.1	Detailed Description	408
29.17.2	Constructor & Destructor Documentation	408
29.17.2.1	BaryCoordSystem()	408
29.17.2.2	BaryCoordSystem(const Array< double, Dim > *corners)	408
29.17.3	Member Function Documentation	408
29.17.3.1	baryToCart(const Array< T, Dim+1 > &bary_pt) const	408
29.17.3.2	cartToBary(const Array< T, Dim > &cart_pt) const	408
29.17.3.3	corner(int i) const	409
29.17.3.4	read(std::istream &is)	409
29.17.3.5	setCorners(const Array< double, Dim > *corners)	409
29.17.3.6	write(std::ostream &os) const	409
29.18	Go::BaryCoordSystemTriangle3D Class Reference	409
29.18.1	Detailed Description	410
29.18.2	Constructor & Destructor Documentation	410
29.18.2.1	BaryCoordSystemTriangle3D()	410
29.18.2.2	BaryCoordSystemTriangle3D(const Array< double, 3 > *corners)	410
29.18.3	Member Function Documentation	410
29.18.3.1	baryToCart(const Array< T, 3 > &bary_pt) const	410
29.18.3.2	cartToBary(const Array< T, 3 > &cart_pt) const	410
29.19	RBD_COMMON::BaseException Class Reference	411
29.19.1	Detailed Description	412
29.19.2	Constructor & Destructor Documentation	412

29.19.2.1 BaseException(const char *a_what=0)	412
29.19.3 Member Function Documentation	412
29.19.3.1 AddInt(int value)	412
29.19.3.2 AddMessage(const char *a_what)	412
29.19.3.3 what()	412
29.19.4 Member Data Documentation	412
29.19.4.1 LastOne	412
29.19.4.2 Select	412
29.19.4.3 SoFar	412
29.19.4.4 what_error	413
29.20NEWMAT::BaseMatrix Class Reference	413
29.20.1 Detailed Description	416
29.20.2 Member Function Documentation	416
29.20.2.1 AsColumn() const	416
29.20.2.2 AsDiagonal() const	416
29.20.2.3 AsMatrix(int, int) const	416
29.20.2.4 AsRow() const	416
29.20.2.5 AsScalar() const	416
29.20.2.6 BandWidth() const	416
29.20.2.7 CleanUp()	416
29.20.2.8 Column(int) const	417
29.20.2.9 Columns(int, int) const	417
29.20.2.10Determinant() const	417
29.20.2.11Evaluate(MatrixType mt=MatrixTypeUnSp)=0	417
29.20.2.12() const	417
29.20.2.13EQND() const	417
29.20.2.14LogDeterminant() const	417
29.20.2.15Maximum() const	417
29.20.2.16Maximum1(int &i) const	418
29.20.2.17Maximum2(int &i, int &j) const	418

29.20.2.18	MaximumAbsoluteValue() const	418
29.20.2.19	MaximumAbsoluteValue1(int &i) const	418
29.20.2.20	MaximumAbsoluteValue2(int &i, int &j) const	418
29.20.2.21	Minimum() const	418
29.20.2.22	Minimum1(int &i) const	418
29.20.2.23	Minimum2(int &i, int &j) const	418
29.20.2.24	MinimumAbsoluteValue() const	419
29.20.2.25	MinimumAbsoluteValue1(int &i) const	419
29.20.2.26	MinimumAbsoluteValue2(int &i, int &j) const	419
29.20.2.27	Norm1() const	419
29.20.2.28	NormFrobenius() const	419
29.20.2.29	NormInfinity() const	419
29.20.2.30	operator&(const BaseMatrix &) const	419
29.20.2.31	operator*(const BaseMatrix &) const	419
29.20.2.32	operator*(Real) const	419
29.20.2.33	operator+(const BaseMatrix &) const	419
29.20.2.34	operator+(Real) const	419
29.20.2.35	operator-(const BaseMatrix &) const	419
29.20.2.36	operator-(Real) const	419
29.20.2.37	operator-() const	419
29.20.2.38	operator/(Real) const	419
29.20.2.39	operator" (const BaseMatrix &) const	420
29.20.2.40	Reverse() const	420
29.20.2.41	Row(int) const	420
29.20.2.42	Rows(int, int) const	420
29.20.2.43	search(const BaseMatrix *) const =0	420
29.20.2.44	SubMatrix(int, int, int, int) const	420
29.20.2.45	Sum() const	420
29.20.2.46	SumAbsoluteValue() const	420
29.20.2.47	SumSquare() const	420

29.20.2.48SymSubMatrix(int, int) const	421
29.20.2.49() const	421
29.20.2.50Trace() const	421
29.20.3 Friends And Related Function Documentation	421
29.20.3.1 AddedMatrix	421
29.20.3.2 BandMatrix	421
29.20.3.3 ColedMatrix	421
29.20.3.4 ColumnVector	421
29.20.3.5 ConcatenatedMatrix	421
29.20.3.6 CroutMatrix	421
29.20.3.7 DiagedMatrix	422
29.20.3.8 DiagonalMatrix	422
29.20.3.9 GeneralMatrix	422
29.20.3.10GenericMatrix	422
29.20.3.11GetSubMatrix	422
29.20.3.12InvertedMatrix	422
29.20.3.13KPMatrix	422
29.20.3.14LinearEquationSolver	422
29.20.3.15LowerBandMatrix	422
29.20.3.16LowerTriangularMatrix	422
29.20.3.17MatedMatrix	423
29.20.3.18Matrix	423
29.20.3.19MultipliedMatrix	423
29.20.3.20NegatedMatrix	423
29.20.3.21NegShiftedMatrix	423
29.20.3.22hricMatrix	423
29.20.3.23ReturnMatrixX	423
29.20.3.24ReversedMatrix	423
29.20.3.25RowedMatrix	423
29.20.3.26RowVector	423

29.20.3.27	ScaledMatrix	424
29.20.3.28	ShiftedMatrix	424
29.20.3.29	SolvedMatrix	424
29.20.3.30	SPMatrix	424
29.20.3.31	StackedMatrix	424
29.20.3.32	SubtractedMatrix	424
29.20.3.33	SymmetricBandMatrix	424
29.20.3.34	SymmetricMatrix	424
29.20.3.35	TransposedMatrix	424
29.20.3.36	UpperBandMatrix	424
29.20.3.37	UpperTriangularMatrix	425
29.21	Go::BaseSurface Class Reference	425
29.21.1	Detailed Description	425
29.22	Go::BasisDerivs Struct Reference	425
29.22.1	Detailed Description	426
29.22.2	Member Function Documentation	426
29.22.2.1	prepareDerivs(double u, double v, double w, int idx_u, int idx_v, int idx_w, int size)	426
29.22.3	Member Data Documentation	426
29.22.3.1	basisDerivs_u	426
29.22.3.2	basisDerivs_v	426
29.22.3.3	basisDerivs_w	426
29.22.3.4	basisValues	426
29.22.3.5	left_idx	426
29.22.3.6	param	427
29.23	Go::BasisDerivs2 Struct Reference	427
29.23.1	Detailed Description	428
29.23.2	Member Function Documentation	428
29.23.2.1	prepareDerivs(double u, double v, double w, int idx_u, int idx_v, int idx_w, int size)	428
29.23.3	Member Data Documentation	428
29.23.3.1	basisDerivs_u	428

29.23.3.2	basisDerivs_uu	428
29.23.3.3	basisDerivs_uv	428
29.23.3.4	basisDerivs_uw	428
29.23.3.5	basisDerivs_v	428
29.23.3.6	basisDerivs_vv	429
29.23.3.7	basisDerivs_vw	429
29.23.3.8	basisDerivs_w	429
29.23.3.9	basisDerivs_ww	429
29.23.3.10	basisValues	429
29.23.3.11	left_idx	429
29.23.3.12	param	429
29.24	Go::BasisDerivsSf Struct Reference	430
29.24.1	Detailed Description	430
29.24.2	Member Function Documentation	430
29.24.2.1	prepareDerivs(double u, double v, int idx_u, int idx_v, int size)	430
29.24.3	Member Data Documentation	430
29.24.3.1	basisDerivs_u	430
29.24.3.2	basisDerivs_v	430
29.24.3.3	basisValues	431
29.24.3.4	left_idx	431
29.24.3.5	param	431
29.25	Go::BasisDerivsSf2 Struct Reference	431
29.25.1	Detailed Description	432
29.25.2	Member Function Documentation	432
29.25.2.1	prepareDerivs(double u, double v, int idx_u, int idx_v, int size)	432
29.25.3	Member Data Documentation	432
29.25.3.1	basisDerivs_u	432
29.25.3.2	basisDerivs_uu	432
29.25.3.3	basisDerivs_uv	432
29.25.3.4	basisDerivs_v	432

29.25.3.5 basisDerivs_vv	432
29.25.3.6 basisValues	433
29.25.3.7 left_idx	433
29.25.3.8 param	433
29.26Go::BasisPts Struct Reference	433
29.26.1 Detailed Description	433
29.26.2 Member Function Documentation	434
29.26.2.1 preparePts(double u, double v, double w, int idx_u, int idx_v, int idx_w, int size)	434
29.26.3 Member Data Documentation	434
29.26.3.1 basisValues	434
29.26.3.2 left_idx	434
29.26.3.3 param	434
29.27Go::BasisPtsSf Struct Reference	434
29.27.1 Detailed Description	435
29.27.2 Member Function Documentation	435
29.27.2.1 preparePts(double u, double v, int idx_u, int idx_v, int size)	435
29.27.3 Member Data Documentation	435
29.27.3.1 basisValues	435
29.27.3.2 left_idx	435
29.27.3.3 param	435
29.28Go::BdCondFunctor Class Reference	436
29.28.1 Detailed Description	436
29.28.2 Constructor & Destructor Documentation	436
29.28.2.1 BdCondFunctor()	436
29.28.2.2 ~BdCondFunctor()	436
29.28.3 Member Function Documentation	436
29.28.3.1 evaluate(const Point &geom_pos)=0	436
29.29Go::BernsteinMulti Class Reference	436
29.29.1 Detailed Description	438
29.29.2 Constructor & Destructor Documentation	438

29.29.2.1 BernsteinMulti()	438
29.29.2.2 BernsteinMulti(int m, int n, const std::vector< double > &coefs)	438
29.29.2.3 BernsteinMulti(int m, int n, ForwardIterator begin, ForwardIterator end)	439
29.29.2.4 BernsteinMulti(double coef)	439
29.29.3 Member Function Documentation	439
29.29.3.1 bindU(double u) const	439
29.29.3.2 bindV(double v) const	439
29.29.3.3 blossom(const std::vector< double > &uvec, const std::vector< double > &vvec) const	439
29.29.3.4 coefsBegin()	440
29.29.3.5 coefsBegin() const	440
29.29.3.6 coefsEnd()	440
29.29.3.7 coefsEnd() const	440
29.29.3.8 degreeElevate(int du, int dv)	440
29.29.3.9 degreeU() const	441
29.29.3.10 degreeV() const	441
29.29.3.11 deriv(int der1, int der2) const	441
29.29.3.12 detHess() const	441
29.29.3.13 isNonNegative(const double eps=0.0) const	441
29.29.3.14 isStrictlyNegative(const double eps=0.0) const	442
29.29.3.15 isStrictlyPositive(const double eps=0.0) const	442
29.29.3.16 isZero(const double eps=0.0) const	442
29.29.3.17 mean() const	442
29.29.3.18 norm() const	443
29.29.3.19 normalize()	443
29.29.3.20 operator()(double u, double v) const	443
29.29.3.21 operator()(const Vector2D &pt) const	443
29.29.3.22 operator()(const std::vector< double > &pt) const	443
29.29.3.23 operator*=(const BernsteinMulti &multi)	444
29.29.3.24 operator*=(double c)	444
29.29.3.25 operator+=(const BernsteinMulti &multi)	444

29.29.3.26	operator+=(double c)	444
29.29.3.27	operator==(const BernsteinMulti &multi)	444
29.29.3.28	operator==(double c)	444
29.29.3.29	operator/=(double c)	444
29.29.3.30	operator[](int num) const	444
29.29.3.31	operator[](int num)	445
29.29.3.32	pickDomain(double u0, double u1, double v0, double v1) const	445
29.29.3.33	pickLine(Array< double, 2 > a, Array< double, 2 > b) const	445
29.29.3.34	read(std::istream &is)	445
29.29.3.35	traceHess() const	445
29.29.3.36	write(std::ostream &os) const	445
29.30	Go::BernsteinPoly Class Reference	446
29.30.1	Detailed Description	447
29.30.2	Constructor & Destructor Documentation	447
29.30.2.1	BernsteinPoly()	447
29.30.2.2	BernsteinPoly(const std::vector< double > &coefs)	447
29.30.2.3	BernsteinPoly(ForwardIterator begin, ForwardIterator end)	447
29.30.2.4	BernsteinPoly(double coef)	447
29.30.3	Member Function Documentation	448
29.30.3.1	blossom(const std::vector< double > &tvec) const	448
29.30.3.2	coefsBegin()	448
29.30.3.3	coefsBegin() const	448
29.30.3.4	coefsEnd()	448
29.30.3.5	coefsEnd() const	448
29.30.3.6	degree() const	449
29.30.3.7	degreeElevate(int d)	449
29.30.3.8	deriv(int der) const	449
29.30.3.9	integral(double a=0.0, double b=1.0) const	449
29.30.3.10	isStrictlyPositive(const double eps=0.0) const	449
29.30.3.11	isZero(const double eps=0.0) const	450

29.30.3.12	<code>norm()</code> const	450
29.30.3.13	<code>normalize()</code>	450
29.30.3.14	<code>operator()(double t)</code> const	450
29.30.3.15	<code>operator*=(const BernsteinPoly &poly)</code>	450
29.30.3.16	<code>operator*=(double c)</code>	451
29.30.3.17	<code>operator+=(const BernsteinPoly &poly)</code>	451
29.30.3.18	<code>operator+=(double c)</code>	451
29.30.3.19	<code>operator-=(const BernsteinPoly &poly)</code>	451
29.30.3.20	<code>operator-=(double c)</code>	451
29.30.3.21	<code>operator/=(double c)</code>	451
29.30.3.22	<code>operator[](int num)</code> const	451
29.30.3.23	<code>operator[](int num)</code>	451
29.30.3.24	<code>pickInterval(double a, double b)</code> const	451
29.30.3.25	<code>read(std::istream &is)</code>	452
29.30.3.26	<code>write(std::ostream &os)</code> const	452
29.31	Go::BernsteinTetrahedralPoly Class Reference	452
29.31.1	Detailed Description	453
29.31.2	Constructor & Destructor Documentation	453
29.31.2.1	<code>BernsteinTetrahedralPoly()</code>	453
29.31.2.2	<code>BernsteinTetrahedralPoly(int deg, const std::vector< double > &coefs)</code>	453
29.31.2.3	<code>BernsteinTetrahedralPoly(int deg, ForwardIterator begin, ForwardIterator end)</code>	454
29.31.2.4	<code>BernsteinTetrahedralPoly(double coef)</code>	454
29.31.3	Member Function Documentation	454
29.31.3.1	<code>blossom(const std::vector< Array< T, 4 > > &uvec)</code> const	454
29.31.3.2	<code>degree()</code> const	454
29.31.3.3	<code>deriv(int der, const Vector4D &d, BernsteinTetrahedralPoly &btP)</code> const	455
29.31.3.4	<code>norm()</code> const	455
29.31.3.5	<code>normalize()</code>	455
29.31.3.6	<code>operator()(const Array< T, 4 > &u)</code> const	455
29.31.3.7	<code>operator*=(const BernsteinTetrahedralPoly &poly)</code>	456

29.31.3.8	operator*=(double c)	456
29.31.3.9	operator+=(const BernsteinTetrahedralPoly &poly)	456
29.31.3.10	operator+=(double c)	456
29.31.3.11	operator-=(const BernsteinTetrahedralPoly &poly)	456
29.31.3.12	operator-=(double c)	456
29.31.3.13	operator/=(double c)	456
29.31.3.14	operator[](int num) const	456
29.31.3.15	operator[](int num)	457
29.31.3.16	pickLine(const Array< double, 4 > &a, const Array< double, 4 > &b) const	457
29.31.3.17	read(std::istream &is)	457
29.31.3.18	write(std::ostream &os) const	457
29.32	Go::BernsteinTriangularPoly Class Reference	457
29.32.1	Detailed Description	458
29.32.2	Constructor & Destructor Documentation	458
29.32.2.1	BernsteinTriangularPoly()	458
29.32.2.2	BernsteinTriangularPoly(int deg, const std::vector< double > &coefs)	458
29.32.2.3	BernsteinTriangularPoly(int deg, ForwardIterator begin, ForwardIterator end)	459
29.32.2.4	BernsteinTriangularPoly(double coef)	459
29.32.3	Member Function Documentation	459
29.32.3.1	blossom(const std::vector< Array< T, 3 > > &uvec) const	459
29.32.3.2	degree() const	460
29.32.3.3	deriv(int der, const Vector3D &d, BernsteinTriangularPoly &btp) const	460
29.32.3.4	norm() const	460
29.32.3.5	normalize()	460
29.32.3.6	operator()(const Array< T, 3 > &u) const	460
29.32.3.7	operator*=(const BernsteinTriangularPoly &poly)	461
29.32.3.8	operator*=(double c)	461
29.32.3.9	operator+=(const BernsteinTriangularPoly &poly)	461
29.32.3.10	operator+=(double c)	461
29.32.3.11	operator-=(const BernsteinTriangularPoly &poly)	461

29.32.3.12	operator==(double c)	461
29.32.3.13	operator/=(double c)	461
29.32.3.14	operator[](int num) const	462
29.32.3.15	operator[](int num)	462
29.32.3.16	read(std::istream &is)	462
29.32.3.17	write(std::ostream &os) const	462
29.33	Go::BezierTriangle< N > Class Template Reference	462
29.33.1	Detailed Description	463
29.33.2	Constructor & Destructor Documentation	463
29.33.2.1	BezierTriangle()	463
29.33.2.2	BezierTriangle(int deg, const Array< double, N > *cp)	463
29.33.3	Member Function Documentation	463
29.33.3.1	eval(Vector3D beta)	463
29.33.3.2	evalderiv(Vector3D beta, Vector3D dir)	463
29.33.3.3	interpolate(int deg, const Vector3D *nodes, const Array< double, N > *values)	463
29.34	Go::Binomial Class Reference	464
29.34.1	Detailed Description	464
29.34.2	Member Typedef Documentation	464
29.34.2.1	iter	464
29.34.3	Constructor & Destructor Documentation	464
29.34.3.1	Binomial()	464
29.34.3.2	Binomial(int n)	464
29.34.4	Member Function Documentation	465
29.34.4.1	operator()(int n, int i)	465
29.34.4.2	operator[](int n)	465
29.34.4.3	quadrinomial(int n, int i, int j, int k)	465
29.34.4.4	trinomial(int n, int i, int j)	466
29.35	Go::BlockBoundaryCondition Class Reference	466
29.35.1	Detailed Description	467
29.35.2	Constructor & Destructor Documentation	467

29.35.2.1	BlockBoundaryCondition(BdConditionType type)	467
29.35.2.2	~BlockBoundaryCondition()	467
29.35.3	Member Function Documentation	467
29.35.3.1	getBdCoefficients(std::vector< std::pair< int, Point > > &coefs)=0	467
29.35.3.2	getBdCoefficients(std::vector< std::pair< int, Point > > &coefs_bd, std::vector< std::pair< int, Point > > &coefs_inner)=0	468
29.35.3.3	getBdConditionType() const	468
29.35.3.4	getCoefficientsEnumeration(std::vector< int > &local_enumeration)=0	468
29.35.3.5	getCoefficientsEnumeration(std::vector< int > &local_enumeration_bd, std::vector< int > &local_enumeration_bd2)=0	468
29.35.3.6	getSolutionSpaceIdx() const	468
29.35.3.7	getTolerances() const =0	468
29.35.3.8	isDirichlet() const	468
29.35.3.9	update()=0	468
29.35.3.10	updateBoundaryValue(BdCondFuncor *fbd)=0	468
29.35.4	Member Data Documentation	468
29.35.4.1	type_	468
29.36	Go::BlockPointBdCond Class Reference	469
29.36.1	Detailed Description	469
29.36.2	Constructor & Destructor Documentation	469
29.36.2.1	BlockPointBdCond()	469
29.36.2.2	~BlockPointBdCond()	469
29.36.3	Member Function Documentation	469
29.36.3.1	getCoefficientsEnumeration(std::vector< int > &local_enumeration) const =0	469
29.36.3.2	getConditionValue() const =0	470
29.36.3.3	getInterpolationFactors(std::vector< std::pair< int, double > > &factors) const =0	470
29.36.3.4	getSolutionSpaceIdx() const	470
29.37	Go::BlockSolution Class Reference	470
29.37.1	Detailed Description	471
29.37.2	Constructor & Destructor Documentation	471
29.37.2.1	~BlockSolution()	471

29.37.3 Member Function Documentation	471
29.37.3.1 asSfSolution()	471
29.37.3.2 asVolSolution()	471
29.37.3.3 basis(int paddir) const =0	471
29.37.3.4 degree(int paddir) const =0	471
29.37.3.5 dimension() const =0	472
29.37.3.6 distinctKnots(int paddir) const =0	472
29.37.3.7 erasePreEvaluatedBasisFunctions()=0	472
29.37.3.8 getBoundaryCoefficients(int boundary, std::vector< int > &enumeration) const =0	472
29.37.3.9 getBoundaryCoefficients(int boundary, std::vector< int > &enumeration_bd, std::vector< int > &enumeration_b2) const =0	472
29.37.3.10getJacobian(std::vector< int > &index_of_Gauss_point) const =0	472
29.37.3.11getMatchingCoefficients(BlockSolution *other, std::vector< std::pair< int, int > > &enumeration, int match_pos=0) const =0	472
29.37.3.12getNmbOfPointBdConditions() const =0	472
29.37.3.13getTolerances() const =0	472
29.37.3.14increaseDegree(int new_degree, int paddir)=0	472
29.37.3.15insertKnots(const std::vector< int > &knot_intervals, int paddir)=0	473
29.37.3.16insertKnots(const std::vector< double > &knots, int paddir)=0	473
29.37.3.17knots(int paddir) const =0	473
29.37.3.18makeMatchingSplineSpace(BlockSolution *other)=0	473
29.37.3.19matchingSplineSpace(BlockSolution *other) const =0	473
29.37.3.20nmbCoefs() const =0	473
29.37.3.21nmbCoefs(int paddir) const =0	473
29.37.3.22performPreEvaluation(std::vector< std::vector< double > > &Gauss_par)=0	473
29.37.3.23setSolutionCoefficients(const std::vector< double > &coefs)=0	473
29.37.3.24updateConditions()=0	473
29.37.3.25valuesInGaussPoint(const std::vector< int > &index_of_Gauss_point, std::vector< Point > &derivs) const =0	474
29.38Go::Body Class Reference	474
29.38.1 Detailed Description	476
29.38.2 Constructor & Destructor Documentation	476

29.38.2.1	Body()	476
29.38.2.2	Body(std::vector< shared_ptr< SurfaceModel > > &shells, int material_id=-1)	476
29.38.2.3	Body(shared_ptr< SurfaceModel > shell, int material_id=-1)	476
29.38.2.4	~Body()	476
29.38.3	Member Function Documentation	476
29.38.3.1	addBodyPointers()	476
29.38.3.2	addshell(shared_ptr< SurfaceModel > shell)	476
29.38.3.3	areNeighbours(Body *other, shared_ptr< ftSurface > &bd_face1, shared_ptr< ftSurface > &bd_face2, int adj_idx=0) const	476
29.38.3.4	boundingBox() const	476
29.38.3.5	eraseBodyAdjacency()	477
29.38.3.6	getAdjacentBodies(std::vector< Body * > &neighbours)	477
29.38.3.7	getAllShells() const	477
29.38.3.8	getMaterial() const	477
29.38.3.9	getOuterShell() const	477
29.38.3.10	getShell(int idx) const	477
29.38.3.11	getShell(ftSurface *face) const	477
29.38.3.12	getTolerances()	477
29.38.3.13	hasMaterialInfo() const	477
29.38.3.14	isInside(const Point &pnt) const	478
29.38.3.15	nmbOfFaces() const	478
29.38.3.16	nmbOfShells() const	478
29.38.3.17	setMaterial(int material_id)	478
29.38.3.18	vertices() const	478
29.38.4	Member Data Documentation	478
29.38.4.1	material_id_	478
29.38.4.2	shells_	478
29.38.4.3	toptol_	478
29.39	bool Class Reference	479
29.39.1	Detailed Description	479
29.39.2	Constructor & Destructor Documentation	479

29.39.2.1	<code>bool(const int b)</code>	479
29.39.2.2	<code>bool(const void *b)</code>	479
29.39.2.3	<code>bool()</code>	479
29.39.3	Member Function Documentation	479
29.39.3.1	<code>operator int() const</code>	479
29.39.3.2	<code>operator"!() const</code>	479
29.40	<code>Go::BoundaryFunctionInt</code> Struct Reference	480
29.40.1	Detailed Description	480
29.40.2	Constructor & Destructor Documentation	480
29.40.2.1	<code>BoundaryFunctionInt(shared_ptr< ParamFunctionInt > bd_obj, int dir, double par)</code>	480
29.40.3	Member Function Documentation	481
29.40.3.1	<code>getDir()</code>	481
29.40.3.2	<code>getObject()</code>	481
29.40.3.3	<code>getPar()</code>	481
29.40.4	Member Data Documentation	481
29.40.4.1	<code>bd_obj_</code>	481
29.40.4.2	<code>par_</code>	482
29.40.4.3	<code>pardir_</code>	482
29.41	<code>Go::BoundaryGeomInt</code> Struct Reference	482
29.41.1	Detailed Description	483
29.41.2	Constructor & Destructor Documentation	483
29.41.2.1	<code>BoundaryGeomInt(shared_ptr< ParamGeomInt > bd_obj, int dir, double par)</code>	483
29.41.2.2	<code>~BoundaryGeomInt()</code>	483
29.41.3	Member Function Documentation	483
29.41.3.1	<code>getDir()</code>	483
29.41.3.2	<code>getObject()</code>	483
29.41.3.3	<code>getPar()</code>	484
29.41.4	Member Data Documentation	484
29.41.4.1	<code>bd_obj_</code>	484
29.41.4.2	<code>par_</code>	484

29.41.4.3 paddir_	484
29.42Go::BoundaryIntersectionData Struct Reference	484
29.42.1 Detailed Description	485
29.42.2 Member Data Documentation	485
29.42.2.1 dir	485
29.42.2.2 par	485
29.42.2.3 pts	485
29.43Go::BoundedCurve Class Reference	485
29.43.1 Detailed Description	487
29.43.2 Constructor & Destructor Documentation	488
29.43.2.1 BoundedCurve()	488
29.43.2.2 BoundedCurve(shared_ptr< ParamCurve > curve, bool prefer_bd_par, double start_par, double end_par, Point start_pt, Point end_pt)	488
29.43.2.3 BoundedCurve(shared_ptr< ParamCurve > curve, Point start_pt, Point end_pt)	488
29.43.2.4 BoundedCurve(shared_ptr< ParamCurve > curve, double start_par, double end_par)	488
29.43.2.5 ~BoundedCurve()	488
29.43.3 Member Function Documentation	488
29.43.3.1 appendCurve(ParamCurve *cv, bool reparam=true)	488
29.43.3.2 appendCurve(ParamCurve *cv, int continuity, double &dist, bool reparam=true)	488
29.43.3.3 boundingBox() const	489
29.43.3.4 classType()	489
29.43.3.5 clone() const	489
29.43.3.6 closestPoint(const Point &pt, double tmin, double tmax, double &clo_t, Point &clo_pt, double &clo_dist, double const *seed=0) const	489
29.43.3.7 dimension() const	489
29.43.3.8 directionCone() const	490
29.43.3.9 endparam() const	490
29.43.3.10geometryCurve()	490
29.43.3.11instanceType() const	490
29.43.3.12sAxisRotational(Point ¢re, Point &axis, Point &vec, double &angle)	490
29.43.3.13sDegenerate(double degenerate_epsilon)	490

29.43.3.14	<code>isInPlane(const Point &loc, const Point &axis, double eps, Point &normal) const</code>	491
29.43.3.15	<code>isLinear(Point &dir, double tol)</code>	491
29.43.3.16	<code>length(double tol)</code>	491
29.43.3.17	<code>point(Point &pt, double tpar) const</code>	491
29.43.3.18	<code>point(std::vector< Point > &pts, double tpar, int derivs, bool from_right=true) const</code>	492
29.43.3.19	<code>read(std::istream &is)</code>	492
29.43.3.20	<code>reverseParameterDirection(bool switchparam=false)</code>	492
29.43.3.21	<code>setParamBounds(double startpar, double endpar)</code>	492
29.43.3.22	<code>setParameterInterval(double t1, double t2)</code>	493
29.43.3.23	<code>startparam() const</code>	493
29.43.3.24	<code>subCurve(double from_par, double to_par, double fuzzy=DEFAULT_PARAMETER_EPSILON) const</code>	493
29.43.3.25	<code>underlyingCurve() const</code>	494
29.43.3.26	<code>write(std::ostream &os) const</code>	494
29.44	Go::BoundedSurface Class Reference	494
29.44.1	Detailed Description	498
29.44.2	Constructor & Destructor Documentation	498
29.44.2.1	<code>BoundedSurface()</code>	498
29.44.2.2	<code>BoundedSurface(shared_ptr< ParamSurface > surf, std::vector< shared_ptr< CurveOnSurface > > loop, double space_epsilon, bool fix_trim_cvs=true)</code>	498
29.44.2.3	<code>BoundedSurface(shared_ptr< ParamSurface > surf, std::vector< std::vector< shared_ptr< CurveOnSurface > > > loops, double space_epsilon, bool fix_trim_cvs=true)</code>	498
29.44.2.4	<code>BoundedSurface(shared_ptr< ParamSurface > surf, std::vector< std::vector< shared_ptr< CurveOnSurface > > > loops, std::vector< double > space_ε epsilons, bool fix_trim_cvs=true)</code>	499
29.44.2.5	<code>BoundedSurface(shared_ptr< ParamSurface > surf, double space_epsilon)</code>	499
29.44.2.6	<code>BoundedSurface(shared_ptr< ParamSurface > surf, std::vector< CurveLoop > &loops)</code>	499
29.44.2.7	<code>BoundedSurface(shared_ptr< ParamSurface > surf, std::vector< shared_ptr< CurveLoop > > &loops)</code>	499
29.44.2.8	<code>~BoundedSurface()</code>	499
29.44.3	Member Function Documentation	500

29.44.3.1	<code>absolutelyAllBoundaryLoops()</code> const	500
29.44.3.2	<code>allBoundaryLoops(double degenerate_epsilon=DEFAULT_SPACE_EPSILON)</code> const	500
29.44.3.3	<code>allIsSpline()</code> const	500
29.44.3.4	<code>allSplineCopy()</code> const	500
29.44.3.5	<code>analyzeLoops()</code>	500
29.44.3.6	<code>area(double tol)</code> const	500
29.44.3.7	<code>asSplineSurface()</code>	501
29.44.3.8	<code>boundingBox()</code> const	501
29.44.3.9	<code>classType()</code>	501
29.44.3.10	<code>clone()</code> const	501
29.44.3.11	<code>closestBoundaryPoint(const Point &pt, double &clo_u, double &clo_v, Point &clo_pt, double &clo_dist, double epsilon, const RectDomain *domain_of_← interest=NULL, double *seed=0)</code> const	501
29.44.3.12	<code>closestInDomain(double u, double v)</code> const	502
29.44.3.13	<code>closestPoint(const Point &pt, double &clo_u, double &clo_v, Point &clo_pt, double &clo_dist, double epsilon, const RectDomain *domain_of_interest=NULL, double *seed=0)</code> const	502
29.44.3.14	<code>closeToUnderlyingBoundary(double upar, double vpar, double domain_← fraction=0.01)</code> const	503
29.44.3.15	<code>constParamCurve(double parameter, bool direction_is_u)</code> const	503
29.44.3.16	<code>constParamCurves(double parameter, bool pardir_is_u)</code> const	503
29.44.3.17	<code>containingDomain()</code> const	504
29.44.3.18	<code>dimension()</code> const	504
29.44.3.19	<code>ElementBoundaryStatus(int elem_ix, double eps)</code>	504
29.44.3.20	<code>ElementOnBoundary(int elem_ix, double eps)</code>	504
29.44.3.21	<code>evalGrid(int num_u, int num_v, double umin, double umax, double vmin, double vmax, std::vector< double > &points, double nodata_val=-9999)</code> const	505
29.44.3.22	<code>fixInvalidSurface(double &max_loop_gap, double max_tol_mult=1.0)</code>	505
29.44.3.23	<code>getBoundaryInfo(Point &pt1, Point &pt2, double epsilon, SplineCurve *&cv, SplineCurve *&crosscv, double knot_tol=1e-05)</code> const	505
29.44.3.24	<code>getBoundaryInfo(Point &pt1, Point &pt2, std::vector< shared_ptr< CurveOn_← Surface > > &bd_cvs)</code> const	506
29.44.3.25	<code>getCornerPoints(std::vector< std::pair< Point, Point > > &corners)</code> const	506

29.44.3.26	getDegenerateCorners(std::vector< Point > °_corners, double tol) const . . .	506
29.44.3.27	getEpsGeo() const	506
29.44.3.28	getInternalPoint(double &u, double &v) const	506
29.44.3.29	getIsoTrimSurface(double tol) const	506
29.44.3.30	getSurfaceParameter(int loop_idx, int cv_idx, double bd_par) const	507
29.44.3.31	hasUnderlyingSpline(shared_ptr< SplineSurface > &srf)	507
29.44.3.32	inDomain(double u, double v, double eps=1.0e-4) const	507
29.44.3.33	inDomain2(double u, double v, double eps=1.0e-4) const	507
29.44.3.34	instanceType() const	507
29.44.3.35	sAxisRotational(Point ¢re, Point &axis, Point &vec, double &angle)	507
29.44.3.36	sDegenerate(bool &b, bool &r, bool &t, bool &l, double tolerance) const	507
29.44.3.37	sIsoTrimmed(double tol) const	508
29.44.3.38	sLinear(Point &dir1, Point &dir2, double tol)	508
29.44.3.39	sPlanar(Point &normal, double tol)	508
29.44.3.40	sValid(int &valid_state) const	508
29.44.3.41	loop(int idx)	508
29.44.3.42	makeBoundaryCurvesG1(double kink)	508
29.44.3.43	makeUnderlyingSpline()	509
29.44.3.44	maxLoopGap()	509
29.44.3.45	maxLoopSfDist(int loop_ind, int nmb_seg_samples=100)	509
29.44.3.46	nextSegmentVal(int dir, double par, bool forward, double tol) const	509
29.44.3.47	normal(Point &n, double upar, double vpar) const	509
29.44.3.48	normalCone() const	510
29.44.3.49	numberOfLoops() const	510
29.44.3.50	onBoundary(double u, double v, double eps=1.0e-4) const	510
29.44.3.51	orientationIsSet()	510
29.44.3.52	orientationOK()	510
29.44.3.53	outerBoundaryLoop(double degenerate_epsilon=DEFAULT_SPACE_EPSILON) const	510
29.44.3.54	parameterDomain() const	511
29.44.3.55	point(Point &pt, double upar, double vpar) const	511

29.44.3.56	<code>point(std::vector< Point > &pts, double upar, double vpar, int derivs, bool u_↔ from_right=true, bool v_from_right=true, double resolution=1.0e-12) const</code>	511
29.44.3.57	<code>read(std::istream &is)</code>	512
29.44.3.58	<code>read(std::istream &is, bool fix_trim_cvs)</code>	512
29.44.3.59	<code>removeMismatchCurves(double max_tol_mult)</code>	512
29.44.3.60	<code>removeSmallBoundaryCurves(double gap, double neighbour, double kink)</code>	512
29.44.3.61	<code>reverseParameterDirection(bool direction_is_u)</code>	512
29.44.3.62	<code>setIterator(IteratorType type)</code>	513
29.44.3.63	<code>setOrientationOK()</code>	513
29.44.3.64	<code>setParameterDomain(double u1, double u2, double v1, double v2)</code>	513
29.44.3.65	<code>simplifyBdLoops(double tol, double ang_tol, double &max_dist)</code>	513
29.44.3.66	<code>splitSingleLoops()</code>	513
29.44.3.67	<code>subSurfaces(double from_upar, double from_vpar, double to_upar, double to_↔ vpar, double fuzzy=DEFAULT_PARAMETER_EPSILON) const</code>	513
29.44.3.68	<code>swapParameterDirection()</code>	514
29.44.3.69	<code>tangentCone(bool pardir_is_u) const</code>	514
29.44.3.70	<code>turnLoopOrientation(int idx)</code>	514
29.44.3.71	<code>turnOrientation()</code>	514
29.44.3.72	<code>underlyingSurface()</code>	515
29.44.3.73	<code>underlyingSurface() const</code>	515
29.44.3.74	<code>write(std::ostream &os) const</code>	515
29.44.4	Friends And Related Function Documentation	515
29.44.4.1	<code>GeometryTools::setParameterDomain</code>	515
29.45	<code>Go::BoundingBox</code> Class Reference	515
29.45.1	Detailed Description	516
29.45.2	Constructor & Destructor Documentation	516
29.45.2.1	<code>BoundingBox()</code>	516
29.45.2.2	<code>BoundingBox(int dim)</code>	517
29.45.2.3	<code>BoundingBox(const Point &low, const Point &high)</code>	517
29.45.2.4	<code>~BoundingBox()</code>	517
29.45.3	Member Function Documentation	517

29.45.3.1 addUnionWith(const Point &pt)	517
29.45.3.2 addUnionWith(const BoundingBox &box)	517
29.45.3.3 check() const	517
29.45.3.4 containsBox(const BoundingBox &box, double tol=0.0) const	517
29.45.3.5 containsPoint(const Point &pt, double tol=0.0) const	517
29.45.3.6 dimension() const	517
29.45.3.7 getOverlap(const BoundingBox &box, double &overlap, double tol=0.0) const	518
29.45.3.8 high() const	518
29.45.3.9 lineIntersect(const Point &p1, const Point &dir) const	518
29.45.3.10 low() const	518
29.45.3.11 overlaps(const BoundingBox &box, double tol=0.0) const	518
29.45.3.12 read(std::istream &is)	518
29.45.3.13 setFromArray(const FloatType *start, const FloatType *end, int dim)	518
29.45.3.14 setFromArray(ForwardIterator start, ForwardIterator end, int dim)	518
29.45.3.15 setFromPoints(const Point &low, const Point &high)	518
29.45.3.16 setFromPoints(const std::vector< Point > &points)	519
29.45.3.17 unset()	519
29.45.3.18 valid() const	519
29.45.3.19 write(std::ostream &os) const	519
29.46 Go::boxStructuring::BoundingBoxStructure Class Reference	519
29.46.1 Detailed Description	520
29.46.2 Member Function Documentation	520
29.46.2.1 addBox(shared_ptr< SubSurfaceBoundingBox > box)	520
29.46.2.2 addSurface(shared_ptr< SurfaceData > surface)	520
29.46.2.3 big_vox_low() const	520
29.46.2.4 boxes_in_voxel(int i, int j, int k) const	521
29.46.2.5 BuildVoxelStructure(BoundingBox bigbox, double volume_reduction)	521
29.46.2.6 closestPoint(int box_idx, bool any_tested, double best_dist, bool isInside, const Point &pt, double &clo_u, double &clo_v, Point &clo_pt, double &clo_dist, double epsilon, const RectDomain *domain_of_interest, double *seed) const	521
29.46.2.7 getBox(int i) const	521

29.46.2.8	<code>getSurface(int i) const</code>	521
29.46.2.9	<code>n_boxes() const</code>	521
29.46.2.10	<code>n_surfaces() const</code>	521
29.46.2.11	<code>n_voxels_x() const</code>	522
29.46.2.12	<code>n_voxels_y() const</code>	522
29.46.2.13	<code>n_voxels_z() const</code>	522
29.46.2.14	<code>setSurfaceCopies(int nmb_copies)</code>	522
29.46.2.15	<code>voxel_length() const</code>	522
29.47	<code>Go::LRSplineSurface::BSKey</code> Struct Reference	522
29.47.1	Detailed Description	523
29.47.2	Member Function Documentation	523
29.47.2.1	<code>operator<(const BSKey &rhs) const</code>	523
29.47.3	Member Data Documentation	523
29.47.3.1	<code>u_max</code>	523
29.47.3.2	<code>u_min</code>	523
29.47.3.3	<code>u_mult1</code>	523
29.47.3.4	<code>u_mult2</code>	523
29.47.3.5	<code>v_max</code>	523
29.47.3.6	<code>v_min</code>	524
29.47.3.7	<code>v_mult1</code>	524
29.47.3.8	<code>v_mult2</code>	524
29.48	<code>Go::BsplineBasis</code> Class Reference	524
29.48.1	Detailed Description	526
29.48.2	Constructor & Destructor Documentation	526
29.48.2.1	<code>BsplineBasis()</code>	526
29.48.2.2	<code>BsplineBasis(int number, int order, RandomIterator knotstart)</code>	526
29.48.2.3	<code>BsplineBasis(int order, RandomIterator knotstart, RandomIterator knotend)</code>	527
29.48.2.4	<code>~BsplineBasis()</code>	527
29.48.3	Member Function Documentation	527
29.48.3.1	<code>begin() const</code>	527

29.48.3.2 begin()	527
29.48.3.3 check() const	527
29.48.3.4 cNDiscontinuities(std::vector< double > &cNDisconts, int depth) const	527
29.48.3.5 coefsAffectingParam(double tpar, int &first_coef, int &last_coef) const	528
29.48.3.6 computeBasisValues(double t, int derivs=0) const	528
29.48.3.7 computeBasisValues(double t, double *basisvals_start, int derivs=0, double resolution=1.0e-12) const	528
29.48.3.8 computeBasisValues(const double *parvals_start, const double *parvals_end, double *basisvals_start, int *knotinter_start, int derivs=0) const	529
29.48.3.9 computeBasisValuesLeft(double tval, int derivs) const	529
29.48.3.10 computeBasisValuesLeft(const double *parvals_start, const double *parvals_end, double *basisvals_start, int *knotinter_start, int derivs) const	529
29.48.3.11 end() const	529
29.48.3.12 end()	530
29.48.3.13 endMultiplicity(bool atstart) const	530
29.48.3.14 endparam() const	530
29.48.3.15 extendedBasis(int order) const	530
29.48.3.16 getKnots()	530
29.48.3.17 getMinContinuity() const	530
29.48.3.18 grevilleParameter(int index) const	530
29.48.3.19 increaseOrder(int order)	531
29.48.3.20 indistinctKnots(double tol, std::vector< double > &first_knotval) const	531
29.48.3.21 insertKnot(double apar)	531
29.48.3.22 insertKnot(const std::vector< double > &new_knots)	531
29.48.3.23 sKreg() const	531
29.48.3.24 sOK() const	532
29.48.3.25 knotInterval(double t) const	532
29.48.3.26 knotIntervalFuzzy(double &t, double tol=DEFAULT_PARAMETER_EPSILON) const	532
29.48.3.27 knotMultiplicities(std::vector< int > &multiplicities) const	532
29.48.3.28 knotMultiplicity(const double parval) const	532
29.48.3.29 knotsSimple(std::vector< double > &result) const	533

29.48.3.30	lastKnotInterval() const	533
29.48.3.31	missingKnots(const BsplineBasis &other, double tol=DEFAULT_PARAMETER← _EPSILON) const	533
29.48.3.32	numCoefs() const	533
29.48.3.33	numElem() const	533
29.48.3.34	order() const	533
29.48.3.35	read(std::istream &is)	533
29.48.3.36	read_bin(std::istream &is)	534
29.48.3.37	removeKnot(double old_knot)	534
29.48.3.38	rescale(double new_start, double new_end)	534
29.48.3.39	reverseParameterDirection()	534
29.48.3.40	sameSplineSpace(const BsplineBasis &other, double tol=DEFAULT_PARAME← TER_EPSILON) const	534
29.48.3.41	setData(int number, int order, RandomIterator knotstart)	535
29.48.3.42	startparam() const	535
29.48.3.43	subBasis(double tmin, double tmax, double knot_diff_tol=1e-05) const	535
29.48.3.44	swap(BsplineBasis &other)	535
29.48.3.45	write(std::ostream &os) const	535
29.48.3.46	write_bin(std::ostream &os) const	536
29.48.4	Friends And Related Function Documentation	536
29.48.4.1	computeBasisValuesLeft(double tval, double *basisvals_start, int derivs, double resolution=1.0e-12) const	536
29.49	Go::CachedInterval Struct Reference	536
29.49.1	Detailed Description	537
29.49.2	Constructor & Destructor Documentation	537
29.49.2.1	CachedInterval()	537
29.49.3	Member Data Documentation	537
29.49.3.1	cached	537
29.49.3.2	inside	537
29.49.3.3	outside	537
29.50	NEWMAT::CannotBuildException Class Reference	538
29.50.1	Detailed Description	539

29.50.2 Constructor & Destructor Documentation	539
29.50.2.1 CannotBuildException(const char *matrix)	539
29.50.3 Member Data Documentation	539
29.50.3.1 Select	539
29.51 Go::CellDivision Class Reference	539
29.51.1 Detailed Description	540
29.51.2 Constructor & Destructor Documentation	540
29.51.2.1 CellDivision()	540
29.51.2.2 CellDivision(std::vector< ftSurface * > faces, int n1, int n2)	540
29.51.2.3 CellDivision(std::vector< ftSurface * > faces, int n1, int n2, int n3)	540
29.51.2.4 CellDivision(std::vector< ftSurface * > faces, std::vector< int > n)	540
29.51.2.5 ~CellDivision()	540
29.51.3 Member Function Documentation	540
29.51.3.1 addFace(ftSurface *face)	540
29.51.3.2 big_box() const	540
29.51.3.3 constructCells()	540
29.51.3.4 constructCells(int n1, int n2)	540
29.51.3.5 constructCells(int n1, int n2, int n3)	540
29.51.3.6 constructCells(std::vector< int > n)	540
29.51.3.7 getCell(int idx) const	540
29.51.3.8 numCells() const	540
29.51.3.9 removeFace(ftSurface *face)	541
29.52 Go::Circle Class Reference	541
29.52.1 Detailed Description	543
29.52.2 Constructor & Destructor Documentation	544
29.52.2.1 Circle()	544
29.52.2.2 Circle(double radius, Point centre, Point normal, Point x_axis, bool is← Reversed=false)	544
29.52.2.3 ~Circle()	544
29.52.3 Member Function Documentation	544
29.52.3.1 appendCurve(ParamCurve *cv, bool reparam=true)	544

29.52.3.2	<code>appendCurve(ParamCurve *cv, int continuity, double &dist, bool reparam=true)</code>	544
29.52.3.3	<code>boundingBox() const</code>	545
29.52.3.4	<code>classType()</code>	545
29.52.3.5	<code>clone() const</code>	545
29.52.3.6	<code>closestPoint(const Point &pt, double tmin, double tmax, double &clo_t, Point &clo_pt, double &clo_dist, double const *seed=0) const</code>	545
29.52.3.7	<code>createSplineCurve() const</code>	545
29.52.3.8	<code>dimension() const</code>	545
29.52.3.9	<code>directionCone() const</code>	546
29.52.3.10	<code>endparam() const</code>	546
29.52.3.11	<code>geometryCurve()</code>	546
29.52.3.12	<code>getCentre() const</code>	546
29.52.3.13	<code>getNormal() const</code>	546
29.52.3.14	<code>getRadius() const</code>	546
29.52.3.15	<code>getXAxis() const</code>	547
29.52.3.16	<code>instanceType() const</code>	547
29.52.3.17	<code>isAxisRotational(Point &centre, Point &axis, Point &vec, double &angle)</code>	547
29.52.3.18	<code>isClosed() const</code>	547
29.52.3.19	<code>isDegenerate(double degenerate_epsilon)</code>	547
29.52.3.20	<code>isInPlane(const Point &loc, const Point &axis, double eps, Point &normal) const</code>	547
29.52.3.21	<code>isInPlane(const Point &norm, double eps, Point &pos) const</code>	547
29.52.3.22	<code>length(double tol)</code>	548
29.52.3.23	<code>point(Point &pt, double tpar) const</code>	549
29.52.3.24	<code>point(std::vector< Point > &pts, double tpar, int derivs, bool from_right=true) const</code>	549
29.52.3.25	<code>read(std::istream &is)</code>	549
29.52.3.26	<code>setParamBounds(double startpar, double endpar)</code>	550
29.52.3.27	<code>setParameterInterval(double t1, double t2)</code>	550
29.52.3.28	<code>setSpanningVectors()</code>	550
29.52.3.29	<code>startparam() const</code>	550
29.52.3.30	<code>subCurve(double from_par, double to_par, double fuzzy=DEFAULT_PARAMETER_EPSILON) const</code>	550

29.52.3.31	swapParameters2D()	551
29.52.3.32	translateCurve(const Point &dir)	551
29.52.3.33	write(std::ostream &os) const	551
29.52.4	Member Data Documentation	551
29.52.4.1	centre_	551
29.52.4.2	endparam_	551
29.52.4.3	normal_	551
29.52.4.4	radius_	551
29.52.4.5	startparam_	552
29.52.4.6	vec1_	552
29.52.4.7	vec2_	552
29.53	Go::ClosestPointCalculator Class Reference	552
29.53.1	Detailed Description	552
29.53.2	Constructor & Destructor Documentation	552
29.53.2.1	ClosestPointCalculator(shared_ptr< ParamObjectInt > obj)	552
29.53.3	Member Function Documentation	552
29.53.3.1	compute(const Point &pt, double *unknown_par, double eps)	552
29.54	NEWMAT::ColedMatrix Class Reference	553
29.54.1	Detailed Description	554
29.54.2	Constructor & Destructor Documentation	554
29.54.2.1	~ColedMatrix()	554
29.54.3	Member Function Documentation	555
29.54.3.1	BandWidth() const	555
29.54.3.2	Evaluate(MatrixType mt=MatrixTypeUnSp)	555
29.54.4	Friends And Related Function Documentation	555
29.54.4.1	BaseMatrix	555
29.55	NEWMAT::ColumnVector Class Reference	555
29.55.1	Detailed Description	558
29.55.2	Constructor & Destructor Documentation	558
29.55.2.1	ColumnVector()	558

29.55.2.2	<code>~ColumnVector()</code>	558
29.55.2.3	<code>ColumnVector(ArrayLengthSpecifier n)</code>	558
29.55.2.4	<code>ColumnVector(const BaseMatrix &)</code>	558
29.55.2.5	<code>ColumnVector(const ColumnVector &gm)</code>	558
29.55.3	Member Function Documentation	558
29.55.3.1	<code>CleanUp()</code>	558
29.55.3.2	<code>element(int)</code>	558
29.55.3.3	<code>element(int) const</code>	558
29.55.3.4	<code>nric() const</code>	558
29.55.3.5	<code>operator()(int)</code>	559
29.55.3.6	<code>operator()(int) const</code>	559
29.55.3.7	<code>operator=(const BaseMatrix &)</code>	559
29.55.3.8	<code>operator=(Real f)</code>	559
29.55.3.9	<code>operator=(const ColumnVector &m)</code>	559
29.55.3.10	<code>ReSize(int)</code>	559
29.55.3.11	<code>ReSize(int, int)</code>	559
29.55.3.12	<code>ReSize(const GeneralMatrix &A)</code>	559
29.55.3.13	<code>Transpose(TransposedMatrix *, MatrixType)</code>	559
29.55.3.14	<code>Type() const</code>	559
29.56	<code>Go::CompleteEdgeNet</code> Class Reference	560
29.56.1	Detailed Description	560
29.56.2	Constructor & Destructor Documentation	560
29.56.2.1	<code>CompleteEdgeNet(shared_ptr< SurfaceModel > sfmodel, bool perform_step2, bool smooth_connections)</code>	560
29.56.2.2	<code>~CompleteEdgeNet()</code>	560
29.56.3	Member Function Documentation	560
29.56.3.1	<code>getMissingEdges()</code>	561
29.56.3.2	<code>getRegularizedModel()</code>	561
29.56.3.3	<code>perform(std::vector< std::pair< Point, Point > > &corr_vx_pts)</code>	561
29.57	<code>Go::ComplexityInfo</code> Class Reference	561
29.57.1	Detailed Description	562

29.57.2 Constructor & Destructor Documentation	562
29.57.2.1 ComplexityInfo()	562
29.57.2.2 ~ComplexityInfo()	562
29.57.3 Member Function Documentation	562
29.57.3.1 getBoxOverlap()	562
29.57.3.2 getConeOverlap()	562
29.57.3.3 getNmbIntpts()	563
29.57.3.4 getNmbSingpts()	563
29.57.3.5 isComplex()	563
29.57.3.6 setBoxOverlap(double overlap)	563
29.57.3.7 setComplex()	563
29.57.3.8 setConeOverlap(double overlap)	564
29.57.3.9 setNmbIntpts(int nmbpts)	564
29.57.3.10setNmbSingpts(int nmbpts)	564
29.58Go::CompositeBox Class Reference	564
29.58.1 Detailed Description	565
29.58.2 Constructor & Destructor Documentation	565
29.58.2.1 CompositeBox(RandomAccessIterator start, int dim, int num_u, int num_v)	565
29.58.2.2 CompositeBox(const Point &low, const Point &high)	565
29.58.2.3 ~CompositeBox()	566
29.58.3 Member Function Documentation	566
29.58.3.1 containsBox(const CompositeBox &box, double toli=0.0, double tole=0.0) const	566
29.58.3.2 containsPoint(const Point &pt, double toli=0.0, double tole=0.0) const	566
29.58.3.3 dimension() const	566
29.58.3.4 edge() const	566
29.58.3.5 getOverlap(const CompositeBox &box, double &overlap, double toli=0.0, double tole=0.0) const	566
29.58.3.6 high(double toli=0.0, double tole=0.0) const	566
29.58.3.7 inner() const	566
29.58.3.8 low(double toli=0.0, double tole=0.0) const	567
29.58.3.9 overlaps(const CompositeBox &box, double toli=0.0, double tole=0.0) const	567

29.58.3.10	read(std::istream &is)	567
29.58.3.11	setFromArray(RandomAccessIterator start, int dim, int num_u, int num_v)	567
29.58.3.12	setFromPoints(const Point &low, const Point &high)	567
29.58.3.13	write(std::ostream &os) const	567
29.59	Go::CompositeCurve Class Reference	568
29.59.1	Detailed Description	569
29.59.2	Constructor & Destructor Documentation	569
29.59.2.1	CompositeCurve(double gap, double neighbour, double kink, double bend, std::vector< shared_ptr< ParamCurve > > &curves)	569
29.59.2.2	~CompositeCurve()	570
29.59.3	Member Function Documentation	570
29.59.3.1	append(shared_ptr< ParamCurve > curve)	570
29.59.3.2	boundingBox()	570
29.59.3.3	boundingBox(int idx) const	570
29.59.3.4	clone() const	570
29.59.3.5	closestPoint(Point &pnt, Point &clo_pnt, int &idx, double clo_par[], double &dist)	571
29.59.3.6	closestPoint(Point &pnt)	571
29.59.3.7	curvature(int idx, double *par) const	571
29.59.3.8	evaluate(int idx, double par[], Point &pnt) const	572
29.59.3.9	evaluate(int idx, double par[], int nder, std::vector< Point > &der) const	573
29.59.3.10	evaluateCurve(double par, Point &pnt) const	573
29.59.3.11	evaluateCurve(double par, int nder, std::vector< Point > &der) const	573
29.59.3.12	extremalPoint(Point &dir, Point &ext_pnt, int &idx, double ext_par[])	574
29.59.3.13	getCurve(int idx) const	574
29.59.3.14	getGlobalPar(int idx, double par) const	574
29.59.3.15	getIndex(ParamCurve *curve) const	575
29.59.3.16	getLocalPar(double global_par, int &idx, double &local_par) const	575
29.59.3.17	hit(const Point &point, const Point &dir, PointOnCurve &result)	575
29.59.3.18	intersect(const ftLine &line)	576
29.59.3.19	intersect_plane(const ftPlane &plane)	576
29.59.3.20	sDegenerate(int idx) const	576

29.59.3.21	nmbEntities() const	577
29.59.3.22	parameterRange(double &start, double &end) const	577
29.59.3.23	tessellate(std::vector< shared_ptr< GeneralMesh > > &meshes) const	577
29.59.3.24	tessellate(int resolution[], std::vector< shared_ptr< GeneralMesh > > &meshes) const	577
29.59.3.25	tessellate(const std::vector< shared_ptr< ParamCurve > > &curves, int resolution[], std::vector< shared_ptr< GeneralMesh > > &meshes) const	578
29.59.3.26	tessellate(double density, std::vector< shared_ptr< GeneralMesh > > &meshes) const	578
29.59.3.27	tessellate(const std::vector< shared_ptr< ParamCurve > > &curves, double density, std::vector< shared_ptr< GeneralMesh > > &meshes) const	578
29.59.3.28	tessellatedCtrPolygon(std::vector< shared_ptr< LineCloud > > &ctr_pol) const	579
29.59.3.29	tessellatedCtrPolygon(const std::vector< shared_ptr< ParamCurve > > &curves, std::vector< shared_ptr< LineCloud > > &ctr_pol) const	579
29.59.3.30	turn(int idx)	579
29.59.3.31	turn()	579
29.60	Go::CompositeModel Class Reference	580
29.60.1	Detailed Description	581
29.60.2	Constructor & Destructor Documentation	581
29.60.2.1	CompositeModel(double gap, double neighbour, double kink, double bend)	581
29.60.2.2	~CompositeModel()	581
29.60.3	Member Function Documentation	582
29.60.3.1	asSurfaceModel()	582
29.60.3.2	boundingBox()=0	582
29.60.3.3	boundingBox(int idx) const =0	582
29.60.3.4	boxExtreme(const BoundingBox &box, const Point &dir, const Point &curr_pnt) const	582
29.60.3.5	boxVecDist(const BoundingBox &b, const Point &v) const	582
29.60.3.6	clone() const	582
29.60.3.7	closestPoint(Point &pnt, Point &clo_pnt, int &idx, double clo_par[], double &dist)=0	583
29.60.3.8	curvature(int idx, double *par) const =0	583
29.60.3.9	evaluate(int idx, double par[], Point &pnt) const =0	583
29.60.3.10	evaluate(int idx, double par[], int nder, std::vector< Point > &der) const =0	584

29.60.3.11	extremalPoint(Point &dir, Point &ext_pnt, int &idx, double ext_par[])=0	584
29.60.3.12	getTolerances()	584
29.60.3.13	intersect(const ftLine &line)=0	585
29.60.3.14	intersect_plane(const ftPlane &plane)=0	585
29.60.3.15	sDegenerate(int idx) const =0	585
29.60.3.16	nmbEntities() const =0	585
29.60.3.17	setTolerances(double gap, double neighbour, double kink, double bend)	586
29.60.3.18	tesselate(std::vector< shared_ptr< GeneralMesh > > &meshes) const =0	586
29.60.3.19	tesselate(int resolution[], std::vector< shared_ptr< GeneralMesh > > &meshes) const =0	586
29.60.3.20	tesselate(double density, std::vector< shared_ptr< GeneralMesh > > &meshes) const =0	587
29.60.3.21	tesselatedCtrPolygon(std::vector< shared_ptr< LineCloud > > &ctr_pol) const =0	587
29.60.3.22	turn(int idx)=0	587
29.60.3.23	turn()=0	587
29.60.4	Member Data Documentation	587
29.60.4.1	closest_idx_	587
29.60.4.2	toptol_	588
29.61	Go::CompositeModelFactory Class Reference	588
29.61.1	Detailed Description	589
29.61.2	Constructor & Destructor Documentation	589
29.61.2.1	CompositeModelFactory(double approxtol, double gap, double neighbour, double kink, double bend)	589
29.61.2.2	~CompositeModelFactory()	589
29.61.3	Member Function Documentation	589
29.61.3.1	createCircularArc(Point centre, Point start_pt, double angle, Point axis)	589
29.61.3.2	createEllipticArc(Point centre, Point direction, double r1, double r2, double start- par, double angle, Point axis)	589
29.61.3.3	createEmpty()	589
29.61.3.4	createFromBox(Point corner, Point side_vec, Point plane_vec, double side1_↔ length, double side2_length, double side3_length)	589
29.61.3.5	createFromCylinder(Point bottom_pos, Point cylinder_axis, Point major_axis, Point minor_axis)	590

29.61.3.6	<code>createFromG2(std::istream &is, bool prefer_surfacemodel=true)</code>	590
29.61.3.7	<code>createFromIges(std::istream &is, bool use_filetol=false, bool prefer_surfacemodel=true)</code>	590
29.61.3.8	<code>createFromSisl(std::vector< SISLSurf * > &surfaces)</code>	590
29.61.3.9	<code>createFromSisl(std::vector< SISLCurve * > &curves)</code>	590
29.61.3.10	<code>createFromSphere(Point centre, double radius)</code>	590
29.61.3.11	<code>createFromSphere(Point centre, Point axis, Point equator, int latitude, int longitude)</code>	590
29.61.3.12	<code>createLineSegment(Point startpt, Point endpt)</code>	590
29.61.3.13	<code>getModelsFromG2(std::istream &is, bool use_filetol=false)</code>	591
29.61.3.14	<code>getModelsFromIges(std::istream &is, bool use_filetol=false)</code>	591
29.61.3.15	<code>interpolateCurves(const std::vector< shared_ptr< SplineCurve > > &curves, std::vector< double > &parvals, int open, int degree=3)</code>	591
29.61.3.16	<code>interpolateCurves(const std::vector< shared_ptr< SplineCurve > > &curves, std::vector< int > &crv_type, std::vector< double > &parvals, int open, int degree=3)</code>	591
29.61.3.17	<code>interpolateCurves2(const std::vector< shared_ptr< SplineCurve > > &curves, std::vector< double > &param, int open, int degree=3)</code>	591
29.61.3.18	<code>interpolateCurves2(const std::vector< shared_ptr< SplineCurve > > &curves, std::vector< int > &crv_type, std::vector< double > &param, int open, int degree=3)</code>	591
29.62	<code>Go::CompositeModelFileHandler Class Reference</code>	592
29.62.1	Detailed Description	593
29.62.2	Constructor & Destructor Documentation	593
29.62.2.1	<code>CompositeModelFileHandler()</code>	593
29.62.2.2	<code>~CompositeModelFileHandler()</code>	593
29.62.3	Member Function Documentation	593
29.62.3.1	<code>clear()</code>	593
29.62.3.2	<code>createGeomObject(const ObjectHeader &obj_header) const</code>	593
29.62.3.3	<code>readBody(const char *filein, int id=-1)</code>	593
29.62.3.4	<code>readFaces(const char *filein)</code>	593
29.62.3.5	<code>readGeomObj(const char *filein)</code>	593
29.62.3.6	<code>readShell(const char *filein, int id=-1)</code>	593
29.62.3.7	<code>readSurfModel(const char *filein, int id=-1)</code>	593

29.62.3.8	<code>writeBody(shared_ptr< Body > &body, std::ostream &os, int body_id=-1)</code>	593
29.62.3.9	<code>writeEnd(std::ostream &os)</code>	594
29.62.3.10	<code>writeFaces(std::ostream &os)</code>	594
29.62.3.11	<code>writeGeomObj(const std::vector< shared_ptr< GeomObject > > &geom_obj, const std::vector< int > &obj_id, std::ostream &os)</code>	594
29.62.3.12	<code>writeHeader(const std::string &file_content_info, std::ostream &os)</code>	594
29.62.3.13	<code>writeStart(std::ostream &os)</code>	594
29.62.3.14	<code>writeSurfModel(Go::SurfaceModel &surf_model, std::ostream &os, int surfmodel_id=-1, bool faces=true)</code>	594
29.62.4	Member Data Documentation	594
29.62.4.1	<code>edges_</code>	594
29.62.4.2	<code>faces2_</code>	594
29.62.4.3	<code>faces_</code>	594
29.62.4.4	<code>geom_objects2_</code>	594
29.62.4.5	<code>geom_objects_</code>	594
29.62.4.6	<code>indent_</code>	594
29.62.4.7	<code>loops_</code>	595
29.62.4.8	<code>MAJOR_VERSION_</code>	595
29.62.4.9	<code>MINOR_VERSION_</code>	595
29.62.4.10	<code>shells_</code>	595
29.62.4.11	<code>vertices_</code>	595
29.63	<code>Go::CompositeSurface</code> Class Reference	595
29.63.1	Detailed Description	597
29.63.2	Constructor & Destructor Documentation	597
29.63.2.1	<code>CompositeSurface()</code>	597
29.63.2.2	<code>CompositeSurface(const std::vector< SplineSurface > &domain_maps, const SplineSurface &surf)</code>	598
29.63.2.3	<code>~CompositeSurface()</code>	598
29.63.3	Member Function Documentation	598
29.63.3.1	<code>allBoundaryLoops(double degenerate_epsilon=DEFAULT_SPACE_EPSILON) const</code>	598
29.63.3.2	<code>area(double tol) const</code>	598

29.63.3.3 boundingBox() const	599
29.63.3.4 classType()	599
29.63.3.5 clone() const	599
29.63.3.6 closestBoundaryPoint(const Point &pt, double &clo_u, double &clo_v, Point &clo_pt, double &clo_dist, double epsilon, const RectDomain *rd=NULL, double *seed=0) const	599
29.63.3.7 closestInDomain(double u, double v) const	599
29.63.3.8 closestPoint(const Point &pt, double &clo_u, double &clo_v, Point &clo_pt, double &clo_dist, double epsilon, const RectDomain *domain_of_interest=NULL, double *seed=0) const	599
29.63.3.9 compositeBox() const	600
29.63.3.10 constParamCurves(double parameter, bool paddir_is_u) const	600
29.63.3.11 containingDomain() const	601
29.63.3.12 dimension() const	601
29.63.3.13 getBoundaryInfo(Point &pt1, Point &pt2, double epsilon, SplineCurve *&cv, SplineCurve *&crosscv, double knot_tol=1e-05) const	601
29.63.3.14 getCornerPoints(std::vector< std::pair< Point, Point > > &corners) const	601
29.63.3.15 getDegenerateCorners(std::vector< Point > °_corners, double tol) const	602
29.63.3.16 inDomain(double u, double v, double eps=1.0e-4) const	602
29.63.3.17 inDomain2(double u, double v, double eps=1.0e-4) const	602
29.63.3.18 instanceType() const	602
29.63.3.19 sDegenerate(bool &b, bool &r, bool &t, bool &l, double tolerance) const	602
29.63.3.20 nextSegmentVal(int dir, double par, bool forward, double tol) const	603
29.63.3.21 normal(Point &n, double upar, double vpar) const	603
29.63.3.22 normalCone() const	604
29.63.3.23 onBoundary(double u, double v, double eps=1.0e-4) const	604
29.63.3.24 outerBoundaryLoop(double degenerate_epsilon=DEFAULT_SPACE_EPSILON) const	604
29.63.3.25 parameterDomain() const	605
29.63.3.26 point(Point &pt, double upar, double vpar) const	605
29.63.3.27 point(std::vector< Point > &pts, double upar, double vpar, int derivs, bool u_↔_from_right=true, bool v_from_right=true, double resolution=1.0e-12) const	605
29.63.3.28 read(std::istream &is)	606

29.63.3.29	<code>reverseParameterDirection(bool direction_is_u)</code>	606
29.63.3.30	<code>setParameterDomain(double u1, double u2, double v1, double v2)</code>	606
29.63.3.31	<code>subSurfaces(double from_upar, double from_vpar, double to_upar, double to_vpar, double fuzzy=DEFAULT_PARAMETER_EPSILON) const</code>	607
29.63.3.32	<code>swapParameterDirection()</code>	607
29.63.3.33	<code>tangentCone(bool padir_is_u) const</code>	607
29.63.3.34	<code>turnOrientation()</code>	608
29.63.3.35	<code>write(std::ostream &os) const</code>	608
29.64	<code>NEWMAT::ConcatenatedMatrix</code> Class Reference	608
29.64.1	Detailed Description	610
29.64.2	Constructor & Destructor Documentation	611
29.64.2.1	<code>ConcatenatedMatrix(const BaseMatrix *bm1x, const BaseMatrix *bm2x)</code>	611
29.64.2.2	<code>~ConcatenatedMatrix()</code>	611
29.64.3	Member Function Documentation	611
29.64.3.1	<code>BandWidth() const</code>	611
29.64.3.2	<code>Evaluate(MatrixType mt=MatrixTypeUnSp)</code>	611
29.64.4	Friends And Related Function Documentation	611
29.64.4.1	<code>BaseMatrix</code>	611
29.64.4.2	<code>GeneralMatrix</code>	611
29.64.4.3	<code>GenericMatrix</code>	611
29.65	<code>Go::ConcreteCreator< T ></code> Class Template Reference	612
29.65.1	Detailed Description	612
29.65.2	Constructor & Destructor Documentation	613
29.65.2.1	<code>ConcreteCreator()</code>	613
29.65.3	Member Function Documentation	613
29.65.3.1	<code>create()</code>	613
29.66	<code>Go::Cone</code> Class Reference	613
29.66.1	Detailed Description	616
29.66.2	Constructor & Destructor Documentation	616
29.66.2.1	<code>Cone()</code>	616

29.66.2.2 Cone(double radius, Point location, Point z_axis, Point x_axis, double cone_angle, bool isSwapped=false)	616
29.66.2.3 ~Cone()	616
29.66.3 Member Function Documentation	616
29.66.3.1 allBoundaryLoops(double degenerate_epsilon=DEFAULT_SPACE_EPSILON) const	616
29.66.3.2 boundingBox() const	617
29.66.3.3 classType()	617
29.66.3.4 clone() const	617
29.66.3.5 closestBoundaryPoint(const Point &pt, double &clo_u, double &clo_v, Point &clo_pt, double &clo_dist, double epsilon, const RectDomain *rd=NULL, double *seed=0) const	617
29.66.3.6 closestPoint(const Point &pt, double &clo_u, double &clo_v, Point &clo_pt, double &clo_dist, double epsilon, const RectDomain *domain_of_interest=NULL, double *seed=0) const	617
29.66.3.7 constParamCurve(double parameter, bool paddir_is_u, double from, double to) const	618
29.66.3.8 constParamCurves(double parameter, bool paddir_is_u) const	618
29.66.3.9 createSplineSurface() const	618
29.66.3.10 dimension() const	618
29.66.3.11 geometrySurface() const	618
29.66.3.12 getBoundaryInfo(Point &pt1, Point &pt2, double epsilon, SplineCurve *&cv, SplineCurve *&crosscv, double knot_tol=1e-05) const	618
29.66.3.13 getCircle(double par) const	619
29.66.3.14 getConeAngle() const	619
29.66.3.15 getCoordinateAxes(Point &x_axis, Point &y_axis, Point &z_axis) const	619
29.66.3.16 getDegenerateCorners(std::vector< Point > °_corners, double tol) const	619
29.66.3.17 getElementaryParamCurve(ElementaryCurve *space_crv, double tol, const Point *start_par_pt=NULL, const Point *end_par_pt=NULL) const	619
29.66.3.18 getLine(double upar) const	620
29.66.3.19 getLocation() const	620
29.66.3.20 getRadius() const	620
29.66.3.21 instanceType() const	620
29.66.3.22 sAxisRotational(Point ¢re, Point &axis, Point &vec, double &angle)	620

29.66.3.23	<code>isBounded()</code> const	620
29.66.3.24	<code>isClosed(bool &closed_dir_u, bool &closed_dir_v)</code> const	620
29.66.3.25	<code>isDegenerate(bool &b, bool &r, bool &t, bool &l, double tolerance)</code> const	620
29.66.3.26	<code>nextSegmentVal(int dir, double par, bool forward, double tol)</code> const	621
29.66.3.27	<code>normal(Point &n, double upar, double vpar)</code> const	621
29.66.3.28	<code>normalCone()</code> const	622
29.66.3.29	<code>parameterDomain()</code> const	622
29.66.3.30	<code>point(Point &pt, double upar, double vpar)</code> const	622
29.66.3.31	<code>point(std::vector< Point > &pts, double upar, double vpar, int derivs, bool u_↔ from_right=true, bool v_from_right=true, double resolution=1.0e-12)</code> const	622
29.66.3.32	<code>read(std::istream &is)</code>	623
29.66.3.33	<code>setCoordinateAxes()</code>	623
29.66.3.34	<code>setParamBoundsU(double from_upar, double to_upar)</code>	623
29.66.3.35	<code>setParamBoundsV(double from_vpar, double to_vpar)</code>	623
29.66.3.36	<code>setParameterBounds(double from_upar, double from_vpar, double to_upar, dou- ble to_vpar)</code>	623
29.66.3.37	<code>subSurface(double from_upar, double from_vpar, double to_upar, double to_vpar, double fuzzy=DEFAULT_PARAMETER_EPSILON)</code> const	623
29.66.3.38	<code>subSurfaces(double from_upar, double from_vpar, double to_upar, double to_↔ vpar, double fuzzy=DEFAULT_PARAMETER_EPSILON)</code> const	623
29.66.3.39	<code>tangentCone(bool padir_is_u)</code> const	624
29.66.3.40	<code>write(std::ostream &os)</code> const	624
29.66.4	Member Data Documentation	624
29.66.4.1	<code>cone_angle_</code>	624
29.66.4.2	<code>domain_</code>	625
29.66.4.3	<code>location_</code>	625
29.66.4.4	<code>orientedDomain_</code>	625
29.66.4.5	<code>radius_</code>	625
29.66.4.6	<code>x_axis_</code>	625
29.66.4.7	<code>y_axis_</code>	625
29.66.4.8	<code>z_axis_</code>	625
29.67	<code>Go::ConeVolume</code> Class Reference	626

29.67.1 Detailed Description	628
29.67.2 Constructor & Destructor Documentation	628
29.67.2.1 ConeVolume()	628
29.67.2.2 ConeVolume(double radius, Point location, Point z_axis, Point x_axis, double cone_angle)	628
29.67.2.3 ~ConeVolume()	628
29.67.3 Member Function Documentation	629
29.67.3.1 boundingBox() const	629
29.67.3.2 classType()	629
29.67.3.3 clone() const	629
29.67.3.4 closestPoint(const Point &pt, double &clo_u, double &clo_v, double &clo_w, Point &clo_pt, double &clo_dist, double epsilon, double *seed=0) const	629
29.67.3.5 dimension() const	629
29.67.3.6 geometryVolume() const	630
29.67.3.7 getAllBoundarySurfaces() const	630
29.67.3.8 instanceType() const	630
29.67.3.9 nextSegmentVal(int dir, double par, bool forward, double tol) const	630
29.67.3.10 parameterSpan() const	630
29.67.3.11 point(Point &pt, double upar, double vpar, double wpar) const	631
29.67.3.12 point(std::vector< Point > &pts, double upar, double vpar, double wpar, int derivs, bool u_from_right=true, bool v_from_right=true, bool w_from_right=true, double resolution=1.0e-12) const	631
29.67.3.13 read(std::istream &is)	631
29.67.3.14 reverseParameterDirection(int pdir)	632
29.67.3.15 setParameters(double from_par, double to_par, int pdir)	632
29.67.3.16 swapParameterDirection(int pdir1, int pdir2)	632
29.67.3.17 tangentCone(int pdir) const	632
29.67.3.18 translate(const Point &vec)	633
29.67.3.19 useCentreDegen()	633
29.67.3.20 useCornerDegen()	633
29.67.3.21 write(std::ostream &os) const	633
29.68 Go::ConnectionFunctor Class Reference	633

29.68.1 Detailed Description	634
29.68.2 Constructor & Destructor Documentation	634
29.68.2.1 ConnectionFunctor(const std::vector< shared_ptr< IntersectionPoint > > &ipoints)	634
29.68.3 Member Function Documentation	634
29.68.3.1 operator()(int a, int b)	634
29.69 ControlWord Class Reference	634
29.69.1 Detailed Description	635
29.69.2 Constructor & Destructor Documentation	635
29.69.2.1 ControlWord()	635
29.69.2.2 ControlWord(int i)	635
29.69.3 Member Function Documentation	635
29.69.3.1 operator"!() const	635
29.69.3.2 operator*(ControlWord i) const	635
29.69.3.3 operator*=(ControlWord i)	636
29.69.3.4 operator+(ControlWord i) const	636
29.69.3.5 operator+() const	636
29.69.3.6 operator+=(ControlWord i)	636
29.69.3.7 operator-(ControlWord i) const	636
29.69.3.8 operator-=(ControlWord i)	636
29.69.3.9 operator<=(ControlWord i) const	636
29.69.3.10 operator>=(ControlWord i) const	636
29.69.3.11 operator^(ControlWord i) const	636
29.69.3.12 operator~() const	636
29.69.4 Member Data Documentation	637
29.69.4.1 cw	637
29.70 NEWMAT::ConvergenceException Class Reference	637
29.70.1 Detailed Description	638
29.70.2 Constructor & Destructor Documentation	638
29.70.2.1 ConvergenceException(const GeneralMatrix &A)	638
29.70.2.2 ConvergenceException(const char *c)	638

29.70.3 Member Data Documentation	638
29.70.3.1 Select	638
29.71 Go::CoordinateSystem< Dim > Class Template Reference	638
29.71.1 Detailed Description	639
29.71.2 Member Typedef Documentation	639
29.71.2.1 Matrix	639
29.71.2.2 Vector	639
29.71.3 Constructor & Destructor Documentation	639
29.71.3.1 CoordinateSystem()	639
29.71.3.2 CoordinateSystem(const Matrix &rot, const Vector &trans)	640
29.71.4 Member Function Documentation	640
29.71.4.1 operator*(const Vector &v)	640
29.71.4.2 operator*(const CoordinateSystem &c)	640
29.71.4.3 rot()	640
29.71.4.4 rot() const	640
29.71.4.5 tr()	640
29.71.4.6 tr() const	640
29.72 Go::CPUclock Class Reference	641
29.72.1 Detailed Description	641
29.72.2 Constructor & Destructor Documentation	641
29.72.2.1 CPUclock()	641
29.72.3 Member Function Documentation	641
29.72.3.1 getInterval()	641
29.72.3.2 getTime()	641
29.72.3.3 initTime()	641
29.73 Go::Creator Class Reference	642
29.73.1 Detailed Description	642
29.73.2 Constructor & Destructor Documentation	642
29.73.2.1 ~Creator()	642
29.73.3 Member Function Documentation	642

29.73.3.1 create()=0	642
29.74Go::CrossesValue Class Reference	643
29.74.1 Detailed Description	643
29.74.2 Member Typedef Documentation	643
29.74.2.1 argument_type	643
29.74.2.2 result_type	643
29.74.3 Constructor & Destructor Documentation	643
29.74.3.1 CrossesValue(int par_dir, double value, double eps)	643
29.74.4 Member Function Documentation	643
29.74.4.1 operator()(const shared_ptr< IntersectionLink > &l) const	643
29.75Go::CrossTangentOffset Class Reference	644
29.75.1 Detailed Description	645
29.75.2 Constructor & Destructor Documentation	645
29.75.2.1 CrossTangentOffset(shared_ptr< SplineCurve > &poscurve, shared_ptr< SplineCurve > &tangcv1, shared_ptr< SplineCurve > &tangcv2, shared_ptr< SplineCurve > &blend1, shared_ptr< SplineCurve > &blend2)	645
29.75.2.2 ~CrossTangentOffset()	645
29.75.3 Member Function Documentation	645
29.75.3.1 approximationOK(double par, Point approxpos, double tol1, double tol2) const	645
29.75.3.2 dim() const	646
29.75.3.3 end() const	646
29.75.3.4 eval(double t) const	646
29.75.3.5 eval(double t, int n, Point der[]) const	646
29.75.3.6 start() const	647
29.76Go::CrossTanOffDist Class Reference	647
29.76.1 Detailed Description	648
29.76.2 Constructor & Destructor Documentation	648
29.76.2.1 CrossTanOffDist(shared_ptr< SplineCurve > &poscurve, shared_ptr< SplineCurve > &tangcv1, shared_ptr< SplineCurve > &tangcv2, shared_ptr< SplineCurve > &blend1, shared_ptr< SplineCurve > &blend2, shared_ptr< SplineCurve > &opposite1, shared_ptr< SplineCurve > &opposite2, double factor)	648
29.76.2.2 ~CrossTanOffDist()	649
29.76.3 Member Function Documentation	649

29.76.3.1 approximationOK(double par, Point approxpos, double tol1, double tol2) const	649
29.76.3.2 dim() const	649
29.76.3.3 end() const	650
29.76.3.4 eval(double t) const	650
29.76.3.5 eval(double t, int n, Point der[]) const	650
29.76.3.6 start() const	650
29.77NEWMAT::CroutMatrix Class Reference	651
29.77.1 Detailed Description	652
29.77.2 Constructor & Destructor Documentation	653
29.77.2.1 CroutMatrix(const BaseMatrix &)	653
29.77.2.2 ~CroutMatrix()	653
29.77.3 Member Function Documentation	653
29.77.3.1 CleanUp()	653
29.77.3.2 GetCol(MatrixRowCol &)	653
29.77.3.3 GetCol(MatrixColX &c)	653
29.77.3.4 GetRow(MatrixRowCol &)	653
29.77.3.5 IsEqual(const GeneralMatrix &) const	653
29.77.3.6 IsSingular() const	653
29.77.3.7 LogDeterminant() const	654
29.77.3.8 lubksb(Real *, int=0)	654
29.77.3.9 MakeSolver()	654
29.77.3.10operator=(const BaseMatrix &)	654
29.77.3.11operator=(const CroutMatrix &m)	654
29.77.3.12Solver(MatrixColX &, const MatrixColX &)	654
29.77.3.13Type() const	654
29.78Cube Class Reference	655
29.78.1 Detailed Description	655
29.79Go::CurveBoundedDomain Class Reference	655
29.79.1 Detailed Description	657
29.79.2 Constructor & Destructor Documentation	657

29.79.2.1	CurveBoundedDomain()	657
29.79.2.2	CurveBoundedDomain(std::vector< shared_ptr< CurveLoop > > loops)	657
29.79.2.3	CurveBoundedDomain(shared_ptr< CurveLoop > ccw_loop)	657
29.79.2.4	~CurveBoundedDomain()	657
29.79.3	Member Function Documentation	658
29.79.3.1	clipWithDomain(int pdir, double parval, double tolerance, shared_ptr< ParamSurface > srf, std::vector< shared_ptr< CurveOnSurface > > &trim_pieces) const	658
29.79.3.2	closestInDomain(const Array< double, 2 > &point, Array< double, 2 > &clo_pt, double tolerance) const	658
29.79.3.3	closestOnBoundary(const Array< double, 2 > &point, Array< double, 2 > &clo_bd_pt, double tolerance) const	659
29.79.3.4	containingDomain() const	659
29.79.3.5	findPcurveInsideSegments(const SplineCurve &curve, double tolerance, std::vector< double > ¶ms_start_end_interval) const	659
29.79.3.6	findPcurveInsideSegments(const SplineCurve &curve, double tolerance, std::vector< double > ¶ms_start_end_interval, std::vector< double > &boundary_params, std::vector< int > &boundary_loops, std::vector< int > &boundary_curves) const	660
29.79.3.7	getInsideIntervals(int pdir, double parval1, double parval2, double tolerance, std::vector< std::pair< double, double > > &insideInts) const	660
29.79.3.8	getInternalPoint(double &upar, double &vpar) const	660
29.79.3.9	isInDomain(const Array< double, 2 > &point, double tolerance) const	660
29.79.3.10	isInDomain2(const Array< double, 2 > &point, double tolerance) const	661
29.79.3.11	isOnBoundary(const Array< double, 2 > &point, double tolerance) const	661
29.79.3.12	isOnCorner(const Array< double, 2 > &point, double tolerance) const	662
29.79.3.13	isPositionPointInDomain(int pdir, double parval1, double parval2, double tolerance) const	662
29.80	Go::CurveLoop Class Reference	662
29.80.1	Detailed Description	663
29.80.2	Constructor & Destructor Documentation	663
29.80.2.1	CurveLoop()	663
29.80.2.2	CurveLoop(const std::vector< shared_ptr< ParamCurve > > &curves, double space_epsilon)	663
29.80.2.3	~CurveLoop()	663

29.80.3 Member Function Documentation	663
29.80.3.1 begin() const	663
29.80.3.2 begin()	664
29.80.3.3 closestParPoint(const Point &pt, int &clo_ind, double &clo_par, Point &clo_pt, double &clo_dist) const	664
29.80.3.4 closestPoint(const Point &pt, int &clo_ind, double &clo_par, Point &clo_pt, double &clo_dist) const	664
29.80.3.5 end() const	664
29.80.3.6 end()	665
29.80.3.7 fixInvalidLoop(double &max_gap)	665
29.80.3.8 getCorners() const	665
29.80.3.9 getCurves()	665
29.80.3.10 getMaxCurveDist() const	665
29.80.3.11 getSmoothCurves(std::vector< std::vector< shared_ptr< ParamCurve > > &curves, double angtol)	665
29.80.3.12 getSpaceEpsilon() const	665
29.80.3.13 isValid() const	665
29.80.3.14 operator[](int index) const	665
29.80.3.15 setCurves(const std::vector< shared_ptr< ParamCurve > > &curves)	666
29.80.3.16 setSpaceEpsilon(const double space_epsilon)	666
29.80.3.17 simplify(double tol, double ang_tol, double &max_dist)	666
29.80.3.18 size() const	666
29.80.3.19 swap(CurveLoop &other)	666
29.80.3.20 turnOrientation()	667
29.81 Go::CurveModel Class Reference	667
29.81.1 Detailed Description	668
29.81.2 Constructor & Destructor Documentation	668
29.81.2.1 CurveModel(double gap, double neighbour, double kink, double bend, std::vector< shared_ptr< ParamCurve > > &curves)	668
29.81.2.2 ~CurveModel()	669
29.81.3 Member Function Documentation	669
29.81.3.1 boundingBox()	669

29.81.3.2 boundingBox(int idx) const	669
29.81.3.3 clone() const	669
29.81.3.4 closestPoint(Point &pnt, Point &clo_pnt, int &idx, double clo_par[], double &dist)	669
29.81.3.5 curvature(int idx, double *par) const	670
29.81.3.6 evaluate(int idx, double par[], Point &pnt) const	670
29.81.3.7 evaluate(int idx, double par[], int nder, std::vector< Point > &der) const	671
29.81.3.8 extremalPoint(Point &dir, Point &ext_pnt, int &idx, double ext_par[])	671
29.81.3.9 fetchCompositeCurves() const	671
29.81.3.10 getCurve(int idx) const	671
29.81.3.11 getIndex(ParamCurve *curve) const	672
29.81.3.12 intersect(const ftLine &line)	672
29.81.3.13 intersect_plane(const ftPlane &plane)	672
29.81.3.14 isDegenerate(int idx) const	673
29.81.3.15 nmbEntities() const	673
29.81.3.16 tessellate(std::vector< shared_ptr< GeneralMesh > > &meshes) const	673
29.81.3.17 tessellate(int resolution[], std::vector< shared_ptr< GeneralMesh > > &meshes) const	673
29.81.3.18 tessellate(double density, std::vector< shared_ptr< GeneralMesh > > &meshes) const	674
29.81.3.19 tessellatedCtrPolygon(std::vector< shared_ptr< LineCloud > > &ctr_pol) const	674
29.81.3.20 turn(int idx)	674
29.81.3.21 turn()	675
29.82 Go::CurveOnSurface Class Reference	675
29.82.1 Detailed Description	678
29.82.2 Constructor & Destructor Documentation	678
29.82.2.1 CurveOnSurface()	678
29.82.2.2 CurveOnSurface(shared_ptr< ParamSurface > surf, shared_ptr< ParamCurve > curve, bool preferparameter)	678
29.82.2.3 CurveOnSurface(shared_ptr< ParamSurface > surf, shared_ptr< ParamCurve > curve, int constdir, double constpar, int boundary)	679
29.82.2.4 CurveOnSurface(shared_ptr< ParamSurface > surf, int constdir, double const- par, double par1, double par2, int boundary)	679

29.82.2.5	CurveOnSurface(shared_ptr< ParamSurface > surf, shared_ptr< ParamCurve > pcurve, shared_ptr< ParamCurve > spacecurve, bool preferparameter, int ccm, int constdir, double constpar, int boundary, bool same_orientation)	679
29.82.2.6	CurveOnSurface(shared_ptr< ParamSurface > surf, shared_ptr< ParamCurve > pcurve, shared_ptr< ParamCurve > spacecurve, bool preferparameter, int ccm=0)	679
29.82.2.7	CurveOnSurface(const CurveOnSurface &surface_curve)	680
29.82.2.8	~CurveOnSurface()	680
29.82.3	Member Function Documentation	680
29.82.3.1	appendCurve(ParamCurve *cv, bool reparam=true)	680
29.82.3.2	appendCurve(ParamCurve *cv, int continuity, double &dist, bool reparam=true)	680
29.82.3.3	boundingBox() const	680
29.82.3.4	classType()	681
29.82.3.5	clone() const	681
29.82.3.6	closestPoint(const Point &pt, double tmin, double tmax, double &clo_t, Point &clo_pt, double &clo_dist, double const *seed=0) const	681
29.82.3.7	containingDomain() const	681
29.82.3.8	curveCreationMethod() const	681
29.82.3.9	dimension() const	682
29.82.3.10	directionCone() const	682
29.82.3.11	enableSameOrientation()	682
29.82.3.12	endparam() const	682
29.82.3.13	ensureParCrvExistence(double epsgeo, const RectDomain *domain_of_interest=NULL, const Point *start_par_pt=NULL, const Point *end_par_pt=NULL)	682
29.82.3.14	ensureSpaceCrvExistence(double tol)	682
29.82.3.15	faceParameter(double crv_par, const RectDomain *domain_of_interest=NULL) const	682
29.82.3.16	geometryCurve()	683
29.82.3.17	getConstantCurveInfo(int &constdir, double &constval, int &bd, bool &same_orientation) const	683
29.82.3.18	getCurveTypeInfo()	683
29.82.3.19	instanceType() const	683
29.82.3.20	sAxisRotational(Point ¢re, Point &axis, Point &vec, double &angle)	683

29.82.3.21	isConstantCurve() const	683
29.82.3.22	isConstantCurve(double tol, int &parDir, double &parVal) const	683
29.82.3.23	isDegenerate(double degenerate_epsilon)	683
29.82.3.24	isInPlane(const Point &loc, const Point &axis, double eps, Point &normal) const	684
29.82.3.25	isInPlane(const Point &norm, double eps, Point &pos) const	684
29.82.3.26	isLinear(Point &dir, double tol)	684
29.82.3.27	length(double tol)	684
29.82.3.28	makeCurvesConsistent(bool prefer_parameter)	685
29.82.3.29	makeParameterCurve(double tol, const Point &par1, const Point &par2)	685
29.82.3.30	maxTraceDiff(int nmb_sample=5) const	685
29.82.3.31	nextSegmentVal(double par, bool forward, double tol) const	685
29.82.3.32	operator=(const CurveOnSurface &other)	685
29.82.3.33	parameterCurve()	685
29.82.3.34	parameterCurve() const	685
29.82.3.35	parPref() const	686
29.82.3.36	point(Point &pt, double tpar) const	686
29.82.3.37	point(std::vector< Point > &pts, double tpar, int derivs, bool from_right=true) const	686
29.82.3.38	read(std::istream &is)	686
29.82.3.39	reverseParameterDirection(bool switchparam=false)	686
29.82.3.40	sameCurve(double tol, int nmb_sample=5) const	687
29.82.3.41	sameOrientation() const	687
29.82.3.42	sameParameterDomain() const	687
29.82.3.43	sameTrace(double tol, int nmb_sample=5) const	687
29.82.3.44	setCurves(shared_ptr< ParamCurve > spacecurve, shared_ptr< ParamCurve > parametercurve)	687
29.82.3.45	setCurveTypeInfo(int ccm)	687
29.82.3.46	setDomainParCrv(double umin, double umax, double vmin, double vmax, double uminprev, double umaxprev, double vminprev, double vmaxprev)	687
29.82.3.47	setParameterCurve(shared_ptr< ParamCurve > parametercurve)	688
29.82.3.48	setParameterInterval(double t1, double t2)	688
29.82.3.49	setParPref(bool prefer)	688

29.82.3.50	setSpaceCurve(shared_ptr< ParamCurve > spacecurve)	688
29.82.3.51	setUnderlyingSurface(shared_ptr< ParamSurface > surface)	688
29.82.3.52	spaceCurve()	688
29.82.3.53	spaceCurve() const	689
29.82.3.54	split(double param, double fuzzy=DEFAULT_PARAMETER_EPSILON) const	689
29.82.3.55	startparam() const	689
29.82.3.56	subCurve(double from_par, double to_par, double fuzzy=DEFAULT_PARAMETER_EPSILON) const	689
29.82.3.57	translateParameterCurve(const Point &dir)	690
29.82.3.58	translateSwapParameterCurve(const Point &dir, double sgn, int pdir)	690
29.82.3.59	underlyingSurface()	690
29.82.3.60	underlyingSurface() const	690
29.82.3.61	unsetParameterCurve()	690
29.82.3.62	unsetSpaceCurve()	690
29.82.3.63	updateCurves(double epsge)	690
29.82.3.64	updateCurves(Point vx1, Point vx2, double epsge)	691
29.82.3.65	updateIsoCurves()	691
29.82.3.66	updateIsoCurves(int constdir, double constpar, int boundary)	691
29.82.3.67	whichBoundary(double tol, bool &same_orientation) const	691
29.82.3.68	write(std::ostream &os) const	691
29.83	Go::CurveOnVolume Class Reference	691
29.83.1	Detailed Description	694
29.83.2	Constructor & Destructor Documentation	694
29.83.2.1	CurveOnVolume()	694
29.83.2.2	CurveOnVolume(shared_ptr< ParamVolume > vol, shared_ptr< ParamCurve > curve, bool preferparameter)	694
29.83.2.3	CurveOnVolume(shared_ptr< ParamVolume > vol, shared_ptr< ParamCurve > pcurve, shared_ptr< ParamCurve > spacecurve, bool preferparameter)	694
29.83.2.4	CurveOnVolume(const CurveOnVolume &volume_curve)	694
29.83.2.5	~CurveOnVolume()	695
29.83.3	Member Function Documentation	695
29.83.3.1	appendCurve(ParamCurve *cv, bool reparam=true)	695

29.83.3.2	<code>appendCurve(ParamCurve *cv, int continuity, double &dist, bool reparam=true)</code>	695
29.83.3.3	<code>boundingBox() const</code>	695
29.83.3.4	<code>classType()</code>	696
29.83.3.5	<code>clone() const</code>	696
29.83.3.6	<code>closestPoint(const Point &pt, double tmin, double tmax, double &clo_t, Point &clo_pt, double &clo_dist, double const *seed=0) const</code>	696
29.83.3.7	<code>containingDomain() const</code>	696
29.83.3.8	<code>dimension() const</code>	696
29.83.3.9	<code>directionCone() const</code>	697
29.83.3.10	<code>endparam() const</code>	697
29.83.3.11	<code>geometryCurve()</code>	697
29.83.3.12	<code>instanceType() const</code>	697
29.83.3.13	<code>sDegenerate(double degenerate_epsilon)</code>	697
29.83.3.14	<code>length(double tol)</code>	698
29.83.3.15	<code>nextSegmentVal(double par, bool forward, double tol) const</code>	698
29.83.3.16	<code>operator=(const CurveOnVolume &other)</code>	698
29.83.3.17	<code>parameterCurve()</code>	698
29.83.3.18	<code>parameterCurve() const</code>	699
29.83.3.19	<code>parPref() const</code>	699
29.83.3.20	<code>point(Point &pt, double tpar) const</code>	699
29.83.3.21	<code>point(std::vector< Point > &pts, double tpar, int derivs, bool from_right=true) const</code>	699
29.83.3.22	<code>read(std::istream &is)</code>	699
29.83.3.23	<code>reverseParameterDirection(bool switchparam=false)</code>	700
29.83.3.24	<code>setCurves(shared_ptr< ParamCurve > spacecurve, shared_ptr< ParamCurve > parametercurve)</code>	700
29.83.3.25	<code>setParameterCurve(shared_ptr< ParamCurve > parametercurve)</code>	700
29.83.3.26	<code>setParameterInterval(double t1, double t2)</code>	700
29.83.3.27	<code>setParPref(bool prefer)</code>	700
29.83.3.28	<code>setSpaceCurve(shared_ptr< ParamCurve > spacecurve)</code>	700
29.83.3.29	<code>setUnderlyingVolume(shared_ptr< ParamVolume > volume)</code>	700
29.83.3.30	<code>spaceCurve()</code>	701

29.83.3.31	spaceCurve() const	701
29.83.3.32	split(double param, double fuzzy=DEFAULT_PARAMETER_EPSILON) const	701
29.83.3.33	startparam() const	701
29.83.3.34	subCurve(double from_par, double to_par, double fuzzy=DEFAULT_PARAMETER_EPSILON) const	701
29.83.3.35	underlyingVolume()	702
29.83.3.36	underlyingVolume() const	702
29.83.3.37	unsetParameterCurve()	702
29.83.3.38	unsetSpaceCurve()	702
29.83.3.39	volumeParameter(double crv_par, const RectDomain *domain_of_interest=NULL) const	702
29.83.3.40	write(std::ostream &os) const	702
29.83.4	Member Data Documentation	703
29.83.4.1	pcurve_	703
29.83.4.2	prefer_parameter_	703
29.83.4.3	spacecurve_	703
29.83.4.4	volume_	703
29.84	CurveResolutionSheet Class Reference	704
29.84.1	Detailed Description	705
29.84.2	Constructor & Destructor Documentation	705
29.84.2.1	CurveResolutionSheet(int res=500)	705
29.84.3	Member Function Documentation	705
29.84.3.1	createSheet(QWidget *parent, gvObserver *obs)	705
29.84.3.2	ok	705
29.84.3.3	return_value	705
29.85	Go::CurveTesselator Class Reference	705
29.85.1	Detailed Description	706
29.85.2	Constructor & Destructor Documentation	706
29.85.2.1	CurveTesselator(const ParamCurve &curve)	706
29.85.2.2	~CurveTesselator()	706
29.85.3	Member Function Documentation	707

29.85.3.1	changeRes(int n)	707
29.85.3.2	getMesh()	707
29.85.3.3	getRes(int &n)	707
29.85.3.4	tessellate()	707
29.86Go::CvCvIntersector	Class Reference	707
29.86.1	Detailed Description	708
29.86.2	Constructor & Destructor Documentation	708
29.86.2.1	CvCvIntersector(shared_ptr< ParamGeomInt > curve1, shared_ptr< ParamGeomInt > curve2, double epsge, Intersector *prev=0, int eliminated_parameter=-1, double eliminated_value=0)	708
29.86.2.2	CvCvIntersector(shared_ptr< ParamGeomInt > curve1, shared_ptr< ParamGeomInt > curve2, shared_ptr< GeoTol > epsge, Intersector *prev=0, int eliminated_parameter=-1, double eliminated_value=0)	709
29.86.2.3	~CvCvIntersector()	709
29.86.3	Member Function Documentation	709
29.86.3.1	checkCoincidence()	709
29.86.3.2	doSubdivide()	709
29.86.3.3	foundIntersectionNearBoundary()	709
29.86.3.4	linearCase()	710
29.86.3.5	lowerOrderIntersector(shared_ptr< ParamGeomInt > obj1, shared_ptr< ParamGeomInt > obj2, Intersector *prev=0, int eliminated_parameter=-1, double eliminated_value=0)	710
29.86.3.6	microCase()	710
29.86.3.7	numParams() const	710
29.86.3.8	performRotatedBoxTest(double eps1, double eps2)	710
29.86.3.9	repairIntersections()	710
29.86.3.10	simpleCase2(Point &axis1, Point &axis2)	710
29.86.3.11	updateIntersections()	710
29.86.3.12	writeOut()	711
29.87Go::CvPtIntersector	Class Reference	711
29.87.1	Detailed Description	712
29.87.2	Constructor & Destructor Documentation	712

29.87.2.1	<code>CvPtIntersector(shared_ptr< ParamGeomInt > obj1, shared_ptr< ParamGeomInt > obj2, shared_ptr< GeoTol > epsge, Intersector *prev=0, int eliminated_parameter=-1, double eliminated_value=0)</code>	712
29.87.2.2	<code>CvPtIntersector(shared_ptr< ParamGeomInt > obj1, shared_ptr< ParamGeomInt > obj2, double epsge, Intersector *prev=0, int eliminated_parameter=-1, double eliminated_value=0)</code>	712
29.87.2.3	<code>~CvPtIntersector()</code>	713
29.87.3	Member Function Documentation	713
29.87.3.1	<code>checkCoincidence()</code>	713
29.87.3.2	<code>doSubdivide()</code>	713
29.87.3.3	<code>linearCase()</code>	713
29.87.3.4	<code>lowerOrderIntersector(shared_ptr< ParamGeomInt > obj1, shared_ptr< ParamGeomInt > obj2, Intersector *prev=0, int eliminated_parameter=-1, double eliminated_value=0)</code>	713
29.87.3.5	<code>microCase()</code>	713
29.87.3.6	<code>numParams() const</code>	713
29.87.3.7	<code>repairIntersections()</code>	714
29.87.3.8	<code>updateIntersections()</code>	714
29.88Go::cvSetConstraint	Struct Reference	714
29.88.1	Detailed Description	715
29.88.2	Constructor & Destructor Documentation	715
29.88.2.1	<code>cvSetConstraint(int cv1_id, double cv1_par, int cv1_der, int cv2_id, double cv2_par, int cv2_der, bool opp)</code>	715
29.88.3	Member Data Documentation	715
29.88.3.1	<code>cv1_der_</code>	715
29.88.3.2	<code>cv1_id_</code>	715
29.88.3.3	<code>cv1_par_</code>	715
29.88.3.4	<code>cv2_der_</code>	715
29.88.3.5	<code>cv2_id_</code>	715
29.88.3.6	<code>cv2_par_</code>	716
29.88.3.7	<code>opp_</code>	716
29.89Go::Cylinder	Class Reference	716
29.89.1	Detailed Description	719

29.89.2 Constructor & Destructor Documentation	719
29.89.2.1 Cylinder()	719
29.89.2.2 Cylinder(double radius, Point location, Point z_axis, Point x_axis, bool is← Swapped=false)	719
29.89.2.3 ~Cylinder()	719
29.89.3 Member Function Documentation	719
29.89.3.1 allBoundaryLoops(double degenerate_epsilon=DEFAULT_SPACE_EPSILON) const	719
29.89.3.2 boundingBox() const	720
29.89.3.3 classType()	720
29.89.3.4 clone() const	720
29.89.3.5 closestBoundaryPoint(const Point &pt, double &clo_u, double &clo_v, Point &clo_pt, double &clo_dist, double epsilon, const RectDomain *rd=NULL, double *seed=0) const	720
29.89.3.6 closestPoint(const Point &pt, double &clo_u, double &clo_v, Point &clo_pt, double &clo_dist, double epsilon, const RectDomain *domain_of_interest=NULL, double *seed=0) const	720
29.89.3.7 constParamCurve(double parameter, bool paddir_is_u, double from, double to) const	721
29.89.3.8 constParamCurves(double parameter, bool paddir_is_u) const	721
29.89.3.9 createSplineSurface() const	721
29.89.3.10 dimension() const	721
29.89.3.11 geometrySurface() const	722
29.89.3.12 getBoundaryInfo(Point &pt1, Point &pt2, double epsilon, SplineCurve *&cv, SplineCurve *&crosscv, double knot_tol=1e-05) const	722
29.89.3.13 getCircle(double par) const	722
29.89.3.14 getCoordinateAxes(Point &x_axis, Point &y_axis, Point &z_axis) const	722
29.89.3.15 getDegenerateCorners(std::vector< Point > °_corners, double tol) const	723
29.89.3.16 getElementaryParamCurve(ElementaryCurve *space_crv, double tol, const Point *start_par_pt=NULL, const Point *end_par_pt=NULL) const	723
29.89.3.17 getLocation() const	723
29.89.3.18 getRadius() const	723
29.89.3.19 instanceType() const	723
29.89.3.20 sAxisRotational(Point ¢re, Point &axis, Point &vec, double &angle)	723

29.89.3.21	isBounded() const	723
29.89.3.22	isClosed(bool &closed_dir_u, bool &closed_dir_v) const	724
29.89.3.23	isDegenerate(bool &b, bool &r, bool &t, bool &l, double tolerance) const	724
29.89.3.24	isLinear(Point &dir1, Point &dir2, double tol)	724
29.89.3.25	nextSegmentVal(int dir, double par, bool forward, double tol) const	724
29.89.3.26	normal(Point &n, double upar, double vpar) const	725
29.89.3.27	normalCone() const	725
29.89.3.28	parameterDomain() const	725
29.89.3.29	point(Point &pt, double upar, double vpar) const	725
29.89.3.30	point(std::vector< Point > &pts, double upar, double vpar, int derivs, bool u_↔ from_right=true, bool v_from_right=true, double resolution=1.0e-12) const	726
29.89.3.31	read(std::istream &is)	726
29.89.3.32	rotate(double rot_ang_rad)	726
29.89.3.33	setCoordinateAxes()	726
29.89.3.34	setParamBoundsU(double from_upar, double to_upar)	726
29.89.3.35	setParamBoundsV(double from_vpar, double to_vpar)	727
29.89.3.36	setParameterBounds(double from_upar, double from_vpar, double to_upar, dou- ble to_vpar)	727
29.89.3.37	subSurface(double from_upar, double from_vpar, double to_upar, double to_vpar, double fuzzy=DEFAULT_PARAMETER_EPSILON) const	727
29.89.3.38	subSurfaces(double from_upar, double from_vpar, double to_upar, double to_↔ vpar, double fuzzy=DEFAULT_PARAMETER_EPSILON) const	727
29.89.3.39	tangentCone(bool padir_is_u) const	727
29.89.3.40	write(std::ostream &os) const	728
29.89.4	Member Data Documentation	728
29.89.4.1	domain_	728
29.89.4.2	location_	728
29.89.4.3	orientedDomain_	728
29.89.4.4	radius_	728
29.89.4.5	x_axis_	728
29.89.4.6	y_axis_	728
29.89.4.7	z_axis_	729

29.90Go::CylinderInt Class Reference	729
29.90.1 Detailed Description	730
29.90.2 Constructor & Destructor Documentation	730
29.90.2.1 CylinderInt()	730
29.90.2.2 CylinderInt(Point ax_pt, Point ax_dir, double radius)	730
29.90.2.3 ~CylinderInt()	730
29.90.3 Member Function Documentation	730
29.90.3.1 ax_dir() const	730
29.90.3.2 ax_pt() const	731
29.90.3.3 radius() const	731
29.90.3.4 read(std::istream &is)	731
29.90.3.5 surface(Point bottom_pos, double height) const	731
29.91Go::CylinderVolume Class Reference	732
29.91.1 Detailed Description	734
29.91.2 Constructor & Destructor Documentation	734
29.91.2.1 CylinderVolume()	734
29.91.2.2 CylinderVolume(Point centre, double radius, Point normal, Point x_axis)	734
29.91.2.3 ~CylinderVolume()	734
29.91.3 Member Function Documentation	735
29.91.3.1 boundingBox() const	735
29.91.3.2 classType()	735
29.91.3.3 clone() const	735
29.91.3.4 closestPoint(const Point &pt, double &clo_u, double &clo_v, double &clo_w, Point &clo_pt, double &clo_dist, double epsilon, double *seed=0) const	735
29.91.3.5 dimension() const	735
29.91.3.6 geometryVolume() const	736
29.91.3.7 getAllBoundarySurfaces() const	736
29.91.3.8 instanceType() const	736
29.91.3.9 nextSegmentVal(int dir, double par, bool forward, double tol) const	736
29.91.3.10parameterSpan() const	736
29.91.3.11point(Point &pt, double upar, double vpar, double wpar) const	737

29.91.3.12	<code>point(std::vector< Point > &pts, double upar, double vpar, double wpar, int derivs, bool u_from_right=true, bool v_from_right=true, bool w_from_right=true, double resolution=1.0e-12) const</code>	737
29.91.3.13	<code>read(std::istream &is)</code>	737
29.91.3.14	<code>reverseParameterDirection(int paddir)</code>	738
29.91.3.15	<code>setParameters(double from_par, double to_par, int paddir)</code>	738
29.91.3.16	<code>swapParameterDirection(int paddir1, int paddir2)</code>	738
29.91.3.17	<code>tangentCone(int paddir) const</code>	738
29.91.3.18	<code>translate(const Point &vec)</code>	739
29.91.3.19	<code>useCentreDegen()</code>	739
29.91.3.20	<code>useCornerDegen()</code>	739
29.91.3.21	<code>write(std::ostream &os) const</code>	739
29.92	<code>hed::Dart Class Reference</code>	739
29.92.1	Detailed Description	740
29.92.2	Constructor & Destructor Documentation	740
29.92.2.1	<code>Dart()</code>	740
29.92.2.2	<code>Dart(Edge *edge, bool dir=true)</code>	740
29.92.2.3	<code>Dart(const Dart &dart)</code>	741
29.92.2.4	<code>~Dart()</code>	741
29.92.3	Member Function Documentation	741
29.92.3.1	<code>alpha0()</code>	741
29.92.3.2	<code>alpha1()</code>	741
29.92.3.3	<code>alpha2()</code>	741
29.92.3.4	<code>getEdge() const</code>	741
29.92.3.5	<code>getNode() const</code>	741
29.92.3.6	<code>getOppositeNode() const</code>	741
29.92.3.7	<code>init(Edge *edge, bool dir=true)</code>	742
29.92.3.8	<code>isCounterClockWise() const</code>	742
29.92.3.9	<code>operator!=(const Dart &dart) const</code>	742
29.92.3.10	<code>operator=(const Dart &dart)</code>	742
29.92.3.11	<code>operator==(const Dart &dart) const</code>	742

29.92.3.12x() const	742
29.92.3.13y() const	742
29.93hetriang::Dart Class Reference	742
29.93.1 Detailed Description	743
29.93.2 Constructor & Destructor Documentation	744
29.93.2.1 Dart()	744
29.93.2.2 Dart(Edge *edge, bool dir=true)	744
29.93.2.3 Dart(const Dart &dart)	744
29.93.2.4 ~Dart()	744
29.93.3 Member Function Documentation	744
29.93.3.1 alpha0()	744
29.93.3.2 alpha1()	744
29.93.3.3 alpha2()	744
29.93.3.4 controllConstDart(Dart &dart)	745
29.93.3.5 getEdge() const	745
29.93.3.6 getNode() const	745
29.93.3.7 getOppositeNode() const	745
29.93.3.8 init(Edge *edge, bool dir=true)	745
29.93.3.9 isCounterClockWise() const	745
29.93.3.10makeCopy()	745
29.93.3.11operator!=(const Dart &dart) const	745
29.93.3.12operator=(const Dart &dart)	745
29.93.3.13operator==(const Dart &dart) const	746
29.93.3.14printDart()	746
29.93.3.15x() const	746
29.93.3.16y() const	746
29.94DataHandler Class Reference	746
29.94.1 Detailed Description	747
29.94.2 Constructor & Destructor Documentation	747
29.94.2.1 DataHandler()	747

29.94.2.2 <code>~DataHandler()</code>	748
29.94.3 Member Function Documentation	748
29.94.3.1 <code>clear()</code>	748
29.94.3.2 <code>create(shared_ptr< Go::GeomObject > obj, const gvColor &col, int id)=0</code>	748
29.94.3.3 <code>paintable()</code>	748
29.94.3.4 <code>propertySheet()</code>	748
29.94.3.5 <code>tesselator()</code>	748
29.94.4 Member Data Documentation	748
29.94.4.1 <code>paintable_</code>	748
29.94.4.2 <code>property_sheet_</code>	748
29.94.4.3 <code>tesselator_</code>	748
29.95DataHandlerVolAndLR Class Reference	749
29.95.1 Detailed Description	749
29.95.2 Constructor & Destructor Documentation	750
29.95.2.1 <code>DataHandlerVolAndLR()</code>	750
29.95.2.2 <code>~DataHandlerVolAndLR()</code>	750
29.95.3 Member Function Documentation	750
29.95.3.1 <code>create(shared_ptr< Go::GeomObject > obj, const gvColor &col, int id)</code>	750
29.96DefaultDataHandler Class Reference	750
29.96.1 Detailed Description	751
29.96.2 Constructor & Destructor Documentation	751
29.96.2.1 <code>DefaultDataHandler()</code>	751
29.96.2.2 <code>~DefaultDataHandler()</code>	751
29.96.3 Member Function Documentation	751
29.96.3.1 <code>create(shared_ptr< Go::GeomObject > obj, const gvColor &col, int id)</code>	751
29.97Go::ParamSurface::degenerate_info Struct Reference	752
29.97.1 Detailed Description	752
29.97.2 Constructor & Destructor Documentation	753
29.97.2.1 <code>degenerate_info()</code>	753
29.97.3 Member Data Documentation	753

29.97.3.1 b_	753
29.97.3.2 is_set_	753
29.97.3.3 l_	753
29.97.3.4 r_	753
29.97.3.5 t_	753
29.97.3.6 tol_	753
29.98Go::DegeneratedIntersectionCurve Class Reference	754
29.98.1 Detailed Description	755
29.98.2 Constructor & Destructor Documentation	755
29.98.2.1 ~DegeneratedIntersectionCurve()	755
29.98.3 Member Function Documentation	755
29.98.3.1 evaluateAt(double pval, Point &pos, Point &tan)	755
29.98.3.2 getCurve() const	755
29.98.3.3 getParamCurve(int obj_nmb) const	756
29.98.3.4 getParamSpan(double &start, double &end) const	756
29.98.3.5 isDegenerated() const	756
29.98.3.6 isIsocurve() const	756
29.98.3.7 refine(const double &pos_tol, const double &angle_tol)	757
29.98.4 Friends And Related Function Documentation	757
29.98.4.1 constructIntersectionCurve	757
29.99NEWMAT::DiagedMatrix Class Reference	757
29.99.1 Detailed Description	758
29.99.2 Constructor & Destructor Documentation	758
29.99.2.1 ~DiagedMatrix()	758
29.99.3 Member Function Documentation	759
29.99.3.1 BandWidth() const	759
29.99.3.2 Evaluate(MatrixType mt=MatrixTypeUnSp)	759
29.99.4 Friends And Related Function Documentation	759
29.99.4.1 BaseMatrix	759
29.10NEWMAT::DiagonalMatrix Class Reference	759

29.100.1	Detailed Description	761
29.100.2	Constructor & Destructor Documentation	761
29.100.2.1	DiagonalMatrix()	761
29.100.2.2	~DiagonalMatrix()	761
29.100.2.3	DiagonalMatrix(ArrayLengthSpecifier)	761
29.100.2.4	DiagonalMatrix(const BaseMatrix &)	761
29.100.2.5	DiagonalMatrix(const DiagonalMatrix &gm)	761
29.100.3	Member Function Documentation	761
29.100.3.1	BandWidth() const	761
29.100.3.2	element(int, int)	762
29.100.3.3	element(int)	762
29.100.3.4	element(int, int) const	762
29.100.3.5	element(int) const	762
29.100.3.6	GetCol(MatrixRowCol &)	762
29.100.3.7	GetCol(MatrixColX &)	762
29.100.3.8	GetRow(MatrixRowCol &)	762
29.100.3.9	LogDeterminant() const	762
29.100.3.10	MakeSolver()	762
29.100.3.11	NextCol(MatrixRowCol &)	762
29.100.3.12	NextCol(MatrixColX &)	762
29.100.3.13	NextRow(MatrixRowCol &)	763
29.100.3.14	tric() const	763
29.100.3.15	operator()(int, int)	763
29.100.3.16	operator()(int)	763
29.100.3.17	operator()(int, int) const	763
29.100.3.18	operator()(int) const	763
29.100.3.19	operator=(const BaseMatrix &)	763
29.100.3.20	operator=(Real f)	763
29.100.3.21	operator=(const DiagonalMatrix &m)	763
29.100.3.22	ReSize(int)	763

29.100.3.23	GetSize(const GeneralMatrix &A)	763
29.100.3.24	Solver(MatrixColX &, const MatrixColX &)	763
29.100.3.25	Trace() const	764
29.100.3.26	Transpose(TransposedMatrix *, MatrixType)	764
29.100.3.27	Type() const	764
29.10	Dijkstra Class Reference	764
29.101.1	Detailed Description	765
29.101.2	Constructor & Destructor Documentation	765
29.101.2.1	Dijkstra()	765
29.101.2.2	~Dijkstra()	765
29.101.3	Member Function Documentation	766
29.101.3.1	closestNeighbour(int node_idx)	766
29.101.3.2	getDistance(int node_idx)	766
29.101.3.3	getDistance(int node_idx, int neighbour)	766
29.101.3.4	initialize()	766
29.101.3.5	insertInCandidates(int node_idx)	766
29.101.3.6	isFinished(int node_idx)	766
29.101.3.7	run()	766
29.101.3.8	run(double radius)	766
29.101.3.9	run(int target)	767
29.101.3.10	setFinished(int node_idx)	767
29.101.3.11	setGraph(const GraphType *tri)	767
29.101.3.12	setSource(int node_idx, double dist=0)	767
29.101.4	Member Data Documentation	767
29.101.4.1	back_trace_	767
29.101.4.2	distances_	767
29.101.4.3	flags_	767
29.101.4.4	graph_	767
29.101.4.5	large_distance_	768
29.101.4.6	queue_	768

29.102.0	Go::DirectionCone Class Reference	768
29.102.1	Detailed Description	768
29.102.2	Constructor & Destructor Documentation	769
29.102.2.1	DirectionCone()	769
29.102.2.2	DirectionCone(const Point &pt)	769
29.102.2.3	DirectionCone(const Point ¢re, double angle)	769
29.102.2.4	~DirectionCone()	769
29.102.3	Member Function Documentation	769
29.102.3.1	addUnionWith(const Point &pt)	769
29.102.3.2	addUnionWith(const DirectionCone &cone)	769
29.102.3.3	angle() const	769
29.102.3.4	centre() const	770
29.102.3.5	containsDirection(const Point &pt, double tol=0.0) const	770
29.102.3.6	dimension() const	770
29.102.3.7	greaterThanPi() const	770
29.102.3.8	overlaps(const DirectionCone &cone) const	770
29.102.3.9	perpendicularOverlaps(const DirectionCone &cone) const	770
29.102.3.10	read(std::istream &is)	770
29.102.3.11	setFromArray(const double *start, const double *end, int dim)	770
29.102.3.12	write(std::ostream &os) const	771
29.103.0	Go::Disc Class Reference	771
29.103.1	Detailed Description	773
29.103.2	Constructor & Destructor Documentation	774
29.103.2.1	Disc()	774
29.103.2.2	Disc(Point centre, double radius, Point x_axis, Point normal, bool isSwapped=false)	774
29.103.2.3	~Disc()	774
29.103.3	Member Function Documentation	774
29.103.3.1	allBoundaryLoops(double degenerate_epsilon=DEFAULT_SPACE_EPSILON) const	774
29.103.3.2	boundingBox() const	774
29.103.3.3	classType()	775

29.103.3.4	<code>clone() const</code>	775
29.103.3.5	<code>closestBoundaryPoint(const Point &pt, double &clo_u, double &clo_v, Point &clo_pt, double &clo_dist, double epsilon, const RectDomain *rd=NULL, double *seed=0) const</code>	775
29.103.3.6	<code>closestPoint(const Point &pt, double &clo_u, double &clo_v, Point &clo_pt, double &clo_dist, double epsilon, const RectDomain *domain_of_interest=NULL, double *seed=0) const</code>	775
29.103.3.7	<code>constParamCurves(double parameter, bool pardir_is_u) const</code>	776
29.103.3.8	<code>createSplineSurface() const</code>	776
29.103.3.9	<code>dimension() const</code>	776
29.103.3.10	<code>geometrySurface() const</code>	776
29.103.3.11	<code>getBoundaryInfo(Point &pt1, Point &pt2, double epsilon, SplineCurve *&cv, SplineCurve *&crosscv, double knot_tol=1e-05) const</code>	777
29.103.3.12	<code>getDegenerateCorners(std::vector< Point > &deg_corners, double tol) const</code>	777
29.103.3.13	<code>instanceType() const</code>	777
29.103.3.14	<code>Bounded() const</code>	777
29.103.3.15	<code>Closed(bool &closed_dir_u, bool &closed_dir_v) const</code>	778
29.103.3.16	<code>Degenerate(bool &b, bool &r, bool &t, bool &l, double tolerance) const</code>	778
29.103.3.17	<code>nextSegmentVal(int dir, double par, bool forward, double tol) const</code>	778
29.103.3.18	<code>normal(Point &n, double upar, double vpar) const</code>	779
29.103.3.19	<code>normalCone() const</code>	779
29.103.3.20	<code>parameterDomain() const</code>	779
29.103.3.21	<code>point(Point &pt, double upar, double vpar) const</code>	779
29.103.3.22	<code>point(std::vector< Point > &pts, double upar, double vpar, int derivs, bool u_from_right=true, bool v_from_right=true, double resolution=1.0e-12) const</code>	780
29.103.3.23	<code>read(std::istream &is)</code>	780
29.103.3.24	<code>setParameterBounds(double from_upar, double from_vpar, double to_upar, double to_vpar)</code>	780
29.103.3.25	<code>subSurface(double from_upar, double from_vpar, double to_upar, double to_vpar, double fuzzy=DEFAULT_PARAMETER_EPSILON) const</code>	780
29.103.3.26	<code>subSurfaces(double from_upar, double from_vpar, double to_upar, double to_vpar, double fuzzy=DEFAULT_PARAMETER_EPSILON) const</code>	781
29.103.3.27	<code>tangentCone(bool pardir_is_u) const</code>	781
29.103.3.28	<code>useCentreDegen()</code>	781

29.103.3.2	UseCornerDegen()	781
29.103.3.3	Write(std::ostream &os) const	782
29.104	Go::Domain Class Reference	782
29.104.1	Detailed Description	782
29.104.2	Constructor & Destructor Documentation	783
29.104.2.1	~Domain()	783
29.104.3	Member Function Documentation	783
29.104.3.1	ClosestInDomain(const Array< double, 2 > &point, Array< double, 2 > &clo_pt, double tolerance) const =0	783
29.104.3.2	ClosestOnBoundary(const Array< double, 2 > &point, Array< double, 2 > &clo_pt, double tolerance) const =0	783
29.104.3.3	IsInDomain(const Array< double, 2 > &point, double tolerance) const =0	783
29.104.3.4	IsInDomain2(const Array< double, 2 > &point, double tolerance) const =0	784
29.104.3.5	IsOnBoundary(const Array< double, 2 > &point, double tolerance) const =0	784
29.105	RBD_COMMON::Domain_error Class Reference	785
29.105.1	Detailed Description	785
29.105.2	Constructor & Destructor Documentation	786
29.105.2.1	Domain_error(const char *a_what=0)	786
29.105.3	Member Data Documentation	786
29.105.3.1	Select	786
29.106	Go::LRSplineSurface::double_pair_hash Struct Reference	786
29.106.1	Detailed Description	786
29.106.2	Member Function Documentation	786
29.106.2.1	operator()(const std::pair< double, double > &dp) const	786
29.107	Red::Edge Class Reference	787
29.107.1	Detailed Description	788
29.107.2	Constructor & Destructor Documentation	788
29.107.2.1	Edge()	788
29.107.2.2	~Edge()	788
29.107.3	Member Function Documentation	788
29.107.3.1	getNextEdgeInFace() const	788

29.107.3.2	getSourceNode()	788
29.107.3.3	getTargetNode()	788
29.107.3.4	getTwinEdge() const	788
29.107.3.5	isConstrained() const	789
29.107.3.6	isLeadingEdge() const	789
29.107.3.7	setAsLeadingEdge(bool val=true)	789
29.107.3.8	setConstrained(bool val=true)	789
29.107.3.9	setNextEdgeInFace(Edge *edge)	789
29.107.3.10	setSourceNode(Node *node)	789
29.107.3.11	setTwinEdge(Edge *edge)	789
29.107.4	Member Data Documentation	789
29.107.4.1	isConstrained_	789
29.107.4.2	isLeadingEdge_	790
29.108	Triang::Edge Class Reference	790
29.108.1	Detailed Description	790
29.108.2	Constructor & Destructor Documentation	790
29.108.2.1	Edge()	790
29.108.2.2	~Edge()	790
29.108.3	Member Function Documentation	791
29.108.3.1	getNextEdgeInFace() const	791
29.108.3.2	getSourceNode()	791
29.108.3.3	getTargetNode()	791
29.108.3.4	getTwinEdge() const	791
29.108.3.5	isLeadingEdge() const	791
29.108.3.6	setAsLeadingEdge(bool val=true)	791
29.108.3.7	setNextEdgeInFace(Edge *edge)	791
29.108.3.8	setSourceNode(shared_ptr< Node > node)	791
29.108.3.9	setTwinEdge(Edge *edge)	791
29.109	EdgeType Class Reference	792
29.109.1	Detailed Description	792

29.109.2	Member Function Documentation	792
29.109.2.1	initEdgeType()	792
29.109.2.2	isVertex(int node)	792
29.109.3	Member Data Documentation	792
29.109.3.1	vertex_	792
29.110	Go::EdgeVertex Class Reference	792
29.110.1	Detailed Description	793
29.110.2	Constructor & Destructor Documentation	793
29.110.2.1	EdgeVertex(std::vector< ftEdge * > edges)	793
29.110.2.2	EdgeVertex(ftEdge *edge)	794
29.110.2.3	~EdgeVertex()	794
29.110.3	Member Function Documentation	794
29.110.3.1	addEdge(ftEdge *edge)	794
29.110.3.2	addEdgeVertex(EdgeVertex *other)	794
29.110.3.3	allEdges() const	794
29.110.3.4	allEdges(Body *bd)	794
29.110.3.5	averageSplineEdges(double eps)	794
29.110.3.6	checkRadialEdgeTopology()	794
29.110.3.7	disconnectTwin(ftEdge *e1, ftEdge *e2)	794
29.110.3.8	getAdjacentBodies() const	794
29.110.3.9	getAdjacentFaces() const	795
29.110.3.10	getAdjacentFaces(Body *bd) const	795
29.110.3.11	getEdge(int idx)	795
29.110.3.12	isEdge(ftEdge *edge) const	795
29.110.3.13	isEdgeSingle(ftEdge *edge) const	795
29.110.3.14	isUniqueEdges()	795
29.110.3.15	isUniqueEdges(Body *bd) const	795
29.110.3.16	organizeTwins()	795
29.110.3.17	removeEdge(ftEdge *edge)	795
29.110.3.18	Organize()	796

29.110.3.19	splitAtVertex(shared_ptr< Vertex > v1, shared_ptr< Vertex > v2, shared_ptr< Vertex > split_vx)	796
29.110.3.20	uniqueEdges()	796
29.110.3.21	uniqueEdges(Body *bd)	796
29.111	Go::Element2D Class Reference	796
29.111.1	Detailed Description	798
29.111.2	Constructor & Destructor Documentation	798
29.111.2.1	Element2D()	798
29.111.2.2	Element2D(double start_u, double start_v, double stop_u, double stop_v)	798
29.111.2.3	~Element2D()	798
29.111.3	Member Function Documentation	798
29.111.3.1	addDataPoints(std::vector< double >::iterator start, std::vector< double >::iterator end, bool sort_in_u)	798
29.111.3.2	addDataPoints(std::vector< double >::iterator start, std::vector< double >::iterator end, int del, bool sort_in_u)	798
29.111.3.3	addGhostPoints(std::vector< double >::iterator start, std::vector< double >::iterator end, bool sort_in_u)	798
29.111.3.4	addGhostPoints(std::vector< double >::iterator start, std::vector< double >::iterator end, int del, bool sort_in_u)	799
29.111.3.5	addSupportFunction(LRBSpline2D *f)	799
29.111.3.6	area() const	799
29.111.3.7	contains(double upar, double vpar)	799
29.111.3.8	copy()	799
29.111.3.9	curveOnElement(double start_u, double start_v, double end_u, double end_v) const	799
29.111.3.10	eraseDataPoints()	799
29.111.3.11	eraseDataPoints(std::vector< double >::iterator start, std::vector< double >::iterator end)	799
29.111.3.12	eraseGhostPoints()	800
29.111.3.13	getAccumulatedError()	800
29.111.3.14	getAccuracyInfo(double &average_error, double &max_error, int &nmb_outside_tol)	800
29.111.3.15	getAverageError()	800
29.111.3.16	getDataBoundingBox(double bb[])	800

29.111.3.17	getDataPoints()	800
29.111.3.18	getGhostPoints()	800
29.111.3.19	getLSMatrix(double *&LSmat, double *&LSright, int &ncond)	800
29.111.3.20	getMaxError()	800
29.111.3.21	getNmbOutsideTol()	801
29.111.3.22	getOutsideGhostPoints(std::vector< double > &points, Direction2D d, bool &sort_in_u)	801
29.111.3.23	getOutsidePoints(std::vector< double > &points, Direction2D d, bool &sort_in_u)	801
29.111.3.24	getOverloadCount() const	801
29.111.3.25	getSupport() const	801
29.111.3.26	hasAccuracyInfo()	801
29.111.3.27	hasDataPoints()	801
29.111.3.28	hasLSMatrix()	801
29.111.3.29	hasSupportFunction(LRBSpline2D *f)	801
29.111.3.30	incrementOverloadCount()	801
29.111.3.31	isModified()	802
29.111.3.32	isOverloaded() const	802
29.111.3.33	makeDataPoints3D()	802
29.111.3.34	nmbBasisFunctions() const	802
29.111.3.35	nmbDataPoints()	802
29.111.3.36	nmbGhostPoints()	802
29.111.3.37	moveSupportFunction(LRBSpline2D *f)	802
29.111.3.38	setAccuracyInfo()	802
29.111.3.39	setModificationFlag()	802
29.111.3.40	setOverloadCount()	802
29.111.3.41	setAccuracyInfo(double accumulated_error, double average_error, double max_error, int nmb_outside_tol)	802
29.111.3.42	setLSMatrix()	803
29.111.3.43	setModified()	803
29.111.3.44	setUmax(double u)	803
29.111.3.45	setUmin(double u)	803

29.111.3.46	setVmax(double v)	803
29.111.3.47	setVmin(double v)	803
29.111.3.48	split(bool split_u, double par_value)	803
29.111.3.49	sumOfScaledBsplines(double upar, double vpar)	803
29.111.3.50	supportBegin()	803
29.111.3.51	supportBegin() const	803
29.111.3.52	supportEnd()	803
29.111.3.53	supportEnd() const	804
29.111.3.54	supportFunction(int i)	804
29.111.3.55	swapParameterDirection()	804
29.111.3.56	max() const	804
29.111.3.57	min() const	804
29.111.3.58	initSquareBernsteinBasis() const	804
29.111.3.59	updateAccuracyInfo()	804
29.111.3.60	updateBasisPointers(std::vector< LRBSpline2D * > &basis)	804
29.111.3.61	updateLSDataParDomain(double u1, double u2, double v1, double v2, double u1new, double u2new, double v1new, double v2new)	804
29.111.3.62	max() const	804
29.111.3.63	min() const	804
29.112	Bo::ElementaryCurve Class Reference	805
29.112.1	Detailed Description	806
29.112.2	Constructor & Destructor Documentation	806
29.112.2.1	ElementaryCurve()	806
29.112.2.2	~ElementaryCurve()	806
29.112.3	Member Function Documentation	806
29.112.3.1	clone() const =0	806
29.112.3.2	createSplineCurve() const =0	807
29.112.3.3	getReversedParameter(double &t) const	807
29.112.3.4	isReversed() const	807
29.112.3.5	reverseParameterDirection(bool switchparam=false)	807
29.112.3.6	setParamBounds(double startpar, double endpar)=0	807

29.112.3.7	subCurve(double from_par, double to_par, double fuzzy=DEFAULT_PARAMETER_EPSILON) const =0	807
29.112.3.8	swapParameters2D()=0	808
29.112.3.9	translateCurve(const Point &dir)=0	808
29.112.4	Member Data Documentation	808
29.112.4.1	isReversed_	808
29.113	ElementarySurface Class Reference	808
29.113.1	Detailed Description	810
29.113.2	Constructor & Destructor Documentation	810
29.113.2.1	ElementarySurface()	810
29.113.2.2	~ElementarySurface()	810
29.113.3	Member Function Documentation	810
29.113.3.1	area(double tol) const	810
29.113.3.2	asSplineSurface()	811
29.113.3.3	clone() const =0	811
29.113.3.4	closestInDomain(double u, double v) const	811
29.113.3.5	containingDomain() const	811
29.113.3.6	createSplineSurface() const =0	811
29.113.3.7	geometrySurface() const =0	812
29.113.3.8	getCornerPoints(std::vector< std::pair< Point, Point > > &corners) const	812
29.113.3.9	getElementaryParamCurve(ElementaryCurve *space_crv, double tol, const Point *start_par_pt=NULL, const Point *end_par_pt=NULL) const	812
29.113.3.10	getOrientedParameters(double &u, double &v) const	812
29.113.3.11	inDomain(double u, double v, double eps=1.0e-4) const	812
29.113.3.12	inDomain2(double u, double v, double eps=1.0e-4) const	812
29.113.3.13	isBounded() const	812
29.113.3.14	isClosed(bool &closed_dir_u, bool &closed_dir_v) const	812
29.113.3.15	isSwapped() const	813
29.113.3.16	inBoundary(double u, double v, double eps=1.0e-4) const	813
29.113.3.17	outerBoundaryLoop(double degenerate_epsilon=DEFAULT_SPACE_EPSILON) const	813
29.113.3.18	reverseParameterDirection(bool direction_is_u)	813

29.113.3.1	GetParameterBounds(double from_upar, double from_vpar, double to_upar, double to_vpar)=0	813
29.113.3.2	SwapParameterDirection()	813
29.113.3.2	TurnOrientation()	814
29.113.4	Member Data Documentation	814
29.113.4.1	isSwapped_	814
29.114	Go::ElementaryVolume Class Reference	814
29.114.1	Detailed Description	815
29.114.2	Constructor & Destructor Documentation	815
29.114.2.1	~ElementaryVolume()	815
29.114.3	Member Function Documentation	816
29.114.3.1	clone() const =0	816
29.114.3.2	geometryVolume() const =0	816
29.115	Go::LRSplineSurface::ElemKey Struct Reference	816
29.115.1	Detailed Description	816
29.115.2	Member Function Documentation	817
29.115.2.1	operator<(const ElemKey &rhs) const	817
29.115.3	Member Data Documentation	817
29.115.3.1	u_min	817
29.115.3.2	v_min	817
29.116	Go::Ellipse Class Reference	817
29.116.1	Detailed Description	819
29.116.2	Constructor & Destructor Documentation	820
29.116.2.1	Ellipse()	820
29.116.2.2	Ellipse(Point centre, Point direction, Point normal, double r1, double r2, bool isReversed=false)	820
29.116.2.3	~Ellipse()	820
29.116.3	Member Function Documentation	820
29.116.3.1	appendCurve(ParamCurve *cv, bool reparam=true)	820
29.116.3.2	appendCurve(ParamCurve *cv, int continuity, double &dist, bool reparam=true)	820
29.116.3.3	boundingBox() const	821

29.116.3.4	<code>classType()</code>	821
29.116.3.5	<code>clone() const</code>	821
29.116.3.6	<code>closestPoint(const Point &pt, double tmin, double tmax, double &clo_t, Point &clo_pt, double &clo_dist, double const *seed=0) const</code>	821
29.116.3.7	<code>createSplineCurve() const</code>	821
29.116.3.8	<code>dimension() const</code>	821
29.116.3.9	<code>directionCone() const</code>	822
29.116.3.10	<code>endparam() const</code>	822
29.116.3.11	<code>geometryCurve()</code>	822
29.116.3.12	<code>getCentre() const</code>	822
29.116.3.13	<code>getNormal() const</code>	822
29.116.3.14	<code>getRadius1() const</code>	822
29.116.3.15	<code>getRadius2() const</code>	823
29.116.3.16	<code>getXAxis() const</code>	823
29.116.3.17	<code>instanceType() const</code>	823
29.116.3.18	<code>Closed() const</code>	823
29.116.3.19	<code>Degenerate(double degenerate_epsilon)</code>	823
29.116.3.20	<code>InPlane(const Point &norm, double eps, Point &pos) const</code>	823
29.116.3.21	<code>length(double tol)</code>	823
29.116.3.22	<code>point(Point &pt, double tpar) const</code>	824
29.116.3.23	<code>point(std::vector< Point > &pts, double tpar, int derivs, bool from_right=true) const</code>	824
29.116.3.24	<code>read(std::istream &is)</code>	824
29.116.3.25	<code>setParamBounds(double startpar, double endpar)</code>	824
29.116.3.26	<code>setParameterInterval(double t1, double t2)</code>	825
29.116.3.27	<code>setSpanningVectors()</code>	825
29.116.3.28	<code>startparam() const</code>	825
29.116.3.29	<code>subCurve(double from_par, double to_par, double fuzzy=DEFAULT_PARAMETER_EPSILON) const</code>	825
29.116.3.30	<code>swapParameters2D()</code>	826
29.116.3.31	<code>translateCurve(const Point &dir)</code>	826
29.116.3.32	<code>write(std::ostream &os) const</code>	826

29.116.4	Member Data Documentation	826
29.116.4.1	centre_	826
29.116.4.2	endparam_	826
29.116.4.3	normal_	826
29.116.4.4	r1_	826
29.116.4.5	r2_	826
29.116.4.6	startparam_	827
29.116.4.7	vec1_	827
29.116.4.8	vec2_	827
29.117	Go::EntityList Struct Reference	827
29.117.1	Detailed Description	827
29.117.2	Constructor & Destructor Documentation	828
29.117.2.1	EntityList()	828
29.117.3	Member Function Documentation	829
29.117.3.1	validEntity(int ent)	829
29.118	Go::Streamable::EofException Class Reference	829
29.118.1	Detailed Description	829
29.119	Go::EvalCurve Class Reference	829
29.119.1	Detailed Description	831
29.119.2	Constructor & Destructor Documentation	831
29.119.2.1	~EvalCurve()	831
29.119.3	Member Function Documentation	831
29.119.3.1	approximationOK(double par, Point approxpos, double tol1, double tol2) const =0	831
29.119.3.2	dim() const =0	831
29.119.3.3	end() const =0	832
29.119.3.4	eval(double t) const =0	832
29.119.3.5	eval(double t, int n, Point der[]) const =0	832
29.119.3.6	start() const =0	833
29.119.3.7	write(std::ostream &out) const	833
29.120	Go::EvalCurveSet Class Reference	833

29.120.1	Detailed Description	834
29.120.2	Constructor & Destructor Documentation	834
29.120.2.1	~EvalCurveSet()	834
29.120.3	Member Function Documentation	834
29.120.3.1	approximationOK(double par, const std::vector< Point > &approxpos, double tol1, double tol2)=0	834
29.120.3.2	dim()=0	834
29.120.3.3	end()=0	835
29.120.3.4	eval(double t)=0	835
29.120.3.5	eval(double t, int n, std::vector< std::vector< Point > > &der)=0	835
29.120.3.6	mbCvs()=0	835
29.120.3.7	resetErr()	836
29.120.3.8	start()=0	836
29.120	Go::EvalFunctorCurve Class Reference	836
29.121.1	Detailed Description	837
29.121.2	Constructor & Destructor Documentation	837
29.121.2.1	EvalFunctorCurve(BdCondFunctor *fbd, shared_ptr< SplineCurve > geo_curve, int dimension)	837
29.121.2.2	~EvalFunctorCurve()	837
29.121.3	Member Function Documentation	837
29.121.3.1	approximationOK(double par, Point approxpos, double tol1, double tol2) const	837
29.121.3.2	dim() const	837
29.121.3.3	end() const	838
29.121.3.4	eval(double t) const	838
29.121.3.5	eval(double t, int n, Point der[]) const	838
29.121.3.6	start() const	838
29.122	Go::EvalFunctorSurface Class Reference	839
29.122.1	Detailed Description	840
29.122.2	Constructor & Destructor Documentation	840
29.122.2.1	EvalFunctorSurface(BdCondFunctor *fbd, shared_ptr< SplineSurface > geo_↵ surface, int dimension)	840
29.122.2.2	~EvalFunctorSurface()	840

29.122.3	Member Function Documentation	840
29.122.3.1	approximationOK(double par_u, double par_v, Point approxpos, double tol1, double tol2) const	840
29.122.3.2	dim() const	840
29.122.3.3	end_u() const	841
29.122.3.4	end_v() const	841
29.122.3.5	eval(double u, double v) const	841
29.122.3.6	eval(double u, double v, int n, Point der[]) const	841
29.122.3.7	start_u() const	842
29.122.3.8	start_v() const	842
29.122.3	Go::EvalOffsetSurface Class Reference	842
29.123.1	Detailed Description	843
29.123.2	Constructor & Destructor Documentation	843
29.123.2.1	EvalOffsetSurface(shared_ptr< ftFaceBase > base_sf, double offset_dist, double epsgeo)	843
29.123.2.2	~EvalOffsetSurface()	843
29.123.3	Member Function Documentation	843
29.123.3.1	approximationOK(double par_u, double par_v, Point approxpos, double tol1, double tol2) const	843
29.123.3.2	dim() const	844
29.123.3.3	end_u() const	844
29.123.3.4	end_v() const	844
29.123.3.5	eval(double u, double v) const	844
29.123.3.6	eval(double u, double v, int n, Point der[]) const	845
29.123.3.7	getProjKinkCurves(std::vector< pair< shared_ptr< ParamCurve >, shared_ptr< ParamCurve > > &par_cvs, std::vector< pair< shared_ptr< ParamSurface >, shared_ptr< ParamSurface > > &sfs)	845
29.123.3.8	gridKinks(const HermiteGrid2D &grid, const std::vector< shared_ptr< SplineCurve > > &kink_cvs_2d, std::vector< int > &grid_kinks) const	845
29.123.3.9	gridSelfIntersections(const HermiteGrid2D &grid, std::vector< int > &grid_self_intersections, std::vector< double > &radius_of_curv) const	845
29.123.3.10	start_u() const	845
29.123.3.11	start_v() const	845

29.124	Go::EvalParamCurve Class Reference	846
29.124.1	Detailed Description	847
29.124.2	Constructor & Destructor Documentation	847
29.124.2.1	EvalParamCurve(shared_ptr< Go::ParamCurve > &crv)	847
29.124.2.2	~EvalParamCurve()	847
29.124.3	Member Function Documentation	847
29.124.3.1	approximationOK(double par, Point approxpos, double tol1, double tol2) const	847
29.124.3.2	dim() const	847
29.124.3.3	end() const	848
29.124.3.4	eval(double t) const	848
29.124.3.5	eval(double t, int n, Point der[]) const	848
29.124.3.6	start() const	848
29.124.3.7	write(std::ostream &out) const	849
29.125	Go::EvalSurface Class Reference	849
29.125.1	Detailed Description	850
29.125.2	Constructor & Destructor Documentation	850
29.125.2.1	~EvalSurface()	850
29.125.3	Member Function Documentation	850
29.125.3.1	approximationOK(double par_u, double par_v, Point approxpos, double tol1, double tol2) const =0	850
29.125.3.2	dim() const =0	850
29.125.3.3	end_u() const =0	851
29.125.3.4	end_v() const =0	851
29.125.3.5	eval(double u, double v) const =0	851
29.125.3.6	eval(double u, double v, int n, Point der[]) const =0	851
29.125.3.7	start_u() const =0	852
29.125.3.8	start_v() const =0	852
29.125.3.9	write(std::ostream &out) const	852
29.126	Go::FaceAdjacency< edgeType, faceType > Class Template Reference	852
29.126.1	Detailed Description	853
29.126.2	Constructor & Destructor Documentation	853

29.126.2.1	FaceAdjacency(double tol_gap, double tol_neighbour, double tol_kink, double tol_bend)	853
29.126.2.2	FaceAdjacency(const tpTolerances &tol)	854
29.126.2.3	~FaceAdjacency()	854
29.126.3	Member Function Documentation	854
29.126.3.1	computeAdjacency(const std::vector< shared_ptr< faceType > > &faces, int first_idx)	854
29.126.3.2	computeAdjacency(const std::vector< shared_ptr< faceType > > &faces, std::vector< std::pair< faceType *, faceType * > > &orient_inconsistent, int first_idx)	854
29.126.3.3	computeFaceAdjacency(std::vector< shared_ptr< faceType > > faces, shared_ptr< faceType > new_face)	855
29.126.3.4	computeFaceAdjacency(std::vector< faceType * > faces, faceType *new_face)	855
29.126.3.5	computeFaceAdjacency(std::vector< shared_ptr< faceType > > faces, shared_ptr< faceType > new_face, std::vector< std::pair< faceType *, faceType * > > &orient_inconsistent)	855
29.126.3.6	computeFaceAdjacency(std::vector< faceType * > faces, faceType *new_face, std::vector< std::pair< faceType *, faceType * > > &orient_inconsistent)	855
29.126.3.7	connectTwins(edgeType *e1, edgeType *e2, double min1, double max1, double min2, double max2, int &status)	856
29.126.3.8	getTolerances()	856
29.126.3.9	releaseFaceAdjacency(shared_ptr< faceType > face)	856
29.126.3.10	getConnectivity(const std::vector< shared_ptr< faceType > > &faces)	856
29.126.3.11	getConnectivity(const std::vector< faceType * > &faces)	856
29.126.3.12	getTolerances(const tpTolerances &tol)	856
29.126.3.13	updateConnectivity(edgeType *e1, edgeType *e2)	857
29.126.4	Member Data Documentation	857
29.126.4.1	new_edges_	857
29.126.4.2	tol_	857
29.127	Go::FaceConnectivity< edgeType > Struct Template Reference	857
29.127.1	Detailed Description	858
29.127.2	Constructor & Destructor Documentation	858
29.127.2.1	FaceConnectivity(edgeType *e1, edgeType *e2)	858
29.127.3	Member Function Documentation	858
29.127.3.1	BestStatus() const	858

29.127.3.2	setEdges(edgeType *e1, edgeType *e2)	859
29.127.3.3	WorstStatus() const	859
29.127.4	Member Data Documentation	859
29.127.4.1	e1_	859
29.127.4.2	e2_	859
29.127.4.3	parameters_	859
29.127.4.4	status_	859
29.128	Go::FaceConnectivityUtils< edgeType, faceType > Class Template Reference	860
29.128.1	Detailed Description	860
29.128.2	Member Function Documentation	860
29.128.2.1	BoundaryLoops(const std::vector< shared_ptr< faceType > > &faces, std::vector< std::vector< edgeType * > > &loopvec)	860
29.128.2.2	BoundaryLoops(const std::vector< shared_ptr< faceType > > &faces, std::vector< std::vector< shared_ptr< edgeType > > > &loopvec)	860
29.128.2.3	cornersAndKinks(const std::vector< shared_ptr< faceType > > &faces, std::vector< edgeType * > &vec)	861
29.128.2.4	disjointObjects(const std::vector< shared_ptr< faceType > > &faces, std::vector< std::vector< faceType * > > &grouped_faces)	861
29.128.2.5	edgesBoundingFace(faceType *face)	861
29.128.2.6	smoothEdges(const std::vector< shared_ptr< faceType > > &faces, std::vector< edgeType * > &vec)	861
29.129	Go::FaceSetQuality Class Reference	861
29.129.1	Detailed Description	864
29.129.2	Constructor & Destructor Documentation	864
29.129.2.1	FaceSetQuality(double gap, double kink, double approx)	864
29.129.2.2	FaceSetQuality(const tpTolerances &toptol, double approx)	864
29.129.2.3	FaceSetQuality(shared_ptr< SurfaceModel > sfmodel)	864
29.129.2.4	~FaceSetQuality()	864
29.129.3	Member Function Documentation	864
29.129.3.1	acuteEdgeAngle(std::vector< std::pair< ftEdge *, ftEdge * > > &edge_acute)	864
29.129.3.2	acuteFaceAngle(std::vector< std::pair< ftSurface *, ftSurface * > > &face_acute)	864
29.129.3.3	attach(shared_ptr< SurfaceModel > sfmodel)	864

29.129.3.4	<code>cvC1G1Discontinuity(std::vector< shared_ptr< ParamCurve > > &c1_discont, std::vector< shared_ptr< ParamCurve > > &g1_discont)</code>	864
29.129.3.5	<code>CurvatureRadius(std::vector< std::pair< shared_ptr< PointOnCurve >, double > > &small_curv_rad, std::pair< shared_ptr< PointOnCurve >, double > &minimum_curv_rad)</code>	864
29.129.3.6	<code>degenerateSfCorners(std::vector< shared_ptr< ftPoint > > &deg_corners)</code>	865
29.129.3.7	<code>degenSurfaces(std::vector< shared_ptr< ParamSurface > > &deg_sfs)</code>	865
29.129.3.8	<code>edgePosAndTangDiscontinuity(std::vector< std::pair< ftEdge *, ftEdge * > > &pos_disconts, std::vector< std::pair< ftEdge *, ftEdge * > > &tang_disconts)</code>	865
29.129.3.9	<code>edgeVertexDistance(std::vector< std::pair< ftEdge *, shared_ptr< Vertex > > > &edge_vertices)</code>	865
29.129.3.10	<code>faceEdgeDistance(std::vector< std::pair< ftSurface *, ftEdge * > > &face_edges)</code>	865
29.129.3.11	<code>faceNormalConsistency(std::vector< shared_ptr< ftSurface > > &inconsistent_faces)</code>	865
29.129.3.12	<code>facePositionDiscontinuity(std::vector< std::pair< ftEdge *, ftEdge * > > &pos_disconts)</code>	865
29.129.3.13	<code>faceTangentDiscontinuity(std::vector< std::pair< ftEdge *, ftEdge * > > &tangent_disconts)</code>	866
29.129.3.14	<code>faceVertexDistance(std::vector< std::pair< ftSurface *, shared_ptr< Vertex > > > &face_vertices)</code>	866
29.129.3.15	<code>getAssociatedSfModel()</code>	866
29.129.3.16	<code>identicalOrEmbeddedEdges(std::vector< std::pair< shared_ptr< ftEdge >, shared_ptr< ftEdge > > > &identical_edges, std::vector< std::pair< shared_ptr< ftEdge >, shared_ptr< ftEdge > > > &embedded_edges)</code>	866
29.129.3.17	<code>identicalOrEmbeddedFaces(std::vector< std::pair< shared_ptr< ftSurface >, shared_ptr< ftSurface > > > &identical_faces, std::vector< std::pair< shared_ptr< ftSurface >, shared_ptr< ftSurface > > > &embedded_faces)</code>	866
29.129.3.18	<code>identicalVertices(std::vector< std::pair< shared_ptr< Vertex >, shared_ptr< Vertex > > > &identical_vertices)</code>	866
29.129.3.19	<code>indistinctKnots(std::vector< shared_ptr< ParamCurve > > &cv_knots, std::vector< shared_ptr< ParamSurface > > &sf_knots, double tol=1.0e-8)</code>	866
29.129.3.20	<code>loopIntersection(std::vector< std::pair< shared_ptr< PointOnEdge >, shared_ptr< PointOnEdge > > > &loop_intersection)</code>	867
29.129.3.21	<code>loopOrientationConsistency(std::vector< shared_ptr< Loop > > &inconsistent_loops)</code>	867
29.129.3.22	<code>loopSelfIntersection(std::vector< std::pair< shared_ptr< PointOnEdge >, shared_ptr< PointOnEdge > > > &loop_self_intersection)</code>	867
29.129.3.23	<code>miniEdges(std::vector< shared_ptr< ftEdge > > &mini_edges)</code>	867
29.129.3.24	<code>miniSurfaces(std::vector< shared_ptr< ftSurface > > &mini_surfaces)</code>	867

29.129.3.25	ArrowRegion(std::vector< std::pair< shared_ptr< PointOnEdge >, shared_ptr< PointOnEdge > > &narrow_regions)	867
29.129.3.26	FC1Discontinuity(std::vector< shared_ptr< ftSurface > > &discont_sfs)	867
29.129.3.27	CurvatureRadius(std::vector< std::pair< shared_ptr< ftPoint >, double > > &small_curv_rad, std::pair< shared_ptr< ftPoint >, double > &minimum_curv_rad)	868
29.129.3.28	G1Discontinuity(std::vector< shared_ptr< ftSurface > > &discont_sfs)	868
29.129.3.29	SliverSurfaces(std::vector< shared_ptr< ParamSurface > > &sliver_sfs, double thickness, double factor=2.0)	868
29.129.3.30	VanishingCurveTangent(std::vector< shared_ptr< PointOnCurve > > &sing_points, std::vector< std::pair< shared_ptr< PointOnCurve >, shared_ptr< PointOnCurve > > &sing_curves)	868
29.129.3.31	VanishingSurfaceNormal(std::vector< shared_ptr< ftPoint > > &singular_points, std::vector< shared_ptr< ftCurve > > &singular_curves)	868
29.130	Go::FaceSetRepair Class Reference	869
29.130.1	Detailed Description	870
29.130.2	Constructor & Destructor Documentation	870
29.130.2.1	FaceSetRepair(shared_ptr< SurfaceModel > sfmodel)	870
29.130.2.2	FaceSetRepair(shared_ptr< FaceSetQuality > quality)	870
29.130.2.3	~FaceSetRepair()	870
29.130.3	Member Function Documentation	870
29.130.3.1	consistentFaceNormal()	870
29.130.3.2	getAssociatedSfModel()	870
29.130.3.3	identicalAndEmbeddedFaces()	871
29.130.3.4	identicalVertices()	871
29.130.3.5	mendEdgeDistance()	871
29.130.3.6	mendGaps()	871
29.130.3.7	optimizeVertexPosition()	871
29.130	Go::Factorial< N > Struct Template Reference	871
29.131.1	Detailed Description	872
29.131.2	Member Enumeration Documentation	872
29.131.2.1	anonymous enum	872
29.130	Go::Factorial< 1 > Struct Template Reference	872

29.132.1	Detailed Description	872
29.132.2	Member Enumeration Documentation	873
29.132.2.1	anonymous enum	873
29.133	Go::Factory Class Reference	873
29.133.1	Detailed Description	873
29.133.2	Constructor & Destructor Documentation	873
29.133.2.1	~Factory()	873
29.133.3	Member Function Documentation	873
29.133.3.1	createObject(ClassType class_type)	873
29.133.3.2	globalFactory()	874
29.133.3.3	registerClass(ClassType class_type, Creator *c)	874
29.134	NEWMAT::FFT_Controller Class Reference	874
29.134.1	Detailed Description	875
29.134.2	Member Function Documentation	875
29.134.2.1	ar_1d_ft(int PTS, Real *X, Real *Y)	875
29.134.2.2	CanFactor(int PTS)	875
29.134.3	Member Data Documentation	875
29.134.3.1	OnlyOldFFT	875
29.135	NEWMAT::FindMaximum2 Class Reference	876
29.135.1	Detailed Description	876
29.135.2	Constructor & Destructor Documentation	876
29.135.2.1	~FindMaximum2()	876
29.135.3	Member Function Documentation	876
29.135.3.1	Fit(ColumnVector &, int)	876
29.136	NEWMAT::FloatingPointPrecision Class Reference	877
29.136.1	Detailed Description	877
29.136.2	Member Function Documentation	877
29.136.2.1	Dig()	877
29.136.2.2	Epsilon()	877
29.136.2.3	LnMaximum()	877

29.136.2.4	<code>LnMinimum()</code>	877
29.136.2.5	<code>Mantissa()</code>	878
29.136.2.6	<code>Maximum()</code>	878
29.136.2.7	<code>MaximumDecimalExponent()</code>	878
29.136.2.8	<code>MaximumExponent()</code>	878
29.136.2.9	<code>Minimum()</code>	878
29.136.2.10	<code>MinimumDecimalExponent()</code>	878
29.136.2.11	<code>MinimumExponent()</code>	878
29.136.2.12	<code>Radix()</code>	878
29.136.2.13	<code>Bounds()</code>	878
29.137	<code>Go::ftCell</code> Class Reference	879
29.137.1	Detailed Description	879
29.137.2	Constructor & Destructor Documentation	880
29.137.2.1	<code>ftCell()</code>	880
29.137.2.2	<code>~ftCell()</code>	880
29.137.3	Member Function Documentation	880
29.137.3.1	<code>addFace(ftSurface *f)</code>	880
29.137.3.2	<code>addFaceBox(BoundingBox &box)</code>	880
29.137.3.3	<code>box() const</code>	880
29.137.3.4	<code>face(int i) const</code>	880
29.137.3.5	<code>faceBox(int i) const</code>	880
29.137.3.6	<code>num_faces() const</code>	880
29.137.3.7	<code>removeFace(ftSurface *face)</code>	880
29.137.3.8	<code>setBox(const BoundingBox &box)</code>	881
29.137.4	Member Data Documentation	881
29.137.4.1	<code>box_</code>	881
29.137.4.2	<code>face_boxes_</code>	881
29.137.4.3	<code>faces_</code>	881
29.138	<code>Go::ftCellInfo</code> Class Reference	881
29.138.1	Detailed Description	881

29.138.2	Constructor & Destructor Documentation	882
29.138.2.1	ftCellInfo()	882
29.138.2.2	ftCellInfo(int i, double d)	882
29.138.3	Member Function Documentation	882
29.138.3.1	operator<(const ftCellInfo &ci) const	882
29.138.4	Member Data Documentation	882
29.138.4.1	dist_	882
29.138.4.2	index_	882
29.139	Go::ftChartSurface Class Reference	883
29.139.1	Detailed Description	884
29.139.2	Constructor & Destructor Documentation	885
29.139.2.1	ftChartSurface(const std::vector< shared_ptr< ParamSurface > > &surfaces, tpTolerances &topeps, double approxeps, double curvature_tol, int m=0, int n=0)	885
29.139.2.2	~ftChartSurface()	885
29.139.3	Member Function Documentation	885
29.139.3.1	createGrid(RotationInfo *rot_info=NULL, ftCurve *outer_bd=NULL)	885
29.139.3.2	createInitialEdges(double degenerate_epsilon=DEFAULT_SPACE_EPSILON, double kink=0.00015, bool no_split=false)	885
29.139.3.3	createSurf(double &max_error, double &mean_error)	885
29.139.3.4	getEdgeGrid(int edge_ind)	885
29.139.3.5	getError(double &max_error, double &mean_error)	885
29.139.3.6	getInnerEdgeCont() const	886
29.139.3.7	getPrioType() const	886
29.139.3.8	gridCreated(int &m, int &n) const	886
29.139.3.9	normal(double u, double v) const	886
29.139.3.10	point(double u, double v) const	886
29.139.3.11	point(double &u, double &v, shared_ptr< ftFaceBase > &face, double *seed=NULL, bool use_input_face=false) const	886
29.139.3.12	prepareGrid(RotationInfo *rot_info=NULL, ftCurve *outer_bd=NULL)	886
29.139.3.13	sampleOnlyEdges() const	886
29.139.3.14	setEdgeScales(std::vector< std::pair< Point, double > > &edge_scales)	887
29.139.3.15	setGridResolution(int m, int n)	887

29.139.3.16	setPrioType(ftTangPriority type)	887
29.139.3.17	setSymmDistrFunctions(bool symm_distr_functions)	887
29.139.3.18	writeDebugGrid(std::ostream &os) const	887
29.139.3.19	writeGrid(std::ostream &os) const	887
29.139.4	Member Data Documentation	887
29.139.4.1	curvature_tol_	887
29.139.4.2	edge_scales_	887
29.139.4.3	graph_	887
29.139.4.4	grid_distr_functions_	887
29.139.4.5	grid_pts_	887
29.139.4.6	m_	887
29.139.4.7	maxerror_	888
29.139.4.8	meanerror_	888
29.139.4.9	n_	888
29.139.4.10	prio_type_	888
29.139.4.11	secn_distr_	888
29.139.4.12	symm_distr_functions_	888
29.140	Go::ftCurve Class Reference	888
29.140.1	Detailed Description	890
29.140.2	Constructor & Destructor Documentation	890
29.140.2.1	ftCurve()	890
29.140.2.2	ftCurve(ftCurveType t)	890
29.140.3	Member Function Documentation	890
29.140.3.1	appendCurve(const ftCurve &cv)	890
29.140.3.2	appendSegment(const ftCurveSegment &seg)	890
29.140.3.3	arcLength(double t1, int seg1, double t2, int seg2) const	891
29.140.3.4	chopOff(const BoundingBox &box)	891
29.140.3.5	curveType() const	891
29.140.3.6	deleteSpaceCurveRepresentation()	891
29.140.3.7	endOfSegment(int segment) const	891

29.140.3.8	JoinSegments(double gap_tol, double neighbour_tol, double kink_tol, double bend_tol)	891
29.140.3.9	JointAfter(int segment) const	891
29.140.3.10	Normal(double t, int segment, int side, Point &normal, double eps) const	891
29.140.3.11	NumDisjointSubcurves() const	892
29.140.3.12	NumSegments() const	892
29.140.3.13	Operator+=(const ftCurve &cv)	892
29.140.3.14	OrientSegments(double neighbour_tol, bool assume_manifold=true)	892
29.140.3.15	Point(double t, int segment, Point &pt) const	892
29.140.3.16	PrependCurve(const ftCurve &cv)	892
29.140.3.17	PrependSegment(const ftCurveSegment &seg)	892
29.140.3.18	Reparametrize(double eps_geo=1.0e-6)	893
29.140.3.19	Reverse()	893
29.140.3.20	Segment(int segment) const	893
29.140.3.21	StartOfSegment(int segment) const	893
29.140.3.22	Tangent(double t, int segment, Point &tan) const	893
29.140.3.23	Triassellate(std::vector< shared_ptr< LineStrip > > &meshes) const	893
29.140.3.24	Triassellate(int resolution, std::vector< shared_ptr< LineStrip > > &meshes) const	893
29.140.3.25	Triassellate(double density, std::vector< shared_ptr< LineStrip > > &meshes) const	893
29.140.3.26	WorstJointType() const	894
29.140.3.27	Write(std::ostream &os) const	894
29.140.3.28	WriteSpaceCurve(std::ostream &os) const	894
29.140.4	Member Data Documentation	894
29.140.4.1	segments_	894
29.140.4.2	type_	894
29.141	Go::ftCurveSegment Class Reference	894
29.141.1	Detailed Description	896
29.141.2	Constructor & Destructor Documentation	896
29.141.2.1	ftCurveSegment(ftCurveType type, tpJointType joint, ftFaceBase *f0, ftFaceBase *f1, shared_ptr< ParamCurve > paramcv0, shared_ptr< ParamCurve > paramcv1, shared_ptr< ParamCurve > spacecv, double eps_geo=1.0e-6)	896

29.141.3	Member Function Documentation	896
29.141.3.1	arcLength(double t1, double t2) const	896
29.141.3.2	chopOff(const BoundingBox &box, bool &erase)	896
29.141.3.3	deleteSpaceCurveRepresentation()	896
29.141.3.4	endOfSegment() const	896
29.141.3.5	endPoint() const	896
29.141.3.6	face(int number) const	896
29.141.3.7	jointAfter() const	897
29.141.3.8	normal(double t, int side, Point &normal, double eps) const	897
29.141.3.9	paramcurvePoint(int number, double t, Point &pt) const	897
29.141.3.10	paramcurveTangent(int number, double t, Point &pt) const	897
29.141.3.11	parameterCurve(int number) const	897
29.141.3.12	point(double t, Point &pt) const	897
29.141.3.13	defineSpaceCurve(double eps_go)	897
29.141.3.14	parametrize(double eps_go=1.0e-6)	897
29.141.3.15	reverse()	897
29.141.3.16	segmentType() const	898
29.141.3.17	setJointAfter(tpJointType jt)	898
29.141.3.18	spaceCurve() const	898
29.141.3.19	startOfSegment() const	898
29.141.3.20	startPoint() const	898
29.141.3.21	tangent(double t, Point &tan) const	898
29.141.3.22	resellate(int resolution) const	898
29.141.4	Member Data Documentation	898
29.141.4.1	joint_	898
29.141.4.2	parameter_curve_	899
29.141.4.3	segment_type_	899
29.141.4.4	space_curve_	899
29.141.4.5	underlying_face_	899
29.141.5	Bo::ftEdge Class Reference	899

29.142.1	Detailed Description	902
29.142.2	Constructor & Destructor Documentation	902
29.142.2.1	ftEdge(ftFaceBase *face, shared_ptr< ParamCurve > cv, double tmin, double tmax, int entry_id=-1)	902
29.142.2.2	ftEdge(shared_ptr< ParamCurve > cv, double tmin, double tmax, int entry_id=-1)	903
29.142.2.3	ftEdge(ftFaceBase *face, shared_ptr< ParamCurve > cv, shared_ptr< Vertex > v1, shared_ptr< Vertex > v2, bool is_reversed=false, int entry_id=-1)	903
29.142.2.4	ftEdge(shared_ptr< ParamCurve > cv, shared_ptr< Vertex > v1, shared_ptr< Vertex > v2, bool is_reversed=false, int entry_id=-1)	903
29.142.2.5	~ftEdge()	903
29.142.3	Member Function Documentation	903
29.142.3.1	addEdgeMultiplicityInstance(ftEdge *other)	903
29.142.3.2	boundingBox()	903
29.142.3.3	checkEdgeTopology()	903
29.142.3.4	closeLoop(ftEdgeBase *last)	904
29.142.3.5	closestPoint(const Point &pt, double &clo_t, Point &clo_pt, double &clo_dist, double const *seed=0) const	904
29.142.3.6	commonVertex(ftEdge *other) const	904
29.142.3.7	connectAfter(ftEdgeBase *edge)	904
29.142.3.8	connectTwin(ftEdgeBase *twin, int &status)	904
29.142.3.9	disconnectThis()	904
29.142.3.10	disconnectTwin()	904
29.142.3.11	entryId()	904
29.142.3.12	estimatedCurveLength()	905
29.142.3.13	estimatedCurveLength(double min_par, double max_par)	905
29.142.3.14	face()	905
29.142.3.15	faceParameter(double t, double *seed=NULL) const	905
29.142.3.16	geomCurve()	905
29.142.3.17	geomEdge()	905
29.142.3.18	getAdjacentFaces() const	905
29.142.3.19	getAllAdjacentFaces() const	905
29.142.3.20	getCommonVertex(ftEdge *other)	905

29.142.3.21	getCurveIndex() const	906
29.142.3.22	getEdgeMultiplicityInstance()	906
29.142.3.23	getOtherVertex(const Vertex *vx)	906
29.142.3.24	getVertex(bool at_start)	906
29.142.3.25	getVertices(shared_ptr< Vertex > &v1, shared_ptr< Vertex > &v2)	906
29.142.3.26	hasCommonRadialEdge(ftEdge *other) const	906
29.142.3.27	hasCommonVertices(ftEdge *other) const	906
29.142.3.28	hasEdgeMultiplicity()	906
29.142.3.29	hasVertex(Vertex *vx)	907
29.142.3.30	Reversed()	907
29.142.3.31	inEdgeVertex(shared_ptr< EdgeVertex > radial_edge)	907
29.142.3.32	inVertex(shared_ptr< Vertex > this_vertex, shared_ptr< Vertex > other_vertex)	907
29.142.3.33	inVertices(ftEdgeBase *twin)	907
29.142.3.34	normal(double t) const	907
29.142.3.35	normal(double t, Point &face_par_pt, double *face_seed) const	907
29.142.3.36	orientationOK() const	907
29.142.3.37	parAtVertex(const Vertex *vx) const	908
29.142.3.38	point(double t) const	908
29.142.3.39	point(double t, int der, std::vector< Point > &derivs) const	908
29.142.3.40	removeEdgeVertex()	908
29.142.3.41	replaceVertex(shared_ptr< Vertex > &this_vertex, shared_ptr< Vertex > &other_vertex)	908
29.142.3.42	reverseGeomCurve()	908
29.142.3.43	setEdgeVertex(shared_ptr< EdgeVertex > radial_edge)	908
29.142.3.44	setEntryId(int id)	908
29.142.3.45	setFace(ftFaceBase *face)	908
29.142.3.46	setGeomCurve(shared_ptr< ParamCurve > geom_curve)	908
29.142.3.47	setReversed(bool is_reversed)	909
29.142.3.48	setVertices(shared_ptr< Vertex > v1, shared_ptr< Vertex > v2)	909
29.142.3.49	split(double t)	909
29.142.3.50	split2(double t)	909

29.142.3.51	splitAtVertex(shared_ptr< Vertex > vx)	909
29.142.3.52	tangent(double t) const	909
29.142.3.53	Max() const	909
29.142.3.54	Min() const	910
29.142.3.55	updateEdgeInfo(double tol)	910
29.142.3.56	updateGeomCurve(double tol)	910
29.143	Go::ftEdgeBase Class Reference	910
29.143.1	Detailed Description	912
29.143.2	Constructor & Destructor Documentation	912
29.143.2.1	ftEdgeBase()	912
29.143.2.2	~ftEdgeBase()	913
29.143.3	Member Function Documentation	913
29.143.3.1	adjacentEdges(bool at_start_of_edge, std::vector< ftEdgeBase * > &adjacent, std::vector< bool > &at_start)	913
29.143.3.2	boundingBox()=0	913
29.143.3.3	checkContinuity(ftEdgeBase *nextedge, double neighbour, double gap, double bend, double kink) const	913
29.143.3.4	checkOverlap(ftEdgeBase *other, double tol, int nmbsample, double &t1, double &t2, double &t3, double &t4, bool &same_dir, bool no_snap=true) const	913
29.143.3.5	closeLoop(ftEdgeBase *last)	913
29.143.3.6	closestPoint(const Go::Point &pt, double &clo_t, Go::Point &clo_pt, double &clo← _dist, double const *seed=0) const =0	913
29.143.3.7	connectAfter(ftEdgeBase *edge)	914
29.143.3.8	connectTwin(ftEdgeBase *twin, int &status)	914
29.143.3.9	disconnectThis()	914
29.143.3.10	disconnectTwin()	914
29.143.3.11	entryId()=0	914
29.143.3.12	face()=0	914
29.143.3.13	geomEdge()=0	914
29.143.3.14	getConnectivityInfo()	915
29.143.3.15	hasConnectivityInfo()	915
29.143.3.16	Reversed()=0	915

29.143.3.17	text()	915
29.143.3.18	normal(double t) const =0	915
29.143.3.19	normal(double t, Go::Point &face_par_pt, double *face_seed) const =0	915
29.143.3.20	onBoundary()	915
29.143.3.21	orientationOK() const	916
29.143.3.22	point(double t) const =0	916
29.143.3.23	prev()	916
29.143.3.24	resetConnectivityInfo()	916
29.143.3.25	setConnectivityInfo(shared_ptr< FaceConnectivity< ftEdgeBase > > info)	916
29.143.3.26	setEntryId(int id)=0	916
29.143.3.27	setFace(ftFaceBase *face)=0	916
29.143.3.28	setReversed(bool is_reversed)=0	916
29.143.3.29	split(double t)=0	917
29.143.3.30	tangent(double t) const =0	917
29.143.3.31	Max() const =0	917
29.143.3.32	Min() const =0	917
29.143.3.33	win()	917
29.143.4	Member Data Documentation	917
29.143.4.1	connectivity_info_	917
29.143.4.2	next_	917
29.143.4.3	prev_	918
29.143.4.4	twin_	918
29.144	Go::ftFaceBase Class Reference	918
29.144.1	Detailed Description	919
29.144.2	Constructor & Destructor Documentation	919
29.144.2.1	ftFaceBase()	919
29.144.2.2	ftFaceBase(int id)	919
29.144.2.3	~ftFaceBase()	920
29.144.3	Member Function Documentation	920
29.144.3.1	asFtSurface()	920

29.144.3.2	<code>boundingBox()</code> =0	920
29.144.3.3	<code>clearInitialEdges()</code>	920
29.144.3.4	<code>closestPoint(const Point &pt, double &clo_u, double &clo_v, Point &clo_pt, double &clo_dist, double epsilon) const =0</code>	920
29.144.3.5	<code>createInitialEdges(double degenerate_epsilon=DEFAULT_SPACE_EPSILON, double kink=0.00015, bool no_split=false)=0</code>	920
29.144.3.6	<code>createSurf(double &max_error, double &mean_error)=0</code>	920
29.144.3.7	<code>getError(double &max_error, double &mean_error)=0</code>	921
29.144.3.8	<code>getId()</code>	921
29.144.3.9	<code>getPrioType() const =0</code>	921
29.144.3.10	<code>isolateFace()</code>	921
29.144.3.11	<code>normal(double u, double v) const =0</code>	921
29.144.3.12	<code>point(double u, double v) const =0</code>	921
29.144.3.13	<code>removeGap(ftEdgeBase *e1, ftEdgeBase *e2, ftFaceBase *other)</code>	921
29.144.3.14	<code>setId(int id)</code>	922
29.144.3.15	<code>startEdges()</code> =0	922
29.144.3.16	<code>surface()</code> =0	922
29.144.3.17	<code>updateBoundaryLoops(shared_ptr< ftEdgeBase > new_edge)</code>	922
29.144.4	Member Data Documentation	922
29.144.4.1	<code>id_</code>	922
29.145	<code>Go::ftGraphEdge</code> Class Reference	922
29.145.1	Detailed Description	923
29.145.2	Constructor & Destructor Documentation	923
29.145.2.1	<code>ftGraphEdge()</code>	923
29.145.2.2	<code>ftGraphEdge(ftSurfaceSetPoint *&lower, ftSurfaceSetPoint *&upper)</code>	923
29.145.2.3	<code>~ftGraphEdge()</code>	923
29.145.3	Member Function Documentation	923
29.145.3.1	<code>endparam() const</code>	923
29.145.3.2	<code>endPoint() const</code>	923
29.145.3.3	<code>getFaces() const</code>	923
29.145.3.4	<code>lower() const</code>	924

29.145.3.5	point(double v_par)	924
29.145.3.6	point(double v_par, shared_ptr< ftFaceBase > face)	924
29.145.3.7	startparam() const	924
29.145.3.8	startPoint() const	924
29.145.3.9	upper() const	924
29.146	Go::ftGroupGeom Class Reference	924
29.146.1	Detailed Description	924
29.146.2	Constructor & Destructor Documentation	925
29.146.2.1	ftGroupGeom()	925
29.146.2.2	~ftGroupGeom()	925
29.146.3	Member Function Documentation	925
29.146.3.1	addGeomObj(shared_ptr< GeomObject > obj)	925
29.146.3.2	getType() const	925
29.146.3.3	operator[](int idx) const	925
29.146.3.4	setType(ftTangPriority type)	925
29.146.3.5	size() const	925
29.146.4	Member Data Documentation	926
29.146.4.1	geomobj_	926
29.146.4.2	type_	926
29.147	Go::ftLine Class Reference	926
29.147.1	Detailed Description	927
29.147.2	Constructor & Destructor Documentation	927
29.147.2.1	ftLine()	927
29.147.2.2	ftLine(const Point &dir, const Point &pnt)	927
29.147.2.3	~ftLine()	927
29.147.3	Member Function Documentation	927
29.147.3.1	closeToScaledPoint(const Point &pt, const Point &scale, double dist2) const	927
29.147.3.2	direction() const	927
29.147.3.3	getTwoPlanes(ftPlane &plane1, ftPlane &plane2) const	927
29.147.3.4	intersectsBox(const BoundingBox &box) const	928

29.147.3.5	intersectsSphereOfBox(const BoundingBox &box) const	928
29.147.3.6	planesIntersectBox(const BoundingBox &box) const	928
29.147.3.7	point() const	928
29.147.3.8	scaled(const Point &scale) const	928
29.147.4	Member Data Documentation	928
29.147.4.1	dir_	928
29.147.4.2	point_	928
29.148	Go::ftMessage Class Reference	928
29.148.1	Detailed Description	929
29.148.2	Constructor & Destructor Documentation	929
29.148.2.1	ftMessage()	929
29.148.2.2	ftMessage(ftmessages message)	929
29.148.2.3	~ftMessage()	929
29.148.3	Member Function Documentation	930
29.148.3.1	addWarning(ftmessages warning)	930
29.148.3.2	getMessage()	930
29.148.3.3	getWarning(int i)	930
29.148.3.4	isOK()	930
29.148.3.5	numberOfWarnings()	930
29.148.3.6	setError(ftmessages error)	930
29.149	Go::ftPlanarGraph Class Reference	930
29.149.1	Detailed Description	931
29.149.2	Constructor & Destructor Documentation	931
29.149.2.1	ftPlanarGraph()	931
29.149.2.2	ftPlanarGraph(std::vector< ftSamplePoint * > &nodes)	931
29.149.2.3	~ftPlanarGraph()	931
29.149.3	Member Function Documentation	931
29.149.3.1	getLocalParameters(double &u, double &v, shared_ptr< ftFaceBase > &face) const	931
29.149.3.2	locateInGraph(double u, double v) const	931
29.149.3.3	setGraph(std::vector< ftSamplePoint * > &nodes)	932

29.150	Go::ftPlane Class Reference	932
29.150.1	Detailed Description	933
29.150.2	Constructor & Destructor Documentation	933
29.150.2.1	ftPlane()	933
29.150.2.2	ftPlane(const Point &n, const Point &p)	933
29.150.2.3	ftPlane(const Point &x, const Point &y, const Point &z)	933
29.150.2.4	ftPlane(const ftPlane &plane)	933
29.150.2.5	~ftPlane()	933
29.150.3	Member Function Documentation	933
29.150.3.1	intersectsBox(const BoundingBox &box) const	933
29.150.3.2	normal() const	934
29.150.3.3	point() const	934
29.150.4	Member Data Documentation	934
29.150.4.1	normal_	934
29.150.4.2	point_	934
29.151	Go::ftPoint Class Reference	934
29.151.1	Detailed Description	935
29.151.2	Constructor & Destructor Documentation	935
29.151.2.1	ftPoint(Point pt, ftSurface *sf=0, double u=0, double v=0)	935
29.151.2.2	ftPoint(double x, double y, double z)	936
29.151.3	Member Function Documentation	936
29.151.3.1	face() const	936
29.151.3.2	normal() const	936
29.151.3.3	position() const	936
29.151.3.4	u() const	936
29.151.3.5	v() const	936
29.151.4	Member Data Documentation	936
29.151.4.1	pt_	936
29.151.4.2	surface_	937
29.151.4.3	u_	937

29.151.4.4_	937
29.152.0::ftPointSet Class Reference	937
29.152.1 Detailed Description	939
29.152.2 Constructor & Destructor Documentation	939
29.152.2.1 ftPointSet()	939
29.152.2.2 ~ftPointSet()	939
29.152.3 Member Function Documentation	940
29.152.3.1 addEntry(shared_ptr< ftSamplePoint > point)	940
29.152.3.2 append(shared_ptr< ftPointSet > triang)	940
29.152.3.3 checkAndUpdateTriangCorners()	940
29.152.3.4 cleanNodeIdentity(double tol)	940
29.152.3.5 computeDist(shared_ptr< ParamSurface > surf)	940
29.152.3.6 computeDistAndRepar(shared_ptr< ParamSurface > surf)	940
29.152.3.7 computeParametricDist(shared_ptr< ParamSurface > surf)	940
29.152.3.8 get3dNode(int i) const	940
29.152.3.9 getMaxDist() const	940
29.152.3.10 getMeanDist() const	941
29.152.3.11 getNeighbours(int i, std::vector< int > &neighbours) const	941
29.152.3.12 getNumNodes() const	941
29.152.3.13 getOrientedTriangles(std::vector< std::vector< int > > &triangles)	941
29.152.3.14 getPoints(std::vector< Vector3D > &positions) const	941
29.152.3.15 getTriangles(std::vector< std::vector< int > > &triangles) const	941
29.152.3.16 getU(int i) const	941
29.152.3.17 getV(int i) const	941
29.152.3.18 identifyBdPnts(std::vector< Point > &points, std::vector< int > &pnt_idx)	941
29.152.3.19 isBoundary(int i) const	942
29.152.3.20 isAdded()	942
29.152.3.21 mergeBoundary(shared_ptr< ftFaceBase > face1, int range1_idx1, int range1_idx2, shared_ptr< ftFaceBase > face2, int range2_idx1, int range2_idx2, double eps)	942
29.152.3.22 operator[](int idx)	942

29.152.3.29	operator[](int idx) const	942
29.152.3.29	orderNeighbours()	942
29.152.3.25	printPoints(std::ostream &os) const	942
29.152.3.26	removePoint(PointIter point)	942
29.152.3.27	repairBdy(shared_ptr< ParamSurface > surf, bool use_seed=true)	942
29.152.3.26	repairInnerPoints(shared_ptr< ParamSurface > surf, bool use_seed=true)	943
29.152.3.29	set3dNode(int i, const Vector3D &p)	943
29.152.3.30	setFirst(PointIter point)	943
29.152.3.31	setSecond(PointIter point)	943
29.152.3.32	setU(int i, double u)	943
29.152.3.33	setV(int i, double v)	943
29.152.3.34	size() const	943
29.152.3.35	write(std::ostream &os) const	943
29.152.3.36	write2D(std::ostream &os) const	944
29.152.4	Member Data Documentation	944
29.152.4.1	first_	944
29.152.4.2	index_to_iter_	944
29.152.4.3	nb_ordered_	944
29.152.4.4	points_	944
29.152.4.5	second_	944
29.153	Go::ftSamplePoint Class Reference	945
29.153.1	Detailed Description	947
29.153.2	Constructor & Destructor Documentation	947
29.153.2.1	ftSamplePoint(Vector3D xyz, int bnd)	947
29.153.2.2	~ftSamplePoint()	947
29.153.3	Member Function Documentation	947
29.153.3.1	addNeighbour(PointIter next)	947
29.153.3.2	asSurfaceSetPoint()	947
29.153.3.3	containsFace(ftFaceBase *face) const	947
29.153.3.4	getAttachedTriangles(std::vector< std::vector< int > > &triangles) const	947

29.153.3.5	getDist() const	948
29.153.3.6	getFirstNeighbour()	948
29.153.3.7	getIndex() const	948
29.153.3.8	getNeighbours() const	948
29.153.3.9	getNmbNeighbour() const	948
29.153.3.10	getPar() const	948
29.153.3.11	getPoint() const	948
29.153.3.12	isConnected(PointIter pnt)	948
29.153.3.13	isOnBoundary() const	949
29.153.3.14	isOnSubSurfaceBoundary() const	949
29.153.3.15	orderNeighbours(ftSamplePoint *nextpoint, bool forward)	949
29.153.3.16	pointDist(ftSamplePoint *other) const	949
29.153.3.17	removeNeighbour(PointIter neighbour)	949
29.153.3.18	setBoundary(int bd_info)	949
29.153.3.19	setDist(double d)	949
29.153.3.20	setIndex(int i)	949
29.153.3.21	setPar(Vector2D uv)	950
29.153.3.22	setPoint(Vector3D xyz)	950
29.153.3.23	write2Dval(std::ostream &os) const	950
29.153.4	Member Data Documentation	950
29.153.4.1	at_boundary_	950
29.153.4.2	dist_	950
29.153.4.3	index_	950
29.153.4.4	next_	950
29.153.4.5	uv_	950
29.153.4.6	xyz_	951
29.154	Go::ftSearchNode Class Reference	951
29.154.1	Detailed Description	951
29.154.2	Constructor & Destructor Documentation	951
29.154.2.1	ftSearchNode(ftSurfaceSetPoint *node)	951

29.154.2.2~ftSearchNode()	951
29.154.3Member Function Documentation	952
29.154.3.1getOrderedSegments() const	952
29.154.3.2node() const	952
29.154.3.3setOrderedSegments(const std::vector< ftGraphEdge > &edges)	952
29.155Go::ftSmoothSurf Class Reference	952
29.155.1Detailed Description	953
29.155.2Constructor & Destructor Documentation	953
29.155.2.1ftSmoothSurf(shared_ptr< SplineSurface > surf, double approxtol, double approx_orig_tol, std::vector< int > ccw_edge_derivs, int maxiter, bool lock_corner_points=false)	953
29.155.2.2~ftSmoothSurf()	953
29.155.3Member Function Documentation	953
29.155.3.1getApproxWeight()	953
29.155.3.2getError(double &max_error, double &mean_error)	954
29.155.3.3refineSurf(ftPointSet &points, bool reparam=true)	954
29.155.3.4setApproxWeight(double weight)	954
29.155.3.5setSmoothU(int k)	954
29.155.3.6setSmoothV(int k)	954
29.155.3.7update(ftPointSet &points, double gapeps, bool reparam=true)	954
29.155.4Member Data Documentation	954
29.155.4.1approx_orig_tol_	954
29.155.4.2approxtol_	954
29.155.4.3ccw_edge_derivs_	954
29.155.4.4init_approx_weight_	955
29.155.4.5lock_corner_points_	955
29.155.4.6max_error_	955
29.155.4.7maxiter_	955
29.155.4.8mean_error_	955
29.155.4.9orig_surf_	955
29.155.4.10sem_	955

29.155.4.1surf_	955
29.156Go::ftSSfEdge Class Reference	956
29.156.1Detailed Description	957
29.156.2Constructor & Destructor Documentation	958
29.156.2.1ftSSfEdge(ftFaceBase *face, ftEdge *edge, int entry_id=-1)	958
29.156.2.2~ftSSfEdge()	958
29.156.3Member Function Documentation	958
29.156.3.1boundingBox()	958
29.156.3.2closestPoint(const Point &pt, double &clo_t, Point &clo_pt, double &clo_dist, double const *seed=0) const	958
29.156.3.3entryId()	958
29.156.3.4face()	958
29.156.3.5geomEdge()	958
29.156.3.6isReversed()	959
29.156.3.7normal(double t) const	959
29.156.3.8normal(double t, Point &face_par_pt, double *face_seed) const	959
29.156.3.9point(double t) const	959
29.156.3.10getEntryId(int id)	959
29.156.3.11setFace(ftFaceBase *face)	959
29.156.3.12setReversed(bool is_reversed)	959
29.156.3.13split(double t)	960
29.156.3.14tangent(double t) const	960
29.156.3.15Max() const	960
29.156.3.16Min() const	960
29.157Go::ftSurface Class Reference	961
29.157.1Detailed Description	965
29.157.2Constructor & Destructor Documentation	965
29.157.2.1ftSurface(shared_ptr< ParamSurface > sf, int id)	965
29.157.2.2ftSurface(shared_ptr< ParamSurface > sf, shared_ptr< Loop > loop, int id=-1)	965
29.157.2.3~ftSurface()	966
29.157.3Member Function Documentation	966

29.157.3.1	addBoundaryLoops(std::vector< shared_ptr< Loop > > &bd_loops)	966
29.157.3.2	addOuterBoundaryLoop(shared_ptr< Loop > outer_loop)	966
29.157.3.3	allRadialEdges() const	966
29.157.3.4	area(double tol) const	966
29.157.3.5	areNeighbours(ftSurface *other, shared_ptr< ftEdge > &edge1, shared_ptr< ftEdge > &edge2, int adj_idx=0) const	966
29.157.3.6	asFtSurface()	966
29.157.3.7	boundingBox()	966
29.157.3.8	checkAndFixBoundaries()	967
29.157.3.9	checkDegAdjacency(ftSurface *other, shared_ptr< Vertex > vx, double tol, shared_ptr< ParamCurve > &bdcv1, shared_ptr< ParamCurve > &bdcv2)	967
29.157.3.10	checkFaceTopology()	967
29.157.3.11	checkLoopOrientation(std::vector< shared_ptr< Loop > > &inconsistent_loops) const	967
29.157.3.12	clearInitialEdges()	967
29.157.3.13	closestBoundaryPoint(const Point &pt, const Point &in_vec, double &clo_u, double &clo_v, Point &clo_pt, double &clo_dist, double &clo_par) const	967
29.157.3.14	closestOuterBoundaryPoint(const Point &pt, const Point &in_vec, double &clo_u, double &clo_v, Point &clo_pt, double &clo_dist, double &clo_par) const	967
29.157.3.15	closestPoint(const Point &pt, double &clo_u, double &clo_v, Point &clo_pt, double &clo_dist, double epsilon) const	967
29.157.3.16	closestPoint(const Point &pt, double &clo_u, double &clo_v, Point &clo_pt, double &clo_dist, double epsilon, const RectDomain *domain_of_interest, double *seed) const	968
29.157.3.17	commonSplineSpace(ftSurface *other, double tol)	968
29.157.3.18	connectTwin(ftSurface *newtwin, double tol, bool no_snap=true)	968
29.157.3.19	connectTwin(ftSurface *newtwin, std::vector< ftEdgeBase * > first1, std::vector< ftEdgeBase * > first2, std::vector< int > forward)	968
29.157.3.20	createInitialEdges(double degenerate_epsilon=DEFAULT_SPACE_EPSILON, double kink=0.00015, bool no_split=false)	968
29.157.3.21	createSurf(double &max_error, double &mean_error)	968
29.157.3.22	disconnectTwin()	968
29.157.3.23	edgeClosestToPoint(double u, double v)	968
29.157.3.24	elementBoundaryStatus(int elem_ix, double eps)	968
29.157.3.25	elementOnBoundary(int elem_ix, double eps)	969

29.157.3.26	FetchCorrespondingFaces() const	969
29.157.3.27	GetAdjacencyInfo(ftSurface *other, double tol, int &bd1, int &bd2, bool &same_orient)	969
29.157.3.28	GetAdjacencyInfo(ftSurface *other, double tol, int adj_idx=0, bool test_corner=false)	970
29.157.3.29	GetAdjacencyInfo(ftEdge *edge, ftSurface *other, double tol)	970
29.157.3.30	GetAdjacentBodies() const	970
29.157.3.31	GetAdjacentFaces(std::vector< ftSurface * > &neighbours) const	970
29.157.3.32	GetAllEdgePtrs() const	970
29.157.3.33	GetAllEdgePtrs(int loop_idx) const	970
29.157.3.34	GetAllEdges() const	970
29.157.3.35	GetAllEdges(int loop_idx) const	970
29.157.3.36	GetBadDistance(std::vector< std::pair< ftSurface *, shared_ptr< Vertex > > > &badPairs, double tol)	970
29.157.3.37	GetBadDistance(std::vector< std::pair< ftSurface *, ftEdge * > > &badPairs, double tol) const	971
29.157.3.38	GetBadDistance(std::vector< std::pair< ftEdge *, shared_ptr< Vertex > > > &badPairs, double tol) const	971
29.157.3.39	GetBody() const	971
29.157.3.40	GetBoundaryCoefEnumeration(int bd, std::vector< int > &enumeration)	971
29.157.3.41	GetBoundaryConditions(int &bd_type, int &bd) const	971
29.157.3.42	GetBoundaryLoop(int idx)	971
29.157.3.43	GetBoundaryPiece(Point &pt1, Point &pt2, double eps)	971
29.157.3.44	GetClosestVertex(const Point &pnt) const	971
29.157.3.45	GetCommonEdges(ftSurface *other) const	971
29.157.3.46	GetCommonVertices(ftSurface *other) const	972
29.157.3.47	GetCornerVertices(double kink) const	972
29.157.3.48	GetCornerVertices(double kink, int loop_idx) const	972
29.157.3.49	GetCorrCoefEnumeration(ftSurface *other, double tol, std::vector< std::pair< int, int > > &enumeration)	972
29.157.3.50	GetCurrEps() const	972
29.157.3.51	GetError(double &max_error, double &mean_error)	972
29.157.3.52	GetFreeBoundaryInfo(double tol, std::vector< int > &free_boundaries)	972
29.157.3.53	GetId()	972

29.157.3.54	getNarrowRegion(double gap_tol, double tol, std::vector< std::pair< shared_ptr< PointOnEdge >, shared_ptr< PointOnEdge > > &narrow_pt)	973
29.157.3.55	getNonCornerVertices(double kink) const	973
29.157.3.56	getNonCornerVertices(double kink, int loop_idx) const	973
29.157.3.57	getPosTangentSurfaceDiscont(std::vector< ftEdge * > &badPos, std::vector< ftEdge * > &badTangent, double tol, double kink, double bend, int leastSurfaceIndex, shared_ptr< SurfaceModel > sm) const	973
29.157.3.58	getPrioType() const	973
29.157.3.59	getRadialEdges() const	973
29.157.3.60	getSurfaceDisconts(double tol, std::vector< double > &disc_u, std::vector< double > &disc_v)	973
29.157.3.61	getSurfaceKinks(double angtol, std::vector< double > &g1_disc_u, std::vector< double > &g1_disc_v)	973
29.157.3.62	getUntrimmed(double gap, double neighbour, double angtol, bool only_corner=false)	973
29.157.3.63	hasAcuteAngle(ftEdge *along_edge, double angtol) const	974
29.157.3.64	hasBody() const	974
29.157.3.65	hasBoundaryConditions() const	974
29.157.3.66	hasRadialEdges() const	974
29.157.3.67	hasRealRadialEdges() const	974
29.157.3.68	hasTwin()	974
29.157.3.69	Close(shared_ptr< Vertex > v, double tol) const	974
29.157.3.70	CornerToCorner(ftSurface *other, double tol, int adj_idx=0)	974
29.157.3.71	isolateFace()	974
29.157.3.72	Spline() const	974
29.157.3.73	makeCommonSplineSpace(ftSurface *other)	975
29.157.3.74	mbAdjacencies(ftSurface *other) const	975
29.157.3.75	mbBoundaryLoops()	975
29.157.3.76	mbEdges() const	975
29.157.3.77	mbNextNeighbours(ftSurface *other) const	975
29.157.3.78	mbOuterBdCrvs(double gap, double neighbour, double angtol) const	975
29.157.3.79	normal(double u, double v) const	975
29.157.3.80	onlyOuterTrim() const	975

29.157.3.81	point(double u, double v) const	975
29.157.3.82	pointInFace(double u, double v, double tol)	976
29.157.3.83	pointInFace2(double u, double v, double tol)	976
29.157.3.84	pointOnBd(double u, double v, double tol)	976
29.157.3.85	removeGap(ftEdgeBase *e1, ftEdgeBase *e2, ftFaceBase *other, double epsge)	976
29.157.3.86	replaceSurf(shared_ptr< ParamSurface > sf)	976
29.157.3.87	setBody(Body *body)	976
29.157.3.88	setBoundaryConditions(int bd_type, int bd)	976
29.157.3.89	setId(int id)	976
29.157.3.90	setPrioType(ftTangPriority type)	977
29.157.3.91	setTwin(ftSurface *newtwin)	977
29.157.3.92	smoothOutFace(int edge_cont, double approx_orig_tol, double deg_tol, double &maxerr, double &meanerr, double approx_weight=0.8)	977
29.157.3.93	splitAlongKinks(double angtol)	977
29.157.3.94	splitAtInternalCorner(ftSurface *other, std::vector< shared_ptr< ftSurface > > &new_face1, std::vector< shared_ptr< ftSurface > > &new_face2, double tol=DEFAULT_SPACE_EPSILON)	977
29.157.3.95	startEdges()	977
29.157.3.96	surface()	977
29.157.3.97	twin()	977
29.157.3.98	updateBoundaryLoops(shared_ptr< ftEdgeBase > new_edge)	978
29.157.3.99	vertices() const	978
29.157.3.100	write(std::ostream &os)	978
29.158.0	ftSurfaceSet Class Reference	978
29.158.1	Detailed Description	980
29.158.2	Constructor & Destructor Documentation	980
29.158.2.1	ftSurfaceSet(const std::vector< shared_ptr< ParamSurface > > &surfaces, tp← Tolerances &topeps, double approxeps)	980
29.158.2.2	~ftSurfaceSet()	980
29.158.3	Member Function Documentation	980
29.158.3.1	boundingBox()	980
29.158.3.2	closestPoint(const Point &pt, double &clo_u, double &clo_v, Point &clo_pt, double &clo_dist, double epsilon) const	981

29.158.3.3	createInitialEdges(double degenerate_epsilon=DEFAULT_SPACE_EPSILON, double kink=0.00015, bool no_split=false)=0	981
29.158.3.4	createSurf(double &max_error, double &mean_error)=0	981
29.158.3.5	estimateSurfSides(SplineSurface *surf)	981
29.158.3.6	fetchSamplePoints(const std::vector< ftEdgeBase * > &edgeloop, std::vector< int > &corner, std::vector< int > &cn, ftPointSet &points)	981
29.158.3.7	getApproxweight()	981
29.158.3.8	getBoundaryConditions(const std::vector< ftEdgeBase * > &edgeloop, std::vector< int > &corner, std::vector< shared_ptr< SplineCurve > > &bd_curves, std::vector< shared_ptr< SplineCurve > > &cross_curves, bool compute_cross_curves, std::vector< BoundaryPiece > &bdpiece, bool require_cord_length_param=false)	981
29.158.3.9	getBoundaryPiece(Point &ptmin, Point &ptmax, double eps)	981
29.158.3.10	getError(double &max_error, double &mean_error)=0	981
29.158.3.11	getId()	982
29.158.3.12	getInitBndData(std::vector< ftEdgeBase * > &edgc, ftPointSet &points, int cn[])	982
29.158.3.13	getInitInnerData(ftPointSet &points, int max_sample)	982
29.158.3.14	getInitInnerData2(ftPointSet &points)	982
29.158.3.15	getNextEdgePieceInfo(std::vector< ftEdgeBase * >::iterator &first_edge, std::vector< ftEdgeBase * >::iterator &last_edge, ftFaceBase * &adjsurf, SplineCurve * &cv, Point &ptmin, Point &ptmax, double &length)	982
29.158.3.16	getOuterLoop(std::vector< ftEdgeBase * > &outer_loop, std::vector< int > &corners, double bend_tol, double kink_tol)	982
29.158.3.17	getPrioType() const =0	982
29.158.3.18	getSurfaces(vector< shared_ptr< ParamSurface > > &surfaces)	982
29.158.3.19	inCrossCurves(std::vector< shared_ptr< SplineCurve > > &cross_curves, double startpar, double endpar)	982
29.158.3.20	merge(double &max_error, double &mean_error, double bend_tol, double kink_tol)	982
29.158.3.21	modifySurface(int maxiter, int max_update_iter, std::vector< int > &edge_derivs, ftPointSet &points, double &max_error, double &mean_error)	982
29.158.3.22	normal(double u, double v) const =0	982
29.158.3.23	point(double u, double v) const =0	983
29.158.3.24	parametrizeSurf(const std::vector< ftEdgeBase * > &first_edges, const std::vector< int > &corners, ftPointSet &points, double umin=0, double umax=0, double vmin=0, double vmax=0)	983
29.158.3.25	sampleOnlyEdges() const =0	983

29.158.3.26	SetAdditionalCornerPts(std::vector< Point > &add_corner_pts)	983
29.158.3.27	SetApproxweight(double approxweight)	983
29.158.3.28	SetId(int id)	983
29.158.3.29	StartEdges()	983
29.158.3.30	Surface()	983
29.158.3.31	TrimWithPlane(const ftPlane &plane)	984
29.158.3.32	UpdatePointTopology(ftPointSet &points)	984
29.158.4	Member Data Documentation	984
29.158.4.1	additional_corner_pts_	984
29.158.4.2	approx_bd_curves_	984
29.158.4.3	approxxtol_	984
29.158.4.4	approxweight_	984
29.158.4.5	boundary_loops_	984
29.158.4.6	faces_	984
29.158.4.7	surf_	984
29.158.4.8	optol_	985
29.159.0	ftSurfaceSetPoint Class Reference	985
29.159.1	Detailed Description	986
29.159.2	Constructor & Destructor Documentation	986
29.159.2.1	ftSurfaceSetPoint(Vector3D xyz, int bnd)	986
29.159.2.2	ftSurfaceSetPoint(Vector3D xyz, int bnd, shared_ptr< ftFaceBase > &face, Vector2D par_pt)	986
29.159.2.3	~ftSurfaceSetPoint()	987
29.159.3	Member Function Documentation	987
29.159.3.1	addFace(shared_ptr< ftFaceBase > &face)	987
29.159.3.2	addInfo(ftSurfaceSetPoint *other)	987
29.159.3.3	addPair(shared_ptr< ftFaceBase > &face, const Vector2D &par_pt)	987
29.159.3.4	asSurfaceSetPoint()	987
29.159.3.5	containsFace(ftFaceBase *face) const	987
29.159.3.6	face(int i)	987
29.159.3.7	getPar(ftFaceBase *face)	987

29.159.3.8	<code>nmbFaces()</code>	988
29.159.3.9	<code>parValue(int i)</code>	988
29.159.3.10	<code>setPosition(Vector3D pos, int bnd)</code>	988
29.159.3.11	<code>setPosition(Vector3D pos)</code>	988
29.159.3.12	<code>write2Dval(std::ostream &os) const</code>	988
29.160	<code>Go::ftVolume</code> Class Reference	988
29.160.1	Detailed Description	990
29.160.2	Constructor & Destructor Documentation	990
29.160.2.1	<code>ftVolume(shared_ptr< ParamVolume > vol, int id=-1)</code>	990
29.160.2.2	<code>ftVolume(shared_ptr< ParamVolume > vol, double gap_eps, double kink_eps, int id=-1)</code>	991
29.160.2.3	<code>ftVolume(shared_ptr< ParamVolume > vol, double gap_eps, double neighbour, double kink_eps, double bend, int id=-1)</code>	991
29.160.2.4	<code>ftVolume(shared_ptr< ParamVolume > vol, shared_ptr< SurfaceModel > shell, int id=-1)</code>	991
29.160.2.5	<code>ftVolume(shared_ptr< ParamVolume > vol, std::vector< shared_ptr< SurfaceModel > > shells, int id=-1)</code>	991
29.160.2.6	<code>ftVolume(shared_ptr< SurfaceModel > shell, int id=-1)</code>	991
29.160.2.7	<code>~ftVolume()</code>	991
29.160.3	Member Function Documentation	991
29.160.3.1	<code>boundingBox() const</code>	991
29.160.3.2	<code>checkBodyTopology()</code>	991
29.160.3.3	<code>checkDegAdjacency(ftVolume *other, shared_ptr< EdgeVertex > evx, double tol, shared_ptr< ParamSurface > &bdsf1, shared_ptr< ParamSurface > &bdsf2)</code>	991
29.160.3.4	<code>closestBoundaryPoint(Point &pt, double &clo_u, double &clo_v, double &clo_w, Point &clo_pt, double &clo_dist, double &clo_par_u, double &clo_par_v, double epsilon) const</code>	992
29.160.3.5	<code>closestPoint(Point &pt, double &clo_u, double &clo_v, double &clo_w, Point &clo_pt, double &clo_dist, double epsilon, double *seed=NULL) const</code>	992
29.160.3.6	<code>commonSplineSpace(ftVolume *other, double tol)</code>	992
29.160.3.7	<code>ElementBoundaryStatus(int elem_ix)</code>	992
29.160.3.8	<code>ElementOnBoundary(int elem_ix)</code>	992
29.160.3.9	<code>getAdjacencyInfo(ftVolume *other, double tol, int &bd1, int &bd2, int &orientation, bool &same_seq)</code>	993

29.160.3.10 [GetAdjacencyInfo\(ftVolume *other, double tol, int adj_idx=0, bool test_corner=false\)](#) 993

29.160.3.11 [GetAdjacentBodies\(std::vector< ftVolume * > &neighbours\)](#) 993

29.160.3.12 [GetBoundaryCoefEnumeration\(int bd, std::vector< int > &enumeration\)](#) 993

29.160.3.13 [GetCommonEdges\(ftVolume *other\) const](#) 993

29.160.3.14 [GetCornerAdjacencyInfo\(ftVolume *other, EdgeVertex *evx, double tol, int adj_idx=0\)](#) 993

29.160.3.15 [GetCornerAdjacencyInfo\(ftVolume *other, double tol, int adj_idx=0\)](#) 993

29.160.3.16 [GetCorrCoefEnumeration\(ftVolume *other, double tol, std::vector< std::pair< int, int > > &enumeration\)](#) 993

29.160.3.17 [GetFreeBoundaryInfo\(double tol, std::vector< int > &free_boundaries\)](#) 994

29.160.3.18 [GetId\(\)](#) 994

29.160.3.19 [GetVertexEnumeration\(shared_ptr< Vertex > vx, Point ¶m, int &corner, int &coef_nmb\) const](#) 994

29.160.3.20 [GetVertexPosition\(shared_ptr< Vertex > vx, Point ¶m\) const](#) 994

29.160.3.21 [GetVolume\(\)](#) 994

29.160.3.22 [BoundaryTrimmed\(\) const](#) 994

29.160.3.23 [CornerToCorner\(shared_ptr< ftVolume > other, double tol\)](#) 994

29.160.3.24 [IsoTrimmed\(\) const](#) 994

29.160.3.25 [Regularized\(\) const](#) 995

29.160.3.26 [Spline\(\) const](#) 995

29.160.3.27 [MakeCommonSplineSpace\(ftVolume *other\)](#) 995

29.160.3.28 [ParamInVolume\(double u, double v, double w\)](#) 995

29.160.3.29 [RadialEdges\(\) const](#) 995

29.160.3.30 [RegularizeBdShells\(std::vector< std::pair< Point, Point > > &corr_vx_pts, std::vector< SurfaceModel * > &modified_adjacent, int split_mode=1, bool pattern_split=false\)](#) 995

29.160.3.31 [ReplaceWithRegVolumes\(int degree, std::vector< SurfaceModel * > &modified_adjacent, bool performe_step2=true, int split_mode=1, bool pattern_split=false\)](#) 995

29.160.3.32 [SplitAtInternalCorner\(ftVolume *other, std::vector< shared_ptr< ftVolume > > &new_vol1, std::vector< shared_ptr< ftVolume > > &new_vol2, double tol=DEFAULT_SPACE_EPSILON\)](#) 995

29.160.3.33 [UniqueNonRadialEdges\(\) const](#) 996

29.160.3.34 [TrimRegular\(int degree\)](#) 996

29.160.3.35 [UpdateBoundaryInfo\(\)](#) 996

29.160	Go::Fun2Fun< Functor > Class Template Reference	996
29.161.1	Detailed Description	996
29.161.2	Constructor & Destructor Documentation	996
29.161.2.1	Fun2Fun(const Functor &f, double a, double b)	996
29.161.3	Member Function Documentation	997
29.161.3.1	maxPar(int n) const	997
29.161.3.2	minPar(int n) const	997
29.161.3.3	operator()(const double *arg) const	997
29.162	Go::FunctionMinimizer< Functor > Class Template Reference	997
29.162.1	Detailed Description	998
29.162.2	Constructor & Destructor Documentation	999
29.162.2.1	FunctionMinimizer(int num_param, const Functor &fun, const double *const seed, double tol=std::sqrt(std::numeric_limits< double >::epsilon()))	999
29.162.2.2	~FunctionMinimizer()	999
29.162.3	Member Function Documentation	999
29.162.3.1	atMax(int param_ix) const	999
29.162.3.2	atMin(int param_ix) const	999
29.162.3.3	fval() const	1000
29.162.3.4	fval(const Point &param) const	1000
29.162.3.5	getPar(int param_ix) const	1000
29.162.3.6	getPar() const	1000
29.162.3.7	grad(Point &result) const	1000
29.162.3.8	grad(const Point &param, Point &result) const	1000
29.162.3.9	inminBrent(const Point &dir, const double *bracket, const double *fval_brak)	1001
29.162.3.10	inimize(const Point &dir, bool &hit_domain_edge, bool rerun=false)	1001
29.162.3.11	moveUV(const Point &dir, double multiplier)	1001
29.162.3.12	numPars() const	1001
29.163	GARCH11_LL Class Reference	1002
29.163.1	Detailed Description	1002
29.163.2	Constructor & Destructor Documentation	1002
29.163.2.1	GARCH11_LL(const ColumnVector &y, const ColumnVector &x)	1002

29.163.3	Member Function Documentation	1003
29.163.3.1	Derivatives()	1003
29.163.3.2	FI()	1003
29.163.3.3	sValid()	1003
29.163.3.4	LogLikelihood()	1003
29.163.3.5	Set(const ColumnVector &p)	1003
29.164	NEWMAT::GeneralMatrix Class Reference	1003
29.164.1	Detailed Description	1007
29.164.2	Constructor & Destructor Documentation	1007
29.164.2.1	GeneralMatrix()	1007
29.164.2.2	GeneralMatrix(ArrayLengthSpecifier)	1007
29.164.2.3	~GeneralMatrix()	1007
29.164.3	Member Function Documentation	1007
29.164.3.1	Add(GeneralMatrix *, Real)	1007
29.164.3.2	Add(Real)	1007
29.164.3.3	BorrowStore(GeneralMatrix *, MatrixType)	1007
29.164.3.4	CheckConversion(const BaseMatrix &)	1007
29.164.3.5	CheckStore() const	1007
29.164.3.6	CleanUp()	1007
29.164.3.7	Eq(const BaseMatrix &, MatrixType)	1008
29.164.3.8	Eq(const BaseMatrix &, MatrixType, bool)	1008
29.164.3.9	Eq2(const BaseMatrix &, MatrixType)	1008
29.164.3.10	Evaluate(MatrixType mt=MatrixTypeUnSp)	1008
29.164.3.11	ForReturn() const	1008
29.164.3.12	GetCol(MatrixRowCol &)=0	1008
29.164.3.13	GetCol(MatrixColX &)=0	1008
29.164.3.14	GetMatrix(const GeneralMatrix *)	1008
29.164.3.15	GetRow(MatrixRowCol &)=0	1008
29.164.3.16	GetStore()	1008
29.164.3.17	Inject(const GeneralMatrix &)	1009

29.164.3.18	Equal(const GeneralMatrix &) const	1009
29.164.3.19	Zero() const	1009
29.164.3.20	logDeterminant() const	1009
29.164.3.21	MakeSolver()	1009
29.164.3.22	Maximum() const	1009
29.164.3.23	Maximum1(int &i) const	1009
29.164.3.24	Maximum2(int &i, int &j) const	1010
29.164.3.25	MaximumAbsoluteValue() const	1010
29.164.3.26	MaximumAbsoluteValue1(int &i) const	1010
29.164.3.27	MaximumAbsoluteValue2(int &i, int &j) const	1010
29.164.3.28	Minimum() const	1010
29.164.3.29	Minimum1(int &i) const	1010
29.164.3.30	Minimum2(int &i, int &j) const	1011
29.164.3.31	MinimumAbsoluteValue() const	1011
29.164.3.32	MinimumAbsoluteValue1(int &i) const	1011
29.164.3.33	MinimumAbsoluteValue2(int &i, int &j) const	1011
29.164.3.34	Multiply(GeneralMatrix *, Real)	1011
29.164.3.35	Multiply(Real)	1011
29.164.3.36	cols() const	1011
29.164.3.37	negAdd(GeneralMatrix *, Real)	1011
29.164.3.38	negAdd(Real)	1011
29.164.3.39	negate(GeneralMatrix *)	1011
29.164.3.40	negate()	1011
29.164.3.41	nextCol(MatrixRowCol &)	1012
29.164.3.42	nextCol(MatrixColX &)	1012
29.164.3.43	nextRow(MatrixRowCol &)	1012
29.164.3.44	rows() const	1012
29.164.3.45	operator ReturnMatrix() const	1012
29.164.3.46	operator&=(const BaseMatrix &)	1012
29.164.3.47	operator*=(const BaseMatrix &)	1012

29.164.3.48	operator*=(Real)	1012
29.164.3.49	operator+=(const BaseMatrix &)	1012
29.164.3.50	operator+=(Real)	1012
29.164.3.51	operator-=(const BaseMatrix &)	1012
29.164.3.52	operator-=(Real r)	1012
29.164.3.53	operator/=(Real r)	1012
29.164.3.54	operator<<(const Real *)	1013
29.164.3.55	operator<<(const BaseMatrix &X)	1013
29.164.3.56	operator<<(Real)	1013
29.164.3.57	operator<<(int f)	1013
29.164.3.58	operator=(Real)	1013
29.164.3.59	operator" ="(const BaseMatrix &)	1013
29.164.3.60	Protect()	1013
29.164.3.61	Release()	1013
29.164.3.62	Release(int t)	1013
29.164.3.63	ReleaseAndDelete()	1013
29.164.3.64	ReSize(int, int, int)	1013
29.164.3.65	ReSize(const GeneralMatrix &A)	1013
29.164.3.66	ReSizeForAdd(const GeneralMatrix &A, const GeneralMatrix &B)	1014
29.164.3.67	ReSizeForSP(const GeneralMatrix &A, const GeneralMatrix &B)	1014
29.164.3.68	RestoreCol(MatrixRowCol &)	1014
29.164.3.69	RestoreCol(MatrixColX &)	1014
29.164.3.70	RestoreRow(MatrixRowCol &)	1014
29.164.3.71	Reuse()	1014
29.164.3.72	ReverseElements()	1014
29.164.3.73	ReverseElements(GeneralMatrix *)	1014
29.164.3.74	SameStorageType(const GeneralMatrix &A) const	1014
29.164.3.75	Search(const BaseMatrix *) const	1015
29.164.3.76	SetParameters(const GeneralMatrix *)	1015
29.164.3.77	SimpleAddOK(const GeneralMatrix *gm)	1015

29.164.3.7	Solver(MatrixColX &, const MatrixColX &)	1015
29.164.3.7	Storage() const	1015
29.164.3.8	Store() const	1015
29.164.3.8	Sum() const	1015
29.164.3.8	SumAbsoluteValue() const	1016
29.164.3.8	SumSquare() const	1016
29.164.3.8	Tag() const	1016
29.164.3.8	Delete()	1016
29.164.3.8	Transpose(TransposedMatrix *, MatrixType)	1016
29.164.3.8	Type() const =0	1016
29.164.4	Friends And Related Function Documentation	1016
29.164.4.1	AddedMatrix	1016
29.164.4.2	BandMatrix	1017
29.164.4.3	BaseMatrix	1017
29.164.4.4	ColedMatrix	1017
29.164.4.5	ColumnVector	1017
29.164.4.6	ConcatenatedMatrix	1017
29.164.4.7	CroutMatrix	1017
29.164.4.8	DiagedMatrix	1017
29.164.4.9	DiagonalMatrix	1017
29.164.4.10	GenericMatrix	1017
29.164.4.10	GetSubMatrix	1017
29.164.4.11	InvertedMatrix	1018
29.164.4.11	IPMatrix	1018
29.164.4.11	LinearEquationSolver	1018
29.164.4.11	LowerBandMatrix	1018
29.164.4.11	LowerTriangularMatrix	1018
29.164.4.11	MatedMatrix	1018
29.164.4.11	Matrix	1018
29.164.4.11	MultipliedMatrix	1018

29.164.4.20	DelegatedMatrix	1018
29.164.4.21	LegShiftedMatrix	1018
29.164.4.22	icMatrix	1019
29.164.4.23	ReturnMatrixX	1019
29.164.4.24	ReversedMatrix	1019
29.164.4.25	RowedMatrix	1019
29.164.4.26	RowVector	1019
29.164.4.27	caledMatrix	1019
29.164.4.28	hiftedMatrix	1019
29.164.4.29	olvedMatrix	1019
29.164.4.30	PMatrix	1019
29.164.4.31	StackedMatrix	1019
29.164.4.32	ubtractedMatrix	1020
29.164.4.33	ymmetricBandMatrix	1020
29.164.4.34	ymmetricMatrix	1020
29.164.4.35	ransposedMatrix	1020
29.164.4.36	pperBandMatrix	1020
29.164.4.37	pperTriangularMatrix	1020
29.164.5	Member Data Documentation	1020
29.164.5.1	ncols	1020
29.164.5.2	nrows	1020
29.164.5.3	storage	1020
29.164.5.4	store	1020
29.164.5.5	tag	1021
29.165	Go::GeneralMesh Class Reference	1021
29.165.1	Detailed Description	1022
29.165.2	Constructor & Destructor Documentation	1022
29.165.2.1	GeneralMesh()	1022
29.165.2.2	~GeneralMesh()	1022
29.165.3	Member Function Documentation	1022

29.165.3.1asGenericTriMesh()	1022
29.165.3.2asLineStrip()	1022
29.165.3.3asRegularMesh()	1022
29.165.3.4atBoundary(int idx)=0	1022
29.165.3.5numTriangles()	1023
29.165.3.6numVertices()=0	1023
29.165.3.7paramArray()=0	1023
29.165.3.8translate(const std::vector< double > &vert_translation)	1023
29.165.3.9triangleIndexArray()=0	1023
29.165.3.10VertexArray()=0	1023
29.165.4Member Data Documentation	1023
29.165.4.1vert_translation_	1023
29.166NEWMAT::GenericMatrix Class Reference	1024
29.166.1Detailed Description	1025
29.166.2Constructor & Destructor Documentation	1025
29.166.2.1GenericMatrix()	1025
29.166.2.2GenericMatrix(const BaseMatrix &bm)	1025
29.166.2.3GenericMatrix(const GenericMatrix &bm)	1025
29.166.2.4~GenericMatrix()	1025
29.166.3Member Function Documentation	1026
29.166.3.1BandWidth() const	1026
29.166.3.2CleanUp()	1026
29.166.3.3Evaluate(MatrixType=MatrixTypeUnSp)	1026
29.166.3.4operator&=(const BaseMatrix &)	1026
29.166.3.5operator*=(const BaseMatrix &)	1026
29.166.3.6operator*=(Real)	1026
29.166.3.7operator+=(const BaseMatrix &)	1026
29.166.3.8operator+=(Real)	1026
29.166.3.9operator-=(const BaseMatrix &)	1026
29.166.3.10operator-=(Real r)	1026

29.166.3.1	operator==(Real r)	1026
29.166.3.1	operator==(const GenericMatrix &)	1027
29.166.3.1	operator==(const BaseMatrix &)	1027
29.166.3.1	operator" ==(const BaseMatrix &)	1027
29.166.3.1	Release()	1027
29.166.4	Friends And Related Function Documentation	1027
29.166.4.1	BaseMatrix	1027
29.167	Go::GenericTriMesh Class Reference	1027
29.167.1	Detailed Description	1029
29.167.2	Constructor & Destructor Documentation	1029
29.167.2.1	GenericTriMesh(int num_vert=0, int num_tri=0, bool use_normals=true, bool use_texcoords=false)	1029
29.167.2.2	~GenericTriMesh()	1029
29.167.3	Member Function Documentation	1029
29.167.3.1	asGenericTriMesh()	1029
29.167.3.2	atBoundary(int idx)	1029
29.167.3.3	boundaryArray()	1029
29.167.3.4	normalArray()	1029
29.167.3.5	numTriangles()	1030
29.167.3.6	numVertices()	1030
29.167.3.7	paramArray()	1030
29.167.3.8	resize(int num_vert, int num_tri)	1030
29.167.3.9	texcoordArray()	1030
29.167.3.10	triangleIndexArray()	1030
29.167.3.11	useNormals()	1030
29.167.3.12	useTexCoords()	1030
29.167.3.13	vertexArray()	1031
29.168	Go::GeomObject Class Reference	1031
29.168.1	Detailed Description	1032
29.168.2	Constructor & Destructor Documentation	1032
29.168.2.1	~GeomObject()	1032

29.168.3	Member Function Documentation	1032
29.168.3.1	boundingBox() const =0	1032
29.168.3.2	classType()	1032
29.168.3.3	clone() const =0	1033
29.168.3.4	dimension() const =0	1033
29.168.3.5	instanceType() const =0	1033
29.168.3.6	writeStandardHeader(std::ostream &os) const	1033
29.169	Geo::GeomObjectInt Class Reference	1034
29.169.1	Detailed Description	1034
29.169.2	Constructor & Destructor Documentation	1034
29.169.2.1	~GeomObjectInt()	1034
29.170	GeoTol Class Reference	1034
29.170.1	Detailed Description	1035
29.170.2	Constructor & Destructor Documentation	1035
29.170.2.1	GeoTol(double epsge, double rel_par_res=1.0e-12, double numerical_tol=1.0e-16)	1035
29.170.2.2	GeoTol(GeoTol *epsge)	1035
29.170.2.3	~GeoTol()	1035
29.170.3	Member Function Documentation	1036
29.170.3.1	getAngleTol() const	1036
29.170.3.2	getEpsBracket() const	1036
29.170.3.3	getEpsge() const	1036
29.170.3.4	getMaxEpsge() const	1036
29.170.3.5	getMinEpsge() const	1036
29.170.3.6	getNumericalTol() const	1036
29.170.3.7	getRefAng() const	1036
29.170.3.8	getRelParRes() const	1036
29.170.3.9	read(std::istream &is)	1037
29.170.3.10	write(std::ostream &os) const	1037
29.171	NEWMAT::GetSubMatrix Class Reference	1037
29.171.1	Detailed Description	1039

29.171.2	Constructor & Destructor Documentation	1039
29.171.2.1	GetSubMatrix(const GetSubMatrix &g)	1039
29.171.2.2	~GetSubMatrix()	1039
29.171.3	Member Function Documentation	1039
29.171.3.1	BandWidth() const	1039
29.171.3.2	Evaluate(MatrixType mt=MatrixTypeUnSp)	1039
29.171.3.3	Inject(const GeneralMatrix &)	1039
29.171.3.4	operator*=(Real)	1039
29.171.3.5	operator+=(const BaseMatrix &)	1040
29.171.3.6	operator+=(Real)	1040
29.171.3.7	operator-=(const BaseMatrix &)	1040
29.171.3.8	operator-=(Real r)	1040
29.171.3.9	operator/=(Real r)	1040
29.171.3.10	operator<<(const BaseMatrix &)	1040
29.171.3.11	operator<<(const Real *)	1040
29.171.3.12	operator<<(Real)	1040
29.171.3.13	operator<<(int f)	1040
29.171.3.14	operator=(const BaseMatrix &)	1040
29.171.3.15	operator=(const GetSubMatrix &m)	1040
29.171.3.16	operator=(Real)	1040
29.171.4	Friends And Related Function Documentation	1040
29.171.4.1	BaseMatrix	1040
29.172	go_iterator_traits< Iterator > Struct Template Reference	1041
29.172.1	Detailed Description	1041
29.172.2	Member Typedef Documentation	1041
29.172.2.1	difference_type	1041
29.172.2.2	pointer	1041
29.172.2.3	reference	1041
29.172.2.4	value_type	1041
29.173	go_iterator_traits< const T * > Struct Template Reference	1042

29.173.1	Detailed Description	1042
29.173.2	Member Typedef Documentation	1042
29.173.2.1	difference_type	1042
29.173.2.2	pointer	1042
29.173.2.3	reference	1042
29.173.2.4	value_type	1042
29.174	Go::go_iterator_traits< T * > Struct Template Reference	1042
29.174.1	Detailed Description	1043
29.174.2	Member Typedef Documentation	1043
29.174.2.1	difference_type	1043
29.174.2.2	pointer	1043
29.174.2.3	reference	1043
29.174.2.4	value_type	1043
29.175	Go::GoTools Class Reference	1043
29.175.1	Detailed Description	1044
29.175.2	Constructor & Destructor Documentation	1044
29.175.2.1	GoTools()	1044
29.175.2.2	~GoTools()	1044
29.175.3	Member Function Documentation	1044
29.175.3.1	className(ClassType class_type)	1044
29.175.3.2	nit()	1044
29.175.3.3	knotEpsilon()	1045
29.175.3.4	parameterEpsilon()	1045
29.175.3.5	setKnotEpsilon(double knot_epsilon)	1045
29.175.3.6	setParameterEpsilon(double parameter_epsilon)	1045
29.175.3.7	setSpaceEpsilon(double space_epsilon)	1045
29.175.3.8	spaceEpsilon()	1045
29.176	Go::GPos Struct Reference	1045
29.176.1	Detailed Description	1046
29.176.2	Constructor & Destructor Documentation	1046

29.176.2.1GPos(int i, int m)	1046
29.176.2.2GPos()	1046
29.176.3Member Data Documentation	1046
29.176.3.1ix	1046
29.176.3.2mult	1046
29.177Go::GpuMatrix< T > Class Template Reference	1046
29.177.1Detailed Description	1047
29.177.2Constructor & Destructor Documentation	1047
29.177.2.1GpuMatrix(int rows, int cols)	1047
29.177.2.2GpuMatrix(int rows, int cols, T *data)	1047
29.177.2.3GpuMatrix(int rows, int cols, T initializer)	1047
29.177.2.4~GpuMatrix()	1047
29.177.3Member Function Documentation	1047
29.177.3.1download(T *data)	1047
29.177.3.2getCols()	1047
29.177.3.3getRows()	1047
29.177.3.4upload(T *data)	1047
29.177.4Friends And Related Function Documentation	1048
29.177.4.1operator<<	1048
29.178gvActionSheet Class Reference	1048
29.178.1Detailed Description	1049
29.178.2Constructor & Destructor Documentation	1049
29.178.2.1~gvActionSheet()	1049
29.178.3Member Function Documentation	1049
29.178.3.1showDialog(gvObserver *observer)=0	1049
29.178.4Member Data Documentation	1049
29.178.4.1form_	1049
29.179gvApplication Class Reference	1049
29.179.1Detailed Description	1051
29.179.2Constructor & Destructor Documentation	1051

29.179.2.1gvApplication(std::auto_ptr< DataHandler > dh, QWidget *parent=0, const char *name=0, Qt::WFlags f=0)	1051
29.179.2.2~gvApplication()	1052
29.179.3Member Function Documentation	1052
29.179.3.1about	1052
29.179.3.2add_group	1052
29.179.3.3add_objects	1052
29.179.3.4assign_texture	1052
29.179.3.5buildGUI()	1052
29.179.3.6changeCurveResolutions	1052
29.179.3.7changeSurfaceResolutions	1052
29.179.3.8close_document	1052
29.179.3.9createObjectToggleBox()	1052
29.179.3.10disable_objects	1052
29.179.3.11dismiss_selections	1052
29.179.3.12display_object_properties	1052
29.179.3.13enable_objects	1052
29.179.3.14getSelectedObjects(std::vector< shared_ptr< Go::GeomObject > > &sel_objs, std::vector< shared_ptr< Go::GeomObject > > ¬_sel_objs)	1052
29.179.3.15group_selected	1052
29.179.3.16open	1053
29.179.3.17quit	1053
29.179.3.18load_last_opened_file	1053
29.179.3.19save_selection_as	1053
29.179.3.20select_all	1053
29.179.3.21select_all_curves	1053
29.179.3.22select_all_surfaces	1053
29.179.3.23select_all_visible	1053
29.179.3.24select_inverse	1053
29.179.3.25select_none	1053
29.179.3.26set_curve_resolutions	1053

29.179.3.27	set_random_color	1053
29.179.3.28	set_surface_resolutions	1053
29.179.3.29	show_control_nets	1053
29.179.3.30	sizeHint() const	1053
29.179.3.31	toggle_blending_mode	1053
29.179.3.32	toggle_enable	1053
29.179.3.33	toggle_multiselect_object	1053
29.179.3.34	toggle_select_object	1053
29.179.3.35	toggle_selection_mode	1053
29.179.3.36	view_axis	1053
29.179.3.37	view_cull	1053
29.179.3.38	view_focus_point	1053
29.179.3.39	view_focus_point_cb	1054
29.179.3.40	view_orthographic	1054
29.179.3.41	view_reset	1054
29.179.3.42	view_reset_visible	1054
29.179.3.43	view_specular	1054
29.179.3.44	view_wireframe	1054
29.179.4	Member Data Documentation	1054
29.179.4.1	actionForm	1054
29.179.4.2	app_name_	1054
29.179.4.3	curr_file_type_	1054
29.179.4.4	data_	1054
29.179.4.5	group_menu_	1054
29.179.4.6	last_file_name_	1054
29.179.4.7	menu_	1054
29.179.4.8	object_menu_	1055
29.179.4.9	select_menu_	1055
29.179.4.10	view_	1055
29.179.4.11	view_menu_	1055

29.180	gvApplicationVolAndLR Class Reference	1055
29.180.1	Detailed Description	1056
29.180.2	Constructor & Destructor Documentation	1057
29.180.2.1	gvApplicationVolAndLR(std::auto_ptr< DataHandler > dh, QWidget *parent=0, const char *name=0, Qt::WFlags f=0)	1057
29.180.2.2	~gvApplicationVolAndLR()	1057
29.180.3	Member Function Documentation	1057
29.180.3.1	buildExtraGUI()	1057
29.180.3.2	move_vertices_to_origin	1057
29.180.3.3	translate_to_origin	1057
29.180.3.4	view_reset	1057
29.181	gvCamera Class Reference	1057
29.181.1	Detailed Description	1059
29.181.2	Constructor & Destructor Documentation	1059
29.181.2.1	gvCamera(const Vector3D &point, int x, int y, int w, int h)	1059
29.181.2.2	~gvCamera()	1059
29.181.3	Member Function Documentation	1059
29.181.3.1	computeWinz()	1059
29.181.3.2	eyeCoordsToObjectCoords(const Vector3D &eyep) const	1059
29.181.3.3	getAxisSize(double &size) const	1059
29.181.3.4	getDistance(double &d) const	1059
29.181.3.5	getFocalPoint(Vector3D &point) const	1059
29.181.3.6	getPerspectiveMode() const	1059
29.181.3.7	getViewPort(int &x, int &y, int &w, int &h) const	1060
29.181.3.8	getWorldRay() const	1060
29.181.3.9	getWorldSide() const	1060
29.181.3.10	getWorldUp() const	1060
29.181.3.11	initializeGL()	1060
29.181.3.12	moveFocalPointRelative(const Vector3D &eyep) const	1060
29.181.3.13	objectCoordsToWindowCoords(const Vector3D &pt) const	1060
29.181.3.14	tick(int x, int y, int w, int h) const	1060

29.181.3.15	setRotation()	1060
29.181.3.16	rotate(double axisx, double axisy, double angle)	1060
29.181.3.17	rotateTransversal(double angle)	1060
29.181.3.18	setAxisSize(double size)	1060
29.181.3.19	setDistance(double d)	1061
29.181.3.20	setFocalPoint(const Vector3D &fpoint)	1061
29.181.3.21	setPerspectiveMode(bool perspective_mode)	1061
29.181.3.22	setViewPort(int x, int y, int w, int h)	1061
29.181.3.23	use() const	1061
29.181.3.24	useModelView() const	1061
29.181.4	Member Data Documentation	1061
29.181.4.1	axisize_	1061
29.181.4.2	distance_	1061
29.181.4.3	focal_point_	1062
29.181.4.4	mv_	1062
29.181.4.5	perspective_mode_	1062
29.181.4.6	viewfield_	1062
29.181.4.7	viewheight_	1062
29.181.4.8	viewwidth_	1062
29.181.4.9	viewx_	1062
29.181.4.10	viewy_	1062
29.181.4.11	winz_	1062
29.182	Color Struct Reference	1063
29.182.1	Detailed Description	1063
29.182.2	Constructor & Destructor Documentation	1063
29.182.2.1	Color()	1063
29.182.2.2	Color(NumericType r, NumericType g, NumericType b)	1063
29.182.2.3	Color(NumericType r, NumericType g, NumericType b, NumericType alpha)	1064
29.182.2.4	Color(const Color &other)	1064
29.182.3	Member Function Documentation	1064

29.182.3.1hsva(NumericType h, NumericType s, NumericType v, NumericType alpha) . . .	1064
29.182.3.2operator=(const gvColor &other)	1064
29.182.4Member Data Documentation	1064
29.182.4.1rgba	1064
29.183gvCurvePaintable Class Reference	1064
29.183.1Detailed Description	1065
29.183.2Constructor & Destructor Documentation	1065
29.183.2.1gvCurvePaintable(Go::LineStrip &line, const gvColor &ncolor, const gvColor &scolor, int id)	1065
29.183.2.2gvCurvePaintable(Go::LineStrip &line, const gvColor &ncolor, int id)	1066
29.183.2.3~gvCurvePaintable()	1066
29.183.3Member Function Documentation	1066
29.183.3.1drawPoints()	1066
29.183.3.2paint(gvTexture *texture)	1066
29.183.3.3setDrawPoints(bool dp)	1066
29.183.4Member Data Documentation	1066
29.183.4.1draw_pts_	1066
29.183.4.2line_	1066
29.184gvData Class Reference	1067
29.184.1Detailed Description	1068
29.184.2Constructor & Destructor Documentation	1068
29.184.2.1gvData(std::auto_ptr< DataHandler > datahandler)	1068
29.184.2.2~gvData()	1068
29.184.3Member Function Documentation	1068
29.184.3.1addGroup(gvGroup group)	1068
29.184.3.2boundingBox()	1068
29.184.3.3boundingBox(const std::vector< int > &objs) const	1069
29.184.3.4clear()	1069
29.184.3.5clearGroup()	1069
29.184.3.6clearLast()	1069
29.184.3.7color(int i)	1069

29.184.3.8	<code>deleteObj(int obj)</code>	1069
29.184.3.9	<code>disableUpdates()</code>	1069
29.184.3.10	<code>enableUpdates()</code>	1069
29.184.3.11	<code>extractSelectedObjects(std::vector< shared_ptr< Go::GeomObject > > &sel_objs)</code>	1069
29.184.3.12	<code>getGroup(int index) const</code>	1069
29.184.3.13	<code>getSelectedStateObject(int id)</code>	1069
29.184.3.14	<code>getTexture(int index)</code>	1069
29.184.3.15	<code>setVisibleStateObject(int id)</code>	1069
29.184.3.16	<code>nbGroups() const</code>	1069
29.184.3.17	<code>numObjects() const</code>	1070
29.184.3.18	<code>object(int i)</code>	1070
29.184.3.19	<code>object(int i) const</code>	1070
29.184.3.20	<code>objectGroups() const</code>	1070
29.184.3.21	<code>painter()</code>	1070
29.184.3.22	<code>propertySheet(int i)</code>	1070
29.184.3.23	<code>readGo(std::istream &is)</code>	1070
29.184.3.24	<code>readGo(const std::vector< shared_ptr< Go::GeomObject > > &new_objects, std::vector< shared_ptr< gvColor > > &new_colors)</code>	1070
29.184.3.25	<code>readIges(std::istream &is)</code>	1070
29.184.3.26	<code>readSisCrvs(std::istream &is)</code>	1070
29.184.3.27	<code>readSisSrfS(std::istream &is)</code>	1070
29.184.3.28	<code>createDataStructure(int nmb_new_objs)</code>	1070
29.184.3.29	<code>registerObserver(gvObserver *obs)</code>	1070
29.184.3.30	<code>setSelectedStateObject(int id, bool state)</code>	1071
29.184.3.31	<code>setTexture(int index, shared_ptr< gvTexture > tex)</code>	1071
29.184.3.32	<code>setVisibleStateObject(int id, bool state)</code>	1071
29.184.3.33	<code>setSelector(int index)</code>	1071
29.184.3.34	<code>updateObservers()</code>	1071
29.184.3.35	<code>writeSelectedGo(std::ostream &os)</code>	1071
29.184.3.36	<code>writeSelectedIges(std::ostream &os)</code>	1071
29.184.4	Member Data Documentation	1071

29.184.4.1object_colors_	1071
29.184.4.2objects_	1071
29.184.4.3painter_	1071
29.185.1gvGenericTriPaintable< FloatType > Class Template Reference	1072
29.185.1Detailed Description	1073
29.185.2Constructor & Destructor Documentation	1073
29.185.2.1gvGenericTriPaintable(GenericTriMesh< FloatType > &tri, const gvColor &ncolor, const gvColor &scolor, int id)	1073
29.185.2.2gvGenericTriPaintable(GenericTriMesh< FloatType > &tri, const gvColor &ncolor, int id)	1073
29.185.2.3~gvGenericTriPaintable()	1073
29.185.3Member Function Documentation	1073
29.185.3.1paint(gvTexture *)	1073
29.185.4Member Data Documentation	1074
29.185.4.1tri_	1074
29.186.1gvGenericTriQuadMesh< FloatType > Class Template Reference	1074
29.186.1Detailed Description	1074
29.186.2Constructor & Destructor Documentation	1074
29.186.2.1gvGenericTriQuadMesh(int num_vert, int num_tri, int num_quads, bool use_↔ normals=true, bool use_texcoords=false)	1074
29.186.3Member Function Documentation	1075
29.186.3.1normalArray()	1075
29.186.3.2numQuads()	1075
29.186.3.3numTriangles()	1075
29.186.3.4numVertices()	1075
29.186.3.5quadIndexArray()	1075
29.186.3.6resize(int num_vert, int num_tri, int num_quads)	1075
29.186.3.7texcoordArray()	1075
29.186.3.8triangleIndexArray()	1075
29.186.3.9useNormals()	1076
29.186.3.10useTexCoords()	1076
29.186.3.11vertexArray()	1076

29.187	gvGenericTriQuadPaintable< FloatType > Class Template Reference	1076
29.187.1	Detailed Description	1077
29.187.2	Constructor & Destructor Documentation	1077
29.187.2.1	gvGenericTriQuadPaintable(gvGenericTriQuadMesh< FloatType > &mesh, const gvColor &ncolor, const gvColor &scolor, int id)	1077
29.187.2.2	gvGenericTriQuadPaintable(gvGenericTriQuadMesh< FloatType > &mesh, const gvColor &ncolor, int id)	1078
29.187.2.3	~gvGenericTriQuadPaintable()	1078
29.187.3	Member Function Documentation	1078
29.187.3.1	paint()	1078
29.187.4	Member Data Documentation	1078
29.187.4.1	mesh_	1078
29.188	gvGroup Class Reference	1078
29.188.1	Detailed Description	1079
29.188.2	Constructor & Destructor Documentation	1079
29.188.2.1	gvGroup(std::vector< int > &members, QString name)	1079
29.188.2.2	~gvGroup()	1079
29.188.3	Member Function Documentation	1079
29.188.3.1	name() const	1079
29.188.3.2	operator[](int index) const	1079
29.188.3.3	size() const	1079
29.189	gvGroupPropertySheet Class Reference	1079
29.189.1	Detailed Description	1080
29.189.2	Constructor & Destructor Documentation	1080
29.189.2.1	gvGroupPropertySheet(std::vector< int > &members, QString &def_name)	1080
29.189.3	Member Function Documentation	1080
29.189.3.1	accept	1080
29.189.3.2	createSheet(QWidget *parent, gvObserver *obs)	1080
29.189.3.3	getMembers()	1081
29.189.3.4	value_changed	1081
29.190	gvLineCloudPaintable Class Reference	1081

29.190.1	Detailed Description	1082
29.190.2	Constructor & Destructor Documentation	1082
29.190.2.1	gvLineCloudPaintable(const Go::LineCloud &lc, const gvColor &ncolor, const gvColor &scolor, int id)	1082
29.190.2.2	gvLineCloudPaintable(const Go::LineCloud &lc, const gvColor &ncolor, int id)	1082
29.190.2.3	~gvLineCloudPaintable()	1082
29.190.3	Member Function Documentation	1082
29.190.3.1	fractionRendered()	1082
29.190.3.2	paint(gvTexture *)	1083
29.190.3.3	setFractionRendered(double f)	1083
29.190.4	Member Data Documentation	1083
29.190.4.1	fractionrendered_	1083
29.190.4.2	c_	1083
29.190	gvNoopPaintable Class Reference	1083
29.191.1	Detailed Description	1084
29.191.2	Constructor & Destructor Documentation	1084
29.191.2.1	gvNoopPaintable(const gvColor &ncolor, const gvColor &scolor, int id)	1084
29.191.2.2	gvNoopPaintable(const gvColor &ncolor, int id)	1084
29.191.2.3	~gvNoopPaintable()	1085
29.191.3	Member Function Documentation	1085
29.191.3.1	paint(gvTexture *texture)	1085
29.190	gvObjectList Class Reference	1085
29.192.1	Detailed Description	1086
29.192.2	Constructor & Destructor Documentation	1086
29.192.2.1	gvObjectList(gvData &data, QWidget *parent=0, const char *name=0, Qt::WindowFlags f=0)	1086
29.192.2.2	~gvObjectList()	1086
29.192.3	Member Function Documentation	1086
29.192.3.1	buildGUI()	1086
29.192.3.2	clicked	1086
29.192.3.3	observedChanged()	1086

29.193.0	gvObserver Class Reference	1087
29.193.1	Detailed Description	1087
29.193.2	Constructor & Destructor Documentation	1087
29.193.2.1	~gvObserver()	1087
29.193.3	Member Function Documentation	1087
29.193.3.1	observedChanged()=0	1087
29.194.0	gvPaintable Class Reference	1088
29.194.1	Detailed Description	1089
29.194.2	Constructor & Destructor Documentation	1090
29.194.2.1	gvPaintable(const gvColor &ncolor, const gvColor &scolor, int id)	1090
29.194.2.2	gvPaintable(const gvColor &ncolor, int id)	1090
29.194.2.3	~gvPaintable()	1090
29.194.3	Member Function Documentation	1090
29.194.3.1	getNormalColor()	1090
29.194.3.2	d() const	1090
29.194.3.3	paint(gvTexture *texture)=0	1090
29.194.3.4	paintGL(QGLWidget *w, gvTexture *texture)	1090
29.194.3.5	selected() const	1090
29.194.3.6	setColor(const gvColor &ncolor, const gvColor &scolor)	1091
29.194.3.7	setColor(const gvColor &ncolor)	1091
29.194.3.8	setId(int id)	1091
29.194.3.9	setSelected(bool state)	1091
29.194.3.10	setVisible(bool state)	1091
29.194.3.11	visible() const	1091
29.194.4	Member Data Documentation	1091
29.194.4.1	id_	1091
29.194.4.2	normal_color_	1091
29.194.4.3	selected_	1091
29.194.4.4	selected_color_	1091
29.194.4.5	visible_	1092

29.195	gvPainter Class Reference	1092
29.195.1	Detailed Description	1092
29.195.2	Constructor & Destructor Documentation	1092
29.195.2.1	gvPainter()	1092
29.195.2.2	~gvPainter()	1093
29.195.3	Member Function Documentation	1093
29.195.3.1	addPaintable(shared_ptr< gvPaintable > pa)	1093
29.195.3.2	drawScene(QGLWidget *w=NULL)	1093
29.195.3.3	getPaintable(int index)	1093
29.195.3.4	getTexture(int index)	1093
29.195.3.5	removeAllPaintables()	1093
29.195.3.6	removeLastPaintable()	1093
29.195.3.7	removePaintable(int id)	1093
29.195.3.8	setTexture(int index, shared_ptr< gvTexture > tex)	1093
29.195.3.9	setTexture(int index)	1094
29.195.4	Member Data Documentation	1094
29.195.4.1	paintables_	1094
29.195.4.2	textures_	1094
29.196	gvParametricSurfacePaintable Class Reference	1094
29.196.1	Detailed Description	1095
29.196.2	Constructor & Destructor Documentation	1095
29.196.2.1	gvParametricSurfacePaintable(genMesh &tri, const gvColor &ncolor, const gvColor &scolor, int id)	1095
29.196.2.2	gvParametricSurfacePaintable(genMesh &tri, const gvColor &ncolor, int id)	1096
29.196.2.3	~gvParametricSurfacePaintable()	1096
29.196.3	Member Function Documentation	1096
29.196.3.1	createSurface()	1096
29.196.3.2	drawSurface()	1096
29.196.3.3	paint(gvTexture *texture)	1096
29.196.4	Member Data Documentation	1096
29.196.4.1	tri_	1096

29.197	gvPointCloudPaintable Class Reference	1096
29.197.1	Detailed Description	1098
29.197.2	Constructor & Destructor Documentation	1098
29.197.2.1	gvPointCloudPaintable(const Go::PointCloud3D &pc, const gvColor &ncolor, const gvColor &scolor, int id, bool paintId=false)	1098
29.197.2.2	gvPointCloudPaintable(const Go::PointCloud3D &pc, const gvColor &ncolor, int id, bool paintId=false)	1098
29.197.2.3	~gvPointCloudPaintable()	1098
29.197.3	Member Function Documentation	1098
29.197.3.1	fractionRendered()	1098
29.197.3.2	getPaintId() const	1098
29.197.3.3	paint(gvTexture *)	1098
29.197.3.4	paintGL(QGLWidget *w, gvTexture *texture)	1099
29.197.3.5	pointSize()	1099
29.197.3.6	setFractionRendered(double f)	1099
29.197.3.7	setPaintId(bool paintId)	1099
29.197.3.8	setPointSize(double sz)	1099
29.197.3.9	translate(const std::vector< double > &vert_translation)	1099
29.197.4	Member Data Documentation	1099
29.197.4.1	fractionrendered_	1099
29.197.4.2	paintId_	1099
29.197.4.3	pc_	1099
29.197.4.4	pointsize_	1100
29.197.4.5	vert_translation_	1100
29.198	gvPropertySheet Class Reference	1100
29.198.1	Detailed Description	1101
29.198.2	Constructor & Destructor Documentation	1101
29.198.2.1	~gvPropertySheet()	1101
29.198.3	Member Function Documentation	1101
29.198.3.1	createSheet(QWidget *parent, gvObserver *observer)=0	1101
29.199	gvQuadsPaintable Class Reference	1101

29.199.1	Detailed Description	1102
29.199.2	Constructor & Destructor Documentation	1102
29.199.2.1	gvQuadsPaintable(Go::QuadMesh &quads, const gvColor &ncolor, const gvColor &scolor, int id)	1102
29.199.2.2	gvQuadsPaintable(Go::QuadMesh &quads, const gvColor &ncolor, int id)	1103
29.199.2.3	~gvQuadsPaintable()	1103
29.199.3	Member Function Documentation	1103
29.199.3.1	paint(gvTexture *texture)	1103
29.199.4	Member Data Documentation	1103
29.199.4.1	quads_	1103
29.200	gvRectangularSurfacePaintable Class Reference	1103
29.200.1	Detailed Description	1104
29.200.2	Constructor & Destructor Documentation	1104
29.200.2.1	gvRectangularSurfacePaintable(Go::RegularMesh &tri, const gvColor &ncolor, const gvColor &scolor, int id)	1104
29.200.2.2	gvRectangularSurfacePaintable(Go::RegularMesh &tri, const gvColor &ncolor, int id)	1105
29.200.2.3	~gvRectangularSurfacePaintable()	1105
29.200.3	Member Function Documentation	1105
29.200.3.1	paint(gvTexture *texture)	1105
29.200.4	Member Data Documentation	1105
29.200.4.1	tri_	1105
29.201	gvRectangularVolumePaintable Class Reference	1105
29.201.1	Detailed Description	1106
29.201.2	Constructor & Destructor Documentation	1106
29.201.2.1	gvRectangularVolumePaintable(Go::RegularVolMesh &tri, const gvColor &ncolor, const gvColor &scolor, int id)	1106
29.201.2.2	gvRectangularVolumePaintable(Go::RegularVolMesh &tri, const gvColor &ncolor, int id)	1107
29.201.2.3	~gvRectangularVolumePaintable()	1107
29.201.3	Member Function Documentation	1107
29.201.3.1	paint(gvTexture *texture)	1107
29.201.4	Member Data Documentation	1107

29.201.4.1	tri_	1107
29.202	gvResolutionDialog Class Reference	1107
29.202.1	Detailed Description	1108
29.202.2	Constructor & Destructor Documentation	1109
29.202.2.1	gvResolutionDialog(int current_res_u, int current_res_v, int minimum_res, int maximum_res, QWidget *parent=0, const char *name=0, bool modal=FALSE, Qt::WFlags f=0)	1109
29.202.2.2	~gvResolutionDialog()	1109
29.202.3	Member Function Documentation	1109
29.202.3.1	accept	1109
29.202.3.2	apply	1109
29.202.3.3	valuesChanged	1109
29.202.4	Member Data Documentation	1109
29.202.4.1	ures_	1109
29.202.4.2	uslide_	1109
29.202.4.3	res_	1109
29.202.4.4	vslide_	1109
29.203	gvStandardMouseHandler Class Reference	1109
29.203.1	Detailed Description	1110
29.203.2	Constructor & Destructor Documentation	1110
29.203.2.1	gvStandardMouseHandler()	1110
29.203.2.2	~gvStandardMouseHandler()	1110
29.204	gvTexture Class Reference	1110
29.204.1	Detailed Description	1111
29.204.2	Constructor & Destructor Documentation	1111
29.204.2.1	gvTexture(bool mipmapped=false)	1111
29.204.2.2	gvTexture(const gvTexture &other)	1111
29.204.2.3	gvTexture(int height, int width, unsigned char *pixels, bool mipmapped=false)	1111
29.204.2.4	gvTexture(std::string filename, bool mipmapped=false)	1111
29.204.2.5	~gvTexture()	1111
29.204.3	Member Function Documentation	1111

29.204.3.1	<code>bind() const</code>	1111
29.204.3.2	<code>genTexture() const</code>	1111
29.204.3.3	<code>getEnvMode()</code>	1111
29.204.3.4	<code>getMagFilter()</code>	1111
29.204.3.5	<code>getMinFilter()</code>	1111
29.204.3.6	<code>getImage(QImage &res, bool withAlpha=false) const</code>	1111
29.204.3.7	<code>height() const</code>	1111
29.204.3.8	<code>operator=(const gvTexture &other)</code>	1111
29.204.3.9	<code>readFile(std::string filename)</code>	1111
29.204.3.10	<code>setAlphaValue(int val)</code>	1111
29.204.3.11	<code>setCenterEdgeTexels(bool enable)</code>	1111
29.204.3.12	<code>setEnvMode(EnvModeSet mode, bool flush=true)</code>	1112
29.204.3.13	<code>setIdentMatrix()</code>	1112
29.204.3.14	<code>setMagFilter(MagFilterSet filter, bool flush=true)</code>	1112
29.204.3.15	<code>setMinFilter(MinFilterSet filter, bool flush=true)</code>	1112
29.204.3.16	<code>setTextureMatrix(const double tm[])</code>	1112
29.204.3.17	<code>setWrapMode(WrapModeSet mode, bool flush=true)</code>	1112
29.204.3.18	<code>width() const</code>	1112
29.205	<code>gvView Class Reference</code>	1112
29.205.1	Detailed Description	1115
29.205.2	Constructor & Destructor Documentation	1115
29.205.2.1	<code>gvView(gvData &data, QWidget *parent=0, const char *name=0, const QGLWidget *shareWidget=0, Qt::WFlags f=0)</code>	1115
29.205.2.2	<code>~gvView()</code>	1115
29.205.2.3	<code>gvView(const QGLFormat &format, gvData &data, QWidget *parent, const char *name, const QGLWidget *shareWidget, Qt::WFlags f)</code>	1115
29.205.3	Member Function Documentation	1115
29.205.3.1	<code>axis()</code>	1115
29.205.3.2	<code>backCull()</code>	1116
29.205.3.3	<code>blendingmode() const</code>	1116
29.205.3.4	<code>drawOverlay()</code>	1116

29.205.3.5	feedback	1116
29.205.3.6	feedbackmode() const	1116
29.205.3.7	focusOnBox()	1116
29.205.3.8	focusOnVisible()	1116
29.205.3.9	get3Dpoint(int mousex, int mousey, Vector3D &objpt)	1116
29.205.3.10	getObjAndParam(int mousex, int mousey, shared_ptr< const Go::ParamSurface > &obj, double &tex_u, double &tex_v)	1116
29.205.3.11	getWindowCoords(const Vector3D &pt, int &mousex, int &mousey) const	1116
29.205.3.12	initializeGL()	1116
29.205.3.13	makeCoarseCheckImage()	1117
29.205.3.14	makeFineCheckImage()	1117
29.205.3.15	minimumSizeHint() const	1117
29.205.3.16	mouseMoveEvent(QMouseEvent *e)	1117
29.205.3.17	mousePressEvent(QMouseEvent *e)	1117
29.205.3.18	mouseReleaseEvent(QMouseEvent *e)	1117
29.205.3.19	objectPicked	1117
29.205.3.20	objectsPicked	1117
29.205.3.21	observedChanged()	1117
29.205.3.22	paintGL()	1117
29.205.3.23	perspective()	1117
29.205.3.24	pick(int mousex, int mousey)	1118
29.205.3.25	pickRegion(int mousex, int mousey, int w, int h)	1118
29.205.3.26	resizeGL(int w, int h)	1118
29.205.3.27	saveSnapshot(int w, int h, const QString &filename)	1118
29.205.3.28	selectionmode()	1118
29.205.3.29	setAxis	1118
29.205.3.30	setBackCull	1118
29.205.3.31	setBlendingmode	1118
29.205.3.32	setCenter	1118
29.205.3.33	setFeedbackmode	1118
29.205.3.34	setGetClickmode	1119

29.205.3.35	setOriginFocalPoint(bool origin)	1119
29.205.3.36	Perspective	1119
29.205.3.37	SelectionMode	1119
29.205.3.38	Specular	1119
29.205.3.39	Wireframe	1119
29.205.3.40	SizeHint() const	1119
29.205.3.41	Specular()	1119
29.205.3.42	CoordInt(int num_objs, int ind, double &min_s, double &max_s, double &min_t, double &max_t)	1119
29.205.3.43	Wireframe()	1119
29.205.4	Member Data Documentation	1120
29.205.4.1	axis_	1120
29.205.4.2	backcull_	1120
29.205.4.3	base_axis_size_	1120
29.205.4.4	blending_mode_	1120
29.205.4.5	box_	1120
29.205.4.6	camera_	1120
29.205.4.7	coarseTex_	1120
29.205.4.8	data_	1120
29.205.4.9	draglength_	1120
29.205.4.10	feedback_mode_	1120
29.205.4.11	fineTex_	1121
29.205.4.12	focus_on_origin_	1121
29.205.4.13	get_click_mode_	1121
29.205.4.14	initialized_	1121
29.205.4.15	last_mouse_pos_	1121
29.205.4.16	lights_camera_	1121
29.205.4.17	mouse_button_	1121
29.205.4.18	mouse_is_active_	1121
29.205.4.19	no_data_	1121
29.205.4.20	painter_	1121

29.205.4.21	<code>selecting_</code>	1122
29.205.4.22	<code>selection_mode_</code>	1122
29.205.4.23	<code>specular_</code>	1122
29.205.4.24	<code>starting_mouse_pos_</code>	1122
29.205.4.25	<code>unitx_</code>	1122
29.205.4.26	<code>unity_</code>	1122
29.205.4.27	<code>wireframe_</code>	1122
29.206	<code>Handle< T ></code> Class Template Reference	1122
29.206.1	Detailed Description	1123
29.206.2	Constructor & Destructor Documentation	1123
29.206.2.1	<code>Handle()</code>	1123
29.206.2.2	<code>Handle(const T &ref)</code>	1123
29.206.2.3	<code>Handle(T *p)</code>	1124
29.206.2.4	<code>Handle(const Handle< T > &ref)</code>	1124
29.206.2.5	<code>~Handle()</code>	1124
29.206.3	Member Function Documentation	1124
29.206.3.1	<code>getPtr() const</code>	1124
29.206.3.2	<code>getPtr()</code>	1124
29.206.3.3	<code>getRef() const</code>	1124
29.206.3.4	<code>getRef()</code>	1124
29.206.3.5	<code>operator!=(const Handle< T > &h) const</code>	1124
29.206.3.6	<code>operator>() const</code>	1124
29.206.3.7	<code>operator>()</code>	1124
29.206.3.8	<code>operator*() const</code>	1125
29.206.3.9	<code>operator*()</code>	1125
29.206.3.10	<code>operator->() const</code>	1125
29.206.3.11	<code>operator->()</code>	1125
29.206.3.12	<code>operator<(const Handle< T > &h) const</code>	1125
29.206.3.13	<code>operator=(const Handle< T > &h)</code>	1125
29.206.3.14	<code>operator=(const T *p)</code>	1125

29.206.3.1	<code>operator=(const T &p)</code>	1125
29.206.3.1	<code>operator==(const Handle< T > &h) const</code>	1125
29.206.3.1	<code>operator>(const Handle< T > &h) const</code>	1125
29.206.3.1	<code>bind(const T *pc)</code>	1126
29.206.3.1	<code>bind(const T &p)</code>	1126
29.206.4	Member Data Documentation	1126
29.206.4.1	<code>classptr</code>	1126
29.207	HandleId Class Reference	1126
29.207.1	Detailed Description	1127
29.207.2	Constructor & Destructor Documentation	1127
29.207.2.1	<code>HandleId()</code>	1127
29.207.2.2	<code>~HandleId()</code>	1127
29.207.3	Member Function Documentation	1127
29.207.3.1	<code>decrement()</code>	1127
29.207.3.2	<code>dynamicObj() const</code>	1127
29.207.3.3	<code>getNoRefs() const</code>	1127
29.207.3.4	<code>increment()</code>	1127
29.207.3.5	<code>isReferenced() const</code>	1127
29.207.3.6	<code>operator delete(void *v)</code>	1128
29.207.3.7	<code>operator new(size_t t)</code>	1128
29.207.3.8	<code>operator new(size_t t, int, const char *file, int line)</code>	1128
29.207.4	Member Data Documentation	1128
29.207.4.1	<code>dynamic_object</code>	1128
29.207.4.2	<code>refcount</code>	1128
29.208	HeapNode Class Reference	1128
29.208.1	Detailed Description	1129
29.208.2	Constructor & Destructor Documentation	1129
29.208.2.1	<code>HeapNode()</code>	1129
29.208.2.2	<code>HeapNode(int idx, double key)</code>	1129
29.208.2.3	<code>~HeapNode()</code>	1129

29.208.3	Member Function Documentation	1129
29.208.3.1	operator>(const HeapNode &x) const	1129
29.208.3.2	operator>=(const HeapNode &x) const	1129
29.208.4	Member Data Documentation	1129
29.208.4.1	idx_	1129
29.208.4.2	key_	1130
29.209	HeapNode2 Class Reference	1130
29.209.1	Detailed Description	1130
29.209.2	Constructor & Destructor Documentation	1130
29.209.2.1	HeapNode2()	1130
29.209.2.2	HeapNode2(int idx, double key)	1130
29.209.2.3	~HeapNode2()	1130
29.209.3	Member Function Documentation	1131
29.209.3.1	operator>(const HeapNode2 &x) const	1131
29.209.3.2	operator>=(const HeapNode2 &x) const	1131
29.209.4	Member Data Documentation	1131
29.209.4.1	idx_	1131
29.209.4.2	key_	1131
29.210	Go::HermiteAppC Class Reference	1131
29.210.1	Detailed Description	1131
29.210.2	Constructor & Destructor Documentation	1131
29.210.2.1	HermiteAppC(EvalCurve *crv, double tolerance1, double tolerance2)	1131
29.210.2.2	HermiteAppC(EvalCurve *crv, double initpars[], int n, double tolerance1, double tolerance2)	1132
29.210.2.3	~HermiteAppC()	1132
29.210.3	Member Function Documentation	1132
29.210.3.1	getCurve()	1132
29.210.3.2	refineApproximation()	1132
29.211	Go::HermiteApprEvalSurf Class Reference	1133
29.211.1	Detailed Description	1133
29.211.2	Constructor & Destructor Documentation	1133

29.211.2.1	HermiteApprEvalSurf(EvalSurface *sf, double tolerance1, double tolerance2)	1133
29.211.2.2	HermiteApprEvalSurf(EvalSurface *sf, double initpars_u[], int mm, double initpars_v[], int nn, double tolerance1, double tolerance2)	1133
29.211.2.3	~HermiteApprEvalSurf()	1134
29.211.3	Member Function Documentation	1134
29.211.3.1	getGrid() const	1134
29.211.3.2	getSurface(bool &method_failed)	1134
29.211.3.3	refineApproximation()	1134
29.211.3.4	removeGridLines(const std::vector< double > &grid_lines_u, const std::vector< double > &grid_lines_v)	1134
29.211.3.5	setNoSplit(const std::vector< shared_ptr< SplineCurve > > &no_split_cvs_2d, double no_split_dist_2d)	1134
29.212	Go::HermiteAppS Class Reference	1135
29.212.1	Detailed Description	1135
29.212.2	Constructor & Destructor Documentation	1135
29.212.2.1	HermiteAppS(EvalCurveSet *surface, double tolerance1, double tolerance2, std::vector< int > dims)	1135
29.212.2.2	HermiteAppS(EvalCurveSet *surface, double initpars[], int n, double tolerance1, double tolerance2, std::vector< int > dims)	1135
29.212.2.3	~HermiteAppS()	1136
29.212.3	Member Function Documentation	1136
29.212.3.1	getCurves()	1136
29.212.3.2	refineApproximation()	1136
29.213	Go::HermiteGrid1D Class Reference	1136
29.213.1	Detailed Description	1137
29.213.2	Constructor & Destructor Documentation	1137
29.213.2.1	HermiteGrid1D(const EvalCurve &crv, double start, double end)	1137
29.213.2.2	HermiteGrid1D(const EvalCurve &crv, double param[], int n)	1137
29.213.2.3	~HermiteGrid1D()	1138
29.213.3	Member Function Documentation	1138
29.213.3.1	addKnot(const EvalCurve &crv, double knot)	1138
29.213.3.2	dim()	1138
29.213.3.3	getData()	1138

29.213.3.4	getKnots()	1138
29.213.3.5	getSegment(int left, int right, double &spar, double &epar, Point bezcoef[4])	1138
29.213.3.6	size()	1139
29.214	Go::HermiteGrid1DMulti Class Reference	1139
29.214.1	Detailed Description	1139
29.214.2	Constructor & Destructor Documentation	1139
29.214.2.1	HermiteGrid1DMulti(EvalCurveSet &surf, double start, double end, std::vector< int > dims)	1139
29.214.2.2	HermiteGrid1DMulti(EvalCurveSet &surf, double param[], int n, std::vector< int > dims)	1140
29.214.2.3	~HermiteGrid1DMulti()	1140
29.214.3	Member Function Documentation	1140
29.214.3.1	addKnot(EvalCurveSet &surf, double knot)	1140
29.214.3.2	dims(int ki)	1140
29.214.3.3	getData()	1141
29.214.3.4	getKnots()	1141
29.214.3.5	getSegment(int left, int right, double &spar, double &epar, std::vector< std::vector< Point > > &bezcoef)	1141
29.214.3.6	size()	1141
29.215	Go::HermiteGrid2D Class Reference	1141
29.215.1	Detailed Description	1142
29.215.2	Constructor & Destructor Documentation	1142
29.215.2.1	HermiteGrid2D(const EvalSurface &sf, double u1, double u2, double v1, double v2)	1142
29.215.2.2	HermiteGrid2D(const EvalSurface &sf, double param_u[], double param_v[], int mm, int nn)	1142
29.215.2.3	~HermiteGrid2D()	1143
29.215.3	Member Function Documentation	1143
29.215.3.1	addKnot(const EvalSurface &sf, double knot, bool dir_is_u)	1143
29.215.3.2	dim() const	1143
29.215.3.3	getData() const	1143
29.215.3.4	getKnots(bool dir_is_u) const	1143
29.215.3.5	getNoSplitStatus(int ind_u, int ind_v)	1143

29.215.3.6	getSegment(int left1, int right1, int left2, int right2, double &spar1, double &epar1, double &spar2, double &epar2, Point bezcoef[4])	1143
29.215.3.7	removeGridLines(const std::vector< int > &grid_lines_u, const std::vector< int > &grid_lines_v)	1144
29.215.3.8	setNoSplitStatus(int ind_u, int ind_v, int no_split_status)	1144
29.215.3.9	size1() const	1144
29.215.3.10	size2() const	1144
29.216	Go::HermiteInterpolator Class Reference	1144
29.216.1	Detailed Description	1145
29.216.2	Constructor & Destructor Documentation	1145
29.216.2.1	HermiteInterpolator()	1145
29.216.2.2	~HermiteInterpolator()	1145
29.216.3	Member Function Documentation	1146
29.216.3.1	basis()	1146
29.216.3.2	Interpolate(int num_points, int dimension, const double *param_start, const double *data_start, std::vector< double > &coefs)	1146
29.216.3.3	Interpolate(const std::vector< Point > &data, const std::vector< double > ¶m, std::vector< double > &coefs)	1146
29.217	Go::Hyperbola Class Reference	1147
29.217.1	Detailed Description	1149
29.217.2	Constructor & Destructor Documentation	1150
29.217.2.1	Hyperbola()	1150
29.217.2.2	Hyperbola(Point location, Point direction, Point normal, double r1, double r2, bool isReversed=false)	1150
29.217.2.3	~Hyperbola()	1150
29.217.3	Member Function Documentation	1150
29.217.3.1	appendCurve(ParamCurve *cv, bool reparam=true)	1150
29.217.3.2	appendCurve(ParamCurve *cv, int continuity, double &dist, bool reparam=true)	1150
29.217.3.3	boundingBox() const	1151
29.217.3.4	classType()	1151
29.217.3.5	clone() const	1151
29.217.3.6	closestPoint(const Point &pt, double tmin, double tmax, double &clo_t, Point &clo_pt, double &clo_dist, double const *seed=0) const	1151

29.217.3.7	<code>createSplineCurve() const</code>	1151
29.217.3.8	<code>dimension() const</code>	1151
29.217.3.9	<code>directionCone() const</code>	1152
29.217.3.10	<code>endparam() const</code>	1152
29.217.3.11	<code>geometryCurve()</code>	1152
29.217.3.12	<code>istanceType() const</code>	1152
29.217.3.13	<code>Bounded() const</code>	1152
29.217.3.14	<code>Degenerate(double degenerate_epsilon)</code>	1152
29.217.3.15	<code>InPlane(const Point &norm, double eps, Point &pos) const</code>	1153
29.217.3.16	<code>length(double tol)</code>	1153
29.217.3.17	<code>point(Point &pt, double tpar) const</code>	1153
29.217.3.18	<code>point(std::vector< Point > &pts, double tpar, int derivs, bool from_right=true) const</code>	1154
29.217.3.19	<code>read(std::istream &is)</code>	1154
29.217.3.20	<code>setParamBounds(double startpar, double endpar)</code>	1154
29.217.3.21	<code>setParameterInterval(double t1, double t2)</code>	1154
29.217.3.22	<code>setSpanningVectors()</code>	1155
29.217.3.23	<code>startparam() const</code>	1155
29.217.3.24	<code>subCurve(double from_par, double to_par, double fuzzy=DEFAULT_PARAMETER_EPSILON) const</code>	1155
29.217.3.25	<code>swapParameters2D()</code>	1155
29.217.3.26	<code>translateCurve(const Point &dir)</code>	1155
29.217.3.27	<code>write(std::ostream &os) const</code>	1155
29.217.4	Member Data Documentation	1156
29.217.4.1	<code>endparam_</code>	1156
29.217.4.2	<code>location_</code>	1156
29.217.4.3	<code>normal_</code>	1156
29.217.4.4	<code>1_</code>	1156
29.217.4.5	<code>2_</code>	1156
29.217.4.6	<code>startparam_</code>	1156
29.217.4.7	<code>vec1_</code>	1156

29.217.4.8	vec2_	1156
29.218	Go::Identity Class Reference	1157
29.218.1	Detailed Description	1157
29.218.2	Member Function Documentation	1157
29.218.2.1	IdenticalCvs(shared_ptr< ParamCurve > cv1, double start1, double end1, shared_ptr< ParamCurve > cv2, double start2, double end2, double tol)	1157
29.218.2.2	IdenticalSfs(shared_ptr< ParamSurface > sf1, shared_ptr< ParamSurface > sf2, double tol)	1157
29.219	NEWMAT::IdentityMatrix Class Reference	1158
29.219.1	Detailed Description	1159
29.219.2	Constructor & Destructor Documentation	1159
29.219.2.1	IdentityMatrix()	1159
29.219.2.2	~IdentityMatrix()	1159
29.219.2.3	IdentityMatrix(ArrayLengthSpecifier n)	1159
29.219.2.4	IdentityMatrix(const IdentityMatrix &gm)	1160
29.219.2.5	IdentityMatrix(const BaseMatrix &)	1160
29.219.3	Member Function Documentation	1160
29.219.3.1	BandWidth() const	1160
29.219.3.2	GetCol(MatrixRowCol &)	1160
29.219.3.3	GetCol(MatrixColX &)	1160
29.219.3.4	GetRow(MatrixRowCol &)	1160
29.219.3.5	LogDeterminant() const	1160
29.219.3.6	MakeSolver()	1160
29.219.3.7	NextCol(MatrixRowCol &)	1160
29.219.3.8	NextCol(MatrixColX &)	1161
29.219.3.9	NextRow(MatrixRowCol &)	1161
29.219.3.10	operator=(const BaseMatrix &)	1161
29.219.3.11	operator=(Real f)	1161
29.219.3.12	ReSize(int n)	1161
29.219.3.13	ReSize(const GeneralMatrix &A)	1161
29.219.3.14	Solver(MatrixColX &, const MatrixColX &)	1161

29.219.3.1	<code>Sum()</code> const	1161
29.219.3.1	<code>SumAbsoluteValue()</code> const	1161
29.219.3.1	<code>SumSquare()</code> const	1162
29.219.3.1	<code>Trace()</code> const	1162
29.219.3.1	<code>Transpose(TransposedMatrix *, MatrixType)</code>	1162
29.219.3.2	<code>Type()</code> const	1162
29.220	<code>Go::IGESconverter</code> Class Reference	1162
29.220.1	Detailed Description	1163
29.220.2	Constructor & Destructor Documentation	1163
29.220.2.1	<code>IGESconverter()</code>	1163
29.220.2.2	<code>IGESconverter(std::istream &is, FileFormat from_type, std::ostream &os, FileFormat to_type)</code>	1164
29.220.2.3	<code>~IGESconverter()</code>	1164
29.220.3	Member Function Documentation	1164
29.220.3.1	<code>addGeom(shared_ptr< Go::GeomObject > sp)</code>	1164
29.220.3.2	<code>getColour(int i)</code>	1164
29.220.3.3	<code>getGoGeom()</code>	1164
29.220.3.4	<code>getGroup()</code>	1164
29.220.3.5	<code>header()</code>	1164
29.220.3.6	<code>minResolution()</code> const	1164
29.220.3.7	<code>num_geom()</code>	1165
29.220.3.8	<code>num_group()</code>	1165
29.220.3.9	<code>readdisp(std::istream &is)</code>	1165
29.220.3.10	<code>readgo(std::istream &is)</code>	1165
29.220.3.11	<code>readIGES(std::istream &is)</code>	1165
29.220.3.12	<code>readislrivs(std::istream &is)</code>	1165
29.220.3.13	<code>readislrfs(std::istream &is)</code>	1165
29.220.3.14	<code>writedisp(std::ostream &os)</code>	1165
29.220.3.15	<code>writego(std::ostream &os)</code>	1165
29.220.3.16	<code>writelGES(std::ostream &os)</code>	1165
29.220	<code>Go::IGESdientry</code> Struct Reference	1166

29.221.1	Detailed Description	1166
29.221.2	Member Data Documentation	1166
29.221.2.1	color	1166
29.221.2.2	entity_label	1166
29.221.2.3	entity_number	1166
29.221.2.4	entity_type_number	1166
29.221.2.5	form	1167
29.221.2.6	label_display	1167
29.221.2.7	level	1167
29.221.2.8	line_count	1167
29.221.2.9	line_font_pattern	1167
29.221.2.10	line_weight	1167
29.221.2.11	param_data_start	1167
29.221.2.12	status	1167
29.221.2.13	structure	1167
29.221.2.14	trans_matrix	1167
29.221.2.15	view	1168
29.222.0	IGESheader Struct Reference	1168
29.222.1	Detailed Description	1169
29.222.2	Constructor & Destructor Documentation	1169
29.222.2.1	IGESheader()	1169
29.222.3	Member Data Documentation	1169
29.222.3.1	application_protocol_description	1169
29.222.3.2	author	1169
29.222.3.3	creator_system_prodid	1169
29.222.3.4	double_prec_magn	1169
29.222.3.5	double_prec_sign	1169
29.222.3.6	drafting_standard	1169
29.222.3.7	IGES_version	1169
29.222.3.8	integer_bits	1170

29.222.3.9	max_coordinate	1170
29.222.3.10	max_line_width	1170
29.222.3.11	max_weight_grads	1170
29.222.3.12	min_resolution	1170
29.222.3.13	model_space_scale	1170
29.222.3.14	num_parameters	1170
29.222.3.15	organisation	1170
29.222.3.16	original_filename	1170
29.222.3.17	ordel	1170
29.222.3.18	preprocessor_version_number	1171
29.222.3.19	rcdel	1171
29.222.3.20	receiving_system_prodid	1171
29.222.3.21	sending_system_prodid	1171
29.222.3.22	angle_prec_magn	1171
29.222.3.23	angle_prec_sign	1171
29.222.3.24	timestamp_file_changed	1171
29.222.3.25	timestamp_model_changed	1171
29.222.3.26	unit_description	1171
29.222.3.27	unit_flag	1171
29.223.0	Go::ImplicitizeCurveAlgo Class Reference	1172
29.223.1	Detailed Description	1172
29.223.2	Constructor & Destructor Documentation	1172
29.223.2.1	ImplicitizeCurveAlgo()	1172
29.223.2.2	ImplicitizeCurveAlgo(int deg)	1172
29.223.2.3	ImplicitizeCurveAlgo(const SplineCurve &curve, int deg)	1173
29.223.3	Member Function Documentation	1173
29.223.3.1	getResultData(BernsteinTriangularPoly &implicit, BaryCoordSystem2D &bc, double &sigma_min)	1173
29.223.3.2	perform()	1173
29.223.3.3	setDegree(int deg)	1173
29.223.3.4	setTolerance(double tol)	1173

29.223.3.5	useSplineCurve(const SplineCurve &curve)	1174
29.224	Go::ImplicitizeCurveAndVectorAlgo Class Reference	1174
29.224.1	Detailed Description	1174
29.224.2	Constructor & Destructor Documentation	1175
29.224.2.1	ImplicitizeCurveAndVectorAlgo()	1175
29.224.2.2	ImplicitizeCurveAndVectorAlgo(int deg)	1175
29.224.2.3	ImplicitizeCurveAndVectorAlgo(const SplineCurve &crv, const Point &pt, int deg)	1175
29.224.3	Member Function Documentation	1175
29.224.3.1	getResultData(BernsteinTetrahedralPoly &implicit, BaryCoordSystem3D &bc, double &sigma_min)	1175
29.224.3.2	perform()	1176
29.224.3.3	setDegree(int deg)	1176
29.224.3.4	setTolerance(double tol)	1176
29.224.3.5	useSplineCurve(const SplineCurve &crv)	1176
29.224.3.6	useVector(const Point &pt)	1176
29.225	Go::ImplicitizePointCloudAlgo Class Reference	1177
29.225.1	Detailed Description	1177
29.225.2	Constructor & Destructor Documentation	1177
29.225.2.1	ImplicitizePointCloudAlgo()	1177
29.225.2.2	ImplicitizePointCloudAlgo(int deg)	1177
29.225.2.3	ImplicitizePointCloudAlgo(const PointCloud3D &cloud, int deg)	1178
29.225.3	Member Function Documentation	1178
29.225.3.1	getResultData(BernsteinTetrahedralPoly &implicit, BaryCoordSystem3D &bc, double &sigma_min)	1178
29.225.3.2	perform()	1178
29.225.3.3	setDegree(int deg)	1178
29.225.3.4	setTolerance(double tol)	1179
29.225.3.5	usePointCloud(const PointCloud3D &cloud)	1179
29.226	Go::ImplicitizeSurfaceAlgo Class Reference	1179
29.226.1	Detailed Description	1180
29.226.2	Constructor & Destructor Documentation	1180

29.226.2.1	ImplicitizeSurfaceAlgo()	1180
29.226.2.2	ImplicitizeSurfaceAlgo(int deg)	1180
29.226.2.3	ImplicitizeSurfaceAlgo(const SplineSurface &surf, int deg)	1180
29.226.3	Member Function Documentation	1180
29.226.3.1	getResultData(BernsteinTetrahedralPoly &implicit, BaryCoordSystem3D &bc, double &sigma_min)	1181
29.226.3.2	perform()	1181
29.226.3.3	setDegree(int deg)	1181
29.226.3.4	setTolerance(double tol)	1181
29.226.3.5	useSplineSurface(const SplineSurface &surf)	1181
29.227	NEWMAT::IncompatibleDimensionsException Class Reference	1182
29.227.1	Detailed Description	1183
29.227.2	Constructor & Destructor Documentation	1183
29.227.2.1	IncompatibleDimensionsException()	1183
29.227.2.2	IncompatibleDimensionsException(const GeneralMatrix &, const GeneralMatrix &)	1183
29.227.3	Member Data Documentation	1183
29.227.3.1	Select	1183
29.228	NEWMAT::IndexException Class Reference	1184
29.228.1	Detailed Description	1185
29.228.2	Constructor & Destructor Documentation	1185
29.228.2.1	IndexException(int i, const GeneralMatrix &A)	1185
29.228.2.2	IndexException(int i, int j, const GeneralMatrix &A)	1185
29.228.2.3	IndexException(int i, const GeneralMatrix &A, bool)	1185
29.228.2.4	IndexException(int i, int j, const GeneralMatrix &A, bool)	1185
29.228.3	Member Data Documentation	1185
29.228.3.1	Select	1185
29.229	Go::IndexMesh2DIterator Class Reference	1185
29.229.1	Detailed Description	1186
29.229.2	Constructor & Destructor Documentation	1186
29.229.2.1	IndexMesh2DIterator(const Mesh2D &m, int kind_u=0, int kmul_u=1, int kind_v=0, int kmul_v=1)	1186

29.229.3	Member Function Documentation	1186
29.229.3.1	operator!=(const IndexMesh2DIterator &rhs) const	1186
29.229.3.2	operator*() const	1186
29.229.3.3	operator++()	1186
29.229.3.4	operator==(const IndexMesh2DIterator &rhs) const	1186
29.230	Go::IntCrvEvaluator Class Reference	1187
29.230.1	Detailed Description	1188
29.230.2	Constructor & Destructor Documentation	1188
29.230.2.1	IntCrvEvaluator(shared_ptr< CurveOnSurface > sfcv1, double start1, double end1, shared_ptr< CurveOnSurface > sfcv2, double start2, double end2, bool same_orientation, int keep_crv=0)	1188
29.230.2.2	~IntCrvEvaluator()	1188
29.230.3	Member Function Documentation	1188
29.230.3.1	approximationOK(double par, const std::vector< Point > &approxpos, double tol1, double tol2)	1188
29.230.3.2	dim()	1188
29.230.3.3	end()	1189
29.230.3.4	eval(double t)	1189
29.230.3.5	eval(double t, int n, std::vector< std::vector< Point > > &der)	1189
29.230.3.6	getMaxErr()	1189
29.230.3.7	nmbCvs()	1189
29.230.3.8	resetErr()	1190
29.230.3.9	start()	1190
29.231	NEWMAT::InternalException Class Reference	1190
29.231.1	Detailed Description	1191
29.231.2	Constructor & Destructor Documentation	1191
29.231.2.1	InternalException(const char *c)	1191
29.231.3	Member Data Documentation	1191
29.231.3.1	Select	1191
29.232	Go::InterpolatedIntersectionCurve Class Reference	1192
29.232.1	Detailed Description	1193
29.232.2	Constructor & Destructor Documentation	1193

29.232.2.1	~InterpolatedIntersectionCurve()	1193
29.232.3	Member Function Documentation	1193
29.232.3.1	evaluateAt(double pval, Point &pos, Point &tan)	1193
29.232.3.2	getCurve() const	1193
29.232.3.3	getGuidePointTangent(shared_ptr< IntersectionPoint > pt, Point &tan, int type) const	1194
29.232.3.4	getParamCurve(int obj_nmb) const	1194
29.232.3.5	getParamSpan(double &start, double &end) const	1194
29.232.3.6	isDegenerated() const	1195
29.232.3.7	isocurve() const	1195
29.232.3.8	read(std::istream &is) const	1195
29.232.3.9	refine(const double &pos_tol, const double &angle_tol)	1195
29.232.3.10	write(std::ostream &os) const	1196
29.232.4	Friends And Related Function Documentation	1196
29.232.4.1	constructIntersectionCurve	1196
29.233	Go::Interpolator Class Reference	1196
29.233.1	Detailed Description	1196
29.233.2	Constructor & Destructor Documentation	1197
29.233.2.1	~Interpolator()	1197
29.233.3	Member Function Documentation	1197
29.233.3.1	basis()=0	1197
29.233.3.2	interpolate(int num_points, int dimension, const double *param_start, const double *data_start, std::vector< double > &coefs)=0	1197
29.234	Go::IntersectionCurve Class Reference	1197
29.234.1	Detailed Description	1199
29.234.2	Constructor & Destructor Documentation	1199
29.234.2.1	~IntersectionCurve()	1199
29.234.2.2	IntersectionCurve(iterator begin, iterator end)	1199
29.234.3	Member Function Documentation	1199
29.234.3.1	evaluateAt(double pval, Point &pos, Point &tan)=0	1199
29.234.3.2	getCurve() const =0	1199

29.234.3.3	getGuidePoint(int index) const	1200
29.234.3.4	getGuidePointTangent(shared_ptr< IntersectionPoint > pt, Point &tan, int type=0) const	1200
29.234.3.5	getParamCurve(int obj_nmb) const =0	1200
29.234.3.6	getParamSpan(double &start, double &end) const =0	1201
29.234.3.7	isDegenerated() const =0	1201
29.234.3.8	islocurve() const =0	1201
29.234.3.9	numGuidePoints() const	1202
29.234.3.10	refine(const double &pos_tol, const double &angle_tol)=0	1202
29.234.3.11	writePointsToStream(std::ostream &os) const	1202
29.234.4	Friends And Related Function Documentation	1202
29.234.4.1	constructIntersectionCurve	1202
29.234.5	Member Data Documentation	1202
29.234.5.1	ipoints_	1202
29.235	IntersectionLink Class Reference	1203
29.235.1	Detailed Description	1203
29.235.2	Constructor & Destructor Documentation	1203
29.235.2.1	IntersectionLink(IntersectionPoint *p1, IntersectionPoint *p2)	1203
29.235.3	Member Function Documentation	1203
29.235.3.1	copyMetaInformation(const IntersectionLink &rhs)	1203
29.235.3.2	crosses(const shared_ptr< IntersectionLink > &link) const	1204
29.235.3.3	delimitsPAC() const	1204
29.235.3.4	dumpToStream(std::ostream &os)	1204
29.235.3.5	getIntersectionPoints(IntersectionPoint *&p1, IntersectionPoint *&p2)	1204
29.235.3.6	getIntersectionPoints(const IntersectionPoint *&p1, const IntersectionPoint *&p2) const	1204
29.235.3.7	getOtherPoint(const IntersectionPoint *p1)	1204
29.235.3.8	isAttachedTo(const IntersectionPoint *ip)	1204
29.235.3.9	isIsoparametric() const	1204
29.235.3.10	isIsoparametricIn(int dir) const	1204
29.235.3.11	linkType() const	1204

29.235.3.11	LinkType()	1205
29.235.3.12	NumParams() const	1205
29.235.3.13	SetIsoparametricIn(int dir, bool iso)	1205
29.235.3.14	SetPAC(bool value)	1205
29.235.3.15	WriteInfo() const	1205
29.236	Go::IntersectionPoint Class Reference	1205
29.236.1	Detailed Description	1207
29.236.2	Constructor & Destructor Documentation	1207
29.236.2.1	IntersectionPoint(const ParamObjectInt *obj1, const ParamObjectInt *obj2, const shared_ptr< GeoTol > epsge, const double *obj1_params, const double *obj2_params)	1207
29.236.2.2	IntersectionPoint(const ParamObjectInt *obj1, const ParamObjectInt *obj2, const shared_ptr< IntersectionPoint > ip, int missing_param)	1208
29.236.2.3	~IntersectionPoint()	1208
29.236.2.4	IntersectionPoint()	1208
29.236.3	Member Function Documentation	1208
29.236.3.1	checkIntersectionPoint(IntPtInfo &int_pt_info) const	1208
29.236.3.2	commonIsoLinks(int *const iso_par_dirs) const	1209
29.236.3.3	connectTo(IntersectionPoint *const point, LinkType type, shared_ptr< IntersectionLink > model_link=shared_ptr< IntersectionLink >(), int added_parameter_dir=-1)	1209
29.236.3.4	connectTo(shared_ptr< IntersectionPoint > point, LinkType type, shared_ptr< IntersectionLink > model_link=shared_ptr< IntersectionLink >(), int added_parameter_dir=-1)	1210
29.236.3.5	containsG2Discontinuity() const	1210
29.236.3.6	differentiatesFromLeft(int pardir) const	1210
29.236.3.7	disconnectFrom(IntersectionPoint *point)	1211
29.236.3.8	endParam(int pardir)	1211
29.236.3.9	fixTangentOrientation(bool flip)	1211
29.236.3.10	IpObjects()	1211
29.236.3.11	getClassification(const ParamSurface *surf1, const ParamSurface *surf2, int branch_num=0) const	1212
29.236.3.12	getClassification(const ParamSurface *par_func, double C, int branch_num=0) const	1212

29.236.3.16	GetDist() const	1212
29.236.3.16	GetInfluenceArea(int dir, bool forward, bool first_outside, double aeps=0.0) const	1213
29.236.3.16	GetIntersectionLink(const IntersectionPoint *point)	1213
29.236.3.16	GetNeighbourLinks(std::vector< shared_ptr< IntersectionLink > > &links) const	1213
29.236.3.16	GetNeighbours(std::vector< IntersectionPoint * > &pnts) const	1214
29.236.3.16	GetObj1() const	1214
29.236.3.16	GetObj2() const	1214
29.236.3.20	GetPar() const	1214
29.236.3.21	GetPar(int idx) const	1214
29.236.3.22	GetPar1() const	1214
29.236.3.23	GetPar1Dir(bool second_branch=false) const	1215
29.236.3.24	GetPar1Point() const	1215
29.236.3.25	GetPar2() const	1215
29.236.3.26	GetPar2Dir(bool second_branch=false) const	1215
29.236.3.27	GetPar2Point() const	1216
29.236.3.28	GetPoint() const	1216
29.236.3.29	GetPoint1() const	1216
29.236.3.30	GetPoint2() const	1216
29.236.3.31	GetSingularityType() const	1216
29.236.3.32	GetTangent(bool second_branch=false) const	1216
29.236.3.33	GetTolerance()	1217
29.236.3.34	GetTolerance() const	1217
29.236.3.35	HasIsoLinks(int *const iso_par_dirs) const	1217
29.236.3.36	HasParentPoint()	1217
29.236.3.37	HasUniqueTangentDirection() const	1217
29.236.3.38	InfluenceArea(int pardir, double par, bool first_outside) const	1218
29.236.3.39	InfluenceAreaBracket(int pardir, double parval)	1218
29.236.3.40	BoundaryPoint(bool &first, bool &second) const	1218
29.236.3.41	ConnectedTo(shared_ptr< IntersectionPoint > p) const	1219
29.236.3.42	ConnectedTo(const IntersectionPoint *point) const	1219

29.236.3.43	Degenerate() const	1219
29.236.3.44	InDomain(double *frompar, double *topar) const	1219
29.236.3.45	NearSingular() const	1220
29.236.3.46	SamePoint(const IntersectionPoint *point) const	1220
29.236.3.47	umBranches() const	1220
29.236.3.48	umNeighbours() const	1220
29.236.3.49	umParams1() const	1220
29.236.3.50	umParams2() const	1220
29.236.3.51	parameterTolerance(int paddir)	1220
29.236.3.52	parentPoint()	1221
29.236.3.53	pointIsSingular() const	1221
29.236.3.54	projectToParamPlanes(const Point &dir, Point &par_1_dir, Point &par_2_dir) const	1221
29.236.3.55	read(std::istream &is)	1221
29.236.3.56	replaceParameter(double *param)	1222
29.236.3.57	setDifferentiateFromLeft(bool_iterator it) const	1222
29.236.3.58	setParentPoint(shared_ptr< IntersectionPoint > p)	1222
29.236.3.59	shareInfluenceAreaWith(shared_ptr< IntersectionPoint > other_pt, int paddir)	1222
29.236.3.60	startParam(int paddir)	1223
29.236.3.61	tangentIsOriented() const	1223
29.236.3.62	tangentPointingInwards(bool &first, bool &second, bool &first_along_boundary, bool &second_along_boundary) const	1223
29.236.3.63	write(std::ostream &os) const	1223
29.236.3.64	writeInfo() const	1224
29.236.3.65	writeParams(std::ostream &os) const	1224
29.236.4	Member Data Documentation	1224
29.236.4.1	tangent_tol	1224
29.237	Go::IntersectionPool Class Reference	1224
29.237.1	Detailed Description	1226
29.237.2	Constructor & Destructor Documentation	1226
29.237.2.1	IntersectionPool(shared_ptr< ParamObjectInt > obj1, shared_ptr< ParamObjectInt > obj2, shared_ptr< IntersectionPool > parent=shared_ptr< IntersectionPool >(), int missing_dir=-1, double missing_value=0)	1226

29.237.2.2~IntersectionPool()	1227
29.237.3 Member Function Documentation	1227
29.237.3.1 addCurve(ip_iterator start, ip_iterator end)	1227
29.237.3.2 addCurves(int nmb_int_cvs, ip_iterator *start_iterators, ip_iterator *end_iterators)	1227
29.237.3.3 addIntersectionPoint(shared_ptr< ParamObjectInt > obj_int1_, shared_ptr< ParamObjectInt > obj_int2_, shared_ptr< GeoTol > epsge, double *par1, double *par2)	1227
29.237.3.4 addParamPoints(int nmb_int_pts, double *pointpar1, double *pointpar2, shared_ptr< GeoTol > epsge)	1228
29.237.3.5 atDifferentBoundary(shared_ptr< IntersectionPoint > pt1, shared_ptr< IntersectionPoint > pt2)	1228
29.237.3.6 atSameBoundary(shared_ptr< IntersectionPoint > pt1, shared_ptr< IntersectionPoint > pt2)	1228
29.237.3.7 checkIfBothPointsLieOnOneEndpointAndNotOfTheSameCurve()	1229
29.237.3.8 checkIntersectionChain(IntersectionPoint *pnt, IntersectionPoint *first, IntersectionPoint *prev=0)	1229
29.237.3.9 checkIntersectionLinks() const	1229
29.237.3.10 checkIntersectionPoints(std::vector< IntPtInfo > &int_pt_info) const	1229
29.237.3.11 checkSortingDir(Point &vec, int sorting_obj)	1229
29.237.3.12 cleanUpPool(int first_idx=0, double epsge=1.0e-15)	1230
29.237.3.13 closestInDomain(double param[], shared_ptr< IntersectionPoint > &pnt) const	1230
29.237.3.14 closestInDomain(const double param[], shared_ptr< IntersectionPoint > &pnt) const	1230
29.237.3.15 endParam(int dir) const	1231
29.237.3.16 existIntersectionPoint(int dir, double par)	1231
29.237.3.17 fetchLoops(std::vector< std::vector< shared_ptr< IntersectionPoint > > > &loop_ints)	1231
29.237.3.18 getAllIntersections(std::vector< shared_ptr< IntersectionPoint > > &result)	1232
29.237.3.19 getAllLinks(std::set< shared_ptr< IntersectionLink > > &links)	1232
29.237.3.20 getBoundaryIntersections(std::vector< shared_ptr< IntersectionPoint > > &bd_ints)	1232
29.237.3.21 getBranchPoints(std::vector< shared_ptr< IntersectionPoint > > &pts)	1232
29.237.3.22 getIntersectionCurves(std::vector< shared_ptr< IntersectionCurve > > &int_curves) const	1232
29.237.3.23 getIntersectionCurves() const	1232

29.237.3.24	<code>getIntersectionPoints(std::vector< shared_ptr< IntersectionPoint > > &int_pts) const</code>	1233
29.237.3.25	<code>getIntersectionPoints()</code>	1233
29.237.3.26	<code>getIntersectionPoints() const</code>	1233
29.237.3.27	<code>getIntersectionPoints(int dir, double par, std::vector< shared_ptr< IntersectionPoint > > &result) const</code>	1233
29.237.3.28	<code>getMidParameter(double *mid)</code>	1234
29.237.3.29	<code>getOrigPoints(std::vector< shared_ptr< IntersectionPoint > > &int_pts) const</code>	1234
29.237.3.30	<code>getResult(std::vector< shared_ptr< IntersectionPoint > > &int_points, std::vector< shared_ptr< IntersectionCurve > > &int_curves)</code>	1234
29.237.3.31	<code>getSortedBdInts(const Point &vec, std::vector< shared_ptr< IntersectionPoint > > &result, int obj_nmb=-1)</code>	1235
29.237.3.32	<code>getSortedInnerInts(int pardir)</code>	1235
29.237.3.33	<code>getSortedIntersections(std::vector< shared_ptr< IntersectionPoint > > &int_pts)</code>	1235
29.237.3.34	<code>getSortedInts(std::vector< shared_ptr< IntersectionPoint > > &result)</code>	1235
29.237.3.35	<code>hasIntersectionPoints() const</code>	1236
29.237.3.36	<code>hasIntersectionPoints(int dir, double par) const</code>	1236
29.237.3.37	<code>hasPointsInInner(int pardir)</code>	1236
29.237.3.38	<code>includeCoveredNeighbourPoints()</code>	1236
29.237.3.39	<code>includeReducedInts(shared_ptr< IntersectionPool > lower_order_pool)</code>	1237
29.237.3.40	<code>InfluenceArea(int pardir, double par, bool first_outside=true)</code>	1238
29.237.3.41	<code>InfluenceArea(int pardir, double par, std::vector< shared_ptr< IntersectionPoint > > &int_pts, bool first_outside=true)</code>	1238
29.237.3.42	<code>insertInInfluenceInterval(shared_ptr< IntersectionPoint > pt_in_pool, double *parvals, int pardir)</code>	1239
29.237.3.43	<code>intersectAlongCommonBoundary(double frac, std::vector< Boundary< IntersectionData > &isects)</code>	1239
29.237.3.44	<code>BoundaryIntersection(shared_ptr< IntersectionPoint > pt1, shared_ptr< IntersectionPoint > pt2)</code>	1239
29.237.3.45	<code>BoundaryPoint(shared_ptr< IntersectionPoint > pt1)</code>	1240
29.237.3.46	<code>BoundaryPoint(IntersectionPoint *pt1)</code>	1240
29.237.3.47	<code>ConnectedInside(shared_ptr< IntersectionPoint > pt1, shared_ptr< IntersectionPoint > pt2)</code>	1240
29.237.3.48	<code>InDomain(IntersectionPoint *pnt) const</code>	1240

29.237.3.49	PAC(std::vector< shared_ptr< IntersectionPoint > > &int_loop)	1241
29.237.3.50	ackingParameter() const	1241
29.237.3.51	ackingParameterValue() const	1241
29.237.3.52	akeIntersectionCurves()	1241
29.237.3.53	irrorIntersectionPoints(size_t first)	1241
29.237.3.54	umIntersectionPoints() const	1241
29.237.3.55	umParams() const	1242
29.237.3.56	umSingularIntersectionPoints()	1242
29.237.3.57	moveBoundaryIntersections(bool single_pt=true)	1242
29.237.3.58	moveDefectLinks()	1242
29.237.3.59	moveDoublePoints()	1242
29.237.3.60	moveFalseCorners()	1242
29.237.3.61	moveIntPoint(shared_ptr< IntersectionPoint > int_point)	1242
29.237.3.62	moveIntPoint2(shared_ptr< IntersectionPoint > int_point)	1243
29.237.3.63	moveIntPoints(double *frompar, double *topar, bool only_inner=false)	1243
29.237.3.64	moveLooseEnds()	1243
29.237.3.65	selfIntersectParamReorganise(shared_ptr< IntersectionPool > sub_pool)	1243
29.237.3.66	otCoincidence(std::vector< shared_ptr< IntersectionPoint > > &int_loop)	1244
29.237.3.67	plitIntersectionLinks(int fixed_dir, double fixed_val)	1244
29.237.3.68	artParam(int dir) const	1244
29.237.3.69	ynchronizePool()	1244
29.237.3.70	alidate() const	1244
29.237.3.71	erifyIntersectionLink(const shared_ptr< IntersectionLink > &link, int recursion← _limit=10) const	1245
29.237.3.72	eedOutClutterPoints()	1245
29.237.3.73	riteDebug(int singular=0)	1245
29.237.3.74	riteIntersectionLinks() const	1245
29.237.3.75	riteIntersectionPoints() const	1245
29.237.4	Friends And Related Function Documentation	1245
29.237.4.1	SfSfIntersector	1245
29.237.6	o::Intersector Class Reference	1246

29.238.1	Detailed Description	1248
29.238.2	Constructor & Destructor Documentation	1248
29.238.2.1	Intersector()	1248
29.238.2.2	Intersector(double epsge, Intersector *prev=0)	1248
29.238.2.3	Intersector(shared_ptr< GeoTol > epsge, Intersector *prev=0)	1248
29.238.2.4	~Intersector()	1248
29.238.3	Member Function Documentation	1249
29.238.3.1	addComplexDomain(RectDomain dom)	1249
29.238.3.2	checkCoincidence()=0	1249
29.238.3.3	complexIntercept()	1249
29.238.3.4	complexityReduced()=0	1249
29.238.3.5	complexSimpleCase()	1249
29.238.3.6	compute(bool compute_at_boundary=true)	1249
29.238.3.7	degTriangleSimple()	1249
29.238.3.8	doPostIterate()	1250
29.238.3.9	doSubdivide()=0	1250
29.238.3.10	getBoundaryIntersections()=0	1250
29.238.3.11	getBoundaryObject(int idx, int bd_idx) const	1250
29.238.3.12	getComplexityInfo()	1250
29.238.3.13	getIntPool()	1251
29.238.3.14	getResult(std::vector< shared_ptr< IntersectionPoint > > &int_points, std::vector< shared_ptr< IntersectionCurve > > &int_curves)	1251
29.238.3.15	getSingularityInfo()	1251
29.238.3.16	getTolerance()	1251
29.238.3.17	handleComplexity()=0	1251
29.238.3.18	hasComplexityInfo()	1252
29.238.3.19	hasSingularityInfo()	1252
29.238.3.20	Linear()=0	1252
29.238.3.21	SelfintCase()	1252
29.238.3.22	SelfIntersection()	1252
29.238.3.23	nearCase()=0	1252

29.238.3.24	microCase()=0	1252
29.238.3.25	mbBdObj(int idx) const	1252
29.238.3.26	mbRecurSIONS()	1253
29.238.3.27	numParams() const =0	1253
29.238.3.28	performInterception()=0	1253
29.238.3.29	print_objs()=0	1253
29.238.3.30	printDebugInfo()=0	1253
29.238.3.31	repairIntersections()=0	1253
29.238.3.32	setHighPriSing(double *par)	1253
29.238.3.33	setSingularityInfo(shared_ptr< SingularityInfo > previous, int missing_dir)	1254
29.238.3.34	simpleCase()=0	1254
29.238.3.35	updateIntersections()=0	1254
29.238.3.36	validateSiblingPools()	1254
29.238.3.37	writeIntersectionPoints() const	1254
29.238.4	Friends And Related Function Documentation	1254
29.238.4.1	IntersectionPool	1254
29.238.4.2	SfSfIntersector	1255
29.238.5	Member Data Documentation	1255
29.238.5.1	complexity_info_	1255
29.238.5.2	epsge_	1255
29.238.5.3	int_results_	1255
29.238.5.4	prev_intersector_	1255
29.238.5.5	singularity_info_	1255
29.238.5.6	sub_intersectors_	1255
29.239	Go::Intersector2Obj Class Reference	1256
29.239.1	Detailed Description	1257
29.239.2	Constructor & Destructor Documentation	1257
29.239.2.1	Intersector2Obj()	1257
29.239.2.2	Intersector2Obj(shared_ptr< ParamGeomInt > obj1, shared_ptr< ParamGeomInt > obj2, shared_ptr< GeoTol > epsge, Intersector *prev=0, int eliminated_parameter=-1, double eliminated_value=0)	1257

29.239.2.3	Intersector2Obj(shared_ptr< ParamGeomInt > obj1, shared_ptr< ParamGeomInt > obj2, double epsge, Intersector *prev=0, int eliminated_parameter=-1, double eliminated_value=0)	1258
29.239.2.4	~Intersector2Obj()	1258
29.239.3	Member Function Documentation	1258
29.239.3.1	atBoundary(const double *par1, std::vector< bool > &boundaries)	1258
29.239.3.2	atSameBoundary(const double *par1, const double *par2)	1258
29.239.3.3	checkCoincidence()=0	1258
29.239.3.4	complexIntercept()	1258
29.239.3.5	complexityReduced()	1259
29.239.3.6	complexSimpleCase()	1259
29.239.3.7	doPostIterate()	1259
29.239.3.8	doSubdivide()=0	1259
29.239.3.9	foundIntersectionNearBoundary()	1259
29.239.3.10	getBoundaryIntersections()	1259
29.239.3.11	getBoundaryObject(int idx, int bd_idx) const	1259
29.239.3.12	getSeedIteration(double seed[])	1260
29.239.3.13	handleComplexity()	1260
29.239.3.14	interceptionBySeparationSurface()	1260
29.239.3.15	isLinear()	1260
29.239.3.16	isSelfintCase()	1260
29.239.3.17	isNearCase()=0	1260
29.239.3.18	lowerOrderIntersector(shared_ptr< ParamGeomInt > obj1, shared_ptr< ParamGeomInt > obj2, Intersector *prev=0, int eliminated_parameter=-1, double eliminated_value=0)=0	1261
29.239.3.19	microCase()=0	1261
29.239.3.20	nmbBdObj(int idx) const	1261
29.239.3.21	performInterception()	1261
29.239.3.22	performInterceptionByImplicitization()	1261
29.239.3.23	performRotatedBoxTest(double eps1, double eps2)	1261
29.239.3.24	postIterate(int nmb_orig, int dir=-1, bool keep_endpt=true)	1262
29.239.3.25	print_objs()	1262

29.239.3.26	PrintDebugInfo()	1262
29.239.3.27	RemoveDegenerateConnections()	1262
29.239.3.28	SetSelfintCase(int type)	1262
29.239.3.29	SimpleCase()	1262
29.239.3.30	SimpleCase2(Point &axis1, Point &axis2)	1262
29.239.3.31	SimpleCaseByImplicitization()	1262
29.239.3.32	UpdateIntersections()=0	1263
29.239.3.33	WriteOut()	1263
29.239.4	Member Data Documentation	1263
29.239.4.1	obj_int_	1263
29.239.4.2	selfint_case_	1263
29.240	Go::IntersectorAlgPar Class Reference	1263
29.240.1	Detailed Description	1265
29.240.2	Constructor & Destructor Documentation	1265
29.240.2.1	IntersectorAlgPar(shared_ptr< AlgObjectInt > alg_obj, shared_ptr< ParamObjectInt > param_obj, shared_ptr< GeoTol > epsge, Intersector *prev=0, int eliminated_parameter=-1, double eliminated_value=0)	1265
29.240.2.2	~IntersectorAlgPar()	1265
29.240.3	Member Function Documentation	1265
29.240.3.1	checkCoincidence()	1265
29.240.3.2	complexityReduced()	1265
29.240.3.3	compute(bool compute_at_boundary=true)	1265
29.240.3.4	doSubdivide()	1266
29.240.3.5	getBoundaryIntersections()	1266
29.240.3.6	getResult(std::vector< shared_ptr< IntersectionPoint > > &int_points, std::vector< shared_ptr< IntersectionCurve > > &int_curves)	1266
29.240.3.7	handleComplexity()	1266
29.240.3.8	isLinear()	1266
29.240.3.9	linearCase()	1266
29.240.3.10	microCase()	1266
29.240.3.11	numParams() const	1267
29.240.3.12	performInterception()	1267

29.240.3.1	print_objs()	1267
29.240.3.1	printDebugInfo()	1267
29.240.3.1	pairIntersections()	1267
29.240.3.1	simpleCase()	1267
29.240.3.1	updateIntersections()	1267
29.240.4	Member Data Documentation	1267
29.240.4.1	algobj_int_	1267
29.240.4.2	int_func_const_	1268
29.240.4.3	param_int_	1268
29.240	Go::IntersectorFuncConst Class Reference	1268
29.241.1	Detailed Description	1269
29.241.2	Constructor & Destructor Documentation	1269
29.241.2.1	IntersectorFuncConst(shared_ptr< ParamFunctionInt > func, shared_ptr< ParamFunctionInt > C, shared_ptr< GeoTol > epsge, Intersector *prev=0, int eliminated_parameter=-1, double eliminated_value=0)	1269
29.241.2.2	~IntersectorFuncConst()	1270
29.241.3	Member Function Documentation	1270
29.241.3.1	checkCoincidence()=0	1270
29.241.3.2	complexityReduced()	1270
29.241.3.3	doSubdivide()=0	1270
29.241.3.4	getBoundaryIntersections()	1270
29.241.3.5	handleComplexity()	1270
29.241.3.6	isLinear()	1270
29.241.3.7	linearCase()	1270
29.241.3.8	lowerOrderIntersector(shared_ptr< ParamFunctionInt > obj1, shared_ptr< ParamFunctionInt > obj2, Intersector *prev=0, int eliminated_parameter=-1, double eliminated_value=0)=0	1271
29.241.3.9	microCase()=0	1271
29.241.3.10	performInterception()	1271
29.241.3.1	print_objs()	1271
29.241.3.1	printDebugInfo()	1271
29.241.3.1	simpleCase()	1271

29.241.3.1	<code>updateIntersections()</code>	1271
29.241.4	Friends And Related Function Documentation	1271
29.241.4.1	<code>IntersectorAlgPar</code>	1271
29.241.5	Member Data Documentation	1272
29.241.5.1	<code>C_</code>	1272
29.241.5.2	<code>func_int_</code>	1272
29.242	<code>Go::IntPtInfo</code> Struct Reference	1272
29.242.1	Detailed Description	1273
29.242.2	Constructor & Destructor Documentation	1273
29.242.2.1	<code>IntPtInfo()</code>	1273
29.242.3	Member Data Documentation	1273
29.242.3.1	<code>direction</code>	1273
29.242.3.2	<code>is_ok</code>	1273
29.242.3.3	<code>location</code>	1273
29.242.3.4	<code>neighbours</code>	1273
29.242.3.5	<code>singularity_type</code>	1273
29.243	<code>Go::IntResultsCompCv</code> Class Reference	1274
29.243.1	Detailed Description	1275
29.243.2	Constructor & Destructor Documentation	1275
29.243.2.1	<code>IntResultsCompCv(CompositeCurve *compcv, const ftLine &line)</code>	1275
29.243.2.2	<code>IntResultsCompCv(CompositeCurve *compcv, const ftPlane &plane)</code>	1275
29.243.2.3	<code>~IntResultsCompCv()</code>	1276
29.243.3	Member Function Documentation	1276
29.243.3.1	<code>addIntCv(shared_ptr< ParamCurve > cv, double *startpar, double *endpar)</code>	1276
29.243.3.2	<code>addIntPt(shared_ptr< ParamCurve > cv, double *parval)</code>	1276
29.243.3.3	<code>getIntersectionCurves(std::vector< std::pair< PointOnCurve, PointOnCurve > > &int_crvs) const</code>	1276
29.243.3.4	<code>getIntersectionPoints(std::vector< PointOnCurve > &int_points) const</code>	1276
29.243.3.5	<code>hasIntCurves() const</code>	1276
29.243.3.6	<code>hasIntersections() const</code>	1276
29.243.3.7	<code>hasIntPoints() const</code>	1276

29.243.3.8	<code>nmbCurveSegments() const</code>	1277
29.243.3.9	<code>nmbIntPoints() const</code>	1277
29.243.3.10	<code>tessellate(std::vector< shared_ptr< LineStrip > > &meshes, PointCloud3D &points) const</code>	1277
29.243.3.11	<code>tessellate(int resolution, std::vector< shared_ptr< LineStrip > > &meshes, PointCloud3D &points) const</code>	1277
29.243.3.12	<code>tessellate(double density, std::vector< shared_ptr< LineStrip > > &meshes, PointCloud3D &points) const</code>	1277
29.244	Go::IntResultsModel Class Reference	1278
29.244.1	Detailed Description	1279
29.244.2	Constructor & Destructor Documentation	1279
29.244.2.1	<code>IntResultsModel(IntersectionType type)</code>	1279
29.244.2.2	<code>~IntResultsModel()</code>	1279
29.244.3	Member Function Documentation	1279
29.244.3.1	<code>addLineInfo(const ftLine &line)</code>	1279
29.244.3.2	<code>addPlaneInfo(const ftPlane &plane)</code>	1280
29.244.3.3	<code>hasIntCurves() const =0</code>	1280
29.244.3.4	<code>hasIntersections() const =0</code>	1280
29.244.3.5	<code>hasIntPoints() const =0</code>	1280
29.244.3.6	<code>nmbCurveSegments() const =0</code>	1280
29.244.3.7	<code>nmbIntPoints() const =0</code>	1280
29.244.3.8	<code>tessellate(std::vector< shared_ptr< LineStrip > > &meshes, PointCloud3D &points) const =0</code>	1280
29.244.3.9	<code>tessellate(int resolution, std::vector< shared_ptr< LineStrip > > &meshes, PointCloud3D &points) const =0</code>	1281
29.244.3.10	<code>tessellate(double density, std::vector< shared_ptr< LineStrip > > &meshes, PointCloud3D &points) const =0</code>	1281
29.244.4	Member Data Documentation	1281
29.244.4.1	<code>line_</code>	1281
29.244.4.2	<code>numpar_</code>	1281
29.244.4.3	<code>plane_</code>	1281
29.244.4.4	<code>type_</code>	1281
29.245	Go::IntResultsSfModel Class Reference	1282

29.245.1	Detailed Description	1283
29.245.2	Constructor & Destructor Documentation	1283
29.245.2.1	IntResultsSfModel(SurfaceModel *sfmodel, const ftLine &line)	1283
29.245.2.2	IntResultsSfModel(SurfaceModel *sfmodel, const ftPlane &plane)	1283
29.245.2.3	~IntResultsSfModel()	1284
29.245.3	Member Function Documentation	1284
29.245.3.1	addIntCvs(ftCurve &cvs)	1284
29.245.3.2	addIntPts(std::vector< ftPoint > &intpts)	1284
29.245.3.3	getIntersectionCurves()	1284
29.245.3.4	getIntersectionPoints()	1284
29.245.3.5	hasIntCurves() const	1284
29.245.3.6	hasIntersections() const	1284
29.245.3.7	hasIntPoints() const	1285
29.245.3.8	nmbCurveSegments() const	1285
29.245.3.9	nmbIntPoints() const	1285
29.245.3.10	resellate(std::vector< shared_ptr< LineStrip > > &meshes, PointCloud3D &points) const	1285
29.245.3.11	resellate(int resolution, std::vector< shared_ptr< LineStrip > > &meshes, PointCloud3D &points) const	1285
29.245.3.12	resellate(double density, std::vector< shared_ptr< LineStrip > > &meshes, PointCloud3D &points) const	1285
29.246	BD_COMMON::Invalid_argument Class Reference	1286
29.246.1	Detailed Description	1286
29.246.2	Constructor & Destructor Documentation	1287
29.246.2.1	Invalid_argument(const char *a_what=0)	1287
29.246.3	Member Data Documentation	1287
29.246.3.1	Select	1287
29.247	Go::InverseFactorial< T, N > Struct Template Reference	1287
29.247.1	Detailed Description	1287
29.247.2	Member Function Documentation	1287
29.247.2.1	ival()	1287
29.248	NEWMAT::InvertedMatrix Class Reference	1288

29.248.1	Detailed Description	1289
29.248.2	Constructor & Destructor Documentation	1290
29.248.2.1	~InvertedMatrix()	1290
29.248.3	Member Function Documentation	1290
29.248.3.1	BandWidth() const	1290
29.248.3.2	Evaluate(MatrixType mt=MatrixTypeUnSp)	1290
29.248.3.3	operator*(const BaseMatrix &) const	1290
29.248.3.4	operator*(Real t) const	1290
29.248.4	Friends And Related Function Documentation	1290
29.248.4.1	BaseMatrix	1290
29.249	Go::IsogeometricBlock Class Reference	1291
29.249.1	Detailed Description	1291
29.249.2	Constructor & Destructor Documentation	1292
29.249.2.1	IsogeometricBlock(IsogeometricModel *model)	1292
29.249.2.2	~IsogeometricBlock()	1292
29.249.3	Member Function Documentation	1292
29.249.3.1	asIsogeometricSfBlock()	1292
29.249.3.2	asIsogeometricVolBlock()	1292
29.249.3.3	basis(int paddir) const =0	1292
29.249.3.4	degree(int paddir) const	1292
29.249.3.5	distinctKnots(int paddir) const	1292
29.249.3.6	erasePreEvaluatedBasisFunctions()=0	1292
29.249.3.7	getNeighbour(int bd_nmb) const =0	1292
29.249.3.8	getNmbOfBoundaryConditions() const =0	1292
29.249.3.9	getNmbOfPointBdConditions() const =0	1292
29.249.3.10	getTolerances() const	1293
29.249.3.11	increaseGeometryDegree(int new_degree, int paddir)=0	1293
29.249.3.12	Neighbour(IsogeometricBlock *other) const =0	1293
29.249.3.13	knots(int paddir) const	1293
29.249.3.14	model()	1293

29.249.3.15	<code>NmbCoefs() const =0</code>	1293
29.249.3.16	<code>NmbCoefs(int paddir) const</code>	1293
29.249.3.17	<code>NmbOfNeighbours() const =0</code>	1293
29.249.3.18	<code>RefineGeometry(std::vector< double > newknots, int paddir)=0</code>	1293
29.249.3.19	<code>RefineGeometry(const BsplineBasis &other_basis, int paddir)=0</code>	1293
29.250	<code>Go::IsogeometricModel Class Reference</code>	1294
29.250.1	Detailed Description	1294
29.250.2	Constructor & Destructor Documentation	1294
29.250.2.1	<code>IsogeometricModel(tpTolerances toptol)</code>	1294
29.250.2.2	<code>~IsogeometricModel()</code>	1294
29.250.3	Member Function Documentation	1294
29.250.3.1	<code>getNmbOfBoundaries() const =0</code>	1294
29.250.3.2	<code>getTolerances() const</code>	1295
29.250.3.3	<code>setMinimumDegree(int degree, int solutionspace_idx)=0</code>	1295
29.250.3.4	<code>setTolerances(tpTolerances t)</code>	1295
29.250.3.5	<code>updateSolutionSplineSpace()=0</code>	1295
29.251	<code>Go::IsogeometricSfBlock Class Reference</code>	1295
29.251.1	Detailed Description	1297
29.251.2	Constructor & Destructor Documentation	1297
29.251.2.1	<code>IsogeometricSfBlock(IsogeometricModel *model, shared_ptr< SplineSurface > geom_sf, std::vector< int > solution_space_dimension, int index)</code>	1297
29.251.2.2	<code>~IsogeometricSfBlock()</code>	1297
29.251.3	Member Function Documentation	1297
29.251.3.1	<code>addNeighbour(shared_ptr< IsogeometricSfBlock > neighbour, int edge_nmb_← this, int edge_nmb_other, bool equ_orient)</code>	1297
29.251.3.2	<code>asIsogeometricSfBlock()</code>	1297
29.251.3.3	<code>basis(int paddir) const</code>	1297
29.251.3.4	<code>erasePreEvaluatedBasisFunctions()</code>	1297
29.251.3.5	<code>geomIsDegenerate(std::vector< int > &degen_bd, double epsge)</code>	1297
29.251.3.6	<code>geomIsPeriodic(int per[], double epsge)</code>	1297
29.251.3.7	<code>getDegenEnumeration(std::vector< int > &degen_bd, std::vector< std::vector< int > > &enumeration, double epsge)</code>	1297

29.251.3.8	<code>getEdgeBoundaryConditions(int edge_number, std::vector< shared_ptr< SurfaceBoundaryCondition > > &bd_cond) const</code>	1297
29.251.3.9	<code>getEdgeOrientation(shared_ptr< ParamCurve > crv, double tol)</code>	1297
29.251.3.10	<code>getEdgePointBdConditions(int edge_number, std::vector< shared_ptr< SurfacePointBdCond > > &bd_cond) const</code>	1297
29.251.3.11	<code>getGeomBoundaryCurve(int edge_number) const</code>	1297
29.251.3.12	<code>getNeighbour(int edge_nmb) const</code>	1297
29.251.3.13	<code>getNeighbourInfo(IsogeometricSfBlock *other, std::vector< int > &edges, std::vector< int > &edges_other, std::vector< bool > &equal_oriented)</code>	1298
29.251.3.14	<code>getNmbOfBoundaryConditions() const</code>	1298
29.251.3.15	<code>getNmbOfPointBdConditions() const</code>	1298
29.251.3.16	<code>getParamOnBdCurve(int edge_number, const Point &position) const</code>	1298
29.251.3.17	<code>getPeriodicEnumeration(int pardir, std::vector< std::pair< int, int > > &enumeration)</code>	1298
29.251.3.18	<code>getSolutionSpace(int solution_index)</code>	1298
29.251.3.19	<code>increaseGeometryDegree(int new_degree, int pardir)</code>	1298
29.251.3.20	<code>Neighbour(IsogeometricBlock *other) const</code>	1298
29.251.3.21	<code>nmbCoefs() const</code>	1298
29.251.3.22	<code>nmbOfNeighbours() const</code>	1298
29.251.3.23	<code>nmbSolutionSpaces() const</code>	1298
29.251.3.24	<code>refineGeometry(std::vector< double > newknots, int pardir)</code>	1298
29.251.3.25	<code>refineGeometry(const BsplineBasis &other_basis, int pardir)</code>	1299
29.251.3.26	<code>setMinimumDegree(int degree, int solutionspace_idx)</code>	1299
29.251.3.27	<code>Surface() const</code>	1299
29.251.3.28	<code>updateGeometry(shared_ptr< SplineCurve > new_boundary, int edge_number)</code>	1299
29.251.3.29	<code>updateSolutionSplineSpace(int solutionspace_idx)</code>	1299
29.252.0	<code>IsogeometricSfModel Class Reference</code>	1299
29.252.1	<code>Detailed Description</code>	1300
29.252.2	<code>Constructor & Destructor Documentation</code>	1300
29.252.2.1	<code>IsogeometricSfModel(shared_ptr< SurfaceModel > sfmodel, std::vector< int > solution_space_dimension)</code>	1300
29.252.2.2	<code>~IsogeometricSfModel()</code>	1300
29.252.3	<code>Member Function Documentation</code>	1300

29.252.3.1	addBoundaryCond(int boundary, std::vector< Point > pos, BdConditionType type, BdCondFunctor *fbd, int solutionspace_idx, double *constant_value=0)	1300
29.252.3.2	addDirichletPointBdCond(int boundary, Point &pos, Point &condition_value, int solutionspace_idx)	1300
29.252.3.3	getBoundary(int idx) const	1300
29.252.3.4	getIsogeometricBlocks(std::vector< shared_ptr< IsogeometricSfBlock > > &sf-block)	1300
29.252.3.5	getNmbOfBoundaries() const	1300
29.252.3.6	getOuterBoundary() const	1300
29.252.3.7	setMinimumDegree(int degree, int solutionspace_idx)	1300
29.252.3.8	updateSolutionSplineSpace()	1301
29.252.3.9	updateSolutionSplineSpace(int solutionspace_idx)	1301
29.253.0	IsogeometricVolBlock Class Reference	1301
29.253.1	Detailed Description	1302
29.253.2	Constructor & Destructor Documentation	1303
29.253.2.1	IsogeometricVolBlock(IsogeometricModel *model, shared_ptr< SplineVolume > geom_vol, std::vector< int > solution_space_dimension, int index)	1303
29.253.2.2	IsogeometricVolBlock()	1303
29.253.3	Member Function Documentation	1303
29.253.3.1	addNeighbour(shared_ptr< IsogeometricVolBlock > neighbour, int face_nmb, int this, int face_nmb_other, int orientation, bool same_dir_order)	1303
29.253.3.2	asIsogeometricVolBlock()	1303
29.253.3.3	basis(int pdir) const	1303
29.253.3.4	erasePreEvaluatedBasisFunctions()	1303
29.253.3.5	geomIsDegenerate(std::vector< std::pair< int, int > > °en_bd, double epsge)	1303
29.253.3.6	geomIsPeriodic(int per[], double epsge)	1303
29.253.3.7	getBoundaryCondition(int index) const	1303
29.253.3.8	getDegenEnumeration(std::vector< std::pair< int, int > > °en_bd, std::vector< std::vector< int > > &enumeration, double epsge)	1303
29.253.3.9	getFaceBoundaryConditions(int face_number, std::vector< shared_ptr< VolBoundaryCondition > > &bd_cond) const	1303
29.253.3.10	getFaceOrientation(shared_ptr< ParamSurface > srf, double tol)	1303
29.253.3.11	getFacePointBdConditions(int face_number, std::vector< shared_ptr< VolPointBdCond > > &bd_cond) const	1303

29.253.3.19	GetGeomBoundarySurface(int face_number) const	1303
29.253.3.19	GetNeighbour(int bd_nmb) const	1303
29.253.3.19	GetNeighbourInfo(IsogeometricVolBlock *other, std::vector< int > &faces, std::vector< int > &faces_other, std::vector< int > &orientation, std::vector< bool > &same_dir_order)	1304
29.253.3.19	GetNmbOfBoundaryConditions() const	1304
29.253.3.19	GetNmbOfPointBdConditions() const	1304
29.253.3.19	GetParamOnBdSurf(int face_number, const Point &position) const	1304
29.253.3.19	GetPeriodicEnumeration(int pardir, std::vector< std::pair< int, int > > &enumeration)	1304
29.253.3.19	GetPointBdCondition(int index) const	1304
29.253.3.20	GetSolutionSpace(int solution_index)	1304
29.253.3.21	IncreaseGeometryDegree(int new_degree, int pardir)	1304
29.253.3.21	IsNeighbour(IsogeometricBlock *other) const	1304
29.253.3.21	NmbCoefs() const	1304
29.253.3.21	NmbOfNeighbours() const	1304
29.253.3.21	NmbSolutionSpaces() const	1304
29.253.3.26	RefineGeometry(std::vector< double > newknots, int pardir)	1304
29.253.3.27	RefineGeometry(const BsplineBasis &other_basis, int pardir)	1305
29.253.3.28	SameDirOrder(int face_nmb) const	1305
29.253.3.29	SetMinimumDegree(int degree, int solutionspace_idx)	1305
29.253.3.30	UpdateGeometry(shared_ptr< SplineSurface > new_boundary, int face_number)	1305
29.253.3.31	UpdateSolutionSplineSpace(int solutionspace_idx)	1305
29.253.3.32	Volume() const	1305
29.254	Go::IsogeometricVolModel Class Reference	1305
29.254.1	Detailed Description	1306
29.254.2	Constructor & Destructor Documentation	1306
29.254.2.1	IsogeometricVolModel(shared_ptr< VolumeModel > volmodel, std::vector< int > solution_space_dimension)	1306
29.254.2.2	~IsogeometricVolModel()	1306
29.254.3	Member Function Documentation	1306

29.254.3.1	addBoundaryCond(std::vector< std::pair< ParamSurface *, Point > > polygon, BdConditionType type, BdCondFuncor *fbd, int solutionspace_idx, double *constant_value=0)	1306
29.254.3.2	addDirichletPointBdCond(ParamSurface *bd_surf, Point &pos, Point &condition_value, int solutionspace_idx)	1306
29.254.3.3	getBoundary(int idx) const	1306
29.254.3.4	getIsogeometricBlocks(std::vector< shared_ptr< IsogeometricVolBlock > > &volblock)	1306
29.254.3.5	getNmbOfBoundaries() const	1306
29.254.3.6	getOuterBoundary() const	1307
29.254.3.7	setMinimumDegree(int degree, int solutionspace_idx)	1307
29.254.3.8	updateSolutionSplineSpace()	1307
29.254.3.9	updateSolutionSplineSpace(int solutionspace_idx)	1307
29.255	IsoparametricIntersectionCurve Class Reference	1307
29.255.1	Detailed Description	1309
29.255.2	Constructor & Destructor Documentation	1309
29.255.2.1	~IsoparametricIntersectionCurve()	1309
29.255.2.2	IsoparametricIntersectionCurve(const iterator begin, const iterator end, int isopar)	1309
29.255.3	Member Function Documentation	1309
29.255.3.1	evaluateAt(double pval, Point &pos, Point &tan)	1309
29.255.3.2	getCurve() const	1310
29.255.3.3	getParamCurve(int obj_nmb) const	1310
29.255.3.4	getParamSpan(double &start, double &end) const	1310
29.255.3.5	isDegenerated() const	1310
29.255.3.6	isIsocurve() const	1311
29.255.3.7	precalculate_iso_curves(int isopar)	1311
29.255.3.8	refine(const double &pos_tol, const double &angle_tol)	1311
29.255.3.9	resolve_isoparametric_directions(const iterator begin, const iterator end)	1311
29.255.4	Friends And Related Function Documentation	1311
29.255.4.1	constructIntersectionCurve	1311
29.255.5	Member Data Documentation	1311
29.255.5.1	isopar_geom_curve_	1311

29.255.5.2	sopar_param_curve_1_	1312
29.255.5.3	sopar_param_curve_2_	1312
29.255.5.4	temp_	1312
29.256	BD_COMMON::Janitor Class Reference	1312
29.256.1	Detailed Description	1313
29.256.2	Constructor & Destructor Documentation	1313
29.256.2.1	Janitor()	1313
29.256.2.2	~Janitor()	1313
29.256.3	Member Function Documentation	1313
29.256.3.1	CleanUp()	1313
29.257	NEWMAT::KPMatrix Class Reference	1314
29.257.1	Detailed Description	1316
29.257.2	Constructor & Destructor Documentation	1316
29.257.2.1	KPMatrix(const BaseMatrix *bm1x, const BaseMatrix *bm2x)	1316
29.257.2.2	~KPMatrix()	1316
29.257.3	Member Function Documentation	1316
29.257.3.1	BandWidth() const	1316
29.257.3.2	Evaluate(MatrixType mt=MatrixTypeUnSp)	1316
29.257.4	Friends And Related Function Documentation	1316
29.257.4.1	BaseMatrix	1316
29.257.4.2	GeneralMatrix	1316
29.257.4.3	GenericMatrix	1317
29.257.4.4	KP	1317
29.258	BD_COMMON::Length_error Class Reference	1317
29.258.1	Detailed Description	1318
29.258.2	Constructor & Destructor Documentation	1318
29.258.2.1	Length_error(const char *a_what=0)	1318
29.258.3	Member Data Documentation	1318
29.258.3.1	Select	1318
29.259	Go::LiftCurve Class Reference	1319

29.259.1	Detailed Description	1320
29.259.2	Constructor & Destructor Documentation	1320
29.259.2.1	LiftCurve(shared_ptr< Go::ParamCurve > ¶meter_crv, shared_ptr< Go::ParamSurface > &surf, double epsgeo)	1320
29.259.2.2	~LiftCurve()	1320
29.259.3	Member Function Documentation	1320
29.259.3.1	approximationOK(double par, Point approxpos, double tol1, double tol2) const	1320
29.259.3.2	dim() const	1321
29.259.3.3	end() const	1321
29.259.3.4	eval(double t) const	1321
29.259.3.5	eval(double t, int n, Point der[]) const	1321
29.259.3.6	start() const	1322
29.260	Go::Line Class Reference	1322
29.260.1	Detailed Description	1324
29.260.2	Constructor & Destructor Documentation	1325
29.260.2.1	Line()	1325
29.260.2.2	Line(Point point, Point direction, bool isReversed=false)	1325
29.260.2.3	Line(Point point, Point direction, double length, bool isReversed=false)	1325
29.260.2.4	Line(Point point1, Point point2, double par1, double par2, bool isReversed=false)	1325
29.260.2.5	~Line()	1325
29.260.3	Member Function Documentation	1325
29.260.3.1	appendCurve(ParamCurve *cv, bool reparam=true)	1325
29.260.3.2	appendCurve(ParamCurve *cv, int continuity, double &dist, bool reparam=true)	1325
29.260.3.3	boundingBox() const	1326
29.260.3.4	classType()	1326
29.260.3.5	clone() const	1326
29.260.3.6	closestPoint(const Point &pt, double tmin, double tmax, double &clo_t, Point &clo_pt, double &clo_dist, double const *seed=0) const	1326
29.260.3.7	createSplineCurve() const	1326
29.260.3.8	dimension() const	1327
29.260.3.9	directionCone() const	1327

29.260.3.16	endparam() const	1327
29.260.3.17	geometryCurve()	1327
29.260.3.18	getDirection()	1327
29.260.3.19	getPoint()	1328
29.260.3.20	instanceType() const	1328
29.260.3.21	isBounded() const	1328
29.260.3.22	isDegenerate(double degenerate_epsilon)	1328
29.260.3.23	InPlane(const Point &loc, const Point &axis, double eps, Point &normal) const	1328
29.260.3.24	InPlane(const Point &norm, double eps, Point &pos) const	1328
29.260.3.25	isLinear(Point &dir, double tol)	1329
29.260.3.26	length(double tol)	1329
29.260.3.27	point(Point &pt, double tpar) const	1329
29.260.3.28	point(std::vector< Point > &pts, double tpar, int derivs, bool from_right=true) const	1329
29.260.3.29	read(std::istream &is)	1330
29.260.3.30	setParamBounds(double startpar, double endpar)	1330
29.260.3.31	setParameterInterval(double t1, double t2)	1330
29.260.3.32	startparam() const	1330
29.260.3.33	subCurve(double from_par, double to_par, double fuzzy=DEFAULT_PARAMETER_EPSILON) const	1330
29.260.3.34	swapParameters2D()	1331
29.260.3.35	translateCurve(const Point &dir)	1331
29.260.3.36	write(std::ostream &os) const	1331
29.260.4	Member Data Documentation	1331
29.260.4.1	dir_	1331
29.260.4.2	endparam_	1331
29.260.4.3	location_	1332
29.260.4.4	startparam_	1332
29.260	Go::Line2DInt Class Reference	1332
29.261.1	Detailed Description	1333
29.261.2	Constructor & Destructor Documentation	1333

29.261.2.1	Line2DInt(Point point, Point dir)	1333
29.261.2.2	Line2DInt(double a, double b, double c)	1334
29.261.2.3	~Line2DInt()	1334
29.261.3	Member Function Documentation	1334
29.261.3.1a()		1334
29.261.3.2b()		1334
29.261.3.3c()		1334
29.262	NEWMAT::LinearEquationSolver Class Reference	1335
29.262.1	Detailed Description	1336
29.262.2	Constructor & Destructor Documentation	1336
29.262.2.1	LinearEquationSolver(const BaseMatrix &bm)	1336
29.262.2.2	~LinearEquationSolver()	1336
29.262.3	Member Function Documentation	1336
29.262.3.1	CleanUp()	1336
29.262.3.2	Evaluate(MatrixType)	1336
29.262.4	Friends And Related Function Documentation	1336
29.262.4.1	BaseMatrix	1336
29.263	Go::LineCloud Class Reference	1337
29.263.1	Detailed Description	1338
29.263.2	Constructor & Destructor Documentation	1338
29.263.2.1	LineCloud()	1338
29.263.2.2	LineCloud(ForwardIterator start, int numlines)	1338
29.263.2.3	~LineCloud()	1339
29.263.3	Member Function Documentation	1339
29.263.3.1	boundingBox() const	1339
29.263.3.2	classType()	1339
29.263.3.3	clone() const	1339
29.263.3.4	dimension() const	1339
29.263.3.5	instanceType() const	1340
29.263.3.6	numLines() const	1340

29.263.3.7	point(int i)	1340
29.263.3.8	point(int i) const	1340
29.263.3.9	rawData()	1341
29.263.3.10	read(std::istream &is)	1341
29.263.3.11	setCloud(const double *points, int numlines)	1341
29.263.3.12	write(std::ostream &os) const	1341
29.264	Go::LineCloudTesselator Class Reference	1342
29.264.1	Detailed Description	1343
29.264.2	Constructor & Destructor Documentation	1343
29.264.2.1	LineCloudTesselator(const Go::LineCloud &lc)	1343
29.264.2.2	~LineCloudTesselator()	1343
29.264.3	Member Function Documentation	1343
29.264.3.1	getRenderCloud()	1343
29.264.3.2	getScale()	1343
29.264.3.3	setScale(double scale)	1343
29.264.3.4	tesselate()	1343
29.265	Go::LineStrip Class Reference	1344
29.265.1	Detailed Description	1345
29.265.2	Constructor & Destructor Documentation	1345
29.265.2.1	LineStrip(int n=200)	1345
29.265.2.2	~LineStrip()	1345
29.265.3	Member Function Documentation	1345
29.265.3.1	asLineStrip()	1345
29.265.3.2	atBoundary(int idx)	1345
29.265.3.3	numVertices()	1345
29.265.3.4	paramArray()	1346
29.265.3.5	resize(int n)	1346
29.265.3.6	stripArray()	1346
29.265.3.7	triangleIndexArray()	1346
29.265.3.8	vertexArray()	1346

29.266	NEWMAT::LL_D_FI Class Reference	1347
29.266.1	Detailed Description	1348
29.266.2	Constructor & Destructor Documentation	1348
29.266.2.1	~LL_D_FI()	1348
29.266.3	Member Function Documentation	1348
29.266.3.1	Derivatives()=0	1348
29.266.3.2	FI()=0	1348
29.266.3.3	IsValid()	1348
29.266.3.4	IsValid(const ColumnVector &X, bool wgx=true)	1348
29.266.3.5	LogLikelihood()=0	1348
29.266.3.6	LogLikelihood(const ColumnVector &X, bool wgx=true)	1348
29.266.3.7	Set(const ColumnVector &X)	1348
29.266.3.8	WG(bool wgx)	1348
29.266.4	Member Data Documentation	1348
29.266.4.1	para	1348
29.266.4.2	wg	1349
29.267	LoadAndStoreFlag Class Reference	1349
29.267.1	Detailed Description	1350
29.267.2	Constructor & Destructor Documentation	1350
29.267.2.1	LoadAndStoreFlag()	1350
29.267.2.2	LoadAndStoreFlag(int i)	1350
29.267.2.3	LoadAndStoreFlag(LSF lsf)	1350
29.267.2.4	LoadAndStoreFlag(const ControlWord &cw)	1350
29.268	Bo::LockedDirDistFunc Class Reference	1350
29.268.1	Detailed Description	1351
29.268.2	Constructor & Destructor Documentation	1351
29.268.2.1	LockedDirDistFunc(const ParamObjectInt *const o1, const ParamObjectInt *const o2, int fixed_dir, double fixed_value)	1351
29.268.3	Member Function Documentation	1351
29.268.3.1	grad(const double *arg, double *grad) const	1351
29.268.3.2	maxPar(int n) const	1351

29.268.3.3	<code>minPar(int n) const</code>	1351
29.268.3.4	<code>operator()(const double *arg) const</code>	1351
29.269	<code>NEWMAT::LogAndSign</code> Class Reference	1351
29.269.1	Detailed Description	1352
29.269.2	Constructor & Destructor Documentation	1352
29.269.2.1	<code>LogAndSign()</code>	1352
29.269.2.2	<code>LogAndSign(Real)</code>	1352
29.269.3	Member Function Documentation	1352
29.269.3.1	<code>ChangeSign()</code>	1352
29.269.3.2	<code>LogValue() const</code>	1352
29.269.3.3	<code>operator*=(Real)</code>	1352
29.269.3.4	<code>PowEq(int k)</code>	1352
29.269.3.5	<code>Sign() const</code>	1352
29.269.3.6	<code>Value() const</code>	1352
29.270	<code>RBD_COMMON::Logic_error</code> Class Reference	1353
29.270.1	Detailed Description	1354
29.270.2	Constructor & Destructor Documentation	1354
29.270.2.1	<code>Logic_error(const char *a_what=0)</code>	1354
29.270.3	Member Data Documentation	1354
29.270.3.1	<code>Select</code>	1354
29.271	<code>Go::Loop</code> Class Reference	1354
29.271.1	Detailed Description	1355
29.271.2	Constructor & Destructor Documentation	1356
29.271.2.1	<code>Loop(ftFaceBase *face, CurveLoop &curve_loop, double kink, bool split_in_↔ kinks=true, bool no_split=false)</code>	1356
29.271.2.2	<code>Loop(ftFaceBase *face, std::vector< shared_ptr< ftEdgeBase > > &edges, dou- ble space_epsilon)</code>	1356
29.271.2.3	<code>Loop(std::vector< shared_ptr< ftEdgeBase > > &edges, double space_epsilon)</code>	1356
29.271.2.4	<code>~Loop()</code>	1356
29.271.3	Member Function Documentation	1356
29.271.3.1	<code>allRadialEdges() const</code>	1356

29.271.3.2	<code>checkConsistency()</code> const	1356
29.271.3.3	<code>closestPoint(const Point &pt, int &clo_ind, double &clo_par, Point &clo_pt, double &clo_dist)</code> const	1356
29.271.3.4	<code>correspondingEdges(shared_ptr< Loop > other, double tol, ftEdgeBase *&first1, ftEdgeBase *&first2, bool &same_dir, bool no_snap=true)</code>	1357
29.271.3.5	<code>getAcuteEdges(std::vector< std::pair< ftEdge *, ftEdge * > > &acute_edges, double angtol)</code> const	1357
29.271.3.6	<code>getBadDistance(std::vector< std::pair< ftSurface *, ftEdge * > > &badPairs, RectDomain *domain, double tol)</code> const	1357
29.271.3.7	<code>getBadDistance(std::vector< std::pair< ftEdge *, shared_ptr< Vertex > > > &badPairs, double tol)</code> const	1357
29.271.3.8	<code>getEdge(size_t idx)</code>	1357
29.271.3.9	<code>getEdges()</code>	1357
29.271.3.10	<code>getFace()</code>	1357
29.271.3.11	<code>getLoopIntersections(shared_ptr< Loop > loop2, double tol, std::vector< std::pair< shared_ptr< PointOnEdge >, shared_ptr< PointOnEdge > > > &int_pt)</code> const	1357
29.271.3.12	<code>getLoopSelfIntersections(double tol, std::vector< std::pair< shared_ptr< PointOnEdge >, shared_ptr< PointOnEdge > > > &int_pt)</code> const	1358
29.271.3.13	<code>getPosTangentSurfaceDiscont(std::vector< ftEdge * > &badPos, std::vector< ftEdge * > &badTangent, double tol, double kink, double bend, int leastSurfaceIndex, shared_ptr< SurfaceModel > sm)</code> const	1358
29.271.3.14	<code>getSeqVertices()</code> const	1358
29.271.3.15	<code>getTol()</code>	1358
29.271.3.16	<code>getVertices()</code> const	1358
29.271.3.17	<code>groupSmoothEdges(double tol, double angtol, std::vector< std::vector< shared_ptr< ftEdgeBase > > > &edge_groups)</code>	1358
29.271.3.18	<code>hasRadialEdges()</code> const	1358
29.271.3.19	<code>Close(ftEdge *edge, RectDomain *domain, double tol)</code> const	1358
29.271.3.20	<code>FaceConsistent()</code>	1358
29.271.3.21	<code>InLoop(ftEdgeBase *edge)</code>	1359
29.271.3.22	<code>isFace(ftFaceBase *face)</code>	1359
29.271.3.23	<code>Size()</code>	1359
29.271.3.24	<code>split(int ind, double par)</code>	1359
29.271.3.25	<code>updateLoop(shared_ptr< ftEdgeBase > new_edge)</code>	1359

29.272	NEWMAT::LowerBandMatrix Class Reference	1359
29.272.1	Detailed Description	1362
29.272.2	Constructor & Destructor Documentation	1362
29.272.2.1	LowerBandMatrix()	1362
29.272.2.2	~LowerBandMatrix()	1362
29.272.2.3	LowerBandMatrix(int n, int lbw)	1362
29.272.2.4	LowerBandMatrix(const BaseMatrix &)	1362
29.272.2.5	LowerBandMatrix(const LowerBandMatrix &gm)	1362
29.272.3	Member Function Documentation	1362
29.272.3.1	element(int, int)	1362
29.272.3.2	element(int, int) const	1362
29.272.3.3	LogDeterminant() const	1362
29.272.3.4	MakeSolver()	1362
29.272.3.5	operator()(int, int)	1363
29.272.3.6	operator()(int, int) const	1363
29.272.3.7	operator=(const BaseMatrix &)	1363
29.272.3.8	operator=(Real f)	1363
29.272.3.9	operator=(const LowerBandMatrix &m)	1363
29.272.3.10	ReSize(int, int, int)	1363
29.272.3.11	ReSize(int n, int lbw)	1363
29.272.3.12	ReSize(const GeneralMatrix &A)	1363
29.272.3.13	Solver(MatrixColX &, const MatrixColX &)	1363
29.272.3.14	Type() const	1363
29.273	NEWMAT::LowerTriangularMatrix Class Reference	1364
29.273.1	Detailed Description	1365
29.273.2	Constructor & Destructor Documentation	1365
29.273.2.1	LowerTriangularMatrix()	1365
29.273.2.2	~LowerTriangularMatrix()	1365
29.273.2.3	LowerTriangularMatrix(ArrayLengthSpecifier)	1366
29.273.2.4	LowerTriangularMatrix(const LowerTriangularMatrix &gm)	1366

29.273.2.5	LowerTriangularMatrix(const BaseMatrix &M)	1366
29.273.3	Member Function Documentation	1366
29.273.3.1	BandWidth() const	1366
29.273.3.2	element(int, int)	1366
29.273.3.3	element(int, int) const	1366
29.273.3.4	GetCol(MatrixRowCol &)	1366
29.273.3.5	GetCol(MatrixColX &)	1366
29.273.3.6	GetRow(MatrixRowCol &)	1366
29.273.3.7	LogDeterminant() const	1366
29.273.3.8	MakeSolver()	1367
29.273.3.9	NextRow(MatrixRowCol &)	1367
29.273.3.10	operator()(int, int)	1367
29.273.3.11	operator()(int, int) const	1367
29.273.3.12	operator=(const BaseMatrix &)	1367
29.273.3.13	operator=(Real f)	1367
29.273.3.14	operator=(const LowerTriangularMatrix &m)	1367
29.273.3.15	ReSize(int)	1367
29.273.3.16	ReSize(const GeneralMatrix &A)	1367
29.273.3.17	RestoreCol(MatrixRowCol &)	1367
29.273.3.18	RestoreCol(MatrixColX &c)	1367
29.273.3.19	Solver(MatrixColX &, const MatrixColX &)	1368
29.273.3.20	Trace() const	1368
29.273.3.21	Type() const	1368
29.274	Go::LRBSpline2D Class Reference	1368
29.274.1	Detailed Description	1370
29.274.2	Constructor & Destructor Documentation	1370
29.274.2.1	LRBSpline2D()	1370
29.274.2.2	LRBSpline2D(const Point &c_g, double weight, int deg_u, int deg_v, Iterator kvec_u_start, Iterator kvec_v_start, double gamma, const Mesh2D *mesh, bool rational=false)	1371
29.274.2.3	LRBSpline2D(const LRBSpline2D &rhs)	1371

29.274.2.4~LRBSpline2D()	1371
29.274.3 Member Function Documentation	1371
29.274.3.1 addSupport(Element2D *el)	1371
29.274.3.2 Coef()	1371
29.274.3.3 Coef() const	1371
29.274.3.4 coefFixed() const	1371
29.274.3.5 coefTimesGamma()	1371
29.274.3.6 coefTimesGamma() const	1371
29.274.3.7 coversCorner(int u_ix, int v_ix) const	1371
29.274.3.8 degree(Direction2D d) const	1372
29.274.3.9 dimension() const	1372
29.274.3.10 endmult_u(bool atstart) const	1372
29.274.3.11 endmult_v(bool atstart) const	1372
29.274.3.12 eval(double u, double v, int u_deriv=0, int v_deriv=0, bool u_at_end=false, bool v_at_end=false) const	1372
29.274.3.13 evalBasisFunc(double u, double v) const	1372
29.274.3.14 evalBasisFunction(double u, double v, int u_deriv=0, int v_deriv=0, bool u_at_end=false, bool v_at_end=false) const	1372
29.274.3.15 evalBasisGridDer(int nmb_der, const std::vector< double > &par1, const std::vector< double > &par2, std::vector< double > &derivs) const	1372
29.274.3.16 evalBasisLineDer(int nmb_der, Direction2D d, const std::vector< double > &parval, std::vector< double > &derivs) const	1372
29.274.3.17 evalpos(double u, double v, double pos[])	1372
29.274.3.18 gamma()	1372
29.274.3.19 gamma() const	1373
29.274.3.20 getExtendedSupport()	1373
29.274.3.21 getGrevilleParameter() const	1373
29.274.3.22 getGrevilleParameter(Direction2D d) const	1373
29.274.3.23 getMesh()	1373
29.274.3.24 getMinimalExtendedSupport()	1373
29.274.3.25 hasSupportedElement(Element2D *el)	1373
29.274.3.26 vec(Direction2D d) const	1373

29.274.3.27	<code>vec(Direction2D d)</code>	1373
29.274.3.28	<code>mbSupportedElements()</code>	1373
29.274.3.29	<code>operator<(const LRBSpline2D &rhs) const</code>	1373
29.274.3.30	<code>operator==(const LRBSpline2D &rhs) const</code>	1373
29.274.3.31	<code>overlaps(Element2D *el) const</code>	1373
29.274.3.32	<code>overlaps(double domain[]) const</code>	1373
29.274.3.33	<code>rational() const</code>	1373
29.274.3.34	<code>read(std::istream &is)</code>	1374
29.274.3.35	<code>removeSupport(Element2D *el)</code>	1374
29.274.3.36	<code>reverseParameterDirection(bool dir_is_u)</code>	1374
29.274.3.37	<code>setCoefAndGamma(Point &coef, double gamma)</code>	1374
29.274.3.38	<code>setFixCoef(int coef_fixed)</code>	1374
29.274.3.39	<code>setMesh(const Mesh2D *mesh)</code>	1374
29.274.3.40	<code>setSupport(std::vector< Element2D * > elements)</code>	1374
29.274.3.41	<code>subtractKnotIdx(int u_del, int v_del)</code>	1374
29.274.3.42	<code>suppMax(Direction2D d) const</code>	1374
29.274.3.43	<code>suppMin(Direction2D d) const</code>	1374
29.274.3.44	<code>supportedElementBegin()</code>	1374
29.274.3.45	<code>supportedElementEnd()</code>	1374
29.274.3.46	<code>supportedElements()</code>	1374
29.274.3.47	<code>swap(LRBSpline2D &rhs)</code>	1375
29.274.3.48	<code>swapParameterDirection()</code>	1375
29.274.3.49	<code>max() const</code>	1375
29.274.3.50	<code>min() const</code>	1375
29.274.3.51	<code>initSquareBernsteinBasis(double start_u, double stop_u, double start_v, double stop_v) const</code>	1375
29.274.3.52	<code>max() const</code>	1375
29.274.3.53	<code>min() const</code>	1375
29.274.3.54	<code>weight()</code>	1376
29.274.3.55	<code>weight() const</code>	1376
29.274.3.56	<code>write(std::ostream &os) const</code>	1376

29.275.0	Go::LRSplineEvalGrid Class Reference	1376
29.275.1	Detailed Description	1377
29.275.2	Member Typedef Documentation	1377
29.275.2.1	param_float_type	1377
29.275.3	Constructor & Destructor Documentation	1377
29.275.3.1	LRSplineEvalGrid()	1377
29.275.3.2	LRSplineEvalGrid(LRSplineSurface &lr_spline)	1377
29.275.4	Member Function Documentation	1377
29.275.4.1	computeBezCoefs(int dim, const double *points, int orderU, int orderV, double *coefs)	1377
29.275.4.2	dim() const	1377
29.275.4.3	elements_begin()	1377
29.275.4.4	elements_end()	1377
29.275.4.5	evaluate(Element2D &elem, double u, double v, double *res) const	1377
29.275.4.6	evaluateGrid(V &element, double *points)	1377
29.275.4.7	high(const Element2D &e, double &u, double &v)	1378
29.275.4.8	low(const Element2D &e, double &u, double &v)	1378
29.275.4.9	numElements() const	1378
29.275.4.10	orderU() const	1378
29.275.4.11	orderV() const	1378
29.275.4.12	origDom()	1378
29.275.4.13	testCoefComputation()	1378
29.276.0	Go::LRSplineSurface Class Reference	1379
29.276.1	Detailed Description	1383
29.276.2	Member Typedef Documentation	1383
29.276.2.1	BSplineMap	1383
29.276.2.2	ElementMap	1383
29.276.3	Constructor & Destructor Documentation	1383
29.276.3.1	LRSplineSurface(int deg_u, int deg_v, int coefs_u, int coefs_v, int dimension, KnotIterator knotvals_u_start, KnotIterator knotvals_v_start, CoefIterator coefs_start, double knot_tol=1.0e-8)	1383

29.276.3.2.LRSplineSurface(int deg_u, int deg_v, int coefs_u, int coefs_v, int dimension, KnotIterator knotvals_u_start, KnotIterator knotvals_v_start, double knot_tol=1e-8)	1384
29.276.3.3.LRSplineSurface(SplineSurface *surf, double knot_tol)	1384
29.276.3.4.LRSplineSurface()	1384
29.276.3.5.LRSplineSurface(const LRSplineSurface &rhs)	1384
29.276.3.6.LRSplineSurface()	1384
29.276.3.7.LRSplineSurface(std::istream &is)	1384
29.276.4.Member Function Documentation	1384
29.276.4.1.addSurface(const LRSplineSurface &other_sf, double fac=1.0)	1384
29.276.4.2.allBoundaryLoops(double degenerate_epsilon=DEFAULT_SPACE_EPSILON) const	1384
29.276.4.3.area(double tol) const	1384
29.276.4.4.asSplineSurface()	1385
29.276.4.5.basisFunctionsBegin() const	1385
29.276.4.6.basisFunctionsBeginNonconst()	1385
29.276.4.7.basisFunctionsEnd() const	1385
29.276.4.8.basisFunctionsEndNonconst()	1385
29.276.4.9.basisFunctionsWithSupportAt(double u, double v) const	1385
29.276.4.10.boundingBox() const	1385
29.276.4.11.splineFromDomain(double start_u, double start_v, double end_u, double end_v, int startmult_u, int startmult_v, int endmult_u, int endmult_v)	1386
29.276.4.12.classType()	1386
29.276.4.13.done() const	1386
29.276.4.14.closestBoundaryPoint(const Point &pt, double &clo_u, double &clo_v, Point &clo_pt, double &clo_dist, double epsilon, const RectDomain *rd=NULL, double *seed=0) const	1386
29.276.4.15.closestInDomain(double u, double v) const	1386
29.276.4.16.closestPoint(const Point &pt, double &clo_u, double &clo_v, Point &clo_pt, double &clo_dist, double epsilon, int maxiter, Element2D *elem=NULL, const RectDomain *rd=NULL, double *seed=NULL) const	1386
29.276.4.17.compositeBox() const	1387
29.276.4.18.computeBasis(double param_u, double param_v, BasisPtsSf &result, int iEI=-1) const	1387

29.276.4.19	computeBasis(double param_u, double param_v, BasisDerivsSf &result, int iEl=-1) const	1387
29.276.4.20	computeBasis(double param_u, double param_v, BasisDerivsSf2 &result, int iEl=-1) const	1387
29.276.4.21	computeBasis(double param_u, double param_v, std::vector< std::vector< double > > &result, int derivs=0, int iEl=-1) const	1387
29.276.4.22	ConstParamCurve(double parameter, bool paddir_is_u) const	1387
29.276.4.23	ConstParamCurve(double parameter, bool paddir_is_u, SplineCurve *&cv, SplineCurve *&crosscv) const	1387
29.276.4.24	ConstParamCurves(double parameter, bool paddir_is_u) const	1387
29.276.4.25	ConstructElementMesh(std::vector< Element2D * > &elements) const	1388
29.276.4.26	ContainingDomain() const	1388
29.276.4.27	CoveringElement(double u, double v) const	1388
29.276.4.28	Degree(Direction2D d) const	1388
29.276.4.29	Dimension() const	1388
29.276.4.30	EdgeCurve(int edge_num) const	1388
29.276.4.31	ElementsBegin() const	1388
29.276.4.32	ElementsEnd() const	1389
29.276.4.33	Endparam_u() const	1389
29.276.4.34	Endparam_v() const	1389
29.276.4.35	EvalGrid(int num_u, int num_v, double umin, double umax, double vmin, double vmax, std::vector< double > &points, double nodata_val=-9999) const	1389
29.276.4.36	ExpandToFullTensorProduct()	1389
29.276.4.37	Generate_key(const LRBSpline2D &b, const Mesh2D &m)	1389
29.276.4.38	Generate_key(const LRBSpline2D &b)	1389
29.276.4.39	Generate_key(const double &, const double &)	1389
29.276.4.40	GetBoundaryBsplines(Direction2D d, bool atstart)	1390
29.276.4.41	GetBoundaryIdx(Point &pt1, Point &pt2, double epsilon, int &bindex, double &par1, double &par2, double knot_tol=1e-05) const	1390
29.276.4.42	GetBoundaryInfo(Point &pt1, Point &pt2, double epsilon, SplineCurve *&cv, SplineCurve *&crosscv, double knot_tol=1e-05) const	1390
29.276.4.43	GetBoundaryInfo(double par1, double par2, int bindex, SplineCurve *&cv, SplineCurve *&crosscv, double knot_tol) const	1391
29.276.4.44	GetCornerPoints(std::vector< std::pair< Point, Point > > &corners) const	1391

29.276.4.45	GetDegenerateCorners(std::vector< Point > °_corners, double tol) const	1391
29.276.4.46	GetElementBds(int num_pts=5) const	1391
29.276.4.47	GetElementContaining(double u, double v) const	1391
29.276.4.48	GetKnotTol()	1391
29.276.4.49	Domain(double u, double v, double eps=1.0e-4) const	1391
29.276.4.50	Domain2(double u, double v, double eps=1.0e-4) const	1391
29.276.4.51	InstanceType() const	1391
29.276.4.52	Degenerate(bool &b, bool &r, bool &t, bool &l, double tolerance) const	1391
29.276.4.53	FullTensorProduct() const	1392
29.276.4.54	Mesh() const	1392
29.276.4.55	MirrorSurface(const Point &pos, const Point &norm) const	1392
29.276.4.56	NextSegmentVal(int dir, double par, bool forward, double tol) const	1392
29.276.4.57	Normal(Point &n, double upar, double vpar) const	1393
29.276.4.58	Normal(Point &n, double upar, double vpar, Element2D *elem) const	1393
29.276.4.59	NormalCone() const	1393
29.276.4.60	NumBasisFunctions() const	1393
29.276.4.61	NumElements() const	1393
29.276.4.62	InBoundary(double u, double v, double eps=1.0e-4) const	1393
29.276.4.63	Operator()(double u, double v, int u_deriv=0, int v_deriv=0) const	1393
29.276.4.64	Operator()(double u, double v, int u_deriv, int v_deriv, Element2D *elem) const	1393
29.276.4.65	Operator=(const LRSplineSurface &other)	1393
29.276.4.66	OuterBoundaryLoop(double degenerate_epsilon=DEFAULT_SPACE_EPSILON) const	1393
29.276.4.67	ParameterDomain() const	1394
29.276.4.68	ParamMax(Direction2D d) const	1394
29.276.4.69	ParamMin(Direction2D d) const	1394
29.276.4.70	Point(Point &pt, double upar, double vpar) const	1394
29.276.4.71	Point(Point &pt, double upar, double vpar, Element2D *elem) const	1394
29.276.4.72	Point(std::vector< Point > &pts, double upar, double vpar, int derivs, bool u_↔ from_right=true, bool v_from_right=true, double resolution=1.0e-12) const	1395

29.276.4.70	Point(std::vector< Point > &pts, double upar, double vpar, int derivs, Element2D *elem, bool u_from_right=true, bool v_from_right=true, double resolution=1.0e-12) const	1395
29.276.4.71	Point() const	1395
29.276.4.72	Read(std::istream &is)	1395
29.276.4.73	Refine(Direction2D d, double fixed_val, double start, double end, int mult=1, bool absolute=false)	1395
29.276.4.74	Refine(const Refinement2D &ref, bool absolute=false)	1395
29.276.4.75	Refine(const std::vector< Refinement2D > &refs, bool absolute=false)	1395
29.276.4.76	RefineBasisFunction(int index)	1396
29.276.4.77	RefineBasisFunction(const std::vector< int > &indices)	1396
29.276.4.78	RefineElement(int index)	1396
29.276.4.79	RefineElement(const std::vector< int > &indices)	1396
29.276.4.80	ReverseParameterDirection(bool direction_is_u)	1396
29.276.4.81	SetCoef(const Point &value, const LRBSpline2D *target)	1396
29.276.4.82	SetCoef(const Point &value, int umin_ix, int vmin_ix, int umax_ix, int vmax_ix, int u_mult=1, int v_mult=1)	1396
29.276.4.83	SetCoefTimesGamma(const Point &value, const LRBSpline2D *target)	1396
29.276.4.84	SetCurrentElement(Element2D *curr_el)	1396
29.276.4.85	SetParameterDomain(double u1, double u2, double v1, double v2)	1396
29.276.4.86	Startparam_u() const	1397
29.276.4.87	Startparam_v() const	1397
29.276.4.88	SubSurface(double from_upar, double from_vpar, double to_upar, double to_vpar, double fuzzy) const	1397
29.276.4.89	SubSurfaces(double from_upar, double from_vpar, double to_upar, double to_vpar, double fuzzy=DEFAULT_PARAMETER_EPSILON) const	1397
29.276.4.90	Swap(LRSplineSurface &rhs)	1397
29.276.4.91	SwapParameterDirection()	1397
29.276.4.92	TangentCone(bool paddir_is_u) const	1398
29.276.4.93	3D()	1398
29.276.4.94	Translate(const Point &vec)	1398
29.276.4.95	TurnOrientation()	1398
29.276.4.96	Write(std::ostream &os) const	1398

29.277.0	Go::LRSurfApprox Class Reference	1398
29.277.1	Detailed Description	1399
29.277.2	Constructor & Destructor Documentation	1399
29.277.2.1	LRSurfApprox(std::vector< double > &points, int dim, double epsge, bool init_mba=false, double mba_level=0.0, bool closest_dist=true, bool repar=false)	1399
29.277.2.2	LRSurfApprox(shared_ptr< SplineSurface > &srf, std::vector< double > &points, double epsge, bool closest_dist=true, bool repar=false)	1400
29.277.2.3	LRSurfApprox(shared_ptr< LRSplineSurface > &srf, std::vector< double > &points, double epsge, bool closest_dist=true, bool repar=false, bool check_init_accuracy=false)	1400
29.277.2.4	LRSurfApprox(int ncoef_u, int order_u, int ncoef_v, int order_v, std::vector< double > &points, int dim, double epsge, bool init_mba=false, double mba_level=0.0, bool closest_dist=true, bool repar=false)	1400
29.277.2.5	LRSurfApprox(int order_u, std::vector< double > &knots_u, int order_v, std::vector< double > &knots_v, std::vector< double > &points, int dim, double epsge, bool init_mba=false, double mba_level=0.0, bool closest_dist=true, bool repar=false)	1401
29.277.2.6	LRSurfApprox(int ncoef_u, int order_u, int ncoef_v, int order_v, std::vector< double > &points, int dim, double domain[4], double epsge, bool init_mba=false, double mba_level=0.0, bool closest_dist=true, bool repar=false)	1401
29.277.2.7	~LRSurfApprox()	1402
29.277.3	Member Function Documentation	1402
29.277.3.1	addLowerConstraint(double minval)	1402
29.277.3.2	addUpperConstraint(double maxval)	1402
29.277.3.3	edgeFix(int edge_fix[])	1402
29.277.3.4	getApproxSurf(double &maxdist, double &avdist_all, double &avdist, int &nmb_out_eps, int max_iter=4)	1402
29.277.3.5	setFixBoundary(bool fix_boundary)	1403
29.277.3.6	setFixCorner(bool fix_corner)	1403
29.277.3.7	setGridInfo(double grid_start[], double cell_size[])	1403
29.277.3.8	setInitMBA(bool initMBA, double start_coef=0.0)	1403
29.277.3.9	setLocalConstraint(double constraint_factor)	1403
29.277.3.10	setMakeGhostPoints(bool make_ghost_points)	1404
29.277.3.11	setMinimumElementSize(double usize_min, double vsize_min)	1404
29.277.3.12	setSmoothBoundary(bool smoothbd)	1404
29.277.3.13	setSmoothingWeight(double smooth)	1404

29.277.3.1	setSwitchToMBA(int iter)	1404
29.277.3.1	setTurn3D(int iter)	1404
29.277.3.1	setUseMBA(bool useMBA)	1404
29.277.3.1	setVerbose(bool verbose)	1405
29.278	Go::LRSurfSmoothLS Class Reference	1405
29.278.1	Detailed Description	1405
29.278.2	Constructor & Destructor Documentation	1405
29.278.2.1	LRSurfSmoothLS(shared_ptr< LRSplineSurface > surf, std::vector< int > &coef_known)	1405
29.278.2.2	LRSurfSmoothLS()	1406
29.278.2.3	~LRSurfSmoothLS()	1406
29.278.3	Member Function Documentation	1406
29.278.3.1	addDataPoints(std::vector< double > &points, bool is_ghost_points=false)	1406
29.278.3.2	equationSolve(shared_ptr< LRSplineSurface > &surf)	1406
29.278.3.3	hasDataPoints() const	1406
29.278.3.4	setInitSf(shared_ptr< LRSplineSurface > surf, std::vector< int > &coef_known)	1406
29.278.3.5	setLeastSquares(const double weight)	1407
29.278.3.6	setLeastSquares(std::vector< double > &points, const double weight)	1407
29.278.3.7	setLeastSquares_omp(const double weight)	1407
29.278.3.8	setOptimize(const double weight1, const double weight2, const double weight3)	1407
29.278.3.9	smoothBoundary(const double weight1, const double weight2, const double weight3)	1407
29.278.3.10	updateLocals()	1408
29.279	Go::LSSmoothData Struct Reference	1408
29.279.1	Detailed Description	1409
29.279.2	Constructor & Destructor Documentation	1409
29.279.2.1	LSSmoothData()	1409
29.279.3	Member Function Documentation	1409
29.279.3.1	addDataPoints(std::vector< double >::iterator start, std::vector< double >::iterator end, bool sort_in_u)	1409
29.279.3.2	addDataPoints(std::vector< double >::iterator start, std::vector< double >::iterator end, int del, bool sort_in_u)	1410

29.279.3.3	<code>addGhostPoints(std::vector< double >::iterator start, std::vector< double >::iterator end, bool sort_in_u)</code>	1410
29.279.3.4	<code>addGhostPoints(std::vector< double >::iterator start, std::vector< double >::iterator end, int del, bool sort_in_u)</code>	1410
29.279.3.5	<code>dataPointSize()</code>	1410
29.279.3.6	<code>eraseDataPoints()</code>	1410
29.279.3.7	<code>eraseDataPoints(std::vector< double >::iterator start, std::vector< double >::iterator end)</code>	1410
29.279.3.8	<code>eraseGhostPoints()</code>	1410
29.279.3.9	<code>getAccumulatedError()</code>	1410
29.279.3.10	<code>getAccuracyInfo(double &average_error, double &max_error, int &nmb_outside_tol)</code>	1410
29.279.3.11	<code>getAverageError()</code>	1411
29.279.3.12	<code>getDataBoundingBox(int dim, double bb[])</code>	1411
29.279.3.13	<code>getDataPoints()</code>	1411
29.279.3.14	<code>getGhostPoints()</code>	1411
29.279.3.15	<code>getLSMatrix(double *LSmat, double *LSright, int &ncond)</code>	1411
29.279.3.16	<code>getMaxError()</code>	1411
29.279.3.17	<code>getNmbOutsideTol()</code>	1411
29.279.3.18	<code>getOutsideGhostPoints(std::vector< double > &ghost, int dim, Direction2D d, double start, double end, bool &sort_in_u)</code>	1411
29.279.3.19	<code>getOutsidePoints(std::vector< double > &points, int dim, Direction2D d, double start, double end, bool &sort_in_u)</code>	1411
29.279.3.20	<code>ghostPointSize()</code>	1411
29.279.3.21	<code>hasAccuracyInfo()</code>	1411
29.279.3.22	<code>hasDataPoints()</code>	1411
29.279.3.23	<code>hasLSMatrix()</code>	1412
29.279.3.24	<code>makeDataPoints3D(int dim)</code>	1412
29.279.3.25	<code>setAccuracyInfo()</code>	1412
29.279.3.26	<code>setAccuracyInfo(double accumulated_error, double average_error, double max_error, int nmb_outside_tol)</code>	1412
29.279.3.27	<code>setLSMatrix(int nmb, int dim)</code>	1412
29.279.3.28	<code>updateAccuracyInfo(int dim)</code>	1412

29.279.3.2	updateLSDataParDomain(double u1, double u2, double v1, double v2, double u1new, double u2new, double v1new, double v2new, int dim)	1412
29.279.4	Member Data Documentation	1412
29.279.4.1	accumulated_error_	1412
29.279.4.2	average_error_	1412
29.279.4.3	data_points_	1412
29.279.4.4	ghost_points_	1412
29.279.4.5	Smat_	1413
29.279.4.6	Sright_	1413
29.279.4.7	max_error_	1413
29.279.4.8	max_error_prev_	1413
29.279.4.9	cond_	1413
29.279.4.10	mb_outside_tol_	1413
29.279.4.11	sort_in_u_	1413
29.279.4.12	sort_in_u_ghost_	1413
29.280	Go::MarchPoint Class Reference	1414
29.280.1	Detailed Description	1414
29.280.2	Constructor & Destructor Documentation	1414
29.280.2.1	MarchPoint(const Go::Point &p, double pa, double pa2, double d, double ca, int s) 1414	
29.280.3	Member Function Documentation	1415
29.280.3.1	operator<(const MarchPoint &other) const	1415
29.280.4	Member Data Documentation	1415
29.280.4.1	cos_ang	1415
29.280.4.2	dist	1415
29.280.4.3	par	1415
29.280.4.4	par2	1415
29.280.4.5	pt	1415
29.280.4.6	status	1415
29.281	NEWMAT::MatedMatrix Class Reference	1416
29.281.1	Detailed Description	1417
29.281.2	Constructor & Destructor Documentation	1417

29.281.2.1~MatedMatrix()	1417
29.281.3Member Function Documentation	1418
29.281.3.1BandWidth() const	1418
29.281.3.2Evaluate(MatrixType mt=MatrixTypeUnSp)	1418
29.281.4Friends And Related Function Documentation	1418
29.281.4.1BaseMatrix	1418
29.282material_appearance Class Reference	1418
29.282.1Detailed Description	1419
29.282.2Constructor & Destructor Documentation	1419
29.282.2.1material_appearance(void)	1419
29.282.3Member Function Documentation	1419
29.282.3.1col_scheme_select(const int i)	1419
29.282.3.2col_scheme_selected(void) const	1419
29.282.3.3Key(unsigned char key)	1419
29.282.3.4redefine_material_f(const vector3t< double > &diff, const vector3t< double > &amb, const vector3t< double > &em, const vector3t< double > &spec, const double shin)	1419
29.282.3.5set_material(void)	1419
29.282.4Member Data Documentation	1419
29.282.4.1ztrans_delta	1419
29.283NEWMAT::Matrix Class Reference	1420
29.283.1Detailed Description	1421
29.283.2Constructor & Destructor Documentation	1421
29.283.2.1Matrix()	1421
29.283.2.2~Matrix()	1422
29.283.2.3Matrix(int, int)	1422
29.283.2.4Matrix(const BaseMatrix &)	1422
29.283.2.5Matrix(const Matrix &gm)	1422
29.283.3Member Function Documentation	1422
29.283.3.1element(int, int)	1422
29.283.3.2element(int, int) const	1422

29.283.3.3	GetCol(MatrixRowCol &)	1422
29.283.3.4	GetCol(MatrixColX &)	1422
29.283.3.5	GetRow(MatrixRowCol &)	1422
29.283.3.6	MakeSolver()	1422
29.283.3.7	Maximum2(int &i, int &j) const	1423
29.283.3.8	MaximumAbsoluteValue2(int &i, int &j) const	1423
29.283.3.9	Minimum2(int &i, int &j) const	1423
29.283.3.10	MinimumAbsoluteValue2(int &i, int &j) const	1423
29.283.3.11	NextCol(MatrixRowCol &)	1423
29.283.3.12	NextCol(MatrixColX &)	1423
29.283.3.13	NextRow(MatrixRowCol &)	1423
29.283.3.14	operator()(int, int)	1424
29.283.3.15	operator()(int, int) const	1424
29.283.3.16	operator=(const BaseMatrix &)	1424
29.283.3.17	operator=(Real f)	1424
29.283.3.18	operator=(const Matrix &m)	1424
29.283.3.19	ReSize(int, int)	1424
29.283.3.20	ReSize(const GeneralMatrix &A)	1424
29.283.3.21	RestoreCol(MatrixRowCol &)	1424
29.283.3.22	RestoreCol(MatrixColX &)	1424
29.283.3.23	Trace() const	1424
29.283.3.24	Type() const	1425
29.283.4	Friends And Related Function Documentation	1425
29.283.4.1	DotProduct	1425
29.284	NEWMAT::MatrixBandWidth Class Reference	1425
29.284.1	Detailed Description	1425
29.284.2	Constructor & Destructor Documentation	1426
29.284.2.1	MatrixBandWidth(const int l, const int u)	1426
29.284.2.2	MatrixBandWidth(const int i)	1426
29.284.3	Member Function Documentation	1426

29.284.3.1	Lower() const	1426
29.284.3.2	minimum(const MatrixBandWidth &) const	1426
29.284.3.3	operator!=(const MatrixBandWidth &bw) const	1426
29.284.3.4	operator*(const MatrixBandWidth &) const	1426
29.284.3.5	operator+(const MatrixBandWidth &) const	1426
29.284.3.6	operator==(const MatrixBandWidth &bw) const	1426
29.284.3.7	() const	1426
29.284.3.8	Upper() const	1427
29.284.4	Member Data Documentation	1427
29.284.4.1	lower	1427
29.284.4.2	upper	1427
29.285	MatrixCol Class Reference	1427
29.285.1	Detailed Description	1428
29.285.2	Constructor & Destructor Documentation	1428
29.285.2.1	MatrixCol(GeneralMatrix *, LoadAndStoreFlag, int=0)	1428
29.285.2.2	MatrixCol(GeneralMatrix *, Real *, LoadAndStoreFlag, int=0)	1428
29.285.2.3	~MatrixCol()	1429
29.285.3	Member Function Documentation	1429
29.285.3.1	Next()	1429
29.286	MatrixColX Class Reference	1429
29.286.1	Detailed Description	1430
29.286.2	Constructor & Destructor Documentation	1430
29.286.2.1	MatrixColX(GeneralMatrix *, Real *, LoadAndStoreFlag, int=0)	1430
29.286.2.2	~MatrixColX()	1430
29.286.3	Member Function Documentation	1431
29.286.3.1	Next()	1431
29.286.4	Member Data Documentation	1431
29.286.4.1	store	1431
29.287	NEWMAT::MatrixInput Class Reference	1431
29.287.1	Detailed Description	1431

29.287.2	Constructor & Destructor Documentation	1431
29.287.2.1	MatrixInput(const MatrixInput &mi)	1431
29.287.2.2	MatrixInput(int nx, Real *rx)	1432
29.287.2.3	~MatrixInput()	1432
29.287.3	Member Function Documentation	1432
29.287.3.1	operator<<(Real)	1432
29.287.3.2	operator<<(int f)	1432
29.287.4	Friends And Related Function Documentation	1432
29.287.4.1	GeneralMatrix	1432
29.288	MatrixRow Class Reference	1432
29.288.1	Detailed Description	1433
29.288.2	Constructor & Destructor Documentation	1433
29.288.2.1	MatrixRow(GeneralMatrix *, LoadAndStoreFlag, int=0)	1433
29.288.2.2	~MatrixRow()	1433
29.288.3	Member Function Documentation	1434
29.288.3.1	Next()	1434
29.289	MatrixRowCol Class Reference	1434
29.289.1	Detailed Description	1436
29.289.2	Constructor & Destructor Documentation	1436
29.289.2.1	MatrixRowCol()	1436
29.289.2.2	~MatrixRowCol()	1436
29.289.3	Member Function Documentation	1436
29.289.3.1	Add(const MatrixRowCol &)	1436
29.289.3.2	Add(const MatrixRowCol &, const MatrixRowCol &)	1436
29.289.3.3	Add(const MatrixRowCol &, Real)	1436
29.289.3.4	Add(Real)	1436
29.289.3.5	AddScaled(const MatrixRowCol &, Real)	1436
29.289.3.6	Check(const MatrixRowCol &)	1437
29.289.3.7	Check()	1437
29.289.3.8	ConCat(const MatrixRowCol &, const MatrixRowCol &)	1437

29.289.3.9	Copy(const MatrixRowCol &)	1437
29.289.3.10	Copy(const Real *&)	1437
29.289.3.10	Copy(Real)	1437
29.289.3.12	CopyCheck(const MatrixRowCol &)	1437
29.289.3.13	Data()	1437
29.289.3.14	crDiag()	1437
29.289.3.15	crId()	1437
29.289.3.16	crLT()	1438
29.289.3.17	crMat()	1438
29.289.3.18	crUT()	1438
29.289.3.19	ject(const MatrixRowCol &)	1438
29.289.3.20	P(const MatrixRowCol &, const MatrixRowCol &)	1438
29.289.3.21	length()	1438
29.289.3.22	length(int i)	1438
29.289.3.23	Maximum1(Real r, int &i)	1438
29.289.3.24	MaximumAbsoluteValue1(Real r, int &i)	1438
29.289.3.25	Minimum1(Real r, int &i)	1438
29.289.3.26	MinimumAbsoluteValue1(Real r, int &i)	1439
29.289.3.27	Multiply(const MatrixRowCol &)	1439
29.289.3.28	Multiply(const MatrixRowCol &, const MatrixRowCol &)	1439
29.289.3.29	Multiply(Real)	1439
29.289.3.30	Multiply(const MatrixRowCol &, Real)	1439
29.289.3.31	negAdd(const MatrixRowCol &, Real)	1439
29.289.3.32	negate(const MatrixRowCol &)	1439
29.289.3.33	evSub(const MatrixRowCol &)	1439
29.289.3.34	kip()	1439
29.289.3.35	kip(int i)	1439
29.289.3.36	Storage()	1440
29.289.3.37	Storage(int i)	1440
29.289.3.38	Sub(const MatrixRowCol &)	1440

29.289.3.3	Sub(const MatrixRowCol &, const MatrixRowCol &)	1440
29.289.3.4	SubRowCol(MatrixRowCol &, int, int) const	1440
29.289.3.4	Sum()	1440
29.289.3.4	SumAbsoluteValue()	1440
29.289.3.4	Zero()	1440
29.289.4	Friends And Related Function Documentation	1440
29.289.4.1	DotProd	1440
29.289.5	Member Data Documentation	1441
29.289.5.1	1cw	1441
29.289.5.2	2data	1441
29.289.5.3	3gm	1441
29.289.5.4	4length	1441
29.289.5.5	5rowcol	1441
29.289.5.6	6skip	1441
29.289.5.7	7storage	1441
29.290	NEWMAT::MatrixType Class Reference	1442
29.290.1	Detailed Description	1443
29.290.2	Member Enumeration Documentation	1443
29.290.2.1	1anonymous enum	1443
29.290.2.2	2Attribute	1444
29.290.3	Constructor & Destructor Documentation	1444
29.290.3.1	1MatrixType()	1444
29.290.3.2	2MatrixType(int i)	1444
29.290.3.3	3MatrixType(int i, bool dlok)	1444
29.290.3.4	4MatrixType(const MatrixType &mt)	1444
29.290.4	Member Function Documentation	1444
29.290.4.1	1AddEqualEl() const	1444
29.290.4.2	2CannotConvert() const	1444
29.290.4.3	3() const	1445
29.290.4.4	4sBand() const	1445

29.290.4.5	Diagonal() const	1445
29.290.4.6	Symmetric() const	1445
29.290.4.7	KP(const MatrixType &) const	1445
29.290.4.8	MultRHS() const	1445
29.290.4.9	New() const	1445
29.290.4.10	New(int, int, BaseMatrix *) const	1445
29.290.4.11	Types()	1445
29.290.4.12	operator"!() const	1445
29.290.4.13	operator"!=(MatrixType t) const	1445
29.290.4.14	operator&(const MatrixType &mt) const	1446
29.290.4.15	operator*(const MatrixType &) const	1446
29.290.4.16	operator+() const	1446
29.290.4.17	operator+(MatrixType mt) const	1446
29.290.4.18	operator<(MatrixType mt) const	1446
29.290.4.19	operator=(const MatrixType &mt)	1446
29.290.4.20	operator==(MatrixType t) const	1446
29.290.4.21	operator>=(MatrixType mt) const	1446
29.290.4.22	operator" (const MatrixType &mt) const	1446
29.290.4.23	SetDataLossOK()	1446
29.290.4.24	KP(const MatrixType &) const	1447
29.290.4.25	sub() const	1447
29.290.4.26	sub() const	1447
29.290.4.27) const	1447
29.290.4.28	value() const	1447
29.290.5	Friends And Related Function Documentation	1447
29.290.5.1	Compare	1447
29.290.5.2	Rectangular	1447
29.290.6	Member Data Documentation	1447
29.290.6.1	attribute	1447
29.290.6.2	DataLossOK	1447

29.291.0	Go::MatrixXD< T, Dim > Class Template Reference	1448
29.291.1	Detailed Description	1449
29.291.2	Constructor & Destructor Documentation	1449
29.291.2.1	MatrixXD()	1449
29.291.2.2	~MatrixXD()	1449
29.291.3	Member Function Documentation	1449
29.291.3.1	det() const	1449
29.291.3.2	frobeniusNorm() const	1449
29.291.3.3	get()	1450
29.291.3.4	identity()	1450
29.291.3.5	mult(const FromConstIteratorType from, const ToIteratorType to) const	1450
29.291.3.6	operator()(int row, int col)	1450
29.291.3.7	operator()(int row, int col) const	1450
29.291.3.8	operator*(const MatrixXD &other) const	1450
29.291.3.9	operator*(T scalar) const	1450
29.291.3.10	operator*(const VectorType &vec) const	1451
29.291.3.11	operator*=(const MatrixXD &other)	1451
29.291.3.12	operator*=(T scalar)	1451
29.291.3.13	operator+(const MatrixXD &other) const	1451
29.291.3.14	operator+=(const MatrixXD &other)	1451
29.291.3.15	operator-() const	1451
29.291.3.16	setToRotation(T angle, T x, T y, T z)	1451
29.291.3.17	setToRotation(const Vector3D &p, const Vector3D &q)	1452
29.291.3.18	setToRotation(double angle, double x, double y, double z)	1452
29.291.3.19	setToRotation(double angle, double x, double y, double z)	1452
29.291.3.20	setToRotation(const Vector3D &p, const Vector3D &q)	1452
29.291.3.21	submatrix(int r, int c) const	1452
29.291.3.22	trace() const	1452
29.291.3.23	transpose()	1452
29.291.3.24	zero()	1453

29.292.0	Bo::Mesh2D Class Reference	1453
29.292.1	Detailed Description	1454
29.292.2	Constructor & Destructor Documentation	1454
29.292.2.1	Mesh2D()	1454
29.292.2.2	Mesh2D(std::istream &is)	1455
29.292.2.3	Mesh2D(Iterator kx_start, Iterator kx_end, Iterator ky_start, Iterator ky_end)	1455
29.292.2.4	Mesh2D(const Array &xknots, const Array &yknots)	1455
29.292.3	Member Function Documentation	1455
29.292.3.1	begin() const	1455
29.292.3.2	end() const	1455
29.292.3.3	extent(Direction2D d, int ix, int start, int mult) const	1455
29.292.3.4	firstMeshVeclx(Direction2D d) const	1455
29.292.3.5	getKnotIdx(Direction2D d, double &par, double eps) const	1455
29.292.3.6	getKnots(Direction2D d, int ix, bool right=true) const	1455
29.292.3.7	incrementMult(Direction2D d, int ix, int start, int end, int mult)	1455
29.292.3.8	indexMeshBegin() const	1455
29.292.3.9	indexMeshEnd() const	1455
29.292.3.10	insertLine(Direction2D d, double kval, int mult=0)	1455
29.292.3.11	knotIntervalFuzzy(Direction2D d, double &par, double eps) const	1455
29.292.3.12	knotsBegin(Direction2D d) const	1455
29.292.3.13	knotsEnd(Direction2D d) const	1455
29.292.3.14	kval(Direction2D d, int ix) const	1456
29.292.3.15	largestMultInLine(Direction2D d, int ix) const	1456
29.292.3.16	lastMeshVeclx(Direction2D d) const	1456
29.292.3.17	maxParam(Direction2D d) const	1456
29.292.3.18	minMultInLine(Direction2D d, int ix) const	1456
29.292.3.19	minParam(Direction2D d) const	1456
29.292.3.20	mu(Direction2D d, int ix, int start, int end) const	1456
29.292.3.21	numDistinctKnots(Direction2D d) const	1456
29.292.3.22	read(std::istream &is)	1456

29.292.3.23	ReverseParameterDirection(bool dir_is_u)	1456
29.292.3.24	Segments(Direction2D dir, int ix, int threshold=1) const	1456
29.292.3.25	SetMult(Direction2D d, int ix, int start, int end, int mult)	1456
29.292.3.26	SetParameterDomain(double u1, double u2, double v1, double v2)	1457
29.292.3.27	SubMesh(int ix1, int ix2, int iy1, int iy2) const	1457
29.292.3.28	Swap(Mesh2D &rhs)	1457
29.292.3.29	SwapParameterDirection()	1457
29.292.3.30	Write(std::ostream &os) const	1457
29.292.3.31	ZeroSegments(Direction2D dir, int ix) const	1457
29.293	Go::Mesh2DIterator Class Reference	1457
29.293.1	Detailed Description	1457
29.293.2	Constructor & Destructor Documentation	1458
29.293.2.1	Mesh2DIterator()	1458
29.293.2.2	Mesh2DIterator(const Mesh2D &m, int u_ix, int v_ix)	1458
29.293.3	Member Function Documentation	1458
29.293.3.1	operator!=(const Mesh2DIterator &rhs) const	1458
29.293.3.2	operator*() const	1458
29.293.3.3	operator++()	1458
29.293.3.4	operator==(const Mesh2DIterator &rhs) const	1458
29.293.3.5	swap(Mesh2DIterator &rhs)	1458
29.294	NEWMAT::MLE_D_FI Class Reference	1459
29.294.1	Detailed Description	1459
29.294.2	Constructor & Destructor Documentation	1459
29.294.2.1	MLE_D_FI(LL_D_FI &ll, int lim=1000, Real criterion=0.0001)	1459
29.294.3	Member Function Documentation	1460
29.294.3.1	Fit(ColumnVector &Parameters)	1460
29.294.3.2	GetCorrelations(SymmetricMatrix &)	1460
29.294.3.3	GetStandardErrors(ColumnVector &)	1460
29.295	Model_3pe Class Reference	1460
29.295.1	Detailed Description	1461

29.295.2	Constructor & Destructor Documentation	1461
29.295.2.1	Model_3pe(const ColumnVector &X_Values)	1461
29.295.3	Member Function Documentation	1461
29.295.3.1	Derivatives()	1461
29.295.3.2	IsValid()	1461
29.295.3.3	operator()(int)	1461
29.296	Go::ModelQuality Class Reference	1462
29.296.1	Detailed Description	1464
29.296.2	Constructor & Destructor Documentation	1464
29.296.2.1	ModelQuality(double gap, double kink, double approx)	1464
29.296.2.2	ModelQuality(const tpTolerances &toptol, double approx)	1464
29.296.2.3	~ModelQuality()	1464
29.296.3	Member Function Documentation	1464
29.296.3.1	acuteEdgeAngle(std::vector< std::pair< ftEdge *, ftEdge * > > &edge_acute)	1464
29.296.3.2	acuteFaceAngle(std::vector< std::pair< ftSurface *, ftSurface * > > &face_acute)	1465
29.296.3.3	cvC1G1Discontinuity(std::vector< shared_ptr< ParamCurve > > &c1_discont, std::vector< shared_ptr< ParamCurve > > &g1_discont)	1465
29.296.3.4	cvCurvatureRadius(std::vector< std::pair< shared_ptr< PointOnCurve >, double > > &small_curv_rad, std::pair< shared_ptr< PointOnCurve >, double > &minimum_curv_rad)	1465
29.296.3.5	degenerateSfCorners(std::vector< shared_ptr< ftPoint > > °_corners)	1465
29.296.3.6	degenSurfaces(std::vector< shared_ptr< ParamSurface > > °_sfs)	1465
29.296.3.7	edgePosAndTangDiscontinuity(std::vector< std::pair< ftEdge *, ftEdge * > > &pos_disconts, std::vector< std::pair< ftEdge *, ftEdge * > > &tang_disconts)	1465
29.296.3.8	edgeVertexDistance(std::vector< std::pair< ftEdge *, shared_ptr< Vertex > > > &edge_vertices)	1465
29.296.3.9	faceEdgeDistance(std::vector< std::pair< ftSurface *, ftEdge * > > &face_edges)	1466
29.296.3.10	faceNormalConsistency(std::vector< shared_ptr< ftSurface > > &inconsistent_faces)	1466
29.296.3.11	facePositionDiscontinuity(std::vector< std::pair< ftEdge *, ftEdge * > > &pos_disconts)	1466
29.296.3.12	faceTangentDiscontinuity(std::vector< std::pair< ftEdge *, ftEdge * > > &tangent_disconts)	1466

29.296.3.16	faceVertexDistance(std::vector< std::pair< ftSurface *, shared_ptr< Vertex > > > &face_vertices)	1466
29.296.3.17	getResults()	1466
29.296.3.18	identicalOrEmbeddedEdges(std::vector< std::pair< shared_ptr< ftEdge >, shared_ptr< ftEdge > > > &identical_edges, std::vector< std::pair< shared_ptr< ftEdge >, shared_ptr< ftEdge > > > &embedded_edges)	1466
29.296.3.19	identicalOrEmbeddedFaces(std::vector< std::pair< shared_ptr< ftSurface >, shared_ptr< ftSurface > > > &identical_faces, std::vector< std::pair< shared_ptr< ftSurface >, shared_ptr< ftSurface > > > &embedded_faces)	1467
29.296.3.20	identicalVertices(std::vector< std::pair< shared_ptr< Vertex >, shared_ptr< Vertex > > > &identical_vertices)	1467
29.296.3.21	isDistinctKnots(std::vector< shared_ptr< ParamCurve > > &cv_knots, std::vector< shared_ptr< ParamSurface > > &sf_knots, double tol=1.0e-8)	1467
29.296.3.22	loopIntersection(std::vector< std::pair< shared_ptr< PointOnEdge >, shared_ptr< PointOnEdge > > > &loop_intersection)	1467
29.296.3.23	loopOrientationConsistency(std::vector< shared_ptr< Loop > > &inconsistent_loops)	1467
29.296.3.24	loopSelfIntersection(std::vector< std::pair< shared_ptr< PointOnEdge >, shared_ptr< PointOnEdge > > > &loop_self_intersection)	1467
29.296.3.25	miniEdges(std::vector< shared_ptr< ftEdge > > &mini_edges)	1467
29.296.3.26	miniSurfaces(std::vector< shared_ptr< ftSurface > > &mini_surfaces)	1468
29.296.3.27	narrowRegion(std::vector< std::pair< shared_ptr< PointOnEdge >, shared_ptr< PointOnEdge > > > &narrow_regions)	1468
29.296.3.28	setCurvatureRadius(double radius)	1468
29.296.3.29	setMiniElementSize(double element_size)	1468
29.296.3.30	setC1Discontinuity(std::vector< shared_ptr< ftSurface > > &discont_sfs)	1468
29.296.3.31	setCurvatureRadius(std::vector< std::pair< shared_ptr< ftPoint >, double > > &small_curv_rad, std::pair< shared_ptr< ftPoint >, double > &minimum_curv_rad)	1468
29.296.3.32	setG1Discontinuity(std::vector< shared_ptr< ftSurface > > &discont_sfs)	1468
29.296.3.33	sliverSurfaces(std::vector< shared_ptr< ParamSurface > > &sliver_sfs, double thickness, double factor=2.0)	1469
29.296.3.34	singularCurveTangent(std::vector< shared_ptr< PointOnCurve > > &singular_points, std::vector< std::pair< shared_ptr< PointOnCurve >, shared_ptr< PointOnCurve > > > &singular_curves)	1469
29.296.3.35	singularSurfaceNormal(std::vector< shared_ptr< ftPoint > > &singular_points, std::vector< shared_ptr< ftCurve > > &singular_curves)	1469
29.296.4	Member Data Documentation	1469

29.296.4.1approx_	1469
29.296.4.2curvature_radius_	1469
29.296.4.3results_	1469
29.296.4.4small_size_	1469
29.296.4.5optol_	1469
29.297.0Go::ModelRepair Class Reference	1470
29.297.1Detailed Description	1471
29.297.2Constructor & Destructor Documentation	1471
29.297.2.1ModelRepair()	1471
29.297.2.2ModelRepair(shared_ptr< QualityResults > results)	1471
29.297.3Member Function Documentation	1471
29.297.3.1consistentFaceNormal()=0	1471
29.297.3.2identicalAndEmbeddedFaces()=0	1471
29.297.3.3identicalVertices()=0	1471
29.297.3.4mendEdgeDistance()=0	1471
29.297.3.5mendGaps()=0	1471
29.297.3.6optimizeVertexPosition()=0	1472
29.297.4Member Data Documentation	1472
29.297.4.1results_	1472
29.298.0MultiDijkstra Class Reference	1472
29.298.1Detailed Description	1473
29.298.2Constructor & Destructor Documentation	1473
29.298.2.1MultiDijkstra()	1473
29.298.2.2~MultiDijkstra()	1473
29.298.3Member Function Documentation	1473
29.298.3.1closestNeighbour(int node_idx)	1473
29.298.3.2getDistance(int node_idx)	1473
29.298.3.3getDistance(int node_idx, int neighbour)	1474
29.298.3.4getLabel(int node_idx)	1474
29.298.3.5initialize()	1474

29.298.3.6insertInCandidates(int node_idx)	1474
29.298.3.7isFinished(int node_idx)	1474
29.298.3.8run()	1474
29.298.3.9run(double radius)	1474
29.298.3.10run(int target)	1474
29.298.3.11setFinished(int node_idx)	1474
29.298.3.12setGraph(GraphType *tri)	1474
29.298.3.13setSource(int node_idx, int node_label, double dist=0)	1474
29.298.4Member Data Documentation	1475
29.298.4.1back_trace_	1475
29.298.4.2distances_	1475
29.298.4.3flags_	1475
29.298.4.4graph_	1475
29.298.4.5label_	1475
29.298.4.6large_distance_	1475
29.298.4.7queue_	1475
29.299NEWMAT::MultipliedMatrix Class Reference	1476
29.299.1Detailed Description	1477
29.299.2Constructor & Destructor Documentation	1477
29.299.2.1MultipliedMatrix(const BaseMatrix *bm1x, const BaseMatrix *bm2x)	1477
29.299.2.2~MultipliedMatrix()	1477
29.299.3Member Function Documentation	1477
29.299.3.1BandWidth() const	1477
29.299.3.2Evaluate(MatrixType mt=MatrixTypeUnSp)	1478
29.299.3.3search(const BaseMatrix *) const	1478
29.299.4Friends And Related Function Documentation	1478
29.299.4.1BaseMatrix	1478
29.299.4.2GeneralMatrix	1478
29.299.4.3GenericMatrix	1478
29.299.5Member Data Documentation	1478

29.299.5.1"@5	1478
29.299.5.2"@7	1478
29.299.5.3m1	1478
29.299.5.4m2	1478
29.299.5.5gm1	1479
29.299.5.6gm2	1479
29.300NEWMAT::MultiRadixCounter Class Reference	1479
29.300.1Detailed Description	1479
29.300.2Constructor & Destructor Documentation	1479
29.300.2.1MultiRadixCounter(int nx, const SimpleIntArray &rx, SimpleIntArray &vx)	1479
29.300.3Member Function Documentation	1479
29.300.3.1Counter() const	1479
29.300.3.2Finish() const	1480
29.300.3.3operator++()	1480
29.300.3.4Reverse() const	1480
29.300.3.5Swap() const	1480
29.301NEWMAT::NegatedMatrix Class Reference	1480
29.301.1Detailed Description	1482
29.301.2Constructor & Destructor Documentation	1482
29.301.2.1NegatedMatrix(const BaseMatrix *bmX)	1482
29.301.2.2~NegatedMatrix()	1482
29.301.3Member Function Documentation	1482
29.301.3.1BandWidth() const	1482
29.301.3.2Evaluate(MatrixType mt=MatrixTypeUnSp)	1482
29.301.3.3search(const BaseMatrix *) const	1482
29.301.4Friends And Related Function Documentation	1482
29.301.4.1BaseMatrix	1482
29.301.5Member Data Documentation	1483
29.301.5.1"@11	1483
29.301.5.2bm	1483

29.301.5.3gm	1483
29.302.1NegShiftedMatrix Class Reference	1483
29.302.1.1Detailed Description	1484
29.302.2Constructor & Destructor Documentation	1485
29.302.2.1NegShiftedMatrix(Real fx, const BaseMatrix *bmx)	1485
29.302.2.2~NegShiftedMatrix()	1485
29.302.3Member Function Documentation	1485
29.302.3.1Evaluate(MatrixType mt=MatrixTypeUnSp)	1485
29.302.4Friends And Related Function Documentation	1485
29.302.4.1BaseMatrix	1485
29.302.4.2GeneralMatrix	1485
29.302.4.3GenericMatrix	1485
29.302.4.4operator-	1485
29.303.1Node Class Reference	1486
29.303.1.1Detailed Description	1487
29.303.2Constructor & Destructor Documentation	1487
29.303.2.1Node()	1487
29.303.2.2Node(double x, double y, double z=0.0)	1487
29.303.2.3~Node()	1487
29.303.3Member Function Documentation	1488
29.303.3.1getFlag() const	1488
29.303.3.2d() const	1488
29.303.3.3nit(double x, double y, double z)	1488
29.303.3.4setFlag(bool flag)	1488
29.303.3.5x() const	1488
29.303.3.6y() const	1488
29.303.3.7z() const	1488
29.304.1Triang::Node Class Reference	1489
29.304.1.1Detailed Description	1489
29.304.2Constructor & Destructor Documentation	1489

29.304.2.1	Node()	1489
29.304.2.2	Node(double x, double y, double z=0.0)	1489
29.304.2.3	Node(Go::PointIter iter, double x, double y, double z=0)	1489
29.304.2.4	~Node()	1489
29.304.3	Member Function Documentation	1490
29.304.3.1	init(double x, double y, double z)	1490
29.304.3.2	pointIter() const	1490
29.304.3.3	x() const	1490
29.304.3.4	y() const	1490
29.304.3.5	z() const	1490
29.305	Go::NonEvaluableIntersectionCurve Class Reference	1490
29.305.1	Detailed Description	1491
29.305.2	Constructor & Destructor Documentation	1491
29.305.2.1	~NonEvaluableIntersectionCurve()	1491
29.305.3	Member Function Documentation	1491
29.305.3.1	evaluateAt(double pval, Point &pos, Point &tan)	1491
29.305.3.2	getCurve() const	1492
29.305.3.3	getParamCurve(int obj_nmb) const	1492
29.305.3.4	getParamSpan(double &start, double &end) const	1492
29.305.3.5	isDegenerated() const	1493
29.305.3.6	isCurve() const	1493
29.305.3.7	refine(const double &pos_tol, const double &angle_tol)	1493
29.305.4	Friends And Related Function Documentation	1494
29.305.4.1	constructIntersectionCurve	1494
29.306	NEWMAT::NonLinearLeastSquares Class Reference	1494
29.306.1	Detailed Description	1495
29.306.2	Constructor & Destructor Documentation	1495
29.306.2.1	NonLinearLeastSquares(R1_Col_I_D &pred, int lim=1000, Real crit=0.0001)	1495
29.306.3	Member Function Documentation	1495
29.306.3.1	Fit(const ColumnVector &, ColumnVector &)	1495

29.306.3.2	GetCorrelations(SymmetricMatrix &)	1495
29.306.3.3	GetHatDiagonal(DiagonalMatrix &) const	1495
29.306.3.4	GetResiduals(ColumnVector &Z) const	1495
29.306.3.5	GetStandardErrors(ColumnVector &)	1495
29.306.3.6	ResidualVariance() const	1496
29.307	Go::NoopTesselator Class Reference	1496
29.307.1	Detailed Description	1497
29.307.2	Constructor & Destructor Documentation	1497
29.307.2.1	~NoopTesselator()	1497
29.307.3	Member Function Documentation	1497
29.307.3.1	tesselate()	1497
29.308	NEWMAT::NotDefinedException Class Reference	1497
29.308.1	Detailed Description	1498
29.308.2	Constructor & Destructor Documentation	1498
29.308.2.1	NotDefinedException(const char *op, const char *matrix)	1498
29.308.3	Member Data Documentation	1498
29.308.3.1	Select	1498
29.309	NEWMAT::NotSquareException Class Reference	1499
29.309.1	Detailed Description	1500
29.309.2	Constructor & Destructor Documentation	1500
29.309.2.1	NotSquareException(const GeneralMatrix &A)	1500
29.309.3	Member Data Documentation	1500
29.309.3.1	Select	1500
29.310	NEWMAT::NPDException Class Reference	1500
29.310.1	Detailed Description	1501
29.310.2	Constructor & Destructor Documentation	1501
29.310.2.1	NPDException(const GeneralMatrix &)	1501
29.310.3	Member Data Documentation	1501
29.310.3.1	Select	1501
29.311	NEWMAT::nricMatrix Class Reference	1502

29.311.1	Detailed Description	1503
29.311.2	Constructor & Destructor Documentation	1504
29.311.2.1	nrncMatrix()	1504
29.311.2.2	nrncMatrix(int m, int n)	1504
29.311.2.3	nrncMatrix(const BaseMatrix &bm)	1504
29.311.2.4	nrncMatrix(const nrncMatrix &gm)	1504
29.311.2.5	~nrncMatrix()	1504
29.311.3	Member Function Documentation	1504
29.311.3.1	CleanUp()	1504
29.311.3.2	nrnc() const	1504
29.311.3.3	operator<<(const BaseMatrix &X)	1504
29.311.3.4	operator=(const BaseMatrix &bm)	1504
29.311.3.5	operator=(Real f)	1505
29.311.3.6	operator=(const nrncMatrix &m)	1505
29.311.3.7	ReSize(int m, int n)	1505
29.311.3.8	ReSize(const GeneralMatrix &A)	1505
29.312	Go::ObjectHeader Class Reference	1505
29.312.1	Detailed Description	1506
29.312.2	Constructor & Destructor Documentation	1506
29.312.2.1	ObjectHeader()	1506
29.312.2.2	ObjectHeader(ClassType t, int major, int minor)	1506
29.312.2.3	ObjectHeader(ClassType t, int major, int minor, const std::vector< int > &auxdata)	1507
29.312.2.4	~ObjectHeader()	1507
29.312.3	Member Function Documentation	1507
29.312.3.1	auxdata(int i) const	1507
29.312.3.2	auxdataSize() const	1507
29.312.3.3	classType() const	1508
29.312.3.4	majorVersion() const	1508
29.312.3.5	minorVersion() const	1508
29.312.3.6	read(std::istream &is)	1508

29.312.3.7	<code>write(std::ostream &os) const</code>	1508
29.313	<code>RBD_COMMON::OneDimSolve</code> Class Reference	1508
29.313.1	Detailed Description	1509
29.313.2	Constructor & Destructor Documentation	1509
29.313.2.1	<code>OneDimSolve(R1_R1 &f, Real AccY=0.0001, Real AccX=0.0)</code>	1509
29.313.3	Member Function Documentation	1509
29.313.3.1	<code>Solve(Real Y, Real X, Real Dev, int Lim=100)</code>	1509
29.314	<code>RBD_COMMON::Out_of_range</code> Class Reference	1509
29.314.1	Detailed Description	1510
29.314.2	Constructor & Destructor Documentation	1510
29.314.2.1	<code>Out_of_range(const char *a_what=0)</code>	1510
29.314.3	Member Data Documentation	1510
29.314.3.1	Select	1510
29.315	<code>RBD_COMMON::Overflow_error</code> Class Reference	1511
29.315.1	Detailed Description	1512
29.315.2	Constructor & Destructor Documentation	1512
29.315.2.1	<code>Overflow_error(const char *a_what=0)</code>	1512
29.315.3	Member Data Documentation	1512
29.315.3.1	Select	1512
29.316	<code>NEWMAT::OverflowException</code> Class Reference	1512
29.316.1	Detailed Description	1513
29.316.2	Constructor & Destructor Documentation	1513
29.316.2.1	<code>OverflowException(const char *c)</code>	1513
29.316.3	Member Data Documentation	1513
29.316.3.1	Select	1513
29.317	<code>Go::Par0FuncIntersector</code> Class Reference	1514
29.317.1	Detailed Description	1515
29.317.2	Constructor & Destructor Documentation	1515
29.317.2.1	<code>Par0FuncIntersector(shared_ptr< ParamFunctionInt > func, shared_ptr< ParamFunctionInt > C, shared_ptr< GeoTol > epsge, Intersector *prev=0, int eliminated_parameter=-1, double eliminated_value=0)</code>	1515

29.317.2.2~Par0FuncIntersector()	1515
29.317.3Member Function Documentation	1515
29.317.3.1checkCoincidence()	1515
29.317.3.2doSubdivide()	1515
29.317.3.3lowerOrderIntersector(shared_ptr< ParamFunctionInt > obj1, shared_ptr< ParamFunctionInt > obj2, Intersector *prev=0, int eliminated_parameter=-1, double eliminated_value=0)	1515
29.317.3.4microCase()	1516
29.317.3.5numParams() const	1516
29.317.3.6repairIntersections()	1516
29.317.3.7updateIntersections()	1516
29.318Go::Par1FuncIntersector Class Reference	1516
29.318.1Detailed Description	1517
29.318.2Constructor & Destructor Documentation	1517
29.318.2.1Par1FuncIntersector(shared_ptr< ParamFunctionInt > func, shared_ptr< ParamFunctionInt > C, shared_ptr< GeoTol > epsge, Intersector *prev=0, int eliminated_parameter=-1, double eliminated_value=0)	1517
29.318.2.2~Par1FuncIntersector()	1518
29.318.3Member Function Documentation	1518
29.318.3.1checkCoincidence()	1518
29.318.3.2doSubdivide()	1518
29.318.3.3lowerOrderIntersector(shared_ptr< ParamFunctionInt > obj1, shared_ptr< ParamFunctionInt > obj2, Intersector *prev=0, int eliminated_parameter=-1, double eliminated_value=0)	1518
29.318.3.4microCase()	1518
29.318.3.5numParams() const	1518
29.318.3.6repairIntersections()	1519
29.318.3.7updateIntersections()	1519
29.319Go::Par2FuncIntersector Class Reference	1519
29.319.1Detailed Description	1520
29.319.2Constructor & Destructor Documentation	1520
29.319.2.1Par2FuncIntersector(shared_ptr< ParamFunctionInt > func, shared_ptr< ParamFunctionInt > C, shared_ptr< GeoTol > epsge, Intersector *prev=0, int eliminated_parameter=-1, double eliminated_value=0)	1520

29.319.2.2~Par2FuncIntersector()	1521
29.319.3 Member Function Documentation	1521
29.319.3.1canConnect(shared_ptr< IntersectionPoint > pt1, shared_ptr< IntersectionPoint > pt2)	1521
29.319.3.2checkCoincidence()	1521
29.319.3.3connectDirected(std::vector< std::pair< shared_ptr< IntersectionPoint >, IntPtClassification > > &bd_ints, int nmbbd)	1521
29.319.3.4doSubdivide()	1521
29.319.3.5sConnected(std::vector< shared_ptr< IntersectionPoint > > bd_ints, int nmbbd)	1521
29.319.3.6sConnected(std::vector< std::pair< shared_ptr< IntersectionPoint >, IntPtClassification > > &bd_ints, int nmb_nottouch)	1521
29.319.3.7lowerOrderIntersector(shared_ptr< ParamFunctionInt > obj1, shared_ptr< ParamFunctionInt > obj2, Intersector *prev=0, int eliminated_parameter=-1, double eliminated_value=0)	1521
29.319.3.8microCase()	1521
29.319.3.9numParams() const	1521
29.319.3.10pairIntersections()	1522
29.319.3.11updateIntersections()	1522
29.320 Go::Parabola Class Reference	1522
29.320.1 Detailed Description	1524
29.320.2 Constructor & Destructor Documentation	1525
29.320.2.1Parabola()	1525
29.320.2.2Parabola(Point location, Point direction, Point normal, double focal_dist, bool isReversed=false)	1525
29.320.2.3~Parabola()	1525
29.320.3 Member Function Documentation	1525
29.320.3.1appendCurve(ParamCurve *cv, bool reparam=true)	1525
29.320.3.2appendCurve(ParamCurve *cv, int continuity, double &dist, bool reparam=true)	1525
29.320.3.3boundingBox() const	1526
29.320.3.4classType()	1526
29.320.3.5clone() const	1526
29.320.3.6closestPoint(const Point &pt, double tmin, double tmax, double &clo_t, Point &clo_pt, double &clo_dist, double const *seed=0) const	1526

29.320.3.7	createSplineCurve() const	1526
29.320.3.8	dimension() const	1526
29.320.3.9	directionCone() const	1527
29.320.3.10	endparam() const	1527
29.320.3.11	geometryCurve()	1527
29.320.3.12	istanceType() const	1527
29.320.3.13	Bounded() const	1527
29.320.3.14	Degenerate(double degenerate_epsilon)	1527
29.320.3.15	InPlane(const Point &norm, double eps, Point &pos) const	1528
29.320.3.16	length(double tol)	1528
29.320.3.17	point(Point &pt, double tpar) const	1528
29.320.3.18	point(std::vector< Point > &pts, double tpar, int derivs, bool from_right=true) const	1529
29.320.3.19	read(std::istream &is)	1529
29.320.3.20	setParamBounds(double startpar, double endpar)	1529
29.320.3.21	setParameterInterval(double t1, double t2)	1529
29.320.3.22	setSpanningVectors()	1530
29.320.3.23	startparam() const	1530
29.320.3.24	subCurve(double from_par, double to_par, double fuzzy=DEFAULT_PARAMETER_EPSILON) const	1530
29.320.3.25	swapParameters2D()	1530
29.320.3.26	translateCurve(const Point &dir)	1530
29.320.3.27	write(std::ostream &os) const	1530
29.320.4	Member Data Documentation	1531
29.320.4.1	endparam_	1531
29.320.4.2	ε_	1531
29.320.4.3	location_	1531
29.320.4.4	normal_	1531
29.320.4.5	startparam_	1531
29.320.4.6	vec1_	1531
29.320.4.7	vec2_	1531

29.320	Go::Parallelepiped Class Reference	1532
29.321	1. Detailed Description	1534
29.321	2. Constructor & Destructor Documentation	1534
29.321.2	1. Parallelepiped()	1534
29.321.2	2. Parallelepiped(Point corner, Point dir_u, Point dir_v, Point dir_w, double len_u, double len_v, double len_w)	1534
29.321.2	3. ~Parallelepiped()	1534
29.321	3. Member Function Documentation	1534
29.321.3	1. boundingBox() const	1534
29.321.3	2. classType()	1535
29.321.3	3. clone() const	1535
29.321.3	4. closestPoint(const Point &pt, double &clo_u, double &clo_v, double &clo_w, Point &clo_pt, double &clo_dist, double epsilon, double *seed=0) const	1535
29.321.3	5. dimension() const	1535
29.321.3	6. geometryVolume() const	1535
29.321.3	7. getAllBoundarySurfaces() const	1536
29.321.3	8. instanceType() const	1536
29.321.3	9. nextSegmentVal(int dir, double par, bool forward, double tol) const	1536
29.321.3	10. parameterSpan() const	1536
29.321.3	11. point(Point &pt, double upar, double vpar, double wpar) const	1536
29.321.3	12. point(std::vector< Point > &pts, double upar, double vpar, double wpar, int derivs, bool u_from_right=true, bool v_from_right=true, bool w_from_right=true, double resolution=1.0e-12) const	1537
29.321.3	13. read(std::istream &is)	1537
29.321.3	14. reverseParameterDirection(int pardir)	1538
29.321.3	15. swapParameterDirection(int pardir1, int pardir2)	1538
29.321.3	16. tangentCone(int pardir) const	1538
29.321.3	17. translate(const Point &vec)	1538
29.321.3	18. write(std::ostream &os) const	1538
29.321	Go::Param0FunctionInt Class Reference	1539
29.322	1. Detailed Description	1541
29.322	2. Constructor & Destructor Documentation	1541

29.322.2.1Param0FunctionInt(double C)	1541
29.322.2.2Param0FunctionInt(double C, ParamFunctionInt *parent)	1541
29.322.2.3~Param0FunctionInt()	1542
29.322.3Member Function Documentation	1542
29.322.3.1boundaryPoint(const double *par, double eps) const	1542
29.322.3.2canDivide(int paddir)	1542
29.322.3.3compositeBox() const	1542
29.322.3.4dimension()	1542
29.322.3.5endParam(int paddir) const	1542
29.322.3.6getBoundaryObjects(std::vector< shared_ptr< BoundaryFunctionInt > > &bd← _objs)	1543
29.322.3.7getCriticalVals(int paddir) const	1543
29.322.3.8getCriticalValsAndKnots(int paddir) const	1543
29.322.3.9getInnerKnotVals(int paddir, bool sort=false) const	1543
29.322.3.10getLengthAndWiggle(double *length, double *wiggle)	1544
29.322.3.11getMesh()	1544
29.322.3.12getMeshSize(int dir)	1544
29.322.3.13getParam0FunctionInt()	1544
29.322.3.14getValue() const	1544
29.322.3.15hasCriticalVals(int paddir) const	1544
29.322.3.16hasCriticalValsOrKnots(int paddir) const	1545
29.322.3.17hasInnerKnots(int paddir) const	1545
29.322.3.18notIntervalFuzzy(double &t, double tol) const	1545
29.322.3.19monotone(Point &dir, double tol=1.0e-15) const	1545
29.322.3.20nextSegmentVal(double par, bool forward) const	1546
29.322.3.21numParams() const	1546
29.322.3.22paramFromMesh(int dir, int idx)	1546
29.322.3.23point(Point &res, const double *par) const	1546
29.322.3.24point(std::vector< Point > &pt, const double *tpar, int derivs, const bool *from← _right=0, double resolution=1.0e-12) const	1546
29.322.3.25startParam(int paddir) const	1547

29.322.3.2	subdivide(int paddir, double par, std::vector< shared_ptr< ParamFunctionInt > > &subdiv_objs, std::vector< shared_ptr< ParamFunctionInt > > &bd_objs)	1547
29.322.4	Member Data Documentation	1547
29.322.4.1	C_	1547
29.322.4.2	dim_	1547
29.322.4.3	parentcurve_	1548
29.323	Go::Param1FunctionInt Class Reference	1548
29.323.1	Detailed Description	1550
29.323.2	Constructor & Destructor Documentation	1550
29.323.2.1	Param1FunctionInt(shared_ptr< ParamCurve > curve)	1550
29.323.2.2	Param1FunctionInt(shared_ptr< ParamCurve > curve, ParamFunctionInt *parent)	1551
29.323.2.3	~Param1FunctionInt()	1551
29.323.3	Member Function Documentation	1551
29.323.3.1	assureInRange(double &t)	1551
29.323.3.2	boundaryPoint(const double *par, double eps) const	1551
29.323.3.3	canDivide(int paddir)	1551
29.323.3.4	compositeBox() const	1552
29.323.3.5	dimension()	1552
29.323.3.6	endParam(int paddir) const	1552
29.323.3.7	endparam() const	1552
29.323.3.8	getBoundaryObjects(std::vector< shared_ptr< BoundaryFunctionInt > > &bd_← _objs)	1552
29.323.3.9	getCriticalVals(int paddir) const	1552
29.323.3.10	getCriticalValsAndKnots(int paddir) const	1553
29.323.3.11	getInnerKnotVals(int paddir, bool sort=false) const	1553
29.323.3.12	getLengthAndWiggle(double *length, double *wiggle)	1553
29.323.3.13	getMesh()	1554
29.323.3.14	getMeshSize(int dir)	1554
29.323.3.15	getParam1FunctionInt()	1554
29.323.3.16	getParamCurve()	1554
29.323.3.17	getParamCurve() const	1554

29.323.3.19	getParentParamCurve(double &start, double &end)	1554
29.323.3.19	getParentParamCurve()	1554
29.323.3.20	hasCriticalVals(int paddir) const	1554
29.323.3.21	hasCriticalValsOrKnots(int paddir) const	1555
29.323.3.22	hasInnerKnots(int paddir) const	1555
29.323.3.23	Degenerate(double epsge, int dir, double *par)	1555
29.323.3.24	isNotIntervalFuzzy(double &t, double tol) const	1555
29.323.3.25	makeIntFunction(shared_ptr< ParamCurve > curve)	1556
29.323.3.26	isMonotone(Point &dir, double tol=1.0e-15) const	1556
29.323.3.27	nextSegmentVal(double par, bool forward) const	1556
29.323.3.28	numParams() const	1557
29.323.3.29	paramFromMesh(int dir, int idx)	1557
29.323.3.30	point(Point &res, const double *par) const	1557
29.323.3.31	point(std::vector< Point > &res, double par, int der)	1557
29.323.3.32	point(std::vector< Point > &pt, const double *tpar, int derivs, const bool *from← _right=0, double resolution=1.0e-12) const	1558
29.323.3.33	startParam(int paddir) const	1558
29.323.3.34	startparam() const	1558
29.323.3.35	subdivide(int paddir, double par, std::vector< shared_ptr< ParamFunctionInt > > &subdiv_objs, std::vector< shared_ptr< ParamFunctionInt > > &bd_objs)	1558
29.323.4	Member Data Documentation	1559
29.323.4.1	curve_	1559
29.323.4.2	deg_tol_	1559
29.323.4.3	dim_	1559
29.323.4.4	mesh_	1559
29.323.4.5	parentcurve_	1559
29.323.4.6	segment_	1559
29.324	Go::Param2FunctionInt Class Reference	1560
29.324.1	Detailed Description	1562
29.324.2	Constructor & Destructor Documentation	1562
29.324.2.1	Param2FunctionInt(shared_ptr< ParamSurface > surf)	1562

29.324.2.2	Param2FunctionInt(shared_ptr< ParamSurface > surf, Param2FunctionInt *parent)	1562
29.324.2.3	~Param2FunctionInt()	1562
29.324.3	Member Function Documentation	1563
29.324.3.1	assureInRange(int pdir, double &t)	1563
29.324.3.2	boundaryPoint(const double *par, double eps) const	1563
29.324.3.3	canDivide(int pdir)	1563
29.324.3.4	compositeBox() const	1563
29.324.3.5	derivs(double u, double v, Point &deriv_u, Point &deriv_v) const	1564
29.324.3.6	dimension()	1564
29.324.3.7	endParam(int pdir) const	1564
29.324.3.8	extractBdCurve(const Go::SplineSurface &, int bidx, int &pdir, double &tpar) const	1564
29.324.3.9	getBoundaryObjects(std::vector< shared_ptr< BoundaryFunctionInt > > &bd← _objs)	1564
29.324.3.10	getConstantParameterCurve(int dir, double par)	1565
29.324.3.11	getCriticalVals(int pdir) const	1565
29.324.3.12	getCriticalValsAndKnots(int pdir) const	1565
29.324.3.13	getInnerKnotVals(int pdir, bool sort=false) const	1565
29.324.3.14	getIsoCurve(double param_start, double param_end, double isoval, bool pdir← _is_u) const	1565
29.324.3.15	getLengthAndWiggle(double *length, double *wiggle)	1566
29.324.3.16	getMesh()	1566
29.324.3.17	getMeshSize(int dir)	1566
29.324.3.18	getMima() const	1566
29.324.3.19	getParam2FunctionInt()	1567
29.324.3.20	getParamSurface()	1567
29.324.3.21	getParamSurface() const	1567
29.324.3.22	getParentParamSurface(RectDomain &domain)	1567
29.324.3.23	getParentParamSurface()	1567
29.324.3.24	getSurface()	1567
29.324.3.25	getSurface() const	1567
29.324.3.26	hasCriticalVals(int pdir) const	1567

29.324.3.27	IsCriticalValsOrKnots(int paddir) const	1568
29.324.3.28	IsInnerKnots(int paddir) const	1568
29.324.3.29	Degenerate(double epsge, int paddir)	1568
29.324.3.30	Degenerate(double epsge, int paddir, double *par)	1568
29.324.3.31	IsolateDegPar(int dir, int deg_edge, double threshold)	1569
29.324.3.32	IsNotIntervalFuzzy(int paddir, double &t, double tol) const	1569
29.324.3.33	MakeIntFunction(shared_ptr< ParamSurface > surf)	1569
29.324.3.34	IsMonotone(Point &dir, double tol=1.0e-15) const	1570
29.324.3.35	NextSegmentVal(int paddir, double par, bool forward) const	1570
29.324.3.36	NumParams() const	1570
29.324.3.37	ParamFromMesh(int dir, int idx)	1570
29.324.3.38	Point(Point &res, const double *par) const	1571
29.324.3.39	Point(std::vector< Point > &pt, const double *tpar, int der, const bool *from_← right=0, double resolution=1.0e-12) const	1571
29.324.3.40	StartParam(int paddir) const	1571
29.324.3.41	Subdivide(int paddir, double par, std::vector< shared_ptr< ParamFunctionInt > > &subdiv_objs, std::vector< shared_ptr< ParamFunctionInt > > &bd_objs)	1572
29.324.4	Member Data Documentation	1572
29.324.4.1	bd_deg_	1572
29.324.4.2	deg_tol_	1572
29.324.4.3	dim_	1572
29.324.4.4	domain_	1572
29.324.4.5	mesh_	1572
29.324.4.6	nmesh_	1572
29.324.4.7	parentfunction_	1572
29.324.4.8	segment_	1573
29.324.4.9	surf_	1573
29.324.4.10	temp_point_array_	1573
29.325	Go::ParamCurve Class Reference	1573
29.325.1	Detailed Description	1575
29.325.2	Constructor & Destructor Documentation	1575

29.325.2.1	~ParamCurve()	1575
29.325.3	Member Function Documentation	1575
29.325.3.1	appendCurve(ParamCurve *cv, bool reparam=true)=0	1575
29.325.3.2	appendCurve(ParamCurve *cv, int continuity, double &dist, bool reparam=true)=0	1575
29.325.3.3	clone() const =0	1576
29.325.3.4	closestPoint(const Point &pt, double tmin, double tmax, double &clo_t, Point &clo_pt, double &clo_dist, double const *seed=0) const =0	1576
29.325.3.5	closestPoint(const Point &pt, double &clo_t, Point &clo_pt, double &clo_dist) const	1576
29.325.3.6	closestPointGeneric(const Point &pt, double tmin, double tmax, double guess_↔ param, double &clo_t, Point &clo_pt, double &clo_dist) const	1576
29.325.3.7	compositeBox() const	1576
29.325.3.8	directionCone() const =0	1577
29.325.3.9	endparam() const =0	1577
29.325.3.10	estimatedCurveLength(int numpts=4) const	1577
29.325.3.11	estimatedCurveLength(double tmin, double tmax, int numpts=4) const	1577
29.325.3.12	geometryCurve()=0	1578
29.325.3.13	AxisRotational(Point ¢re, Point &axis, Point &vec, double &angle)	1578
29.325.3.14	Closed()	1578
29.325.3.15	Degenerate(double degenerate_epsilon)=0	1578
29.325.3.16	InPlane(const Point &loc, const Point &axis, double eps, Point &normal) const	1579
29.325.3.17	InPlane(const Point &norm, double eps, Point &pos) const	1579
29.325.3.18	Linear(Point &dir, double tol)	1579
29.325.3.19	Length(double tol)=0	1579
29.325.3.20	Length(double tol, double tstart, double tend)	1580
29.325.3.21	nextSegmentVal(double par, bool forward, double tol) const	1580
29.325.3.22	point(Point &pt, double tpar) const =0	1580
29.325.3.23	point(std::vector< Point > &pts, double tpar, int derivs, bool from_right=true) const =0	1581
29.325.3.24	point(double tpar) const	1581
29.325.3.25	point(double tpar, int derivs, bool from_right=true) const	1581
29.325.3.26	reverseParameterDirection(bool switchparam=false)=0	1582

29.325.3.27	771(Point pt, double aepsge, double astart, double aend, double anext, double &cpos, int *jstat) const	1582
29.325.3.28	771_s9del(double *eco, double *eco1, double *eco2, int idim) const	1582
29.325.3.29	771_s9point(Point pt, std::vector< Point > val, Point diff, double astart, double aend, int max_it, double *cnext, double *ad, double adel, double *cdist, double aprev, int *jstat) const	1582
29.325.3.30	GetParameterInterval(double t1, double t2)=0	1582
29.325.3.31	Split(double param, double fuzzy=DEFAULT_PARAMETER_EPSILON) const	1582
29.325.3.32	Startparam() const =0	1582
29.325.3.33	SubCurve(double from_par, double to_par, double fuzzy=DEFAULT_PARAMETER_EPSILON) const =0	1583
29.325.3.34	UniformEvaluator(int num, std::vector< Point > &points, std::vector< double > ¶m) const	1583
29.326	Go::ParamCurveInt Class Reference	1583
29.326.1	Detailed Description	1586
29.326.2	Constructor & Destructor Documentation	1586
29.326.2.1	ParamCurveInt(shared_ptr< ParamCurve > curve, ParamGeomInt *parent=0)	1586
29.326.2.2	~ParamCurveInt()	1586
29.326.3	Member Function Documentation	1586
29.326.3.1	assureInRange(double &t)	1586
29.326.3.2	axisFromEndpts(Point &axis) const	1587
29.326.3.3	boundaryPoint(const double *par, double eps) const	1587
29.326.3.4	canDivide(int padir)	1587
29.326.3.5	checkPeriodicity(int padir=0) const	1587
29.326.3.6	compositeBox() const	1588
29.326.3.7	dimension() const	1588
29.326.3.8	directionCone() const	1588
29.326.3.9	endParam(int padir) const	1588
29.326.3.10	endparam() const	1588
29.326.3.11	getBoundaryObjects(std::vector< shared_ptr< BoundaryGeomInt > > &bd_objs)	1589
29.326.3.12	getCriticalVals(int padir) const	1589
29.326.3.13	getCriticalValsAndKnots(int padir) const	1589
29.326.3.14	getInnerKnotVals(int padir, bool sort=false) const	1589

29.326.3.15	<code>GetLengthAndWiggle(double *length, double *wiggle)</code>	1590
29.326.3.16	<code>GetMesh()</code>	1590
29.326.3.17	<code>GetMeshSize(int dir)</code>	1590
29.326.3.18	<code>GetOptimizedConeAngle(Point &axis1, Point &axis2)</code>	1590
29.326.3.19	<code>GetParamCurve()</code>	1591
29.326.3.20	<code>GetParamCurve() const</code>	1591
29.326.3.21	<code>GetParamCurveInt()</code>	1591
29.326.3.22	<code>GetParamSurfaceInt()</code>	1591
29.326.3.23	<code>GetParentParamCurve(double &start, double &end)</code>	1591
29.326.3.24	<code>GetParentParamCurve()</code>	1592
29.326.3.25	<code>GetRotatedBox(std::vector< Point > &axis) const</code>	1592
29.326.3.26	<code>GetSpline()</code>	1592
29.326.3.27	<code>HasCriticalVals(int pdir) const</code>	1592
29.326.3.28	<code>HasCriticalValsOrKnots(int pdir) const</code>	1593
29.326.3.29	<code>HasInnerKnots(int pdir) const</code>	1593
29.326.3.30	<code>Degenerate(double epsge, int dir, double *par)</code>	1593
29.326.3.31	<code>IsSpline()</code>	1593
29.326.3.32	<code>IsNotIntervalFuzzy(double &t, double tol) const</code>	1594
29.326.3.33	<code>MakeIntObject(shared_ptr< ParamCurve > curve)</code>	1594
29.326.3.34	<code>NextSegmentVal(double par, bool forward, double tol) const</code>	1594
29.326.3.35	<code>NumParams() const</code>	1595
29.326.3.36	<code>ParamFromMesh(int dir, int idx)</code>	1595
29.326.3.37	<code>Point(Point &res, const double *par) const</code>	1595
29.326.3.38	<code>Point(std::vector< Point > &res, const double *par, int der, const bool *from_↵ right=0, double resolution=1.0e-12) const</code>	1595
29.326.3.39	<code>SetCriticalVal(int pdir, double par)</code>	1596
29.326.3.40	<code>StartParam(int pdir) const</code>	1596
29.326.3.41	<code>Startparam() const</code>	1596
29.326.3.42	<code>Subdivide(int pdir, double par, std::vector< shared_ptr< ParamGeomInt > > &subdiv_objs, std::vector< shared_ptr< ParamGeomInt > > &bd_objs)</code>	1596
29.326.4	Member Data Documentation	1596

29.326.4.1	<code>curve_</code>	1597
29.326.4.2	<code>dim_</code>	1597
29.326.4.3	<code>length_</code>	1597
29.326.4.4	<code>w_set_</code>	1597
29.326.4.5	<code>mesh_</code>	1597
29.326.4.6	<code>segment_</code>	1597
29.326.4.7	<code>wiggle_</code>	1597
29.327	ParametricSurfacePropertySheet Class Reference	1598
29.327.1	Detailed Description	1599
29.327.2	Constructor & Destructor Documentation	1599
29.327.2.1	<code>ParametricSurfacePropertySheet()</code>	1599
29.327.2.2	<code>ParametricSurfacePropertySheet(Go::ParametricSurfaceTesselator *tess, gv↔ ParametricSurfacePaintable *pable, shared_ptr< Go::ParamSurface > &surf)</code>	1599
29.327.2.3	<code>~ParametricSurfacePropertySheet()</code>	1599
29.327.3	Member Function Documentation	1599
29.327.3.1	<code>apply</code>	1599
29.327.3.2	<code>createSheet(QWidget *parent, gvObserver *obs)</code>	1599
29.328	Go::ParametricSurfaceTesselator Class Reference	1600
29.328.1	Detailed Description	1601
29.328.2	Constructor & Destructor Documentation	1601
29.328.2.1	<code>ParametricSurfaceTesselator(const ParamSurface &surf)</code>	1601
29.328.2.2	<code>~ParametricSurfaceTesselator()</code>	1601
29.328.3	Member Function Documentation	1601
29.328.3.1	<code>changeRes(int n, int m)</code>	1601
29.328.3.2	<code>getMesh()</code>	1601
29.328.3.3	<code>getRes(int &n, int &m)</code>	1601
29.328.3.4	<code>tesselate()</code>	1601
29.329	Go::ParamFunctionInt Class Reference	1602
29.329.1	Detailed Description	1603
29.329.2	Constructor & Destructor Documentation	1603
29.329.2.1	<code>~ParamFunctionInt()</code>	1603

29.329.3	Member Function Documentation	1603
29.329.3.1	getBoundaryObjects(std::vector< shared_ptr< BoundaryFunctionInt > > &bd_obj)=0	1603
29.329.3.2	getMesh()=0	1603
29.329.3.3	getMeshSize(int dir)=0	1604
29.329.3.4	getParam0FunctionInt()	1604
29.329.3.5	getParam1FunctionInt()	1604
29.329.3.6	getParam2FunctionInt()	1604
29.329.3.7	monotone(Point &dir, double tol=1.0e-15) const =0	1604
29.329.3.8	paramFromMesh(int dir, int idx)=0	1605
29.329.3.9	subdivide(int pardir, double par, std::vector< shared_ptr< ParamFunctionInt > > &subdiv_objs, std::vector< shared_ptr< ParamFunctionInt > > &bd_objs)=0	1605
29.329.4	Member Data Documentation	1605
29.329.4.1	boundary_obj_	1605
29.330	Go::ParamGeomInt Class Reference	1606
29.330.1	Detailed Description	1607
29.330.2	Constructor & Destructor Documentation	1607
29.330.2.1	ParamGeomInt(ParamGeomInt *parent=0)	1607
29.330.2.2	~ParamGeomInt()	1608
29.330.3	Member Function Documentation	1608
29.330.3.1	checkPeriodicity(int pardir) const	1608
29.330.3.2	compositeBox() const =0	1608
29.330.3.3	coneLargerThanPi()	1608
29.330.3.4	dimension() const =0	1608
29.330.3.5	directionCone() const =0	1609
29.330.3.6	getBoundaryObject(int bd_idx) const	1609
29.330.3.7	getBoundaryObjects(std::vector< shared_ptr< BoundaryGeomInt > > &bd_obj)=0	1609
29.330.3.8	getMesh()=0	1609
29.330.3.9	getMeshSize(int dir)=0	1609
29.330.3.10	getOptimizedConeAngle(Point &axis1, Point &axis2)=0	1610
29.330.3.11	Degenerate(double epsge, int dir)	1610

29.330.3.112	Degenerate(double epsge, int dir, double *par)=0	1610
29.330.3.113	Linear(double epsge)	1610
29.330.3.114	Spline()=0	1611
29.330.3.115	mbBdObj() const	1611
29.330.3.116	ParamFromMesh(int dir, int idx)=0	1611
29.330.3.117	reducedDirectionCone(bool reduce_at_bd[4], double epsge) const	1611
29.330.3.118	subdivide(int pardir, double par, std::vector< shared_ptr< ParamGeomInt > > &subdiv_objs, std::vector< shared_ptr< ParamGeomInt > > &bd_objs)=0 . . .	1612
29.330.4	Member Data Documentation	1612
29.330.4.1	boundary_obj_	1612
29.330	Go::ParamObjectInt Class Reference	1612
29.331.1	Detailed Description	1614
29.331.2	Constructor & Destructor Documentation	1614
29.331.2.1	ParamObjectInt(ParamObjectInt *parent=0)	1614
29.331.2.2	~ParamObjectInt()	1614
29.331.3	Member Function Documentation	1615
29.331.3.1	boundaryPoint(const double *par, double eps) const =0	1615
29.331.3.2	canDivide(int pardir)=0	1615
29.331.3.3	canDivideTinyTriang(int pardir)	1615
29.331.3.4	compositeBox() const =0	1615
29.331.3.5	endParam(int pardir) const =0	1615
29.331.3.6	getCriticalVals(int pardir) const =0	1616
29.331.3.7	getCriticalValsAndKnots(int pardir) const =0	1616
29.331.3.8	getInnerKnotVals(int pardir, bool sort=false) const =0	1616
29.331.3.9	getLengthAndWiggle(double *length, double *wiggle)=0	1616
29.331.3.10	getParamCurveInt()	1617
29.331.3.11	getParamPointInt()	1617
29.331.3.12	getParamSurfaceInt()	1617
29.331.3.13	getParent() const	1618
29.331.3.14	getSameTypeAncestor() const	1618
29.331.3.15	hasCriticalVals(int pardir) const =0	1618

29.331.3.16	IsCriticalValsOrKnots(int pdir) const =0	1618
29.331.3.17	IsInnerKnots(int pdir) const =0	1618
29.331.3.18	Corner(const double *par, double epspar) const	1619
29.331.3.19	Degenerate(double epsge, int dir, double *par)	1619
29.331.3.20	NumParams() const =0	1619
29.331.3.21	Point(Point &pt, const double *tpar) const =0	1619
29.331.3.22	Point(std::vector< Point > &pt, const double *tpar, int derivs, const bool *from← _right=0, double resolution=1.0e-12) const =0	1620
29.331.3.23	SetCriticalVal(int pdir, double par)	1620
29.331.3.24	SetParent(ParamObjectInt *parent)	1620
29.331.3.25	StartParam(int pdir) const =0	1620
29.331.4	Member Data Documentation	1621
29.331.4.1	parent_	1621
29.332	Co::ParamPointInt Class Reference	1621
29.332.1	Detailed Description	1623
29.332.2	Constructor & Destructor Documentation	1623
29.332.2.1	ParamPointInt(shared_ptr< Point > point, ParamGeomInt *parent=0)	1623
29.332.2.2	~ParamPointInt()	1623
29.332.3	Member Function Documentation	1623
29.332.3.1	boundaryPoint(const double *par, double eps) const	1623
29.332.3.2	canDivide(int pdir)	1624
29.332.3.3	compositeBox() const	1624
29.332.3.4	dimension() const	1624
29.332.3.5	directionCone() const	1624
29.332.3.6	endParam(int pdir) const	1624
29.332.3.7	getBoundaryObjects(std::vector< shared_ptr< BoundaryGeomInt > > &bd_objs)	1625
29.332.3.8	getCriticalVals(int pdir) const	1625
29.332.3.9	getCriticalValsAndKnots(int pdir) const	1625
29.332.3.10	getInnerKnotVals(int pdir, bool sort=false) const	1626
29.332.3.11	getLengthAndWiggle(double *length, double *wiggle)	1626
29.332.3.12	getMesh()	1626

29.332.3.19	GetMeshSize(int dir)	1626
29.332.3.19	GetOptimizedConeAngle(Point &axis1, Point &axis2)	1627
29.332.3.19	GetParamPointInt()	1627
29.332.3.19	GetRotatedBox(std::vector< Point > &axis) const	1627
29.332.3.17	HasCriticalVals(int paddir) const	1627
29.332.3.18	HasCriticalValsOrKnots(int paddir) const	1628
29.332.3.19	HasInnerKnots(int paddir) const	1628
29.332.3.20	Degenerate(double epsge, int dir, double *par)	1628
29.332.3.21	Spline()	1629
29.332.3.22	NumParams() const	1629
29.332.3.23	ParamFromMesh(int dir, int idx)	1629
29.332.3.24	Point(Point &pt, const double *tpar) const	1629
29.332.3.25	Point(std::vector< Point > &pt, const double *tpar, int derivs, const bool *from← _right=0, double resolution=1.0e-12) const	1630
29.332.3.26	StartParam(int paddir) const	1630
29.332.3.27	Subdivide(int paddir, double par, std::vector< shared_ptr< ParamGeomInt > > &subdiv_objs, std::vector< shared_ptr< ParamGeomInt > > &bd_objs)	1630
29.332.4	Member Data Documentation	1631
29.332.4.1	coords_	1631
29.332.4.2	dim_	1631
29.332.4.3	parentcurve_	1631
29.332.4.4	point_	1631
29.333	Go::ParamSurface Class Reference	1631
29.333.1	Detailed Description	1634
29.333.2	Constructor & Destructor Documentation	1634
29.333.2.1	~ParamSurface()	1634
29.333.2.2	ParamSurface()	1634
29.333.3	Member Function Documentation	1634
29.333.3.1	allBoundaryLoops(double degenerate_epsilon=DEFAULT_SPACE_EPSILON) const =0	1634
29.333.3.2	area(double tol) const =0	1635
29.333.3.3	asSplineSurface()	1635

29.333.3.4	clone() const =0	1635
29.333.3.5	closestBoundaryPoint(const Point &pt, double &clo_u, double &clo_v, Point &clo_pt, double &clo_dist, double epsilon, const RectDomain *rd=NULL, double *seed=0) const =0	1636
29.333.3.6	closestInDomain(double u, double v) const =0	1636
29.333.3.7	closestPoint(const Point &pt, double &clo_u, double &clo_v, Point &clo_pt, double &clo_dist, double epsilon, const RectDomain *domain_of_interest=NULL, double *seed=0) const	1636
29.333.3.8	closestPoint(const Point &pt, double &clo_u, double &clo_v, Point &clo_pt, double &clo_dist, double epsilon, int maxiter, const RectDomain *domain_of_interest=NULL, double *seed=0) const	1636
29.333.3.9	compositeBox() const	1637
29.333.3.10	constParamCurves(double parameter, bool pardir_is_u) const =0	1637
29.333.3.11	containingDomain() const =0	1637
29.333.3.12	ElementBoundaryStatus(int elem_ix, double eps)	1637
29.333.3.13	ElementOnBoundary(int elem_ix, double eps)	1638
29.333.3.14	estimateSfSize(double &u_size, double &v_size, int u_nmb=5, int v_nmb=5) const	1638
29.333.3.15	evalGrid(int num_u, int num_v, double umin, double umax, double vmin, double vmax, std::vector< double > &points, double nodata_val=-9999) const	1638
29.333.3.16	getBoundaryInfo(Point &pt1, Point &pt2, double epsilon, SplineCurve *&cv, SplineCurve *&crosscv, double knot_tol=1e-05) const =0	1639
29.333.3.17	getCornerPoints(std::vector< std::pair< Point, Point > > &corners) const =0	1639
29.333.3.18	getDegenerateCorners(std::vector< Point > °_corners, double tol) const =0	1639
29.333.3.19	getInternalPoint(double &u, double &v) const	1639
29.333.3.20	Domain(double u, double v, double eps=1.0e-4) const =0	1640
29.333.3.21	Domain2(double u, double v, double eps=1.0e-4) const =0	1640
29.333.3.22	AxisRotational(Point ¢re, Point &axis, Point &vec, double &angle)	1640
29.333.3.23	Degenerate(bool &b, bool &r, bool &t, bool &l, double tolerance) const	1640
29.333.3.24	IsoTrimmed(double tol) const	1641
29.333.3.25	Linear(Point &dir1, Point &dir2, double tol)	1641
29.333.3.26	Planar(Point &normal, double tol)	1641
29.333.3.27	Spline() const	1641
29.333.3.28	irrorSurface(const Point &pos, const Point &norm) const	1641
29.333.3.29	nextSegmentVal(int dir, double par, bool forward, double tol) const =0	1641

29.333.3.30	<code>normal(Point &n, double upar, double vpar) const =0</code>	1642
29.333.3.31	<code>normalCone() const =0</code>	1642
29.333.3.32	<code>onBoundary(double u, double v, double eps=1.0e-4) const =0</code>	1642
29.333.3.33	<code>outerBoundaryLoop(double degenerate_epsilon=DEFAULT_SPACE_EPSILON) const =0</code>	1642
29.333.3.34	<code>parameterDomain() const =0</code>	1643
29.333.3.35	<code>point(Point &pt, double upar, double vpar) const =0</code>	1643
29.333.3.36	<code>point(std::vector< Point > &pts, double upar, double vpar, int derivs, bool u_↔ from_right=true, bool v_from_right=true, double resolution=1.0e-12) const =0</code>	1643
29.333.3.37	<code>point(double upar, double vpar) const</code>	1644
29.333.3.38	<code>point(double upar, double vpar, int derivs) const</code>	1644
29.333.3.39	<code>reverseParameterDirection(bool direction_is_u)=0</code>	1645
29.333.3.40	<code>1773(const double ppoint[], double aepsge, double estart[], double eend[], dou- ble enext[], double gpos[], int maxiter, int *jstat) const</code>	1645
29.333.3.41	<code>1773_s9corr(double gd[], double acoef1, double acoef2, double astart1, double aend1, double astart2, double aend2) const</code>	1645
29.333.3.42	<code>1773_s9dir(double *cdist, double *cdiff1, double *cdiff2, double PS[], const dou- ble *eval1, std::vector< Point > eval2, double aepsge, int idim, int *jstat) const</code>	1645
29.333.3.43	<code>Iterator(IteratorType type)</code>	1645
29.333.3.44	<code>setParameterDomain(double u1, double u2, double v1, double v2)</code>	1645
29.333.3.45	<code>singularity(double &sing_u, double &sing_v, Point &sing_pt, double &sing_dist, double epsilon, const RectDomain *rd=NULL, double *seed=0) const</code>	1646
29.333.3.46	<code>subSurfaces(double from_upar, double from_vpar, double to_upar, double to_↔ vpar, double fuzzy=DEFAULT_PARAMETER_EPSILON) const =0</code>	1646
29.333.3.47	<code>swapParameterDirection()=0</code>	1646
29.333.3.48	<code>tangentCone(bool pardir_is_u) const =0</code>	1646
29.333.3.49	<code>turnOrientation()=0</code>	1647
29.333.4	Member Data Documentation	1647
29.333.4.1	<code>Idegen_</code>	1647
29.333.4.2	<code>Iterator_</code>	1647
29.334	Go::ParamSurfaceInt Class Reference	1647
29.334.1	Detailed Description	1651
29.334.2	Constructor & Destructor Documentation	1651

29.334.2.1ParamSurfaceInt(shared_ptr< ParamSurface > surf, ParamGeomInt *parent=0)	1651
29.334.2.2~ParamSurfaceInt()	1651
29.334.3Member Function Documentation	1651
29.334.3.1atDegenerateBd(const double *par, double epsge, double epspar)	1651
29.334.3.2axisFromCorners(Point &axis1, Point &axis2) const	1651
29.334.3.3boundaryPoint(const double *par, double eps) const	1651
29.334.3.4canDivide(int paddir)	1652
29.334.3.5canDivideTinyTriang(int paddir)	1652
29.334.3.6canImplicitize()	1652
29.334.3.7canSelfIntersect(double epsge) const	1652
29.334.3.8checkPeriodicity(int paddir) const	1652
29.334.3.9compositeBox() const	1653
29.334.3.10derivs(double u, double v, Point &deriv_u, Point &deriv_v, bool from_right_1=true, bool from_right_2=true) const	1653
29.334.3.11dimension() const	1653
29.334.3.12directionCone() const	1654
29.334.3.13endParam(int paddir) const	1654
29.334.3.14first_fund_form(double u, double v, bool u_from_right, bool v_from_right, double &E, double &F, double &G) const	1654
29.334.3.15getBoundaryObjects(std::vector< shared_ptr< BoundaryGeomInt > > &bd_objs)	1654
29.334.3.16getConstantParameterCurve(int dir, double par)	1655
29.334.3.17getConstantParameterCurve(int dir, double par, double tmin, double tmax)	1655
29.334.3.18getCriticalVals(int paddir) const	1655
29.334.3.19getCriticalValsAndKnots(int paddir) const	1655
29.334.3.20getDegDomain(double epsge)	1655
29.334.3.21getDegTriang()	1656
29.334.3.22getDomain() const	1656
29.334.3.23getImplicit(double tol, double &tol2, AlgObj3DInt &alg_obj_3d_int)	1656
29.334.3.24getInnerKnotVals(int paddir, bool sort=false) const	1656
29.334.3.25getIsoCurve(double param_start, double param_end, double isoval, bool paddir←_is_u) const	1657
29.334.3.26getLengthAndWiggle(double *length, double *wiggle)	1657

29.334.3.27	<code>GetMesh()</code>	1657
29.334.3.28	<code>GetMeshSize(int dir)</code>	1657
29.334.3.29	<code>GetMima() const</code>	1658
29.334.3.30	<code>GetNormalSurface() const</code>	1658
29.334.3.31	<code>GetOptimizedConeAngle(Point &axis1, Point &axis2)</code>	1658
29.334.3.32	<code>GetParamSurface()</code>	1658
29.334.3.33	<code>GetParamSurface() const</code>	1659
29.334.3.34	<code>GetParamSurfaceInt()</code>	1659
29.334.3.35	<code>GetParentParamSurface(RectDomain &domain)</code>	1659
29.334.3.36	<code>GetParentParamSurface()</code>	1659
29.334.3.37	<code>GetParOffBd(int dir, bool atstart, double tol) const</code>	1659
29.334.3.38	<code>GetRotatedBox(std::vector< Point > &axis) const</code>	1660
29.334.3.39	<code>GetSingularity(double eps, double sing_par[], Point &sing_pt, double &sing_val, double *seed)</code>	1660
29.334.3.40	<code>HasCriticalVals(int pardir) const</code>	1660
29.334.3.41	<code>HasCriticalValsOrKnots(int pardir) const</code>	1661
29.334.3.42	<code>HasInnerKnots(int pardir) const</code>	1661
29.334.3.43	<code>Implicitize(double tol)</code>	1661
29.334.3.44	<code>IsCorner(const double *par, double epspar) const</code>	1661
29.334.3.45	<code>IsDegenerate(double epsge)</code>	1662
29.334.3.46	<code>IsDegenerate(double epsge, int dir)</code>	1662
29.334.3.47	<code>IsDegenerate(double epsge, int dir, double *par)</code>	1662
29.334.3.48	<code>IsIsoParametric(ParamCurveInt *curve, int dir, double par, double ptol, double tol)</code>	1662
29.334.3.49	<code>IsolateDegPar(int dir, int deg_edge, double threshold, double *deg_factor=NULL)</code>	1663
29.334.3.50	<code>IsSimple()</code>	1663
29.334.3.51	<code>IsSpline()</code>	1663
29.334.3.52	<code>IsNotIntervalFuzzy(double &u, double &v, double utol, double vtol) const</code>	1663
29.334.3.53	<code>MakeIntCurve(shared_ptr< ParamCurve > crv, ParamGeomInt *parent)</code>	1664
29.334.3.54	<code>MakeIntObject(shared_ptr< ParamSurface > surf)</code>	1664
29.334.3.55	<code>MaxCurvatures(bool dir_is_u, int nmb_params, double iso_from, double iso_to, double guess_param, std::vector< double > &max_curv_params, std::vector< double > &max_curvatures)</code>	1664

29.334.3.56	MinimumAlongCurve(int dir, double par, double tmin, double tmax, Point &pt, double &minpar, Point &minval, double &mindist)	1664
29.334.3.57	NextSegmentVal(int dir, double par, bool forward, double tol) const	1664
29.334.3.58	Normal(double u, double v, Point &normal, bool from_right_1=true, bool from_right_2=true) const	1665
29.334.3.59	NumParams() const	1665
29.334.3.60	ParamFromMesh(int dir, int idx)	1665
29.334.3.61	Point(Point &pt, const double *tpar) const	1665
29.334.3.62	Point(std::vector< Point > &pt, const double *tpar, int derivs, const bool *from_right=0, double resolution=1.0e-12) const	1666
29.334.3.63	Second_fund_form(double u, double v, bool u_from_right, bool v_from_right, double &L, double &M, double &N) const	1666
29.334.3.64	SetDegTriang()	1666
29.334.3.65	SplitAtG0(double angtol, std::vector< shared_ptr< ParamSurfaceInt > > &subG1)	1667
29.334.3.66	StartParam(int pdir) const	1667
29.334.3.67	Subdivide(int pdir, double par, std::vector< shared_ptr< ParamGeomInt > > &subdiv_objs, std::vector< shared_ptr< ParamGeomInt > > &bd_objs)	1667
29.334.3.68	SubSurfaces(double from_upar, double from_vpar, double to_upar, double to_vpar, double fuzzy)	1667
29.334.4	Member Data Documentation	1668
29.334.4.1	bd_deg_	1668
29.334.4.2	cone_	1668
29.334.4.3	deg_domain_	1668
29.334.4.4	deg_tol_	1668
29.334.4.5	deg_triang_	1668
29.334.4.6	dim_	1668
29.334.4.7	domain_	1668
29.334.4.8	impl_deg_	1668
29.334.4.9	impl_sf_algo_	1669
29.334.4.10	implicit_err_	1669
29.334.4.11	implicit_obj_	1669
29.334.4.12	implicit_tol_	1669
29.334.4.13	length_	1669

29.334.4.14	<code>w_set_</code>	1669
29.334.4.15	<code>mesh_</code>	1669
29.334.4.16	<code>mesh_</code>	1669
29.334.4.17	<code>segment_</code>	1669
29.334.4.18	<code>surf_</code>	1669
29.334.4.19	<code>comp_point_array_</code>	1670
29.334.4.20	<code>wiggle_</code>	1670
29.335	<code>ParamVolume Class Reference</code>	1670
29.335.1	Detailed Description	1671
29.335.2	Constructor & Destructor Documentation	1671
29.335.2.1	<code>~ParamVolume()</code>	1671
29.335.3	Member Function Documentation	1671
29.335.3.1	<code>asSplineVolume()</code>	1671
29.335.3.2	<code>clone() const =0</code>	1672
29.335.3.3	<code>closestPoint(const Point &pt, double &clo_u, double &clo_v, double &clo_w, Point &clo_pt, double &clo_dist, double epsilon, double *seed=0) const =0</code>	1672
29.335.3.4	<code>constParamSurface(double parameter, int pardir) const</code>	1672
29.335.3.5	<code>getAllBoundarySurfaces() const =0</code>	1673
29.335.3.6	<code>asSpline() const</code>	1673
29.335.3.7	<code>nextSegmentVal(int dir, double par, bool forward, double tol) const =0</code>	1673
29.335.3.8	<code>parameterSpan() const =0</code>	1673
29.335.3.9	<code>point(Point &pt, double upar, double vpar, double wpar) const =0</code>	1674
29.335.3.10	<code>point(std::vector< Point > &pts, double upar, double vpar, double wpar, int derivs, bool u_from_right=true, bool v_from_right=true, bool w_from_right=true, double resolution=1.0e-12) const =0</code>	1674
29.335.3.11	<code>reverseParameterDirection(int pardir)=0</code>	1675
29.335.3.12	<code>swapParameterDirection(int pardir1, int pardir2)=0</code>	1675
29.335.3.13	<code>tangentCone(int pardir) const =0</code>	1675
29.335.3.14	<code>translate(const Point &vec)=0</code>	1675
29.336	<code>PathType Class Reference</code>	1676
29.336.1	Detailed Description	1676
29.336.2	Member Function Documentation	1676

29.336.2.1	funnelOfPath(PathType prec)	1676
29.336.2.2	initPathType()	1676
29.336.2.3	modification_l_path_(int new_pt, const vector< UnfNodeType > &nodes_unf)	1676
29.336.2.4	modification_r_path_(int new_pt, const vector< UnfNodeType > &nodes_unf)	1677
29.336.2.5	print()	1677
29.336.3	Member Data Documentation	1677
29.336.3.1	funnel_	1677
29.336.3.2	_path_	1677
29.336.3.3	_path_	1677
29.337	Go::Plane Class Reference	1677
29.337.1	Detailed Description	1680
29.337.2	Constructor & Destructor Documentation	1680
29.337.2.1	Plane()	1680
29.337.2.2	Plane(Point location, Point normal, bool isSwapped=false)	1680
29.337.2.3	Plane(Point location, Point normal, Point x_axis, bool isSwapped=false)	1681
29.337.2.4	Plane(double a, double b, double c, double d, bool isSwapped=false)	1681
29.337.2.5	~Plane()	1681
29.337.3	Member Function Documentation	1681
29.337.3.1	allBoundaryLoops(double degenerate_epsilon=DEFAULT_SPACE_EPSILON) const	1681
29.337.3.2	boundingBox() const	1681
29.337.3.3	classType()	1681
29.337.3.4	clone() const	1682
29.337.3.5	closestBoundaryPoint(const Point &pt, double &clo_u, double &clo_v, Point &clo_pt, double &clo_dist, double epsilon, const RectDomain *rd=NULL, double *seed=0) const	1682
29.337.3.6	closestPoint(const Point &pt, double &clo_u, double &clo_v, Point &clo_pt, double &clo_dist, double epsilon, const RectDomain *domain_of_interest=NULL, double *seed=0) const	1682
29.337.3.7	constParamCurves(double parameter, bool pardir_is_u) const	1682
29.337.3.8	createSplineSurface() const	1683
29.337.3.9	dimension() const	1683
29.337.3.10	instance(const Point &pnt) const	1683

29.337.3.1	<code>geometrySurface() const</code>	1683
29.337.3.1	<code>getBoundaryInfo(Point &pt1, Point &pt2, double epsilon, SplineCurve *&cv, SplineCurve *&crosscv, double knot_tol=1e-05) const</code>	1683
29.337.3.1	<code>getDegenerateCorners(std::vector< Point > &deg_corners, double tol) const</code>	1684
29.337.3.1	<code>getNormal()</code>	1684
29.337.3.1	<code>getPoint()</code>	1684
29.337.3.1	<code>getSpanningVectors(Point &axis1, Point &axis2)</code>	1684
29.337.3.1	<code>instanceType() const</code>	1684
29.337.3.1	<code>intersect(const RotatedBox &bd_box) const</code>	1684
29.337.3.1	<code>isBounded() const</code>	1684
29.337.3.2	<code>isClosed(bool &closed_dir_u, bool &closed_dir_v) const</code>	1685
29.337.3.2	<code>isDegenerate(bool &b, bool &r, bool &t, bool &l, double tolerance) const</code>	1685
29.337.3.2	<code>isLinear(Point &dir1, Point &dir2, double tol)</code>	1685
29.337.3.2	<code>isPlanar(Point &normal, double tol)</code>	1685
29.337.3.2	<code>nextSegmentVal(int dir, double par, bool forward, double tol) const</code>	1685
29.337.3.2	<code>normal(Point &n, double upar, double vpar) const</code>	1686
29.337.3.2	<code>normalCone() const</code>	1686
29.337.3.2	<code>parameterDomain() const</code>	1686
29.337.3.2	<code>point(Point &pt, double upar, double vpar) const</code>	1686
29.337.3.2	<code>point(std::vector< Point > &pts, double upar, double vpar, int derivs, bool u_from_right=true, bool v_from_right=true, double resolution=1.0e-12) const</code>	1687
29.337.3.3	<code>projectPoint(const Point &pnt) const</code>	1687
29.337.3.3	<code>read(std::istream &is)</code>	1687
29.337.3.3	<code>setParameterBounds(double from_upar, double from_vpar, double to_upar, double to_vpar)</code>	1688
29.337.3.3	<code>setSpanningVectors()</code>	1688
29.337.3.3	<code>setSpanningVectorsSafe()</code>	1688
29.337.3.3	<code>subSurface(double from_upar, double from_vpar, double to_upar, double to_vpar, double fuzzy=DEFAULT_PARAMETER_EPSILON) const</code>	1688
29.337.3.3	<code>subSurfaces(double from_upar, double from_vpar, double to_upar, double to_vpar, double fuzzy=DEFAULT_PARAMETER_EPSILON) const</code>	1688
29.337.3.3	<code>tangentCone(bool pardir_is_u) const</code>	1688
29.337.3.3	<code>write(std::ostream &os) const</code>	1689

29.337.4	Member Data Documentation	1689
29.337.4.1	domain_	1689
29.337.4.2	location_	1689
29.337.4.3	normal_	1689
29.337.4.4	orientedDomain_	1689
29.337.4.5	vec1_	1689
29.337.4.6	vec2_	1689
29.338	Go::PlaneInt Class Reference	1690
29.338.1	Detailed Description	1691
29.338.2	Constructor & Destructor Documentation	1691
29.338.2.1	PlaneInt()	1691
29.338.2.2	PlaneInt(Point point, Point normal)	1691
29.338.2.3	PlaneInt(double a, double b, double c, double d)	1691
29.338.2.4	~PlaneInt()	1691
29.338.3	Member Function Documentation	1691
29.338.3.1	a() const	1691
29.338.3.2	b() const	1692
29.338.3.3	c() const	1692
29.338.3.4	d() const	1692
29.338.3.5	read(std::istream &is)	1692
29.338.3.6	surface(Point mid_pt, double length_x, double length_y) const	1692
29.339	Go::Point Class Reference	1693
29.339.1	Detailed Description	1695
29.339.2	Constructor & Destructor Documentation	1695
29.339.2.1	Point()	1695
29.339.2.2	Point(int dim)	1695
29.339.2.3	Point(double x, double y)	1695
29.339.2.4	Point(double x, double y, double z)	1695
29.339.2.5	Point(const Array< T, Dim > &v)	1695
29.339.2.6	Point(RandomAccessIterator first, RandomAccessIterator last)	1696

29.339.2.7	Point(double *begin, double *end, bool own)	1696
29.339.2.8	Point(const Point &v)	1696
29.339.2.9	~Point()	1696
29.339.3	Member Function Documentation	1696
29.339.3.1	angle(const Point &v) const	1696
29.339.3.2	angle2(const Point &v) const	1696
29.339.3.3	angle_smallest(const Point &v) const	1696
29.339.3.4	begin() const	1697
29.339.3.5	begin()	1697
29.339.3.6	cosAngle(const Point &v) const	1697
29.339.3.7	cross(const Point &v) const	1697
29.339.3.8	dimension() const	1697
29.339.3.9	dist(const Point &v) const	1697
29.339.3.10	dist2(const Point &v) const	1697
29.339.3.11	distInf(const Point &v) const	1697
29.339.3.12	end() const	1698
29.339.3.13	end()	1698
29.339.3.14	length() const	1698
29.339.3.15	length2() const	1698
29.339.3.16	lengthInf() const	1698
29.339.3.17	normalize()	1698
29.339.3.18	normalize_checked()	1698
29.339.3.19	operator%(const Point &v) const	1698
29.339.3.20	operator*(double d) const	1699
29.339.3.21	operator*(const Point &v) const	1699
29.339.3.22	operator*=(double d)	1699
29.339.3.23	operator+(const Point &v) const	1699
29.339.3.24	operator+=(const Point &v)	1699
29.339.3.25	operator-(const Point &v) const	1699
29.339.3.26	operator-() const	1699

29.339.3.27	operator==(const Point &v)	1699
29.339.3.28	operator/(double d) const	1700
29.339.3.29	operator/=(double d)	1700
29.339.3.30	operator=(const Point &v)	1700
29.339.3.31	operator[](int i) const	1700
29.339.3.32	operator[](int i)	1700
29.339.3.33	read(std::istream &is)	1700
29.339.3.34	setValue(int idx, double val)	1700
29.339.3.35	size(int d)	1700
29.339.3.36	toCrossProd(const Point &u, const Point &v)	1701
29.339.3.37	setValue(double x, double y)	1701
29.339.3.38	setValue(double x, double y, double z)	1701
29.339.3.39	setValue(const double *array)	1701
29.339.3.40	setValue(double val)	1701
29.339.3.41	size() const	1701
29.339.3.42	swap(Point &other)	1701
29.339.3.43	write(std::ostream &os) const	1702
29.340	Go::PointCloud< Dim > Class Template Reference	1702
29.340.1	Detailed Description	1703
29.340.2	Constructor & Destructor Documentation	1703
29.340.2.1	PointCloud()	1703
29.340.2.2	PointCloud(ForwardIterator start, int numpoints)	1703
29.340.2.3	PointCloud(std::vector< Array< double, Dim > > &points)	1704
29.340.2.4	~PointCloud()	1704
29.340.3	Member Function Documentation	1704
29.340.3.1	boundingBox() const	1704
29.340.3.2	classType()	1704
29.340.3.3	clone() const	1704
29.340.3.4	dimension() const	1705
29.340.3.5	instanceType() const	1705

29.340.3.6	numPoints() const	1705
29.340.3.7	point(int i)	1705
29.340.3.8	point(int i) const	1705
29.340.3.9	pointVector()	1706
29.340.3.10	rawData()	1706
29.340.3.11	rawData() const	1706
29.340.3.12	read(std::istream &is)	1706
29.340.3.13	rotate(const Vector3D &p, const Vector3D &q)	1707
29.340.3.14	translate(Array< double, Dim > vec)	1707
29.340.3.15	write(std::ostream &os) const	1707
29.341	PointCloudPropertySheet Class Reference	1707
29.341.1	Detailed Description	1708
29.341.2	Constructor & Destructor Documentation	1709
29.341.2.1	PointCloudPropertySheet(gvPointCloudPaintable *pable)	1709
29.341.3	Member Function Documentation	1709
29.341.3.1	apply	1709
29.341.3.2	createSheet(QWidget *parent, gvObserver *obs)	1709
29.342	Go::PointOnCurve Class Reference	1709
29.342.1	Detailed Description	1709
29.342.2	Constructor & Destructor Documentation	1710
29.342.2.1	PointOnCurve()	1710
29.342.2.2	PointOnCurve(shared_ptr< ParamCurve > curve, double par)	1710
29.342.2.3	PointOnCurve(shared_ptr< ParamCurve > curve, Point pnt)	1710
29.342.2.4	~PointOnCurve()	1710
29.342.3	Member Function Documentation	1710
29.342.3.1	evaluate(int der, std::vector< Point > &deriv) const	1710
29.342.3.2	getCurve() const	1710
29.342.3.3	getPar() const	1710
29.342.3.4	getPos() const	1710
29.342.3.5	setParInterval(double start, double end)	1711

29.343	Go::PointOnEdge Class Reference	1711
29.343.1	Detailed Description	1711
29.343.2	Constructor & Destructor Documentation	1711
29.343.2.1	PointOnEdge(ffEdge *edge, double par)	1711
29.343.2.2	~PointOnEdge()	1711
29.343.3	Member Function Documentation	1712
29.343.3.1	edge() const	1712
29.343.3.2	par() const	1712
29.343.3.3	position() const	1712
29.344	Go::PointSequence Class Reference	1712
29.344.1	Detailed Description	1713
29.344.2	Constructor & Destructor Documentation	1713
29.344.2.1	PointSequence()	1713
29.344.2.2	PointSequence(int dim, int nmb_pts, RandomIterator coefs_start, PointSequenceType pst=PSTPoint)	1713
29.344.2.3	PointSequence(int dim, int nmb_pts_1, int nmb_pts_2, RandomIterator coefs_start, PointSequenceType pst=PSTPoint)	1713
29.344.2.4	PointSequence(int dim, int nmb_pts_1, int nmb_pts_2, int nmb_pts_3, RandomIterator coefs_start, PointSequenceType pst=PSTPoint)	1713
29.344.2.5	~PointSequence()	1714
29.344.3	Member Function Documentation	1714
29.344.3.1	coefs_begin()	1714
29.344.3.2	coefs_begin() const	1714
29.344.3.3	coefs_end()	1714
29.344.3.4	coefs_end() const	1714
29.344.3.5	dimension() const	1714
29.344.3.6	grid_dimension() const	1714
29.344.3.7	grid_length(int i) const	1714
29.344.3.8	type() const	1715
29.345	rBiCGStab Class Reference	1715
29.345.1	Detailed Description	1716
29.345.2	Constructor & Destructor Documentation	1716

29.345.2.1PrBiCGStab()	1716
29.345.2.2~PrBiCGStab()	1716
29.345.3Member Function Documentation	1716
29.345.3.1converged()	1716
29.345.3.2getCPUTime()	1716
29.345.3.3getItCount()	1716
29.345.3.4setMaxIterations(int max_iterations)	1717
29.345.3.5setTolerance(double tolerance=1.0e-6)	1717
29.345.3.6solve(const PrMatrix &A, PrVec &x, const PrVec &b)	1717
29.345.4Member Data Documentation	1717
29.345.4.1converged_	1717
29.345.4.2cpu_time_	1717
29.345.4.3it_count_	1717
29.345.4.4max_iterations_	1717
29.345.4.5tolerance_	1717
29.346PrCellStructure Class Reference	1718
29.346.1Detailed Description	1718
29.346.2Constructor & Destructor Documentation	1718
29.346.2.1PrCellStructure()	1718
29.346.2.2PrCellStructure(int n, double *xyz_points, int max_no_cells=10)	1718
29.346.2.3~PrCellStructure()	1719
29.346.3Member Function Documentation	1719
29.346.3.1attach(int n, const double *xyz_points)	1719
29.346.3.2get3dNode(int i) const	1719
29.346.3.3getBall(const Vector3D &p, double radius, vector< int > &neighbours, int notP=0) const	1719
29.346.3.4getI(int i, int j, int k) const	1720
29.346.3.5getIJK(int ii, int &i, int &j, int &k) const	1720
29.346.3.6getKNearest(const Vector3D &p, int k, vector< int > &neighbours, int notP=0) const	1720
29.346.3.7getNumCells() const	1720

29.346.3.8	getNumNodes() const	1720
29.346.3.9	print(ostream &os)	1721
29.346.3.10	can(istream &is)	1721
29.346.3.11	set3dNode(int i, const Vector3D &p)	1721
29.346.3.12	setNumCells(int max_no_cells)	1721
29.346.3.13	whichCell(const Vector3D &xyz, int &i, int &j, int &k) const	1721
29.347	PrCG Class Reference	1722
29.347.1	Detailed Description	1723
29.347.2	Constructor & Destructor Documentation	1723
29.347.2.1	PrCG()	1723
29.347.2.2	~PrCG()	1723
29.347.3	Member Function Documentation	1723
29.347.3.1	converged()	1723
29.347.3.2	getCPUtime()	1723
29.347.3.3	getItCount()	1723
29.347.3.4	setMaxIterations(int max_iterations)	1723
29.347.3.5	setTolerance(double tolerance=1.0e-6)	1724
29.347.3.6	solve(const PrMatrix &A, PrVec &x, const PrVec &b)	1724
29.347.4	Member Data Documentation	1724
29.347.4.1	converged_	1724
29.347.4.2	cpu_time_	1724
29.347.4.3	it_count_	1724
29.347.4.4	max_iterations_	1724
29.347.4.5	tolerance_	1724
29.348	Go::preEvaluationSf Struct Reference	1724
29.348.1	Detailed Description	1725
29.348.2	Member Data Documentation	1725
29.348.2.1	basisvals_u_	1725
29.348.2.2	basisvals_v_	1725
29.348.2.3	deriv_u_	1725

29.348.2.4	deriv_v_	1725
29.348.2.5	gauss_par1_	1725
29.348.2.6	gauss_par2_	1725
29.348.2.7	left_u_	1726
29.348.2.8	left_v_	1726
29.348.2.9	points_	1726
29.348	Go::preEvaluationVol Struct Reference	1726
29.349.1	Detailed Description	1726
29.349.2	Member Data Documentation	1726
29.349.2.1	basisvals_u_	1726
29.349.2.2	basisvals_v_	1727
29.349.2.3	basisvals_w_	1727
29.349.2.4	deriv_u_	1727
29.349.2.5	deriv_v_	1727
29.349.2.6	deriv_w_	1727
29.349.2.7	gauss_par1_	1727
29.349.2.8	gauss_par2_	1727
29.349.2.9	gauss_par3_	1727
29.349.2.10	ft_u_	1727
29.349.2.11	ft_v_	1727
29.349.2.12	ft_w_	1728
29.349.2.13	points_	1728
29.350	ExplicitConnectivity Class Reference	1728
29.350.1	Detailed Description	1729
29.350.2	Member Function Documentation	1729
29.350.2.1	findFace(int i, int j, std::vector< int > &neighbours, std::vector< int > &face) const	1729
29.350.2.2	findGenus() const	1730
29.350.2.3	findNextEdgeInFace(int &i, int &j, std::vector< int > &neighbours) const	1730
29.350.2.4	findNumFaces() const	1730
29.350.2.5	sTriangulation() const	1730

29.350.2.6	printInfo(std::ostream &os) const	1730
29.350.2.7	printTexture(std::ostream &os) const	1730
29.350.2.8	printUVFaces(std::ostream &os) const	1730
29.350.2.9	printXYZFaces(std::ostream &os) const	1731
29.350.2.10	printXYZFacesML(std::ostream &os) const	1731
29.350.2.11	printXYZFacesVRML(std::ostream &os) const	1731
29.351	PrFaber_F Class Reference	1731
29.351.1	Detailed Description	1732
29.351.2	Constructor & Destructor Documentation	1732
29.351.2.1	PrFaber_F()	1732
29.351.2.2	~PrFaber_F()	1733
29.351.3	Member Function Documentation	1733
29.351.3.1	compose(int jlev, int dim=1)	1733
29.351.3.2	decompose(int jlev, int dim=1)	1733
29.352	PrFastUnorganized_OP Class Reference	1733
29.352.1	Detailed Description	1735
29.352.2	Constructor & Destructor Documentation	1735
29.352.2.1	PrFastUnorganized_OP(int num_cells=10)	1735
29.352.2.2	PrFastUnorganized_OP(int n, int n_int, double *xyz_points, int num_cells=10)	1735
29.352.2.3	~PrFastUnorganized_OP()	1735
29.352.3	Member Function Documentation	1735
29.352.3.1	get3dNode(int i) const	1735
29.352.3.2	getKNearest()	1735
29.352.3.3	getNeighbours(int i, vector< int > &neighbours) const	1735
29.352.3.4	getNumNodes() const	1736
29.352.3.5	getRadius()	1736
29.352.3.6	getU(int i) const	1736
29.352.3.7	getV(int i) const	1736
29.352.3.8	nitNeighbours()	1736
29.352.3.9	sBoundary(int i) const	1736

29.352.3.10	Print(ostream &os)	1736
29.352.3.11	ScanRawData(istream &is, int num_cells=10, double noise=0.0)	1736
29.352.3.12	Set3dNode(int i, const Vector3D &p)	1737
29.352.3.13	SetKNearest(int knearest)	1737
29.352.3.14	SetRadius(double radius)	1737
29.352.3.15	SetU(int i, double u)	1737
29.352.3.16	SetV(int i, double v)	1737
29.352.3.17	UseK()	1738
29.352.3.18	UseRadius()	1738
29.353	PrFilterbank Class Reference	1738
29.353.1	Detailed Description	1739
29.353.2	Constructor & Destructor Documentation	1739
29.353.2.1	~PrFilterbank()	1739
29.353.3	Member Function Documentation	1739
29.353.3.1	attach(shared_ptr< PrNestedTriangulation > t)	1739
29.353.3.2	compose(int jlev, int dim=1)=0	1739
29.353.3.3	composeAll(int dim=1)	1740
29.353.3.4	composeUpTo(int jlev, int dim=1)	1740
29.353.3.5	decompose(int jlev, int dim=1)=0	1740
29.353.3.6	decomposeAll(int dim=1)	1740
29.353.3.7	decomposeFrom(int jlev, int dim=1)	1740
29.353.4	Member Data Documentation	1740
29.353.4.1	neighbours_	1740
29.353.4.2	_	1740
29.354	PrHeap Class Reference	1740
29.354.1	Detailed Description	1741
29.354.2	Constructor & Destructor Documentation	1741
29.354.2.1	PrHeap()	1741
29.354.2.2	PrHeap(int maxsize)	1741
29.354.2.3	PrHeap(int *elems, int n)	1742

29.354.2.4	PrHeap(const PrHeap &heap)	1742
29.354.2.5	~PrHeap()	1742
29.354.3	Member Function Documentation	1742
29.354.3.1	emptyHeap()	1742
29.354.3.2	getMaxSize() const	1742
29.354.3.3	getSize() const	1742
29.354.3.4	modify(double new_key, int element)	1742
29.354.3.5	pop()	1742
29.354.3.6	pop(double &key, int &element)	1742
29.354.3.7	print(std::ostream &os) const	1743
29.354.3.8	push(double key, int element)	1743
29.354.3.9	redim(int size)	1743
29.355	PrintCounter Class Reference	1743
29.355.1	Detailed Description	1743
29.355.2	Constructor & Destructor Documentation	1743
29.355.2.1	~PrintCounter()	1743
29.355.2.2	PrintCounter(const char *sx)	1743
29.355.3	Member Function Documentation	1743
29.355.3.1	operator++()	1743
29.356	PrLevelTriangulation_OP Class Reference	1744
29.356.1	Detailed Description	1745
29.356.2	Constructor & Destructor Documentation	1745
29.356.2.1	PrLevelTriangulation_OP()	1745
29.356.2.2	PrLevelTriangulation_OP(vector< PrNestedNode > *node, vector< PrTriangle > &nt, int level=0)	1745
29.356.2.3	~PrLevelTriangulation_OP()	1746
29.356.3	Member Function Documentation	1746
29.356.3.1	findNumFaces() const	1746
29.356.3.2	get3dNode(int i) const	1746
29.356.3.3	getNeighbours(int i, vector< int > &neighbours) const	1746
29.356.3.4	getNumNodes() const	1746

29.356.3.5	getPrNestedNode(int i)	1747
29.356.3.6	getPrTriangle(int i)	1747
29.356.3.7	getU(int i) const	1747
29.356.3.8	getV(int i) const	1747
29.356.3.9	isBoundary(int i) const	1747
29.356.3.10	print(ostream &os)	1747
29.356.3.11	printRawData(ostream &os)	1747
29.356.3.12	printUV(ostream &os)	1747
29.356.3.13	printUVTriangles(ostream &os, bool num=false)	1748
29.356.3.14	printXYZTriangles(ostream &os, bool num=false)	1748
29.356.3.15	set3dNode(int i, const Vector3D &p)	1748
29.356.3.16	setU(int i, double u)	1748
29.356.3.17	setV(int i, double v)	1748
29.357	PrMat Class Reference	1749
29.357.1	Detailed Description	1750
29.357.2	Constructor & Destructor Documentation	1750
29.357.2.1	PrMat()	1750
29.357.2.2	PrMat(int m, int n, double val=0.0)	1750
29.357.2.3	PrMat()	1750
29.357.3	Member Function Documentation	1750
29.357.3.1	columns() const	1750
29.357.3.2	operator()(int i, int j) const	1751
29.357.3.3	operator()(int i, int j)	1751
29.357.3.4	prod(const PrVec &x, PrVec &y) const	1751
29.357.3.5	read(std::istream &is)	1751
29.357.3.6	redim(int m, int n)	1751
29.357.3.7	rows() const	1751
29.357.4	Member Data Documentation	1751
29.357.4.1	a_	1751
29.357.4.2	m_	1751

29.357.4.3n_	1752
29.358 PrMatrix Class Reference	1752
29.358.1 Detailed Description	1753
29.358.2 Constructor & Destructor Documentation	1753
29.358.2.1 ~PrMatrix()	1753
29.358.3 Member Function Documentation	1753
29.358.3.1 columns() const =0	1753
29.358.3.2 operator()(int i, int j) const =0	1753
29.358.3.3 print(std::ostream &os)	1753
29.358.3.4 prod(const PrVec &x, PrVec &y) const =0	1753
29.358.3.5 read(std::istream &is)=0	1753
29.358.3.6 rows() const =0	1754
29.359 PrMatSparse Class Reference	1754
29.359.1 Detailed Description	1755
29.359.2 Constructor & Destructor Documentation	1756
29.359.2.1 ~PrMatSparse()	1756
29.359.2.2 PrMatSparse()	1756
29.359.2.3 PrMatSparse(int m, int n, int num_nonzero)	1756
29.359.2.4 PrMatSparse(int m, int n, int num_nonzero, const int *irow, const int *jcol, const double *data)	1756
29.359.3 Member Function Documentation	1756
29.359.3.1 columns() const	1756
29.359.3.2 row(int k)	1756
29.359.3.3 row(int k) const	1756
29.359.3.4 col(int k)	1756
29.359.3.5 col(int k) const	1757
29.359.3.6 matProd(PrMatSparse &B, PrMatSparse &C) const	1757
29.359.3.7 operator()(int i, int j) const	1757
29.359.3.8 operator()(int k)	1757
29.359.3.9 operator()(int k) const	1757
29.359.3.10 print(std::ostream &os)	1757

29.359.3.1	printFull(std::ostream &os)	1757
29.359.3.1	prod(const PrVec &x, PrVec &y) const	1757
29.359.3.1	read(std::istream &is)	1758
29.359.3.1	redim(int m, int n, int num_nonzero)	1758
29.359.3.1	rows() const	1758
29.359.3.1	scalProd(double d)	1758
29.359.3.1	setToMatrix(const PrMatrix &m, double tol=0.0)	1758
29.360	PrNestedNode Class Reference	1758
29.360.1	Detailed Description	1759
29.360.2	Constructor & Destructor Documentation	1759
29.360.2.1	PrNestedNode()	1759
29.360.2.2	PrNestedNode(double x, double y, double z, double u, double v, int level)	1760
29.360.2.3	~PrNestedNode()	1760
29.360.3	Member Function Documentation	1760
29.360.3.1	addTrianglePtr(int t)	1760
29.360.3.2	nit(double x, double y, double z, double u, double v, int level)	1760
29.360.3.3	level() const	1760
29.360.3.4	level()	1760
29.360.3.5	pnt() const	1760
29.360.3.6	pnt()	1760
29.360.3.7	print(ostream &os)	1761
29.360.3.8	printUV(ostream &os)	1761
29.360.3.9	printXYZ(ostream &os)	1761
29.360.3.10	tt(int i) const	1761
29.360.3.11	tt(int i)	1761
29.360.3.12	tt() const	1761
29.360.3.13	tt()	1761
29.360.3.14	tt() const	1761
29.360.3.15	tt()	1761
29.360.3.16	tt() const	1761

29.360.3.17)	1761
29.360.3.18) const	1761
29.360.3.19)	1762
29.360.3.20) const	1762
29.360.3.21)	1762
29.361PrNestedTriangulation Class Reference	1762
29.361.1Detailed Description	1763
29.361.2Constructor & Destructor Documentation	1763
29.361.2.1~PrNestedTriangulation()	1763
29.361.3Member Function Documentation	1763
29.361.3.1getFinestLevel()=0	1763
29.361.3.2getNeighbours(int i, int jlev, vector< int > &neighbours)=0	1763
29.361.3.3getNumNodes(int jlev)=0	1764
29.361.3.4getParents(int i, int jlev, int &p1, int &p2)	1764
29.361.3.5getX(int i)=0	1764
29.361.3.6getY(int i)=0	1764
29.361.3.7getZ(int i)=0	1764
29.361.3.8isBoundary(int i)=0	1764
29.361.3.9setX(int i, const double &x)=0	1764
29.361.3.10setY(int i, const double &y)=0	1764
29.361.3.11setZ(int i, const double &z)=0	1765
29.362PrNode Class Reference	1765
29.362.1Detailed Description	1765
29.362.2Constructor & Destructor Documentation	1766
29.362.2.1PrNode()	1766
29.362.2.2PrNode(double x, double y, double z, double u, double v, int tr)	1766
29.362.3Member Function Documentation	1766
29.362.3.1init(double x, double y, double z, double u, double v, int tr)	1766
29.362.3.2point() const	1766
29.362.3.3print(std::ostream &os) const	1766

29.362.3.4	printUV(std::ostream &os) const	1766
29.362.3.5	printXYZ(std::ostream &os) const	1766
29.362.3.6	scan(std::istream &is)	1766
29.362.3.7	r() const	1766
29.362.3.8	r()	1766
29.362.3.9	u() const	1766
29.362.3.10	u()	1767
29.362.3.11	v() const	1767
29.362.3.12	v()	1767
29.362.3.13	v() const	1767
29.362.3.14	v()	1767
29.362.3.15	v() const	1767
29.362.3.16	v()	1767
29.362.3.17	w() const	1767
29.362.3.18	w()	1767
29.363	IEWMAT::ProgramException Class Reference	1768
29.363.1	Detailed Description	1769
29.363.2	Constructor & Destructor Documentation	1769
29.363.2.1	ProgramException()	1769
29.363.2.2	ProgramException(const char *c)	1769
29.363.2.3	ProgramException(const char *c, const GeneralMatrix &)	1769
29.363.2.4	ProgramException(const char *c, const GeneralMatrix &, const GeneralMatrix &)	1769
29.363.2.5	ProgramException(const char *c, MatrixType, MatrixType)	1769
29.363.3	Member Data Documentation	1769
29.363.3.1	Select	1769
29.364	Go::ProjectCurve Class Reference	1770
29.364.1	Detailed Description	1771
29.364.2	Constructor & Destructor Documentation	1771
29.364.2.1	ProjectCurve(shared_ptr< Go::ParamCurve > &space_crv, shared_ptr< Go::ParamSurface > &surf, shared_ptr< Go::Point > &start_par_pt, shared_ptr< Go::Point > &end_par_pt, double epsgeo1, const RectDomain *domain_of_interest=NULL)	1771

29.364.2.2~ProjectCurve()	1771
29.364.3 Member Function Documentation	1771
29.364.3.1approximationOK(double par, Go::Point approxpos, double tol1, double tol2) const	1771
29.364.3.2dim() const	1772
29.364.3.3end() const	1772
29.364.3.4eval(double t) const	1772
29.364.3.5eval(double t, Go::Point seed) const	1772
29.364.3.6eval(double t, int n, Go::Point der[]) const	1772
29.364.3.7start() const	1773
29.365 Go::ProjectCurveAndCrossTan Class Reference	1773
29.365.1 Detailed Description	1774
29.365.2 Constructor & Destructor Documentation	1774
29.365.2.1ProjectCurveAndCrossTan(const SplineCurve &space_crv, const SplineCurve &crosstan_crv, const SplineSurface &surf, const Point *start_par_pt, const Point *end_par_pt, double epsgeo, const RectDomain *domain_of_interest=NULL)	1774
29.365.2.2~ProjectCurveAndCrossTan()	1775
29.365.3 Member Function Documentation	1775
29.365.3.1approximationOK(double par, const std::vector< Go::Point > &approxpos, double tol1, double tol2)	1775
29.365.3.2dim()	1775
29.365.3.3end()	1776
29.365.3.4eval(double t)	1776
29.365.3.5eval(double t, int n, std::vector< std::vector< Go::Point > > &ders)	1776
29.365.3.6mbCvs()	1776
29.365.3.7start()	1777
29.366 Go::ProjectIntersectionCurve Class Reference	1777
29.366.1 Detailed Description	1778
29.366.2 Constructor & Destructor Documentation	1778
29.366.2.1ProjectIntersectionCurve(shared_ptr< SplineCurve > &inters_crv, shared_ptr< SplineCurve > &p_crv, shared_ptr< SplineCurve > &other_p_crv, shared_ptr< ParamSurface > &surf, shared_ptr< ParamSurface > &other_surf, double offset_dist, double other_offset_dist, double epsgeo)	1778
29.366.2.2~ProjectIntersectionCurve()	1778

29.366.3	Member Function Documentation	1778
29.366.3.1	approximationOK(double par, Point approxpos, double tol1, double tol2) const	1779
29.366.3.2	dim() const	1779
29.366.3.3	end() const	1779
29.366.3.4	eval(double t) const	1779
29.366.3.5	eval(double t, int n, Point der[]) const	1780
29.366.3.6	start() const	1780
29.367	PrOrganizedPoints Class Reference	1780
29.367.1	Detailed Description	1782
29.367.2	Constructor & Destructor Documentation	1782
29.367.2.1	~PrOrganizedPoints()	1782
29.367.3	Member Function Documentation	1782
29.367.3.1	findIndex(Vector3D &point) const	1782
29.367.3.2	findNumBdyComponents() const	1782
29.367.3.3	findNumBdyNodes() const	1782
29.367.3.4	findNumComponents() const	1782
29.367.3.5	findNumEdges() const	1782
29.367.3.6	get2Neighbours(int i, std::vector< int > &neighbours) const	1782
29.367.3.7	get3dNode(int i) const =0	1783
29.367.3.8	getCommonNeighbours(int j, int k, std::vector< int > &neighbours) const	1783
29.367.3.9	getNeighbours(int i, std::vector< int > &neighbours) const =0	1783
29.367.3.10	getNumNodes() const =0	1783
29.367.3.11	getU(int i) const =0	1783
29.367.3.12	getV(int i) const =0	1783
29.367.3.13	getIndexComponents(std::vector< int > &component, std::vector< int > &newIndex) const	1783
29.367.3.14	Boundary(int i) const =0	1784
29.367.3.15	Minimum(int i, std::vector< int > &face)	1784
29.367.3.16	labelBdyNode(int i, int ic, std::vector< int > &component) const	1784
29.367.3.17	labelNode(int i, int ic, std::vector< int > &component) const	1784

29.367.3.1	labelNode(int i, int ic, int &index, std::vector< int > &component, std::vector< int > &newIndex) const	1785
29.367.3.19	printInfo(std::ostream &os) const	1785
29.367.3.20	printUVEdges(std::ostream &os) const	1785
29.367.3.21	printUVNodes(std::ostream &os, bool num=0) const	1785
29.367.3.22	printUVXYZNodes(std::ostream &os, bool num=0) const	1785
29.367.3.23	printXYZEdges(std::ostream &os) const	1786
29.367.3.24	printXYZNodes(std::ostream &os, bool num=0) const	1786
29.367.3.25	set3dNode(int i, const Vector3D &p)=0	1786
29.367.3.26	setU(int i, double u)=0	1786
29.367.3.27	setV(int i, double v)=0	1786
29.367.3.28	topologicalDistToBdy(std::vector< int > &label) const	1786
29.368	PrParametrizeBdy Class Reference	1786
29.368.1	Detailed Description	1788
29.368.2	Constructor & Destructor Documentation	1788
29.368.2.1	PrParametrizeBdy()	1788
29.368.2.2	~PrParametrizeBdy()	1788
29.368.3	Member Function Documentation	1788
29.368.3.1	attach(shared_ptr< PrOrganizedPoints > graph)	1788
29.368.3.2	boundaryLength(int i1, int i2)	1788
29.368.3.3	chord(const Vector3D &a, const Vector3D &b)	1788
29.368.3.4	findBdyNode()	1788
29.368.3.5	findCornersFromUV(int *c)	1788
29.368.3.6	findCornersFromXYZ(int *c)	1789
29.368.3.7	getNextBdyNode(int i)	1789
29.368.3.8	parametrize()	1789
29.368.3.9	parametrize(int c1, int c2, int c3, int c4, double umin=0.0, double umax=1.0, double vmin=0.0, double vmax=1.0)	1789
29.368.3.10	parametrizeSide(int i1, int i2)	1789
29.368.3.11	setParamKind(PrBdyParamKind bdyparamtype=PrCHORDLENGTHBDY)	1789
29.368.4	Member Data Documentation	1789

29.368.4.1	bdyparamtype_	1789
29.368.4.2	g_	1789
29.368.4.3	neighbours_	1790
29.369	PrParametrizeInt Class Reference	1790
29.369.1	Detailed Description	1792
29.369.2	Constructor & Destructor Documentation	1792
29.369.2.1	PrParametrizeInt()	1792
29.369.2.2	~PrParametrizeInt()	1792
29.369.3	Member Function Documentation	1792
29.369.3.1	attach(shared_ptr< PrOrganizedPoints > graph)	1792
29.369.3.2	computeWeights()	1792
29.369.3.3	findBarycentre(double &ucentre, double &vcentre)	1792
29.369.3.4	findFixedPntsFromXYZ(vector< int > &fixedPnts)	1792
29.369.3.5	getAllNeighbours() const	1793
29.369.3.6	getAllWeights() const	1793
29.369.3.7	getNumInt2Nghrs(int i, vector< int > &)	1793
29.369.3.8	getNumIntNghrs(int i)	1793
29.369.3.9	isFixed(int, vector< int > &)	1793
29.369.3.10	makeWeights(int i)=0	1793
29.369.3.11	new_parametrize3d(vector< int > &, vector< double > &)	1793
29.369.3.12	parametrize()	1793
29.369.3.13	parametrize3d(vector< int > &, vector< double > &)	1793
29.369.3.14	setBiCGTolerance(double tolerance=1.0e-6)	1793
29.369.3.15	setStartVectorKind(PrParamStartVector svttype=PrBARYCENTRE)	1794
29.369.3.16	smooth(int nmb, vector< int > &fixedPnts)	1794
29.369.4	Member Data Documentation	1794
29.369.4.1	allNeighbours_	1794
29.369.4.2	allWeights_	1794
29.369.4.3	g_	1794
29.369.4.4	neighbours_	1794

29.369.4.5	startvectortype_	1794
29.369.4.6	tolerance_	1794
29.369.4.7	weights_	1794
29.370	PrParametrizeMesh Class Reference	1795
29.370.1	Detailed Description	1795
29.370.2	Constructor & Destructor Documentation	1795
29.370.2.1	PrParametrizeMesh()	1795
29.370.2.2	~PrParametrizeMesh()	1796
29.370.3	Member Function Documentation	1796
29.370.3.1	attach(PrTriangulation_OP *mesh, PrTriangulation_OP *basemesh)	1796
29.370.3.2	castRay(PrTriangulation_OP &triangulation, int vs, double angle, double dMax, const std::set< int > &T, std::vector< int > &t_path, std::vector< Vector3D > &v_path)	1796
29.370.3.3	convertAngle(const PrTriangulation_OP &triangulation, double theta, int v, double &bc, int &t)	1796
29.370.3.4	edgeInPolygon(int n1, int n2, const std::vector< int > &polygon)	1796
29.370.3.5	getConnectedNodes(PrTriangulation_OP &triangulation, const std::vector< int > &polygon, std::set< int > &nodes)	1796
29.370.3.6	getInteriorNeighbours(int v, const std::vector< int > &neighbours, const std::vector< int > &boundary, std::vector< int > &new_nbrs)	1796
29.370.3.7	getIsoline(const PrTriangulation_OP &triangulation, int vs, int vd, std::vector< int > &t_path, std::vector< Vector3D > &v_path)	1796
29.370.3.8	getLeftNode(PrTriangulation_OP &triangulation, int n1, int n2)	1797
29.370.3.9	getNextV(const Vector3D &p1, const Vector3D &p2, const Vector3D &p3, const Vector3D &p4, double a, double b, bool f, double &c)	1797
29.370.3.10	makePath(PrTriangulation_OP &triangulation, int n1, int n2, std::vector< int > &path)	1797
29.370.3.11	makePolygon(PrTriangulation_OP &triangulation, const std::vector< int > &nodes, std::vector< int > &polygon)	1797
29.370.3.12	makeSubTriangulation(PrTriangulation_OP &triangulation, const vector< int > &nodes, PrSubTriangulation &subtriangulation)	1797
29.370.3.13	makeSubTriangulationFromTriangle(int i, PrSubTriangulation &subtri, std::vector< int > &corners)	1797
29.370.3.14	makeSubTriangulationInsidePolygon(PrTriangulation_OP &triangulation, const std::vector< int > &polygon, PrSubTriangulation &subtriangulation)	1797
29.370.3.15	parametrize()	1797

29.370.3.1	parametrizeSubTriangulation(shared_ptr< PrSubTriangulation > sub_tri, std::vector< int > &corners)	1797
29.371	PrParamTriangulation Class Reference	1798
29.371.1	Detailed Description	1798
29.371.2	Constructor & Destructor Documentation	1798
29.371.2.1	PrParamTriangulation()	1798
29.371.2.2	~PrParamTriangulation()	1799
29.371.3	Member Function Documentation	1799
29.371.3.1	attach(PrTriangulation_OP *mesh, PrTriangulation_OP *basemesh)	1799
29.371.3.2	evaluator(PrTriangulation_OP &mesh, int idx, Vector3D &bc)	1799
29.371.3.3	findTriangleContainingPoint(int coarseT, const Vector3D &coarseBC, int &fineT, Vector3D &fineBC)	1799
29.371.3.4	getCoarseMesh()	1799
29.371.3.5	getFineMesh()	1799
29.371.3.6	getParamPoint(int i)	1800
29.371.3.7	getSurfPoint(int coarseTri, const Vector3D &coarseBC)	1800
29.371.3.8	getUV(int node, int tri, double &u, double &v)	1800
29.371.3.9	makeCorrespondences(int idx, PrSubTriangulation &sub_tri)	1801
29.371.3.10	open(istream &is)	1801
29.371.3.11	printBaseTriangles()	1801
29.371.3.12	save(ostream &os)	1801
29.371.3.13	triangleContainsPoint(int fineTri, int coarseTri, const Vector3D &coarseBC, Vector3D &fineBC)	1801
29.372	PrPGNode Class Reference	1802
29.372.1	Detailed Description	1802
29.372.2	Constructor & Destructor Documentation	1802
29.372.2.1	PrPGNode()	1802
29.372.2.2	PrPGNode(double x, double y, double z, double u, double v, int end)	1803
29.372.2.3	PrPGNode(const PrPGNode &p)	1803
29.372.2.4	~PrPGNode()	1803
29.372.3	Member Function Documentation	1803
29.372.3.1	end() const	1803

29.372.3.2end()	1803
29.372.3.3init(double x, double y, double z, double u, double v, int end)	1803
29.372.3.4pnt() const	1803
29.372.3.5pnt()	1803
29.372.3.6print(std::ostream &os)	1804
29.372.3.7printUV(std::ostream &os)	1804
29.372.3.8printXYZ(std::ostream &os)	1804
29.372.3.9scan(std::istream &is)	1804
29.372.3.10() const	1804
29.372.3.11d()	1804
29.372.3.12() const	1804
29.372.3.13()	1804
29.372.3.14() const	1804
29.372.3.15()	1804
29.372.3.16() const	1804
29.372.3.17()	1804
29.372.3.18() const	1805
29.372.3.19()	1805
29.373PrPlanarGraph_OP Class Reference	1805
29.373.1Detailed Description	1806
29.373.2Constructor & Destructor Documentation	1806
29.373.2.1PrPlanarGraph_OP()	1806
29.373.2.2PrPlanarGraph_OP(int npts, double *xyz_nodes, double *uv_nodes, int *end, int *adj)	1807
29.373.2.3PrPlanarGraph_OP(int npts, double *xyz_nodes, int *end, int *adj)	1807
29.373.2.4PrPlanarGraph_OP(PrOrganizedPoints &op)	1807
29.373.2.5~PrPlanarGraph_OP()	1807
29.373.3Member Function Documentation	1807
29.373.3.1get3dNode(int i) const	1807
29.373.3.2getNeighbours(int i, vector< int > &neighbours) const	1807
29.373.3.3getNumNodes() const	1808

29.373.3.4	getU(int i) const	1808
29.373.3.5	getV(int i) const	1808
29.373.3.6	isBoundary(int i) const	1808
29.373.3.7	print(std::ostream &os)	1808
29.373.3.8	scan(std::istream &is)	1808
29.373.3.9	scan2(std::istream &is)	1808
29.373.3.10	set3dNode(int i, const Vector3D &p)	1808
29.373.3.11	setU(int i, double u)	1809
29.373.3.12	setV(int i, double v)	1809
29.374	PrPrewavelet_F Class Reference	1809
29.374.1	Detailed Description	1810
29.374.2	Constructor & Destructor Documentation	1810
29.374.2.1	PrPrewavelet_F()	1810
29.374.2.2	~PrPrewavelet_F()	1811
29.374.3	Member Function Documentation	1811
29.374.3.1	compose(int jlev, int dim=1)	1811
29.374.3.2	decompose(int jlev, int dim=1)	1811
29.374.3.3	setCGTolerance(double tolerance=1.0e-6)	1811
29.375	PrPrmEDDHLS Class Reference	1811
29.375.1	Detailed Description	1812
29.375.2	Constructor & Destructor Documentation	1812
29.375.2.1	PrPrmEDDHLS()	1812
29.375.2.2	~PrPrmEDDHLS()	1812
29.375.3	Member Function Documentation	1813
29.375.3.1	makeWeights(int i)	1813
29.376	PrPrmExperimental Class Reference	1813
29.376.1	Detailed Description	1814
29.376.2	Constructor & Destructor Documentation	1814
29.376.2.1	PrPrmExperimental()	1814
29.376.2.2	~PrPrmExperimental()	1814

29.376.3	Member Function Documentation	1814
29.376.3.1	makeWeights(int i)	1814
29.377	PrPmLeastSquare Class Reference	1815
29.377.1	Detailed Description	1816
29.377.2	Constructor & Destructor Documentation	1816
29.377.2.1	PrPmLeastSquare()	1816
29.377.2.2	~PrPmLeastSquare()	1816
29.377.3	Member Function Documentation	1816
29.377.3.1	makeWeights(int i)	1816
29.378	PrPmMeanValue Class Reference	1816
29.378.1	Detailed Description	1817
29.378.2	Constructor & Destructor Documentation	1817
29.378.2.1	PrPmMeanValue()	1817
29.378.2.2	~PrPmMeanValue()	1817
29.378.3	Member Function Documentation	1818
29.378.3.1	makeWeights(int i)	1818
29.379	PrPmShpPres Class Reference	1818
29.379.1	Detailed Description	1819
29.379.2	Constructor & Destructor Documentation	1819
29.379.2.1	PrPmShpPres()	1819
29.379.2.2	~PrPmShpPres()	1819
29.379.3	Member Function Documentation	1819
29.379.3.1	localParam(int i)	1819
29.379.3.2	makeWeights(int i)	1819
29.379.4	Member Data Documentation	1819
29.379.4.1	alpha_	1819
29.379.4.2	en_	1820
29.379.4.3	u_	1820
29.379.4.4	v_	1820
29.380	PrPmSurface Class Reference	1820

29.380.1	Detailed Description	1821
29.380.2	Constructor & Destructor Documentation	1821
29.380.2.1	PrPrmSurface()	1821
29.380.2.2	~PrPrmSurface()	1821
29.380	PrPrmSymMeanValue Class Reference	1821
29.381.1	Detailed Description	1822
29.381.2	Constructor & Destructor Documentation	1822
29.381.2.1	PrPrmSymMeanValue()	1822
29.381.2.2	~PrPrmSymMeanValue()	1822
29.381.3	Member Function Documentation	1823
29.381.3.1	makeWeights(int i)	1823
29.381.3.2	atanThetaOverTwo(Vector3D &a, Vector3D &b, Vector3D &c)	1823
29.380	PrPrmUniform Class Reference	1823
29.382.1	Detailed Description	1824
29.382.2	Constructor & Destructor Documentation	1824
29.382.2.1	PrPrmUniform()	1824
29.382.2.2	~PrPrmUniform()	1824
29.380	PrPrmWachspress Class Reference	1824
29.383.1	Detailed Description	1825
29.383.2	Constructor & Destructor Documentation	1825
29.383.2.1	PrPrmWachspress()	1825
29.383.2.2	~PrPrmWachspress()	1825
29.383.3	Member Function Documentation	1826
29.383.3.1	makeWeights(int i)	1826
29.383.3.2	atanThetaOverTwo(Vector3D &a, Vector3D &b, Vector3D &c)	1826
29.380	PrRectangularGrid_OP Class Reference	1826
29.384.1	Detailed Description	1828
29.384.2	Constructor & Destructor Documentation	1828
29.384.2.1	PrRectangularGrid_OP()	1828
29.384.2.2	PrRectangularGrid_OP(int numcols, int numrows, const double *grid)	1829

29.384.2.3	PrRectangularGrid_OP(int numcols, int numRows, const double *grid, const double *uvgrid)	1829
29.384.2.4	~PrRectangularGrid_OP()	1829
29.384.3	Member Function Documentation	1829
29.384.3.1	get3dNode(int i) const	1829
29.384.3.2	getCorners(int &c1, int &c2, int &c3, int &c4) const	1829
29.384.3.3	getDim(int &numcols, int &numrows) const	1829
29.384.3.4	getNeighbours(int i, vector< int > &neighbours) const	1829
29.384.3.5	getNumNodes() const	1829
29.384.3.6	getTriangle(double u, double v, int ii, int jj, bool &right, double &tau0, double &tau1, double &tau2) const	1829
29.384.3.7	getU(int i) const	1830
29.384.3.8	getU(int i, int j) const	1830
29.384.3.9	getUVTriangle(double u, double v, int &i, int &j, bool &right, double &tau0, double &tau1, double &tau2) const	1830
29.384.3.10	getUVTriangle(double &u, double &v, int ilast, int jlast, int &i, int &j, bool &right, double &tau0, double &tau1, double &tau2) const	1831
29.384.3.11	getV(int i) const	1831
29.384.3.12	getV(int i, int j) const	1831
29.384.3.13	graphToGrid(int index, int &col, int &row) const	1831
29.384.3.14	gridToGraph(int col, int row) const	1831
29.384.3.15	isBoundary(int i) const	1832
29.384.3.16	print(std::ostream &os)	1832
29.384.3.17	printRawData(std::ostream &os)	1832
29.384.3.18	scan(std::istream &is)	1832
29.384.3.19	scanRawData(std::istream &is)	1832
29.384.3.20	set3dNode(int i, const Vector3D &p)	1832
29.384.3.21	setDim(int numcols, int numRows)	1832
29.384.3.22	setU(int i, double u)	1832
29.384.3.23	setUVVertices(const double *grid)	1833
29.384.3.24	setV(int i, double v)	1833
29.384.3.25	setXYZVertices(const double *grid)	1833

29.384.3.20vData()	1833
29.384.3.21vData() const	1833
29.384.3.22yzData()	1833
29.384.3.23yzData() const	1833
29.385PrSubTriangulation Class Reference	1834
29.385.1Detailed Description	1835
29.385.2Constructor & Destructor Documentation	1836
29.385.2.1PrSubTriangulation()	1836
29.385.2.2PrSubTriangulation(PrTriangulation_OP *graph, vector< int > boundary, vector< int > interior)	1836
29.385.2.3~PrSubTriangulation()	1836
29.385.3Member Function Documentation	1836
29.385.3.1attach(PrTriangulation_OP *graph)	1836
29.385.3.2findNumBdyNodes() const	1836
29.385.3.3get3dNode(int i) const	1836
29.385.3.4getGlobalIndex(int i) const	1837
29.385.3.5getLocalIndex(int i) const	1837
29.385.3.6getNeighbours(int i, vector< int > &neighbours) const	1837
29.385.3.7getNeighbourTriangles(int i, vector< int > &neighbours) const	1837
29.385.3.8getNumNodes() const	1837
29.385.3.9getTriangleIndices(vector< int > &indices) const	1837
29.385.3.10getU(int i) const	1837
29.385.3.11getV(int i) const	1837
29.385.3.12initialize(vector< int > boundary, vector< int > interior)	1838
29.385.3.13Boundary(int i) const	1838
29.385.3.14set3dNode(int i, const Vector3D &p)	1838
29.385.3.15setU(int i, double u)	1838
29.385.3.16setV(int i, double v)	1838
29.385.3.17triangleInThis(int i) const	1838
29.386PrThin Class Reference	1839
29.386.1Detailed Description	1840

29.386.2	Constructor & Destructor Documentation	1840
29.386.2.1	PrThin()	1840
29.386.2.2	~PrThin()	1840
29.386.3	Member Function Documentation	1840
29.386.3.1	attach(PrTriangulation_OP *trn)	1840
29.386.3.2	findError(int i)	1840
29.386.3.3	findNearestNeighbour(int i, int &tr)	1840
29.386.3.4	halfEdgeCollapse(int i)	1841
29.386.3.5	makeHeap()	1841
29.386.3.6	printHeap()	1841
29.386.3.7	removePoint(int i)	1841
29.386.3.8	setError(double error)	1841
29.386.3.9	setStepNumber(int n)	1841
29.386.3.10	in()	1841
29.386.4	Member Data Documentation	1841
29.386.4.1	error_	1841
29.386.4.2	heap_	1841
29.386.4.3	list_ptr	1841
29.386.4.4	steps_	1842
29.386.4.5	list_ptr	1842
29.386.4.6	rn_	1842
29.387	PrThreshold Class Reference	1842
29.387.1	Detailed Description	1843
29.387.2	Constructor & Destructor Documentation	1843
29.387.2.1	PrThreshold()	1843
29.387.2.2	~PrThreshold()	1843
29.387.3	Member Function Documentation	1843
29.387.3.1	attach(shared_ptr< PrNestedTriangulation > t)	1843
29.387.3.2	averageAbsValue()	1843
29.387.3.3	findThreshold(double &wavelet_comp_rate)	1844

29.387.3.4	leaveLevel(int jlev)	1844
29.387.3.5	maxNorm()	1844
29.387.3.6	setThreshold(double threshold=0.0)	1844
29.387.3.7	threshold(double &comp_rate, double &wavelet_comp_rate, int error_out=0)	1844
29.387.3.8	thresholdByCompRate(double &wavelet_comp_rate, int error_out=0)	1844
29.387.3.9	triangleNorm(int i, int j, int k)	1844
29.387.3.10	truncateLevel(int jlev)	1844
29.387.3.11	weightedL2Norm()	1844
29.387.4	Member Data Documentation	1844
29.387.4.1	ft_	1844
29.387.4.2	threshold_	1845
29.388	PrTriangle Class Reference	1845
29.388.1	Detailed Description	1846
29.388.2	Constructor & Destructor Documentation	1846
29.388.2.1	PrTriangle()	1846
29.388.2.2	PrTriangle(int n1, int n2, int n3, int t1, int t2, int t3)	1846
29.388.2.3	PrTriangle(const PrTriangle &t)	1846
29.388.2.4	~PrTriangle()	1846
29.388.3	Member Function Documentation	1846
29.388.3.1	getAnticlockwiseNode(int node) const	1846
29.388.3.2	getClockwiseNode(int node) const	1846
29.388.3.3	getEdge(int triangle, int &n1, int &n2) const	1847
29.388.3.4	getLeftTriangle(int node) const	1847
29.388.3.5	getOppositeTriangle(int node) const	1847
29.388.3.6	getRightTriangle(int node) const	1847
29.388.3.7	nit(int n1, int n2, int n3, int t1, int t2, int t3)	1847
29.388.3.8	sVertex(int node) const	1847
29.388.3.9	n1() const	1847
29.388.3.10	n2() const	1847
29.388.3.11	n3() const	1847
29.388.3.12	t1() const	1847
29.388.3.13	t2() const	1847
29.388.3.14	t3() const	1847

29.388.3.122()	1847
29.388.3.123() const	1848
29.388.3.124()	1848
29.388.3.125rint(std::ostream &os) const	1848
29.388.3.126replaceNode(int n1, int n2)	1848
29.388.3.127replaceTriangle(int t1, int t2)	1848
29.388.3.128scan(std::istream &is)	1848
29.388.3.129() const	1848
29.388.3.130()	1848
29.388.3.131() const	1848
29.388.3.132()	1848
29.388.3.133() const	1849
29.388.3.134()	1849
29.389 PrTriangulation_NT Class Reference	1849
29.389.1 Detailed Description	1850
29.389.2 Constructor & Destructor Documentation	1850
29.389.2.1 PrTriangulation_NT()	1850
29.389.2.2 PrTriangulation_NT(PrTriangulation_OP &t)	1851
29.389.2.3 ~PrTriangulation_NT()	1851
29.389.3 Member Function Documentation	1851
29.389.3.1 getFinestLevel()	1851
29.389.3.2 getLevel(int i)	1851
29.389.3.3 getNeighbours(int i, int jlev, vector< int > &neighbours)	1851
29.389.3.4 getNumNodes(int jlev)	1851
29.389.3.5 getTopLevel()	1852
29.389.3.6 getX(int i)	1852
29.389.3.7 getY(int i)	1852
29.389.3.8 getZ(int i)	1852
29.389.3.9 isBoundary(int i)	1852
29.389.3.10 t(PrParamTriangulation *pt)	1852

29.389.3.1	<code>refine()</code>	1852
29.389.3.1	<code>setX(int i, const double &x)</code>	1853
29.389.3.1	<code>setY(int i, const double &y)</code>	1853
29.389.3.1	<code>setZ(int i, const double &z)</code>	1853
29.390	<code>PrTriangulation_OP</code> Class Reference	1853
29.390.1	Detailed Description	1855
29.390.2	Constructor & Destructor Documentation	1855
29.390.2.1	<code>PrTriangulation_OP()</code>	1855
29.390.2.2	<code>PrTriangulation_OP(const double *xyz_points, int np, const int *triangles, int nt)</code>	1856
29.390.2.3	<code>PrTriangulation_OP(const double *xyz_points, const double *uv_points, int np, const int *triangles, int nt)</code>	1856
29.390.2.4	<code>PrTriangulation_OP(PrExplicitConnectivity &op)</code>	1856
29.390.2.5	<code>~PrTriangulation_OP()</code>	1856
29.390.3	Member Function Documentation	1856
29.390.3.1	<code>findNumFaces() const</code>	1856
29.390.3.2	<code>get3dNode(int i) const</code>	1856
29.390.3.3	<code>getNeighbours(int i, std::vector< int > &neighbours) const</code>	1857
29.390.3.4	<code>getNodeArray()</code>	1857
29.390.3.5	<code>getNumNodes() const</code>	1857
29.390.3.6	<code>getPrNode(int i)</code>	1857
29.390.3.7	<code>getPrNode(int i) const</code>	1857
29.390.3.8	<code>getPrTriangle(int i) const</code>	1857
29.390.3.9	<code>getPrTriangle(int i)</code>	1857
29.390.3.10	<code>getTriangleArray()</code>	1858
29.390.3.11	<code>getTriangles(int i, Go::ScratchVect< int, 20 > &triangles) const</code>	1858
29.390.3.12	<code>getU(int i) const</code>	1858
29.390.3.13	<code>getV(int i) const</code>	1858
29.390.3.14	<code>isBoundary(int i) const</code>	1858
29.390.3.15	<code>print(std::ostream &os) const</code>	1858
29.390.3.16	<code>printRawData(std::ostream &os) const</code>	1858
29.390.3.17	<code>printUV(std::ostream &os) const</code>	1858

29.390.3.19	PrintUVTriangles(std::ostream &os, bool num=false) const	1859
29.390.3.19	PrintXYZTriangles(std::ostream &os, bool num=false) const	1859
29.390.3.20	can(std::istream &is)	1859
29.390.3.21	canRawData(std::istream &is)	1859
29.390.3.22	set3dNode(int i, const Vector3D &p)	1859
29.390.3.23	setU(int i, double u)	1859
29.390.3.24	setV(int i, double v)	1859
29.390.3.25	splitTriangles(int t1, int t2, Vector3D &v)	1859
29.390.3.26	splitVertex(int i, int j, std::vector< int > &new_nodes)	1860
29.390.3.27	swapNodes(int n1, int n2)	1860
29.390.3.28	swapTriangles(int t1, int t2)	1860
29.391	PrUnorganized_OP Class Reference	1860
29.391.1	Detailed Description	1861
29.391.2	Constructor & Destructor Documentation	1862
29.391.2.1	PrUnorganized_OP(int num_cells=10)	1862
29.391.2.2	PrUnorganized_OP(int n, int n_int, double *xyz_points, int num_cells=10)	1862
29.391.2.3	~PrUnorganized_OP()	1862
29.391.3	Member Function Documentation	1862
29.391.3.1	get3dNode(int i)	1862
29.391.3.2	getNeighbours(int i, vector< int > &neighbours)	1862
29.391.3.3	getNumNodes()	1862
29.391.3.4	getRadius()	1862
29.391.3.5	getU(int i)	1862
29.391.3.6	getV(int i)	1863
29.391.3.7	isBoundary(int i)	1863
29.391.3.8	print(std::ostream &os)	1863
29.391.3.9	printRawData(std::ostream &os)	1863
29.391.3.10	can(std::istream &is)	1863
29.391.3.11	canRawData(std::istream &is, int num_cells, double noise)	1863
29.391.3.12	set3dNode(int i, const Vector3D &p)	1863

29.391.3.1	GetKNearest(int knearest)	1863
29.391.3.1	GetRadius(double radius)	1864
29.391.3.1	SetU(int i, double u)	1864
29.391.3.1	SetV(int i, double v)	1864
29.391.3.1	UseK()	1864
29.391.3.1	UseRadius()	1864
29.392	PrVec Class Reference	1864
29.392.1	Detailed Description	1865
29.392.2	Constructor & Destructor Documentation	1865
29.392.2.1	PrVec()	1865
29.392.2.2	PrVec(int n, double fill_with=0.0)	1865
29.392.2.3	PrVec(InputIterator begin, InputIterator end)	1866
29.392.3	Member Function Documentation	1866
29.392.3.1	inner(const PrVec &x)	1866
29.392.3.2	operator()(int i)	1866
29.392.3.3	operator()(int i) const	1866
29.392.3.4	operator[](int i)	1866
29.392.3.5	operator[](int i) const	1866
29.392.3.6	print(std::ostream &os)	1866
29.392.3.7	read(std::istream &is)	1866
29.392.3.8	redim(int n, double fill_with=0.0)	1867
29.392.3.9	size() const	1867
29.392.4	Member Data Documentation	1867
29.392.4.1	a_	1867
29.393	Geo::PtPtIntersector Class Reference	1867
29.393.1	Detailed Description	1868
29.393.2	Constructor & Destructor Documentation	1868
29.393.2.1	PtPtIntersector(shared_ptr< ParamGeomInt > point1, shared_ptr< ParamGeomInt > point2, shared_ptr< GeoTol > epsge, Intersector *prev=0, int eliminated_parameter=-1, double eliminated_value=0)	1868

29.393.2.2	PtPtIntersector(shared_ptr< ParamGeomInt > point1, shared_ptr< ParamGeomInt > point2, double epsge, Intersector *prev=0, int eliminated_parameter=-1, double eliminated_value=0)	1869
29.393.2.3	~PtPtIntersector()	1869
29.393.3	Member Function Documentation	1869
29.393.3.1	checkCoincidence()	1869
29.393.3.2	doSubdivide()	1869
29.393.3.3	linearCase()	1869
29.393.3.4	lowerOrderIntersector(shared_ptr< ParamGeomInt > obj1, shared_ptr< ParamGeomInt > obj2, Intersector *prev=0, int eliminated_parameter=-1, double eliminated_value=0)	1870
29.393.3.5	microCase()	1870
29.393.3.6	numParams() const	1870
29.393.3.7	repairIntersections()	1870
29.393.3.8	updateIntersections()	1870
29.394	Go::QuadMesh Class Reference	1871
29.394.1	Detailed Description	1872
29.394.2	Constructor & Destructor Documentation	1872
29.394.2.1	QuadMesh(int num_vert=0, int num_quads=0, bool use_normals=true, bool use_texcoords=false)	1872
29.394.3	Member Function Documentation	1872
29.394.3.1	atBoundary(int idx)	1872
29.394.3.2	normalArray()	1872
29.394.3.3	numQuads()	1873
29.394.3.4	numTriangles()	1873
29.394.3.5	numVertices()	1873
29.394.3.6	paramArray()	1873
29.394.3.7	quadIndexArray()	1873
29.394.3.8	resize(int num_vert, int num_quads)	1873
29.394.3.9	texcoordArray()	1873
29.394.3.10	triangleIndexArray()	1874
29.394.3.11	useNormals()	1874
29.394.3.12	useTexCoords()	1874

29.394.3.1	VertexArray()	1874
29.395	Go::QualityResults Class Reference	1874
29.395.1	Detailed Description	1875
29.395.2	Constructor & Destructor Documentation	1875
29.395.2.1	~QualityResults()	1875
29.395.3	Friends And Related Function Documentation	1875
29.395.3.1	FaceSetQuality	1875
29.395.3.2	FaceSetRepair	1875
29.395.3.3	ModelQuality	1875
29.395.3.4	ModelRepair	1875
29.396	NEWMAT::R1_Col_I_D Class Reference	1876
29.396.1	Detailed Description	1877
29.396.2	Constructor & Destructor Documentation	1877
29.396.2.1	~R1_Col_I_D()	1877
29.396.3	Member Function Documentation	1877
29.396.3.1	Derivatives()=0	1877
29.396.3.2	IsValid()	1877
29.396.3.3	IsValid(const ColumnVector &X)	1877
29.396.3.4	operator()(int i)=0	1877
29.396.3.5	operator()(int i, const ColumnVector &X)	1877
29.396.3.6	Set(const ColumnVector &X)	1877
29.396.4	Member Data Documentation	1877
29.396.4.1	para	1877
29.397	RBD_COMMON::R1_R1 Class Reference	1878
29.397.1	Detailed Description	1878
29.397.2	Constructor & Destructor Documentation	1879
29.397.2.1	R1_R1()	1879
29.397.2.2	~R1_R1()	1879
29.397.3	Member Function Documentation	1879
29.397.3.1	IsValid(Real X)	1879

29.397.3.2operator Real()	1879
29.397.3.3operator>()=0	1879
29.397.3.4operator()(Real X)	1879
29.397.3.5Set(Real X)	1879
29.397.4Member Data Documentation	1879
29.397.4.1maxX	1879
29.397.4.2maxXinf	1879
29.397.4.3minX	1880
29.397.4.4minXinf	1880
29.397.4.5x	1880
29.397.4.6xSet	1880
29.398RBD_COMMON::Range_error Class Reference	1880
29.398.1Detailed Description	1881
29.398.2Constructor & Destructor Documentation	1881
29.398.2.1Range_error(const char *a_what=0)	1881
29.398.3Member Data Documentation	1881
29.398.3.1Select	1881
29.399ank_info Struct Reference	1882
29.399.1Detailed Description	1882
29.399.2Member Data Documentation	1882
29.399.2.1antgr	1882
29.399.2.2antrem	1882
29.399.2.3groups	1882
29.399.2.4prio	1882
29.400Go::Rational Class Reference	1882
29.400.1Detailed Description	1883
29.400.2Constructor & Destructor Documentation	1883
29.400.2.1Rational()	1883
29.400.2.2Rational(int p)	1883
29.400.2.3Rational(int p, int q)	1884

29.400.3	Member Function Documentation	1884
29.400.3.1	operator!=(const Rational r)	1884
29.400.3.2	operator*=(const Rational &other)	1884
29.400.3.3	operator+=(const Rational &other)	1884
29.400.3.4	operator-() const	1884
29.400.3.5	operator-=(const Rational &other)	1884
29.400.3.6	operator/=(const Rational &other)	1884
29.400.3.7	operator==(const Rational r)	1885
29.400.3.8	simplify()	1885
29.400.3.9	write(std::ostream &os) const	1885
29.400	Go::raw_pointer_comp< T > Struct Template Reference	1885
29.401.1	Detailed Description	1885
29.401.2	Member Function Documentation	1885
29.401.2.1	operator()(shared_ptr< T > A, shared_ptr< T > B)	1885
29.402	RectangularSurfacePropertySheet Class Reference	1886
29.402.1	Detailed Description	1887
29.402.2	Constructor & Destructor Documentation	1887
29.402.2.1	RectangularSurfacePropertySheet(Go::RectangularSurfaceTesselator *tess, gvRectangularSurfacePaintable *pable, shared_ptr< Go::ParamSurface > &surf)	1887
29.402.2.2	~RectangularSurfacePropertySheet()	1887
29.402.3	Member Function Documentation	1887
29.402.3.1	apply	1887
29.402.3.2	createSheet(QWidget *parent, gvObserver *obs)	1887
29.403	Go::RectangularSurfaceTesselator Class Reference	1887
29.403.1	Detailed Description	1888
29.403.2	Constructor & Destructor Documentation	1889
29.403.2.1	RectangularSurfaceTesselator(const ParamSurface &surf, int ures=20, int vres=20, bool iso=false, int uiso=15, int viso=15, int isoires=300)	1889
29.403.2.2	~RectangularSurfaceTesselator()	1889
29.403.3	Member Function Documentation	1889
29.403.3.1	changeIsoLineNumRes(int m, int n, int res)	1889

29.403.3.2	changeIsolines(bool isolines)	1889
29.403.3.3	changeRes(int m, int n)	1889
29.403.3.4	getIsolineNumRes(int &m, int &n, int &res)	1889
29.403.3.5	getIsolines(bool &isolines)	1889
29.403.3.6	getIsolineStrips()	1890
29.403.3.7	getMesh()	1890
29.403.3.8	getRes(int &m, int &n)	1890
29.403.3.9	tesselate()	1890
29.404	RectangularVolumePropertySheet Class Reference	1890
29.404.1	Detailed Description	1891
29.404.2	Constructor & Destructor Documentation	1891
29.404.2.1	RectangularVolumePropertySheet(Go::RectangularVolumeTesselator *tess, gvRectangularVolumePaintable *pable, shared_ptr< Go::ParamVolume > &vol)	1891
29.404.2.2	~RectangularVolumePropertySheet()	1891
29.404.3	Member Function Documentation	1891
29.404.3.1	apply	1891
29.404.3.2	createSheet(QWidget *parent, gvObserver *obs)	1891
29.405	Go::RectangularVolumeTesselator Class Reference	1892
29.405.1	Detailed Description	1893
29.405.2	Constructor & Destructor Documentation	1893
29.405.2.1	RectangularVolumeTesselator(const ParamVolume &vol, int res=50)	1893
29.405.2.2	~RectangularVolumeTesselator()	1893
29.405.3	Member Function Documentation	1893
29.405.3.1	changeRes(int m)	1893
29.405.3.2	getMesh()	1893
29.405.3.3	getRes(int &m)	1893
29.405.3.4	tesselate()	1893
29.406	Go::RectDomain Class Reference	1894
29.406.1	Detailed Description	1895
29.406.2	Constructor & Destructor Documentation	1895
29.406.2.1	RectDomain()	1895

29.406.2.2	RectDomain(const Array< double, 2 > &corner1, const Array< double, 2 > &corner2)	1895
29.406.2.3	~RectDomain()	1895
29.406.3	Member Function Documentation	1895
29.406.3.1	addUnionWith(const RectDomain &rd)	1895
29.406.3.2	closestInDomain(const Array< double, 2 > &point, Array< double, 2 > &clo_pt, double tolerance) const	1896
29.406.3.3	closestOnBoundary(const Array< double, 2 > &point, Array< double, 2 > &clo_pt, double tolerance) const	1896
29.406.3.4	diagLength()	1896
29.406.3.5	intersectWith(const RectDomain &rd)	1896
29.406.3.6	isInDomain(const Array< double, 2 > &point, double tolerance) const	1896
29.406.3.7	isInDomain2(const Array< double, 2 > &point, double tolerance) const	1897
29.406.3.8	isOnBoundary(const Array< double, 2 > &point, double tolerance) const	1897
29.406.3.9	isOnCorner(const Array< double, 2 > &point, double tolerance) const	1897
29.406.3.10	lowerLeft() const	1898
29.406.3.11	max() const	1898
29.406.3.12	min() const	1898
29.406.3.13	upperRight() const	1898
29.406.3.14	max() const	1898
29.406.3.15	min() const	1899
29.406.3.16	whichBoundary(const Array< double, 2 > &point1, const Array< double, 2 > &point2, double tolerance) const	1899
29.407	Go::RectGrid Class Reference	1899
29.407.1	Detailed Description	1901
29.407.2	Constructor & Destructor Documentation	1901
29.407.2.1	RectGrid()	1901
29.407.2.2	RectGrid(int numu, int numv, int dim, double *pts)	1901
29.407.2.3	~RectGrid()	1901
29.407.3	Member Function Documentation	1901
29.407.3.1	boundingBox() const	1901
29.407.3.2	classType()	1901

29.407.3.3	<code>clone() const</code>	1902
29.407.3.4	<code>dimension() const</code>	1902
29.407.3.5	<code>instanceType() const</code>	1902
29.407.3.6	<code>numCoefs_u() const</code>	1902
29.407.3.7	<code>numCoefs_v() const</code>	1902
29.407.3.8	<code>rawData()</code>	1902
29.407.3.9	<code>rawData() const</code>	1903
29.407.3.10	<code>read(std::istream &is)</code>	1903
29.407.3.11	<code>setGrid(int numu, int numv, int dim, const double *pts)</code>	1903
29.407.3.12	<code>swapDirections()</code>	1903
29.407.3.13	<code>write(std::ostream &os) const</code>	1903
29.408	RectGridTesselator Class Reference	1904
29.408.1	Detailed Description	1905
29.408.2	Constructor & Destructor Documentation	1905
29.408.2.1	<code>RectGridTesselator(const RectGrid &rg)</code>	1905
29.408.2.2	<code>~RectGridTesselator()</code>	1905
29.408.3	Member Function Documentation	1905
29.408.3.1	<code>getMesh()</code>	1905
29.408.3.2	<code>tesselate()</code>	1905
29.409	RectMatrixCol Class Reference	1906
29.409.1	Detailed Description	1907
29.409.2	Constructor & Destructor Documentation	1907
29.409.2.1	<code>RectMatrixCol(const Matrix &, int, int, int)</code>	1907
29.409.2.2	<code>RectMatrixCol(const Matrix &, int)</code>	1907
29.409.3	Member Function Documentation	1907
29.409.3.1	<code>Down()</code>	1907
29.409.3.2	<code>Left()</code>	1907
29.409.3.3	<code>Reset(const Matrix &, int, int, int)</code>	1907
29.409.3.4	<code>Reset(const Matrix &, int)</code>	1907
29.409.3.5	<code>Right()</code>	1907

29.409.3.6Up()	1907
29.409.4Friends And Related Function Documentation	1908
29.409.4.1ComplexScale	1908
29.409.4.2Rotate	1908
29.410RectMatrixDiag Class Reference	1908
29.410.1Detailed Description	1909
29.410.2Constructor & Destructor Documentation	1909
29.410.2.1RectMatrixDiag(const DiagonalMatrix &D)	1909
29.410.3Member Function Documentation	1909
29.410.3.1DownDiag()	1909
29.410.3.2operator[](int i)	1909
29.410.3.3UpDiag()	1909
29.411RectMatrixRow Class Reference	1910
29.411.1Detailed Description	1910
29.411.2Constructor & Destructor Documentation	1911
29.411.2.1RectMatrixRow(const Matrix &, int, int, int)	1911
29.411.2.2RectMatrixRow(const Matrix &, int)	1911
29.411.3Member Function Documentation	1911
29.411.3.1Down()	1911
29.411.3.2Left()	1911
29.411.3.3operator[](int i)	1911
29.411.3.4Reset(const Matrix &, int, int, int)	1911
29.411.3.5Reset(const Matrix &, int)	1911
29.411.3.6Right()	1911
29.411.3.7Up()	1911
29.412RectMatrixRowCol Class Reference	1912
29.412.1Detailed Description	1913
29.412.2Constructor & Destructor Documentation	1913
29.412.2.1RectMatrixRowCol(Real *st, int nx, int sp, int sh)	1913
29.412.3Member Function Documentation	1913

29.412.3.1	AddScaled(const RectMatrixRowCol &, Real)	1913
29.412.3.2	Divide(const RectMatrixRowCol &, Real)	1913
29.412.3.3	Divide(Real)	1913
29.412.3.4	DownDiag()	1913
29.412.3.5	First()	1913
29.412.3.6	Negate()	1913
29.412.3.7	operator*(const RectMatrixRowCol &) const	1913
29.412.3.8	operator[](int i)	1914
29.412.3.9	Reset(Real *st, int nx, int sp, int sh)	1914
29.412.3.10	SumSquare() const	1914
29.412.3.11	UpDiag()	1914
29.412.3.12	Zero()	1914
29.412.4	Friends And Related Function Documentation	1914
29.412.4.1	ComplexScale	1914
29.412.4.2	Rotate	1914
29.412.5	Member Data Documentation	1914
29.412.5.1	n	1914
29.412.5.2	shift	1914
29.412.5.3	spacing	1915
29.412.5.4	store	1915
29.413	Go::LRSplineSurface::Refinement2D Struct Reference	1915
29.413.1	Detailed Description	1915
29.413.2	Member Function Documentation	1915
29.413.2.1	setVal(double val, double st, double e, Direction2D dir, int mult)	1915
29.413.3	Member Data Documentation	1915
29.413.3.1	d	1915
29.413.3.2	end	1916
29.413.3.3	kval	1916
29.413.3.4	multiplicity	1916
29.413.3.5	start	1916

29.414	Go::RegistrationInput Struct Reference	1916
29.414.1	Detailed Description	1917
29.414.2	Constructor & Destructor Documentation	1917
29.414.2.1	RegistrationInput()	1917
29.414.3	Member Function Documentation	1917
29.414.3.1	setToleranceWeights(double w_rotation, double w_translation, double w_rescaling)	1917
29.414.4	Member Data Documentation	1918
29.414.4.1	area_tolerance_sq_	1918
29.414.4.2	calculate_tolerance_weights_	1918
29.414.4.3	max_newton_iterations_	1918
29.414.4.4	max_solve_iterations_	1918
29.414.4.5	multi_core_	1918
29.414.4.6	newton_tolerance_	1918
29.414.4.7	solve_tolerance_	1918
29.414.4.8	tolerance_weight_rescale_	1919
29.414.4.9	tolerance_weight_rotation_	1919
29.414.4.10	tolerance_weight_translation_	1919
29.415	Go::RegistrationResult Struct Reference	1919
29.415.1	Detailed Description	1920
29.415.2	Member Function Documentation	1920
29.415.2.1	tok()	1920
29.415.3	Member Data Documentation	1920
29.415.3.1	last_change_	1920
29.415.3.2	last_newton_iteration_	1920
29.415.3.3	rescaling_	1920
29.415.3.4	result_type_	1921
29.415.3.5	rotation_matrix_	1921
29.415.3.6	solve_result_	1921
29.415.3.7	translation_	1921
29.416	Go::Registrator< T > Class Template Reference	1921

29.416.1	Detailed Description	1921
29.416.2	Constructor & Destructor Documentation	1922
29.416.2.1	Registrar()	1922
29.417	Go::RegularizeFace Class Reference	1922
29.417.1	Detailed Description	1923
29.417.2	Constructor & Destructor Documentation	1923
29.417.2.1	RegularizeFace(shared_ptr< ftSurface > face, double epsge, double angtol, double tol2, bool split_in_cand=false)	1923
29.417.2.2	RegularizeFace(shared_ptr< ftSurface > face, double epsge, double angtol, double tol2, double bend, bool split_in_cand=false)	1923
29.417.2.3	RegularizeFace(shared_ptr< ftSurface > face, shared_ptr< SurfaceModel > model, bool split_in_cand=false)	1923
29.417.2.4	~RegularizeFace()	1923
29.417.3	Member Function Documentation	1923
29.417.3.1	classifyVertices()	1923
29.417.3.2	fetchVxPntCorr()	1923
29.417.3.3	getRegularFaces()	1924
29.417.3.4	getSeamJointInfo() const	1924
29.417.3.5	setAxis(Point ¢re, Point &axis)	1924
29.417.3.6	setCandSplit(std::vector< std::pair< std::pair< Point, int >, std::pair< Point, int > > > cand_split)	1924
29.417.3.7	setDivideInT(bool divideInT)	1924
29.417.3.8	setNonTjointFaces(std::vector< shared_ptr< ftSurface > > &faces)	1924
29.417.3.9	setSplitMode(int split_mode)	1924
29.417.3.10	unsetAxis()	1924
29.418	Go::RegularizeFaceSet Class Reference	1925
29.418.1	Detailed Description	1925
29.418.2	Constructor & Destructor Documentation	1925
29.418.2.1	RegularizeFaceSet(std::vector< shared_ptr< ftSurface > > faces, double epsge, double angtol, bool split_in_cand=false)	1925
29.418.2.2	RegularizeFaceSet(std::vector< shared_ptr< ftSurface > > faces, double gap, double neighbour, double kink, double bend, bool split_in_cand=false)	1925
29.418.2.3	RegularizeFaceSet(shared_ptr< SurfaceModel > model, bool split_in_cand=false)	1925

29.418.2.4~RegularizeFaceSet()	1926
29.418.3 Member Function Documentation	1926
29.418.3.1fetchVxPntCorr()	1926
29.418.3.2getModifiedAdjacentModels()	1926
29.418.3.3getRegularFaces(bool reverse_sequence=false)	1926
29.418.3.4getRegularModel(bool reverse_sequence=false)	1926
29.418.3.5setFaceCorrespondance(int idx1, int idx2)	1926
29.418.3.6setSplitMode(int split_mode)	1926
29.419 Go::RegularMesh Class Reference	1927
29.419.1 Detailed Description	1928
29.419.2 Constructor & Destructor Documentation	1928
29.419.2.1RegularMesh(int m=20, int n=20, bool use_normals=true, bool use_↔ texcoords=false)	1928
29.419.2.2~RegularMesh()	1928
29.419.3 Member Function Documentation	1928
29.419.3.1asRegularMesh()	1928
29.419.3.2atBoundary(int idx)	1929
29.419.3.3normalArray()	1929
29.419.3.4numStrips()	1929
29.419.3.5numTriangles()	1929
29.419.3.6numVertices()	1929
29.419.3.7paramArray()	1929
29.419.3.8resize(int m, int n)	1929
29.419.3.9stripArray()	1930
29.419.3.10stripLength()	1930
29.419.3.11texcoordArray()	1930
29.419.3.12anslate(const std::vector< double > &vert_translation)	1930
29.419.3.13angleArray()	1930
29.419.3.14angleIndexArray()	1930
29.419.3.15seNormals()	1930
29.419.3.16seTexCoords()	1930

29.419.3.1	VertexArray()	1931
29.420	Go::RegularVolMesh Class Reference	1931
29.420.1	Detailed Description	1932
29.420.2	Constructor & Destructor Documentation	1932
29.420.2.1	RegularVolMesh(int m=20, bool use_normals=true, bool use_texcoords=false)	1932
29.420.2.2	~RegularVolMesh()	1933
29.420.3	Member Function Documentation	1933
29.420.3.1	asRegularMesh()	1933
29.420.3.2	atBoundary(int idx)	1933
29.420.3.3	normalArray()	1933
29.420.3.4	numStrips()	1933
29.420.3.5	numTriangles()	1933
29.420.3.6	numVertices()	1933
29.420.3.7	paramArray()	1934
29.420.3.8	resize(int m)	1934
29.420.3.9	stripArray()	1934
29.420.3.10	stripLength()	1934
29.420.3.11	texcoordArray()	1934
29.420.3.12	triangleArray()	1934
29.420.3.13	triangleIndexArray()	1934
29.420.3.14	useNormals()	1934
29.420.3.15	useTexCoords()	1935
29.420.3.16	VertexArray()	1935
29.421	NEWMAT::ReturnMatrixX Class Reference	1935
29.421.1	Detailed Description	1936
29.421.2	Constructor & Destructor Documentation	1936
29.421.2.1	~ReturnMatrixX()	1936
29.421.2.2	ReturnMatrixX(const ReturnMatrixX &tm)	1936
29.421.2.3	ReturnMatrixX(const GeneralMatrix *gmx)	1937
29.421.3	Member Function Documentation	1937

29.421.3.1BandWidth() const	1937
29.421.3.2Evaluate(MatrixType mt=MatrixTypeUnSp)	1937
29.421.4Friends And Related Function Documentation	1937
29.421.4.1BaseMatrix	1937
29.422NEWMAT::ReversedMatrix Class Reference	1937
29.422.1Detailed Description	1939
29.422.2Constructor & Destructor Documentation	1939
29.422.2.1~ReversedMatrix()	1939
29.422.3Member Function Documentation	1939
29.422.3.1Evaluate(MatrixType mt=MatrixTypeUnSp)	1939
29.422.4Friends And Related Function Documentation	1939
29.422.4.1BaseMatrix	1939
29.423Go::RotatedBox Class Reference	1939
29.423.1Detailed Description	1940
29.423.2Constructor & Destructor Documentation	1940
29.423.2.1RotatedBox(RandomAccessIterator start, int dim, int num_u, int num_v, const Point *axis)	1940
29.423.2.2RotatedBox(const Point &low, const Point &high, const Point *axis)	1940
29.423.2.3~RotatedBox()	1940
29.423.3Member Function Documentation	1941
29.423.3.1box() const	1941
29.423.3.2containsBox(const RotatedBox &box, double toli=0.0, double tole=0.0) const	1941
29.423.3.3containsPoint(const Point &pt, double toli=0.0, double tole=0.0) const	1941
29.423.3.4dimension() const	1941
29.423.3.5high() const	1941
29.423.3.6high_rot() const	1941
29.423.3.7low() const	1941
29.423.3.8low_rot() const	1942
29.423.3.9overlaps(const RotatedBox &box, double toli=0.0, double tole=0.0) const	1942
29.423.3.10setFromArray(RandomAccessIterator start, int dim, int num_u, int num_v)	1942
29.423.3.11setFromPoints(const Point &low, const Point &high)	1942

29.424	Go::RotationInfo Struct Reference	1942
29.424.1	Detailed Description	1943
29.424.2	Member Data Documentation	1943
29.424.2.1	center_pt_	1943
29.424.2.2	rot_angle_	1943
29.424.2.3	rot_axis_	1943
29.425	NEWMAT::RowedMatrix Class Reference	1943
29.425.1	Detailed Description	1945
29.425.2	Constructor & Destructor Documentation	1945
29.425.2.1	~RowedMatrix()	1945
29.425.3	Member Function Documentation	1945
29.425.3.1	BandWidth() const	1945
29.425.3.2	Evaluate(MatrixType mt=MatrixTypeUnSp)	1945
29.425.4	Friends And Related Function Documentation	1945
29.425.4.1	BaseMatrix	1945
29.426	NEWMAT::RowVector Class Reference	1946
29.426.1	Detailed Description	1948
29.426.2	Constructor & Destructor Documentation	1948
29.426.2.1	RowVector()	1948
29.426.2.2	~RowVector()	1948
29.426.2.3	RowVector(ArrayLengthSpecifier n)	1948
29.426.2.4	RowVector(const BaseMatrix &)	1948
29.426.2.5	RowVector(const RowVector &gm)	1948
29.426.3	Member Function Documentation	1948
29.426.3.1	CleanUp()	1948
29.426.3.2	element(int)	1948
29.426.3.3	element(int) const	1948
29.426.3.4	GetCol(MatrixRowCol &)	1948
29.426.3.5	GetCol(MatrixColX &)	1949
29.426.3.6	NextCol(MatrixRowCol &)	1949

29.426.3.7	NextCol(MatrixColX &)	1949
29.426.3.8	nrnc() const	1949
29.426.3.9	operator()(int)	1949
29.426.3.10	operator()(int) const	1949
29.426.3.11	operator=(const BaseMatrix &)	1949
29.426.3.12	operator=(Real f)	1949
29.426.3.13	operator=(const RowVector &m)	1949
29.426.3.14	ReSize(int)	1949
29.426.3.15	ReSize(int, int)	1949
29.426.3.16	ReSize(const GeneralMatrix &A)	1949
29.426.3.17	RestoreCol(MatrixRowCol &)	1950
29.426.3.18	RestoreCol(MatrixColX &c)	1950
29.426.3.19	Transpose(TransposedMatrix *, MatrixType)	1950
29.426.3.20	Type() const	1950
29.427	RBD_COMMON::Runtime_error Class Reference	1950
29.427.1	Detailed Description	1951
29.427.2	Constructor & Destructor Documentation	1951
29.427.2.1	Runtime_error(const char *a_what=0)	1951
29.427.3	Member Data Documentation	1951
29.427.3.1	Select	1951
29.428	Go::SamplePointData Struct Reference	1952
29.428.1	Detailed Description	1953
29.428.2	Constructor & Destructor Documentation	1953
29.428.2.1	SamplePointData(Point pos, Point norm, double curvature, ftSurface *face, double face_par_u, double face_par_v, ftEdge *edge, double edge_par)	1953
29.428.2.2	SamplePointData(Point pos, Point norm, double curvature, ftSurface *face, double face_par_u, double face_par_v)	1953
29.428.3	Member Data Documentation	1953
29.428.3.1	edge_	1953
29.428.3.2	edge_par_	1953
29.428.3.3	face_	1953

29.428.3.4	face_par_	1953
29.428.3.5	mean_curvature_	1954
29.428.3.6	norm_	1954
29.428.3.7	pos_	1954
29.429	NEWMAT::ScaledMatrix Class Reference	1954
29.429.1	Detailed Description	1955
29.429.2	Constructor & Destructor Documentation	1956
29.429.2.1	~ScaledMatrix()	1956
29.429.3	Member Function Documentation	1956
29.429.3.1	BandWidth() const	1956
29.429.3.2	Evaluate(MatrixType mt=MatrixTypeUnSp)	1956
29.429.4	Friends And Related Function Documentation	1956
29.429.4.1	BaseMatrix	1956
29.429.4.2	GeneralMatrix	1956
29.429.4.3	GenericMatrix	1956
29.429.4.4	operator*	1956
29.430	Go::ScratchVect< T, N > Class Template Reference	1957
29.430.1	Detailed Description	1958
29.430.2	Member Typedef Documentation	1958
29.430.2.1	const_iterator	1958
29.430.2.2	iterator	1958
29.430.3	Constructor & Destructor Documentation	1958
29.430.3.1	ScratchVect()	1958
29.430.3.2	ScratchVect(size_t size)	1958
29.430.3.3	ScratchVect(size_t size, T elem)	1958
29.430.3.4	ScratchVect(const std::vector< T > &vec)	1958
29.430.3.5	ScratchVect(RAlter beg, RAlter end)	1959
29.430.3.6	ScratchVect(const ScratchVect< T, M > &orig)	1959
29.430.4	Member Function Documentation	1959
29.430.4.1	assign(RAlter beg, RAlter end)	1959

29.430.4.2	<code>begin()</code>	1959
29.430.4.3	<code>begin() const</code>	1959
29.430.4.4	<code>clear()</code>	1959
29.430.4.5	<code>end()</code>	1959
29.430.4.6	<code>end() const</code>	1960
29.430.4.7	<code>operator=(const ScratchVect &orig)</code>	1960
29.430.4.8	<code>operator[](int i) const</code>	1960
29.430.4.9	<code>operator[](int i)</code>	1960
29.430.4.10	<code>push_back(const T &e)</code>	1960
29.430.4.11	<code>resize(size_t new_size)</code>	1960
29.430.4.12	<code>size() const</code>	1960
29.430	<code>Go::SecondOrderProperties</code> Struct Reference	1961
29.431.1	Detailed Description	1961
29.431.2	Constructor & Destructor Documentation	1961
29.431.2.1	<code>SecondOrderProperties()</code>	1961
29.431.3	Member Function Documentation	1962
29.431.3.1	<code>clear()</code>	1962
29.431.4	Member Data Documentation	1962
29.431.4.1	<code>singularity_info_is_cached</code>	1962
29.431.4.2	<code>singularity_type</code>	1962
29.431.4.3	<code>tangent_2d_1</code>	1962
29.431.4.4	<code>tangent_2d_2</code>	1962
29.431.4.5	<code>tangent_2d_is_cached</code>	1962
29.431.4.6	<code>tangent_3d</code>	1962
29.431.4.7	<code>tangent_is_oriented</code>	1962
29.432	<code>Go::SfBoundaryCondition</code> Class Reference	1963
29.432.1	Detailed Description	1964
29.432.2	Constructor & Destructor Documentation	1964
29.432.2.1	<code>SfBoundaryCondition(int edge_nmb, BdConditionType type, BdCondFuncor *fbd, std::pair< double, double > end_par, SfSolution *solution)</code>	1964

29.432.2.2	SfBoundaryCondition(int edge_nmb, BdConditionType type, const Point &const← _val, std::pair< double, double > end_par, SfSolution *solution)	1964
29.432.2.3	~SfBoundaryCondition()	1964
29.432.3	Member Function Documentation	1964
29.432.3.1	edgeNumber() const	1964
29.432.3.2	getBasisFunctions(int index_of_Gauss_point, bool &u_dir, std::vector< double > &basisValues, std::vector< double > &basisDerivs) const	1964
29.432.3.3	getBdCoefficients(std::vector< std::pair< int, Point > > &coefs)	1964
29.432.3.4	getBdCoefficients(std::vector< std::pair< int, Point > > &coefs_bd, std::vector< std::pair< int, Point > > &coefs_bd2)	1964
29.432.3.5	getCoefficientsEnumeration(std::vector< int > &local_enumeration)	1965
29.432.3.6	getCoefficientsEnumeration(std::vector< int > &local_enumeration_bd, std← ::vector< int > &local_enumeration_bd2)	1965
29.432.3.7	getSplineApproximation() const	1965
29.432.3.8	getTolerances() const	1965
29.432.3.9	update()	1965
29.432.3.10	updateBoundaryValue(BdCondFuncor *fbd)	1965
29.433	Go::CompositeModelFileHandler::sfvinfo Struct Reference	1965
29.433.1	Detailed Description	1966
29.433.2	Constructor & Destructor Documentation	1966
29.433.2.1	sfvinfo(int ccm, int constdir, double constpar, int bd, int orientation)	1966
29.433.2.2	sfvinfo()	1966
29.433.3	Member Data Documentation	1966
29.433.3.1	bd_	1966
29.433.3.2	ccm_	1966
29.433.3.3	constdir_	1966
29.433.3.4	constpar_	1967
29.433.3.5	infofet_	1967
29.433.3.6	orientation_	1967
29.434	Go::SfCvIntersector Class Reference	1967
29.434.1	Detailed Description	1968
29.434.2	Constructor & Destructor Documentation	1968

29.434.2.1	SfCvIntersector(shared_ptr< ParamGeomInt > obj1, shared_ptr< ParamGeomInt > obj2, double epsge, Intersector *prev=0, int eliminated_parameter=-1, double eliminated_value=0)	1968
29.434.2.2	SfCvIntersector(shared_ptr< ParamGeomInt > obj1, shared_ptr< ParamGeomInt > obj2, shared_ptr< GeoTol > epsge, Intersector *prev=0, int eliminated_parameter=-1, double eliminated_value=0)	1969
29.434.2.3	~SfCvIntersector()	1969
29.434.3	Member Function Documentation	1969
29.434.3.1	checkCoincidence()	1969
29.434.3.2	doSubdivide()	1969
29.434.3.3	foundIntersectionNearBoundary()	1969
29.434.3.4	linearCase()	1969
29.434.3.5	lowerOrderIntersector(shared_ptr< ParamGeomInt > obj1, shared_ptr< ParamGeomInt > obj2, Intersector *prev=0, int eliminated_parameter=-1, double eliminated_value=0)	1969
29.434.3.6	microCase()	1970
29.434.3.7	numParams() const	1970
29.434.3.8	performRotatedBoxTest(double eps1, double eps2)	1970
29.434.3.9	postIterate(int nmb_orig, int dir=-1, bool keep_endpt=true)	1970
29.434.3.10	postIterateBd()	1970
29.434.3.11	repairIntersections()	1970
29.434.3.12	simpleCase()	1970
29.434.3.13	simpleCase2(Point &axis1, Point &axis2)	1970
29.434.3.14	updateIntersections()	1971
29.435	Go::SfPointBdCond Class Reference	1971
29.435.1	Detailed Description	1972
29.435.2	Constructor & Destructor Documentation	1972
29.435.2.1	SfPointBdCond(int edge_nmb, double param, Point &condition_value)	1972
29.435.2.2	~SfPointBdCond()	1972
29.435.3	Member Function Documentation	1972
29.435.3.1	edgeNumber() const	1972
29.435.3.2	getCoefficientsEnumeration(std::vector< int > &local_enumeration) const	1972
29.435.3.3	getConditionValue() const	1972

29.435.3.4	<code>getInterpolationFactors(std::vector< std::pair< int, double > > &factors) const</code>	1972
29.435.3.5	<code>getParam() const</code>	1972
29.436	<code>Go::SfPtIntersector Class Reference</code>	1973
29.436.1	Detailed Description	1974
29.436.2	Constructor & Destructor Documentation	1974
29.436.2.1	<code>SfPtIntersector(shared_ptr< ParamGeomInt > obj1, shared_ptr< ParamGeomInt > obj2, shared_ptr< GeoTol > epsge, Intersector *prev=0, int eliminated_parameter=-1, double eliminated_value=0)</code>	1974
29.436.2.2	<code>~SfPtIntersector()</code>	1974
29.436.3	Member Function Documentation	1974
29.436.3.1	<code>checkCoincidence()</code>	1974
29.436.3.2	<code>doSubdivide()</code>	1974
29.436.3.3	<code>linearCase()</code>	1974
29.436.3.4	<code>lowerOrderIntersector(shared_ptr< ParamGeomInt > obj1, shared_ptr< ParamGeomInt > obj2, Intersector *prev=0, int eliminated_parameter=-1, double eliminated_value=0)</code>	1975
29.436.3.5	<code>microCase()</code>	1975
29.436.3.6	<code>numParams() const</code>	1975
29.436.3.7	<code>performRotatedBoxTest(double eps1, double eps2)</code>	1975
29.436.3.8	<code>repairIntersections()</code>	1975
29.436.3.9	<code>updateIntersections()</code>	1975
29.437	<code>Go::SfSelfIntersector Class Reference</code>	1976
29.437.1	Detailed Description	1977
29.437.2	Constructor & Destructor Documentation	1977
29.437.2.1	<code>SfSelfIntersector(shared_ptr< ParamSurfaceInt > surf, double epsge, Intersector *prev=NULL)</code>	1977
29.437.2.2	<code>SfSelfIntersector(shared_ptr< ParamSurfaceInt > surf, shared_ptr< GeoTol > epsge, Intersector *prev=NULL)</code>	1977
29.437.2.3	<code>~SfSelfIntersector()</code>	1978
29.437.3	Member Function Documentation	1978
29.437.3.1	<code>addComplexDomain(RectDomain dom)</code>	1978
29.437.3.2	<code>checkCoincidence()</code>	1978
29.437.3.3	<code>complexityReduced()</code>	1978

29.437.3.4	<code>compute(bool compute_at_boundary=true)</code>	1978
29.437.3.5	<code>computeG1()</code>	1978
29.437.3.6	<code>doSubdivide()</code>	1979
29.437.3.7	<code>getBoundaryIntersections()</code>	1979
29.437.3.8	<code>getNmbComplexDomain()</code>	1979
29.437.3.9	<code>getNonSelfintersecting()</code>	1979
29.437.3.10	<code>handleComplexity()</code>	1979
29.437.3.11	<code>isLinear()</code>	1979
29.437.3.12	<code>isSelfIntersection()</code>	1979
29.437.3.13	<code>nearCase()</code>	1980
29.437.3.14	<code>microCase()</code>	1980
29.437.3.15	<code>sumParams() const</code>	1980
29.437.3.16	<code>performInterception()</code>	1980
29.437.3.17	<code>print_objs()</code>	1980
29.437.3.18	<code>printDebugInfo()</code>	1980
29.437.3.19	<code>pairIntersections()</code>	1980
29.437.3.20	<code>setMaxRec(int max_rec)</code>	1980
29.437.3.21	<code>simpleCase()</code>	1981
29.437.3.22	<code>updateIntersections()</code>	1981
29.438	<code>SfSfIntersector Class Reference</code>	1981
29.438.1	Detailed Description	1983
29.438.2	Constructor & Destructor Documentation	1983
29.438.2.1	<code>SfSfIntersector()</code>	1983
29.438.2.2	<code>SfSfIntersector(shared_ptr< ParamGeomInt > obj1, shared_ptr< ParamGeomInt > obj2, shared_ptr< GeoTol > epsge, Intersector *prev=0)</code>	1983
29.438.2.3	<code>SfSfIntersector(shared_ptr< ParamGeomInt > obj1, shared_ptr< ParamGeomInt > obj2, double epsge, Intersector *prev=0)</code>	1983
29.438.2.4	<code>~SfSfIntersector()</code>	1984
29.438.3	Member Function Documentation	1984
29.438.3.1	<code>checkCloseEndpoint(shared_ptr< IntersectionPoint > pnt, shared_ptr< IntersectionLink > link)</code>	1984
29.438.3.2	<code>checkCoincidence()</code>	1984

29.438.3.3	complexityReduced()	1984
29.438.3.4	connectIfPossible(shared_ptr< IntersectionPoint > pt1, shared_ptr< IntersectionPoint > pt2)	1984
29.438.3.5	degTriangleSimple()	1984
29.438.3.6	doSubdivide()	1984
29.438.3.7	fixCrossingLinks()	1984
29.438.3.8	foundIntersectionNearBoundary()	1984
29.438.3.9	getApproxImplicit(std::vector< std::vector< shared_ptr< Param2FunctionInt > > &approx_implicit, std::vector< double > &approx_implicit_err, std::vector< double > &approx_implicit_gradsize, std::vector< double > &approx_implicit_gradvar)	1985
29.438.3.10	getSingularityBox(shared_ptr< IntersectionPoint > sing, double frompar[], double topar[])	1985
29.438.3.11	getSubIntersector(double frompar[], double topar[])	1985
29.438.3.12	handleComplexity()	1985
29.438.3.13	interceptionBySeparationSurface()	1985
29.438.3.14	iterateOnIntersectionPoints()	1985
29.438.3.15	linearCase()	1985
29.438.3.16	lowerOrderIntersector(shared_ptr< ParamGeomInt > obj1, shared_ptr< ParamGeomInt > obj2, Intersector *prev=0, int eliminated_parameter=-1, double eliminated_value=0)	1985
29.438.3.17	microCase()	1985
29.438.3.18	numParams() const	1985
29.438.3.19	performInterceptionByImplicitization()	1986
29.438.3.20	performRotatedBoxTest(double eps1, double eps2)	1986
29.438.3.21	postIterate3(int nmb_orig, int dir)	1986
29.438.3.22	moveIsolatedPoints()	1986
29.438.3.23	pairFalseBranchPoints()	1986
29.438.3.24	pairIntersections()	1986
29.438.3.25	pairMissingLinks()	1986
29.438.3.26	pairSingularityBox()	1986
29.438.3.27	simpleCase2(Point &axis1, Point &axis2)	1986
29.438.3.28	simpleCaseByImplicitization()	1986

29.438.3.2	updateIntersections()	1986
29.438.4	Friends And Related Function Documentation	1986
29.438.4.1	IntersectionPool	1986
29.438.4.2	SfSelfIntersector	1987
29.439	SfSolution Class Reference	1987
29.439.1	Detailed Description	1989
29.439.2	Constructor & Destructor Documentation	1989
29.439.2.1	SfSolution(IsogeometricSfBlock *parent, shared_ptr< SplineSurface > sol_surf)	1989
29.439.2.2	~SfSolution()	1989
29.439.3	Member Function Documentation	1989
29.439.3.1	addBoundaryCondition(int edge_nmb, BdConditionType type, BdCondFuncor *fbd, std::pair< double, double > end_par)	1989
29.439.3.2	addBoundaryCondition(int edge_nmb, BdConditionType type, const Point &const_val, std::pair< double, double > end_par)	1989
29.439.3.3	addDirichletPointBdCond(double param[], Point &condition_value)	1989
29.439.3.4	asSfSolution()	1989
29.439.3.5	basis(int paddir) const	1989
29.439.3.6	degree(int paddir) const	1989
29.439.3.7	dimension() const	1989
29.439.3.8	distinctKnots(int paddir) const	1989
29.439.3.9	erasePreEvaluatedBasisFunctions()	1990
29.439.3.10	getBasisFunctions(int index_of_Gauss_point1, int index_of_Gauss_point2, std::vector< double > &basisValues, std::vector< double > &basisDerivs_u, std::vector< double > &basisDerivs_v) const	1990
29.439.3.11	getBasisFunctions(double param1, double param2, std::vector< double > &basisValues, std::vector< double > &basisDerivs_u, std::vector< double > &basisDerivs_v) const	1990
29.439.3.12	getBasisFunctionValues(int basis_func_id_u, int basis_func_id_v, std::vector< int > &index_of_Gauss_points1, std::vector< int > &index_of_Gauss_points2, std::vector< double > &basisValues, std::vector< double > &basisDerivs_u, std::vector< double > &basisDerivs_v) const	1990
29.439.3.13	getBasisFunctionValues(int basis_func_id_u, int basis_func_id_v, int knot_ind_u, int knot_ind_v, std::vector< int > &index_of_Gauss_points1, std::vector< int > &index_of_Gauss_points2, std::vector< double > &basisValues, std::vector< double > &basisDerivs_u, std::vector< double > &basisDerivs_v) const	1990
29.439.3.14	getBoundaryCoefficients(int boundary, std::vector< int > &enumeration) const	1990

29.439.3.15	GetBoundaryCoefficients(int boundary, std::vector< int > &enumeration_bd, std::vector< int > &enumeration_bd2) const	1990
29.439.3.16	GetBoundaryCondition(int index) const	1990
29.439.3.17	GetEdgeBoundaryConditions(int edge_number, std::vector< shared_ptr< SfBoundaryCondition > > &bd_cond) const	1990
29.439.3.18	GetEdgeBoundaryConditions(std::vector< shared_ptr< SfBoundaryCondition > > &bd_cond) const	1990
29.439.3.19	GetEdgePointBdConditions(int edge_number, std::vector< shared_ptr< SfPointBdCond > > &bd_cond) const	1990
29.439.3.20	GetGaussParameter(int index_of_Gauss_point, int pardir) const	1990
29.439.3.21	GetGeometrySurface() const	1990
29.439.3.22	GetJacobian(std::vector< int > &index_of_Gauss_point) const	1990
29.439.3.23	GetMatchingCoefficients(BlockSolution *other, std::vector< std::pair< int, int > > &enumeration, int match_pos=0) const	1991
29.439.3.24	GetNmbOfBoundaryConditions() const	1991
29.439.3.25	GetNmbOfPointBdConditions() const	1991
29.439.3.26	GetPointBdCond(std::vector< shared_ptr< SfPointBdCond > > &bd_cond) const	1991
29.439.3.27	GetPointBdCondition(int index) const	1991
29.439.3.28	GetSolutionSurface() const	1991
29.439.3.29	GetTolerances() const	1991
29.439.3.30	IncreaseDegree(int new_degree, int pardir)	1991
29.439.3.31	InsertKnots(const std::vector< int > &knot_intervals, int pardir)	1991
29.439.3.32	InsertKnots(const std::vector< double > &knots, int pardir)	1991
29.439.3.33	Knots(int pardir) const	1991
29.439.3.34	MakeMatchingSplineSpace(BlockSolution *other)	1991
29.439.3.35	MatchingSplineSpace(BlockSolution *other) const	1992
29.439.3.36	NmbCoefs() const	1992
29.439.3.37	NmbCoefs(int pardir) const	1992
29.439.3.38	PerformPreEvaluation(std::vector< std::vector< double > > &Gauss_par)	1992
29.439.3.39	RefineToGeometry(int pardir)	1992
29.439.3.40	SetMinimumDegree(int degree)	1992
29.439.3.41	SetSolutionCoefficients(const std::vector< double > &coefs)	1992
29.439.3.42	UpdateConditions()	1992

29.439.3.42	valuesInGaussPoint(const std::vector< int > &index_of_Gauss_point, std::vector< Point > &derivs) const	1992
29.440	NEWMAT::ShiftedMatrix Class Reference	1993
29.440.1	Detailed Description	1994
29.440.2	Constructor & Destructor Documentation	1994
29.440.2.1	ShiftedMatrix(const BaseMatrix *bmx, Real fx)	1994
29.440.2.2	~ShiftedMatrix()	1994
29.440.3	Member Function Documentation	1994
29.440.3.1	Evaluate(MatrixType mt=MatrixTypeUnSp)	1994
29.440.3.2	search(const BaseMatrix *) const	1995
29.440.4	Friends And Related Function Documentation	1995
29.440.4.1	BaseMatrix	1995
29.440.4.2	GeneralMatrix	1995
29.440.4.3	GenericMatrix	1995
29.440.4.4	operator+	1995
29.440.5	Member Data Documentation	1995
29.440.5.1	@9	1995
29.440.5.2	bm	1995
29.440.5.3	f	1995
29.440.5.4	gm	1995
29.440	Go::sideConstraint Struct Reference	1996
29.441.1	Detailed Description	1996
29.441.2	Member Data Documentation	1996
29.441.2.1	constant_term_	1996
29.441.2.2	dim_	1996
29.441.2.3	factor_	1996
29.442	Go::sideConstraintSet Struct Reference	1997
29.442.1	Detailed Description	1997
29.442.2	Constructor & Destructor Documentation	1997
29.442.2.1	sideConstraintSet()	1997
29.442.2.2	sideConstraintSet(int dim)	1997

29.442.3	Member Data Documentation	1997
29.442.3.1	constant_term_	1998
29.442.3.2	dim_	1998
29.442.3.3	factor_	1998
29.443	IEWMAT::SimpleIntArray Class Reference	1998
29.443.1	Detailed Description	1999
29.443.2	Constructor & Destructor Documentation	1999
29.443.2.1	SimpleIntArray(int xn)	1999
29.443.2.2	~SimpleIntArray()	1999
29.443.2.3	SimpleIntArray(const SimpleIntArray &b)	2000
29.443.3	Member Function Documentation	2000
29.443.3.1	CleanUp()	2000
29.443.3.2	Data()	2000
29.443.3.3	Data() const	2000
29.443.3.4	operator=(int ai)	2000
29.443.3.5	operator=(const SimpleIntArray &b)	2000
29.443.3.6	operator[](int i)	2000
29.443.3.7	operator[](int i) const	2000
29.443.3.8	ReSize(int i, bool keep=false)	2000
29.443.3.9	Size() const	2000
29.443.4	Member Data Documentation	2001
29.443.4.1a	2001
29.443.4.2a	2001
29.444	Go::SingBox Struct Reference	2001
29.444.1	Detailed Description	2002
29.444.2	Constructor & Destructor Documentation	2002
29.444.2.1	SingBox(std::vector< std::pair< double, int > > box, shared_ptr< Intersection← Point > sing)	2002
29.444.3	Member Data Documentation	2002
29.444.3.1	box_limit_	2002
29.444.3.2	sing_	2002

29.445.5	EWMAT::SingularException Class Reference	2002
29.445.1	Detailed Description	2003
29.445.2	Constructor & Destructor Documentation	2003
29.445.2.1	SingularException(const GeneralMatrix &A)	2003
29.445.3	Member Data Documentation	2003
29.445.3.1	Select	2003
29.446.6	Go::SingularityInfo Class Reference	2004
29.446.1	Detailed Description	2004
29.446.2	Constructor & Destructor Documentation	2004
29.446.2.1	SingularityInfo()	2004
29.446.2.2	SingularityInfo(shared_ptr< SingularityInfo > previous, bool use_previous=false)	2004
29.446.2.3	SingularityInfo(shared_ptr< SingularityInfo > previous, int missing_dir)	2005
29.446.3	Member Function Documentation	2005
29.446.3.1	addIterationCount()	2005
29.446.3.2	addSimpleCount(bool simple1, bool simple2)	2005
29.446.3.3	addSingularPoint(double *par, int nmb_par)	2005
29.446.3.4	cleanUp(std::vector< double > start, std::vector< double > end, double tol)	2006
29.446.3.5	getHighPriSing(int idx)	2006
29.446.3.6	getHighPriSing()	2006
29.446.3.7	getNmbPoints(int nmb_par)	2006
29.446.3.8	getParam(int dir)	2006
29.446.3.9	getPoint(int idx, int nmb_par)	2007
29.446.3.10	hasHighPriSing()	2007
29.446.3.11	hasPoint()	2007
29.446.3.12	iterationDone()	2007
29.446.3.13	iterationSucceed()	2008
29.446.3.14	nmbSimple1()	2008
29.446.3.15	nmbSimple2()	2008
29.446.3.16	setHighPriSing(double *par, int nmb_par)	2008
29.446.3.17	setHighPriSingType(SingularityClassification type)	2008

29.446.3.1	setSingularPoint(double *par, int nmb_par)	2009
29.447	Go::SingUnion Struct Reference	2009
29.447.1	Detailed Description	2009
29.447.2	Constructor & Destructor Documentation	2009
29.447.2.1	SingUnion(double box[], std::vector< int > idx)	2009
29.447.3	Member Function Documentation	2009
29.447.3.1	isInside(double *mima)	2009
29.447.4	Member Data Documentation	2010
29.447.4.1	limit_	2010
29.447.4.2	singbox_idx_	2010
29.448	ISLbox Struct Reference	2010
29.448.1	Detailed Description	2010
29.448.2	Member Data Documentation	2010
29.448.2.1	e2max	2010
29.448.2.2	e2min	2010
29.448.2.3	e3max	2011
29.448.2.4	e3min	2011
29.448.2.5	etol	2011
29.448.2.6	fmax	2011
29.448.2.7	fmin	2011
29.449	ISLCurve Struct Reference	2011
29.449.1	Detailed Description	2012
29.449.2	Member Data Documentation	2012
29.449.2.1	cuopen	2012
29.449.2.2	coef	2012
29.449.2.3	et	2012
29.449.2.4	copy	2012
29.449.2.5	dim	2012
29.449.2.6	k	2012
29.449.2.7	kind	2013

29.449.2.8n 2013

29.449.2.9pbox 2013

29.449.2.10dir 2013

29.449.2.11coef 2013

29.450.1SISLdir Struct Reference 2013

29.450.1.1Detailed Description 2013

29.450.2Member Data Documentation 2014

29.450.2.1aang 2014

29.450.2.2bcoef 2014

29.450.2.3smooth 2014

29.450.2.4gtpi 2014

29.451.1SISLEdge Struct Reference 2014

29.451.1.1Detailed Description 2015

29.451.2Member Data Documentation 2015

29.451.2.1iedge 2015

29.451.2.2point 2015

29.451.2.3rpt 2015

29.452.1SISLIntcurve Struct Reference 2015

29.452.1.1Detailed Description 2016

29.452.2Member Data Documentation 2016

29.452.2.1epar1 2016

29.452.2.2epar2 2016

29.452.2.3par1 2016

29.452.2.4par2 2016

29.452.2.5point 2016

29.452.2.6type 2016

29.452.2.7geom 2017

29.452.2.8ppar1 2017

29.452.2.9ppar2 2017

29.452.2.10retop 2017

29.453	ISLIntdat Struct Reference	2017
29.453.1	Detailed Description	2018
29.453.2	Member Data Documentation	2018
29.453.2.1	filist	2018
29.453.2.2	lmax	2018
29.453.2.3	pmax	2018
29.453.2.4	point	2018
29.453.2.5	vlist	2018
29.453.2.6	vpoint	2018
29.454	ISLIntlist Struct Reference	2019
29.454.1	Detailed Description	2019
29.454.2	Member Data Documentation	2019
29.454.2.1	ind_first	2019
29.454.2.2	ind_last	2020
29.454.2.3	numb	2020
29.454.2.4	type	2020
29.454.2.5	pfirst	2020
29.454.2.6	plast	2020
29.454.2.7	pretop	2020
29.455	ISLIntpt Struct Reference	2020
29.455.1	Detailed Description	2021
29.455.2	Member Data Documentation	2021
29.455.2.1	adist	2021
29.455.2.2	curve_dir	2021
29.455.2.3	edge_1	2021
29.455.2.4	edge_2	2022
29.455.2.5	epar	2022
29.455.2.6	evaluated	2022
29.455.2.7	geo_data_1	2022
29.455.2.8	geo_data_2	2022

29.455.2.9	geo_track_2d_1	2022
29.455.2.10	geo_track_2d_2	2022
29.455.2.11	geo_track_3d	2022
29.455.2.12	inter	2022
29.455.2.13	par	2022
29.455.2.14	side_1	2023
29.455.2.15	side_2	2023
29.455.2.16	ft_obj_1	2023
29.455.2.17	ft_obj_2	2023
29.455.2.18	marker	2023
29.455.2.19	no_of_curves	2023
29.455.2.20	no_of_curves_alloc	2023
29.455.2.21	curve	2023
29.455.2.22	next	2023
29.455.2.23	ght_obj_1	2023
29.455.2.24	ght_obj_2	2024
29.455.2.25	size_1	2024
29.455.2.26	size_2	2024
29.455.2.27	rim	2024
29.456	ISLIntsurf Struct Reference	2024
29.456.1	Detailed Description	2024
29.456.2	Member Data Documentation	2024
29.456.2.1	const_par	2024
29.456.2.2	epar	2025
29.456.2.3	par	2025
29.456.2.4	point	2025
29.457	ISLObject Struct Reference	2025
29.457.1	Detailed Description	2026
29.457.2	Member Data Documentation	2026
29.457.2.1	c1	2026

29.457.2.2	edg	2026
29.457.2.3	obj	2026
29.457.2.4	o1	2026
29.457.2.5	p1	2026
29.457.2.6	simple	2026
29.457.2.7	s1	2026
29.458	ISLPoint Struct Reference	2027
29.458.1	Detailed Description	2027
29.458.2	Member Data Documentation	2027
29.458.2.1	ec	2027
29.458.2.2	coef	2027
29.458.2.3	copy	2027
29.458.2.4	dim	2028
29.458.2.5	box	2028
29.459	ISLPtedge Struct Reference	2028
29.459.1	Detailed Description	2028
29.459.2	Member Data Documentation	2029
29.459.2.1	pnext	2029
29.459.2.2	pt	2029
29.460	ISLSurf Struct Reference	2029
29.460.1	Detailed Description	2030
29.460.2	Member Data Documentation	2030
29.460.2.1	cuopen_1	2030
29.460.2.2	cuopen_2	2030
29.460.2.3	coef	2030
29.460.2.4	et1	2030
29.460.2.5	et2	2030
29.460.2.6	copy	2030
29.460.2.7	dim	2030
29.460.2.8	k1	2030

29.460.2.9k2	2030
29.460.2.10ind	2031
29.460.2.11h1	2031
29.460.2.11a2	2031
29.460.2.12box	2031
29.460.2.13dir	2031
29.460.2.14coef	2031
29.460.2.15se_count	2031
29.461.1SISLTrack Struct Reference	2032
29.461.1.1Detailed Description	2032
29.461.1.2Member Data Documentation	2032
29.461.2.1impli	2032
29.461.2.2exact	2033
29.461.2.3deg	2033
29.461.2.4pcurve_2d_1	2033
29.461.2.5pcurve_2d_2	2033
29.461.2.6pcurve_3d	2033
29.461.2.7pretop	2033
29.461.2.8psurf_1	2033
29.461.2.9psurf_2	2033
29.461.2.10ing_end	2033
29.461.2.11ing_start	2033
29.461.2.12rned	2034
29.462.1SISLTrimpar Struct Reference	2034
29.462.1.1Detailed Description	2034
29.462.1.2Member Data Documentation	2034
29.462.2.1parindex	2034
29.462.2.2ptindex	2034
29.463.1Go::SmoothCurve Class Reference	2034
29.463.1.1Detailed Description	2035

29.463.2	Constructor & Destructor Documentation	2035
29.463.2.1	SmoothCurve(int dim=3)	2035
29.463.2.2	~SmoothCurve()	2035
29.463.3	Member Function Documentation	2035
29.463.3.1	attach(const shared_ptr< SplineCurve > &incurve, int coef_known[], int numSideConstraints=0)	2035
29.463.3.2	equationSolve(shared_ptr< SplineCurve > &curve)	2036
29.463.3.3	setLeastSquares(const std::vector< double > &pnts, const std::vector< double > ¶m_pnts, const std::vector< double > &pnt_weights, double weight)	2036
29.463.3.4	setOptim(const double weight1, const double weight2, const double weight3)	2036
29.463.3.5	setPeriodicity(int cont, double weight)	2037
29.463.3.6	setSideConstraints(std::vector< sideConstraint > &constraints)	2037
29.464	Go::SmoothCurveSet Class Reference	2037
29.464.1	Detailed Description	2038
29.464.2	Constructor & Destructor Documentation	2038
29.464.2.1	SmoothCurveSet()	2038
29.464.2.2	~SmoothCurveSet()	2038
29.464.3	Member Function Documentation	2038
29.464.3.1	attach(std::vector< shared_ptr< SplineCurve > > &incvs, std::vector< int > &seem, std::vector< std::vector< int > > &coef_known, int numSideConstraints=0)	2038
29.464.3.2	equationSolve(std::vector< shared_ptr< SplineCurve > > &curves)	2038
29.464.3.3	setApproxOrig(double weight)	2038
29.464.3.4	setApproxSideConstraints(std::vector< shared_ptr< sideConstraintSet > > &constraints, double weight)	2038
29.464.3.5	setCvSetConstraints(const std::vector< shared_ptr< cvSetConstraint > > &cv_set_constraints, bool appr_constraints, double appr_wgt)	2038
29.464.3.6	setInterpolationConditions(const std::vector< std::vector< double > > &pnts, const std::vector< std::vector< double > > ¶m_pnts, const std::vector< std::vector< int > > &der, bool appr_constraints, double appr_wgt, int *jstat)	2039
29.464.3.7	setLeastSquares(const std::vector< std::vector< double > > &pnts, const std::vector< std::vector< double > > ¶m_pnts, const std::vector< std::vector< double > > &pnt_weights, double weight)	2039
29.464.3.8	setOptimize(double weight1, double weight2, double weight3)	2039
29.464.3.9	setOrthCond(const std::vector< std::vector< double > > &pnts, const std::vector< std::vector< double > > ¶m_pnts, double weight)	2039

29.464.3.1	<code>setSideConstraints(std::vector< shared_ptr< sideConstraintSet > > &constraints, bool replace_constraints)</code>	2039
29.465	<code>Go::SmoothSurf Class Reference</code>	2039
29.465.1	Detailed Description	2041
29.465.2	Constructor & Destructor Documentation	2041
29.465.2.1	<code>SmoothSurf()</code>	2041
29.465.2.2	<code>SmoothSurf(bool copy_coefs)</code>	2041
29.465.2.3	<code>~SmoothSurf()</code>	2041
29.465.3	Member Function Documentation	2041
29.465.3.1	<code>adjustAtSeem()</code>	2042
29.465.3.2	<code>approxOrig(double weight)</code>	2042
29.465.3.3	<code>approxOrigNonrational(double weight)</code>	2042
29.465.3.4	<code>approxOrigRational(double weight)</code>	2042
29.465.3.5	<code>attach(shared_ptr< SplineSurface > &insf, int seem[], int coef_known[], int num_side_constraints=0, int has_normal_cond=0)</code>	2042
29.465.3.6	<code>equationSolve(shared_ptr< SplineSurface > &surf)</code>	2043
29.465.3.7	<code>getBasis(const double *sb1, const double *sb2, int kleft1, int kleft2, int nder, double *sbasis)</code>	2043
29.465.3.8	<code>prepareIntegral()</code>	2043
29.465.3.9	<code>preparePeriodicity(int seem[])</code>	2043
29.465.3.10	<code>releaseScratch()</code>	2043
29.465.3.11	<code>setC1AtSeem(int pardir, double weight)</code>	2044
29.465.3.12	<code>setC2AtSeem(int pardir, double weight)</code>	2044
29.465.3.13	<code>setLeastSquares(const std::vector< double > &pnts, const std::vector< double > &param_pnts, const std::vector< double > &pnt_weights, const double weight)</code>	2044
29.465.3.14	<code>setNormalCond(const std::vector< double > &pnts, const std::vector< double > &param_pnts, const std::vector< double > &pnt_weights, const double weight)</code>	2044
29.465.3.15	<code>setOptimize(const double weight1, const double weight2, const double weight3)</code>	2044
29.465.3.16	<code>setOptimizeNonrational(const double weight1, const double weight2, const double weight3)</code>	2045
29.465.3.17	<code>setOptimizeRational(const double weight1, const double weight2, const double weight3)</code>	2045
29.465.3.18	<code>setPeriodicity(int pardir, int cont, double weight1, double weight2)</code>	2045

29.465.3.19	setRationalCnAtSeem(int pdir, int cn, double weight1, double weight2)	2045
29.465.3.20	setRelaxParam(double omega)	2045
29.465.3.21	setSideConstraints(std::vector< sideConstraint > &constraints)	2046
29.465.4	Member Data Documentation	2047
29.465.4.1	bspline_surface_	2047
29.465.4.2	coef_array_	2047
29.465.4.3	coefknown_	2047
29.465.4.4	cont_seem_	2047
29.465.4.5	copy_coefs_	2047
29.465.4.6	gmat_	2047
29.465.4.7	gright_	2047
29.465.4.8	der_	2047
29.465.4.9	der_scratch_	2048
29.465.4.10	dim1_	2048
29.465.4.11	dim_	2048
29.465.4.12	integralset_	2048
29.465.4.13	ldim_	2048
29.465.4.14	l1_	2048
29.465.4.15	l2_	2048
29.465.4.16	l1_	2048
29.465.4.17	l2_	2048
29.465.4.18	lcond_	2048
29.465.4.19	lconstraint_	2049
29.465.4.20	pointer_	2049
29.465.4.21	form_dim_	2049
29.465.4.22	pivot_	2049
29.465.4.23	rational_	2049
29.465.4.24	coef_	2049
29.465.4.25	f_	2049
29.465.4.26	l1_	2049

29.465.4.2	29.465.4.2	2049
29.466	Go::SmoothSurfSet Class Reference	2050
29.466.1	Detailed Description	2050
29.466.2	Constructor & Destructor Documentation	2051
29.466.2.1	SmoothSurfSet()	2051
29.466.2.2	SmoothSurfSet(bool copy_coefs)	2051
29.466.2.3	~SmoothSurfSet()	2051
29.466.3	Member Function Documentation	2051
29.466.3.1	approxOrig(double weight)	2051
29.466.3.2	attach(std::vector< shared_ptr< SplineSurface > > &insf, std::vector< std::vector< int > > &coef_known, int num_side_constraints=0, int has_normal_cond=0)	2051
29.466.3.3	equationSolve(std::vector< shared_ptr< SplineSurface > > &surfaces)	2052
29.466.3.4	getBasis(double *sb1, double *sb2, int kk1, int kk2, int kleft1, int kleft2, int ider, double *sbasis)	2052
29.466.3.5	setApproxSideConstraints(std::vector< sideConstraintSet > &constraints, double weight)	2052
29.466.3.6	setLeastSquares(std::vector< std::vector< double > > &pnts, std::vector< std::vector< double > > ¶m_pnts, std::vector< std::vector< double > > &pnt_weights, double weight)	2052
29.466.3.7	setNormalCond(std::vector< std::vector< double > > &pnts, std::vector< std::vector< double > > ¶m_pnts, std::vector< std::vector< double > > &pnt_weights, double weight)	2053
29.466.3.8	setOptimize(double weight1, double weight2, double weight3)	2053
29.466.3.9	setSideConstraints(std::vector< sideConstraintSet > &constraints)	2053
29.466.4	Member Data Documentation	2053
29.466.4.1	coef_array_	2053
29.466.4.2	coef_known_	2054
29.466.4.3	copy_coefs_	2054
29.466.4.4	gmat_	2054
29.466.4.5	gright_	2054
29.466.4.6	ider_	2054
29.466.4.7	dim1_	2054
29.466.4.8	dim_	2054

29.466.4.9	<code>dim_</code>	2054
29.466.4.10	<code>cond_</code>	2054
29.466.4.11	<code>constraint_</code>	2054
29.466.4.12	<code>pivot_</code>	2055
29.466.4.13	<code>dfs_</code>	2055
29.467	<code>Go::SmoothTransition Class Reference</code>	2055
29.467.1	Detailed Description	2056
29.467.2	Constructor & Destructor Documentation	2056
29.467.2.1	<code>SmoothTransition(shared_ptr< const SplineCurve > &inters_crv, shared_ptr< const SplineCurve > &p_crv1, shared_ptr< const SplineCurve > &p_crv2, shared_ptr< const ParamSurface > surf1, shared_ptr< const ParamSurface > surf2, double offset_dist1, double offset_dist2, double epsgeo)</code>	2056
29.467.2.2	<code>~SmoothTransition()</code>	2056
29.467.3	Member Function Documentation	2057
29.467.3.1	<code>approximationOK(double par, const std::vector< Point > &approxpos, double tol1, double tol2)</code>	2057
29.467.3.2	<code>dim()</code>	2057
29.467.3.3	<code>end()</code>	2057
29.467.3.4	<code>eval(double t)</code>	2057
29.467.3.5	<code>eval(double t, int n, std::vector< std::vector< Point > > &der)</code>	2058
29.467.3.6	<code>mbCvs()</code>	2058
29.467.3.7	<code>start()</code>	2058
29.468	<code>Go::SmoothVolume Class Reference</code>	2059
29.468.1	Detailed Description	2059
29.468.2	Constructor & Destructor Documentation	2059
29.468.2.1	<code>SmoothVolume()</code>	2059
29.468.2.2	<code>SmoothVolume(bool copy_coefs)</code>	2059
29.468.2.3	<code>~SmoothVolume()</code>	2059
29.468.3	Member Function Documentation	2060
29.468.3.1	<code>attach(shared_ptr< SplineVolume > &in_vol, std::vector< CoefStatus > status)</code>	2060
29.468.3.2	<code>equationSolve(shared_ptr< SplineVolume > &vol)</code>	2060
29.468.3.3	<code>reset()</code>	2060

29.468.3.4	setLeastSquares(std::vector< double > &pnts, std::vector< double > ¶m← _pnts, std::vector< double > &pnt_weights, const double weight)	2060
29.468.3.5	setOptimize(const double weight1, const double weight2, const double weight3)	2060
29.468.3.6	setPeriodicity(int pardir, int cont, double weight1, double weight2)	2061
29.469	BD_COMMON::SolutionException Class Reference	2061
29.469.1	Detailed Description	2062
29.469.2	Constructor & Destructor Documentation	2062
29.469.2.1	SolutionException(const char *a_what=0)	2062
29.469.3	Member Data Documentation	2062
29.469.3.1	Select	2062
29.470	Go::SolveBCG Class Reference	2063
29.470.1	Detailed Description	2063
29.470.2	Constructor & Destructor Documentation	2064
29.470.2.1	SolveBCG(int conv_type, bool symm)	2064
29.470.2.2	~SolveBCG()	2064
29.470.3	Member Function Documentation	2064
29.470.3.1	precond(double relaxfac)	2064
29.470.3.2	solve(double *ex, double *eb, int nn)	2064
29.471	Go::SolveCG Class Reference	2064
29.471.1	Detailed Description	2065
29.471.2	Constructor & Destructor Documentation	2065
29.471.2.1	SolveCG()	2065
29.471.2.2	~SolveCG()	2066
29.471.3	Member Function Documentation	2066
29.471.3.1	attachMatrix(double *gmat, int nn)	2066
29.471.3.2	forwBack(double *r, double *s)	2066
29.471.3.3	getIndex(int ki, int kj)	2066
29.471.3.4	matrixProduct(RandomIterator1 sx, RandomIterator2 sy)	2066
29.471.3.5	precondRILU(double relaxfac)	2066
29.471.3.6	printPrecond()	2067
29.471.3.7	setMaxIterations(int max_iterations)	2067

29.471.3.8	setTolerance(double tolerance=1.0e-6)	2067
29.471.3.9	solve(double *ex, double *eb, int nn)	2067
29.471.3.10	solveRILU(double *ex, double *eb, int nn)	2068
29.471.3.11	solveStd(double *ex, double *eb, int nn)	2068
29.471.3.12	transposedMatrixProduct(double *sx, double *sy)	2068
29.471.4	Member Data Documentation	2068
29.471.4.1A	_	2068
29.471.4.2	diagonal_	2068
29.471.4.3	diagset_	2068
29.471.4.4	row_	2069
29.471.4.5	col_	2069
29.471.4.6	M_	2069
29.471.4.7	max_ iterations_	2069
29.471.4.8	nn_	2069
29.471.4.9	np_	2069
29.471.4.10	omega_	2069
29.471.4.11	tolerance_	2069
29.472	So::SolveCGCO Class Reference	2070
29.472.1	Detailed Description	2070
29.472.2	Constructor & Destructor Documentation	2071
29.472.2.1	SolveCGCO(int m, int n)	2071
29.472.2.2	~SolveCGCO()	2071
29.472.3	Member Function Documentation	2071
29.472.3.1	precondRILU(double relaxfac)	2071
29.473	SEWMAT::SolvedMatrix Class Reference	2071
29.473.1	Detailed Description	2072
29.473.2	Constructor & Destructor Documentation	2073
29.473.2.1	~SolvedMatrix()	2073
29.473.3	Member Function Documentation	2073
29.473.3.1	BandWidth() const	2073

29.473.3.2	<code>Evaluate(MatrixType mt=MatrixTypeUnSp)</code>	2073
29.473.4	Friends And Related Function Documentation	2073
29.473.4.1	<code>BaseMatrix</code>	2073
29.473.4.2	<code>InvertedMatrix</code>	2073
29.474	<code>Go::SpaceIntCrv</code> Class Reference	2074
29.474.1	Detailed Description	2075
29.474.2	Constructor & Destructor Documentation	2075
29.474.2.1	<code>SpaceIntCrv(shared_ptr< ParamCurve > init_crv, int pdir, std::vector< shared_ptr< CurveOnSurface > > &sfcv1, std::vector< double > start1, std::vector< double > end1, std::vector< shared_ptr< CurveOnSurface > > &sfcv2, std::vector< double > start2, std::vector< double > end2, std::vector< bool > opposite, bool same_orient)</code>	2075
29.474.2.2	<code>~SpaceIntCrv()</code>	2075
29.474.3	Member Function Documentation	2075
29.474.3.1	<code>approximationOK(double par, Point approxpos, double tol1, double tol2) const</code>	2075
29.474.3.2	<code>dim() const</code>	2076
29.474.3.3	<code>end() const</code>	2076
29.474.3.4	<code>eval(double t) const</code>	2076
29.474.3.5	<code>eval(double t, int n, Point der[]) const</code>	2076
29.474.3.6	<code>start() const</code>	2077
29.475	<code>Go::Sphere</code> Class Reference	2077
29.475.1	Detailed Description	2080
29.475.2	Constructor & Destructor Documentation	2080
29.475.2.1	<code>Sphere()</code>	2080
29.475.2.2	<code>Sphere(double radius, Point location, Point z_axis, Point x_axis, bool isSwapped=false)</code>	2080
29.475.2.3	<code>~Sphere()</code>	2080
29.475.3	Member Function Documentation	2080
29.475.3.1	<code>allBoundaryLoops(double degenerate_epsilon=DEFAULT_SPACE_EPSILON) const</code>	2080
29.475.3.2	<code>boundingBox() const</code>	2081
29.475.3.3	<code>classType()</code>	2081
29.475.3.4	<code>clone() const</code>	2081

29.475.3.5	closestBoundaryPoint(const Point &pt, double &clo_u, double &clo_v, Point &clo_pt, double &clo_dist, double epsilon, const RectDomain *rd=NULL, double *seed=0) const	2081
29.475.3.6	closestPoint(const Point &pt, double &clo_u, double &clo_v, Point &clo_pt, double &clo_dist, double epsilon, const RectDomain *domain_of_interest=NULL, double *seed=0) const	2081
29.475.3.7	constParamCurves(double parameter, bool paddir_is_u) const	2082
29.475.3.8	createSplineSurface() const	2082
29.475.3.9	dimension() const	2082
29.475.3.10	geometrySurface() const	2082
29.475.3.11	getBoundaryInfo(Point &pt1, Point &pt2, double epsilon, SplineCurve *&cv, SplineCurve *&crosscv, double knot_tol=1e-05) const	2082
29.475.3.12	getCoordinateAxes(Point &x_axis, Point &y_axis, Point &z_axis) const	2083
29.475.3.13	getDegenerateCorners(std::vector< Point > °_corners, double tol) const	2083
29.475.3.14	getElementaryParamCurve(ElementaryCurve *space_crv, double tol, const Point *start_par_pt=NULL, const Point *end_par_pt=NULL) const	2083
29.475.3.15	getLatitudinalCircle(double vpar) const	2083
29.475.3.16	getLocation() const	2084
29.475.3.17	getLongitudinalCircle(double upar) const	2084
29.475.3.18	getRadius() const	2084
29.475.3.19	getDistanceType() const	2084
29.475.3.20	AxisRotational(Point ¢re, Point &axis, Point &vec, double &angle)	2084
29.475.3.21	isBounded() const	2085
29.475.3.22	isClosed(bool &closed_dir_u, bool &closed_dir_v) const	2085
29.475.3.23	isDegenerate(bool &b, bool &r, bool &t, bool &l, double tolerance) const	2085
29.475.3.24	nextSegmentVal(int dir, double par, bool forward, double tol) const	2085
29.475.3.25	normal(Point &n, double upar, double vpar) const	2086
29.475.3.26	normalCone() const	2086
29.475.3.27	parameterDomain() const	2086
29.475.3.28	point(Point &pt, double upar, double vpar) const	2086
29.475.3.29	point(std::vector< Point > &pts, double upar, double vpar, int derivs, bool u_from_right=true, bool v_from_right=true, double resolution=1.0e-12) const	2087
29.475.3.30	read(std::istream &is)	2087

29.475.3.31	setCoordinateAxes()	2087
29.475.3.32	setParameterBounds(double from_upar, double from_vpar, double to_upar, double to_vpar)	2087
29.475.3.33	subSurface(double from_upar, double from_vpar, double to_upar, double to_vpar, double fuzzy=DEFAULT_PARAMETER_EPSILON) const	2088
29.475.3.34	subSurfaces(double from_upar, double from_vpar, double to_upar, double to_vpar, double fuzzy=DEFAULT_PARAMETER_EPSILON) const	2088
29.475.3.35	tangentCone(bool pardir_is_u) const	2088
29.475.3.36	write(std::ostream &os) const	2088
29.475.4	Member Data Documentation	2089
29.475.4.1	domain_	2089
29.475.4.2	location_	2089
29.475.4.3	orientedDomain_	2089
29.475.4.4	radius_	2089
29.475.4.5	x_axis_	2089
29.475.4.6	y_axis_	2089
29.475.4.7	z_axis_	2089
29.476	Go::SphereInt Class Reference	2090
29.476.1	Detailed Description	2091
29.476.2	Constructor & Destructor Documentation	2091
29.476.2.1	SphereInt()	2091
29.476.2.2	SphereInt(Point center, double radius)	2091
29.476.2.3	~SphereInt()	2091
29.476.3	Member Function Documentation	2091
29.476.3.1	center() const	2091
29.476.3.2	radius() const	2091
29.476.3.3	read(std::istream &is)	2091
29.476.3.4	surface() const	2092
29.477	Go::SphereVolume Class Reference	2092
29.477.1	Detailed Description	2094
29.477.2	Constructor & Destructor Documentation	2094
29.477.2.1	SphereVolume()	2094

29.477.2.2	SphereVolume(double radius, Point location, Point z_axis, Point x_axis)	2094
29.477.2.3	~SphereVolume()	2094
29.477.3	Member Function Documentation	2094
29.477.3.1	boundingBox() const	2094
29.477.3.2	classType()	2095
29.477.3.3	clone() const	2095
29.477.3.4	closestPoint(const Point &pt, double &clo_u, double &clo_v, double &clo_w, Point &clo_pt, double &clo_dist, double epsilon, double *seed=0) const	2095
29.477.3.5	dimension() const	2095
29.477.3.6	geometryVolume() const	2095
29.477.3.7	getAllBoundarySurfaces() const	2096
29.477.3.8	instanceType() const	2096
29.477.3.9	nextSegmentVal(int dir, double par, bool forward, double tol) const	2096
29.477.3.10	parameterSpan() const	2096
29.477.3.11	point(Point &pt, double upar, double vpar, double wpar) const	2096
29.477.3.12	point(std::vector< Point > &pts, double upar, double vpar, double wpar, int derivs, bool u_from_right=true, bool v_from_right=true, bool w_from_right=true, double resolution=1.0e-12) const	2097
29.477.3.13	read(std::istream &is)	2097
29.477.3.14	reverseParameterDirection(int pardir)	2098
29.477.3.15	swapParameterDirection(int pardir1, int pardir2)	2098
29.477.3.16	tangentCone(int pardir) const	2098
29.477.3.17	translate(const Point &vec)	2098
29.477.3.18	write(std::ostream &os) const	2098
29.478	Go::Spline1FunctionInt Class Reference	2099
29.478.1	Detailed Description	2100
29.478.2	Constructor & Destructor Documentation	2100
29.478.2.1	Spline1FunctionInt(shared_ptr< ParamCurve > curve)	2100
29.478.2.2	Spline1FunctionInt(shared_ptr< ParamCurve > curve, ParamFunctionInt *parent)	2101
29.478.2.3	~Spline1FunctionInt()	2101
29.478.3	Member Function Documentation	2101
29.478.3.1	getCriticalValsAndKnots(int pardir) const	2101

29.478.3.2	getInnerKnotVals(int pdir, bool sort=false) const	2101
29.478.3.3	getMesh()	2102
29.478.3.4	getMeshSize(int dir)	2102
29.478.3.5	hasCriticalValsOrKnots(int pdir) const	2102
29.478.3.6	hasInnerKnots(int pdir) const	2102
29.478.3.7	knotIntervalFuzzy(double &t, double tol) const	2102
29.478.3.8	makeIntFunction(shared_ptr< ParamCurve > curve)	2103
29.478.3.9	monotone(Point &dir, double tol=1.0e-15) const	2103
29.478.3.10	nextSegmentVal(double par, bool forward) const	2103
29.478.3.11	paramFromMesh(int dir, int idx)	2104
29.478.4	Member Data Documentation	2104
29.478.4.1	spcv_	2104
29.479	Go::Spline2FunctionInt Class Reference	2104
29.479.1	Detailed Description	2106
29.479.2	Constructor & Destructor Documentation	2106
29.479.2.1	Spline2FunctionInt(shared_ptr< SplineSurface > surf)	2106
29.479.2.2	Spline2FunctionInt(shared_ptr< SplineSurface > surf, Param2FunctionInt *parent)	2106
29.479.2.3	~Spline2FunctionInt()	2107
29.479.3	Member Function Documentation	2107
29.479.3.1	createGradSurface() const	2107
29.479.3.2	endParam(int pdir) const	2107
29.479.3.3	getBoundaryObjects(std::vector< shared_ptr< BoundaryFunctionInt > > &bd← _objs)	2107
29.479.3.4	getCriticalValsAndKnots(int pdir) const	2107
29.479.3.5	getInnerKnotVals(int pdir, bool sort=false) const	2108
29.479.3.6	hasCriticalValsOrKnots(int pdir) const	2108
29.479.3.7	hasInnerKnots(int pdir) const	2108
29.479.3.8	knotIntervalFuzzy(int pdir, double &t, double tol) const	2108
29.479.3.9	makeIntFunction(shared_ptr< ParamSurface > surf)	2109
29.479.3.10	monotone(Point &dir, double tol=1.0e-15) const	2109
29.479.3.11	nextSegmentVal(int pdir, double par, bool forward) const	2109

29.479.3.1	startParam(int pdir) const	2109
29.479.3.1	surface3D()	2110
29.479.4	Member Data Documentation	2110
29.479.4.1	spsf_	2110
29.480	Go::SplineApproximator Class Reference	2110
29.480.1	Detailed Description	2111
29.480.2	Constructor & Destructor Documentation	2111
29.480.2.1	SplineApproximator()	2111
29.480.2.2	~SplineApproximator()	2111
29.480.3	Member Function Documentation	2112
29.480.3.1	basis()	2112
29.480.3.2	Interpolate(int num_points, int dimension, const double *param_start, const double *data_start, std::vector< double > &coefs)	2112
29.480.3.3	setNumCoefs(int num)	2112
29.480.3.4	setSplineSpace(const BsplineBasis &basis)	2112
29.481	Go::SplineCurve Class Reference	2113
29.481.1	Detailed Description	2116
29.481.2	Constructor & Destructor Documentation	2116
29.481.2.1	SplineCurve()	2116
29.481.2.2	SplineCurve(int number, int order, RandomIterator1 knotstart, RandomIterator2 coefsstart, int dim, bool rational=false)	2116
29.481.2.3	SplineCurve(const BsplineBasis &basis, RandomIterator coefsstart, int dim, bool rational=false)	2116
29.481.2.4	SplineCurve(const Point &pnt1, const Point &pnt2)	2117
29.481.2.5	SplineCurve(const Point &pnt1, double startpar, const Point &pnt2, double endpar)	2117
29.481.2.6	~SplineCurve()	2117
29.481.3	Member Function Documentation	2117
29.481.3.1	appendCurve(ParamCurve *other_curve, int continuity, double &dist, bool repar=true)	2117
29.481.3.2	appendCurve(ParamCurve *cv, bool repar=true)	2118
29.481.3.3	basis() const	2118
29.481.3.4	basis()	2118

29.481.3.5	<code>boundingBox() const</code>	2119
29.481.3.6	<code>checkElementaryCurve()</code>	2119
29.481.3.7	<code>classType()</code>	2119
29.481.3.8	<code>clone() const</code>	2119
29.481.3.9	<code>closestPoint(const Point &pt, double tmin, double tmax, double &clo_t, Point &clo_pt, double &clo_dist, double const *seed=0) const</code>	2119
29.481.3.10	<code>defs_begin()</code>	2119
29.481.3.11	<code>defs_begin() const</code>	2120
29.481.3.12	<code>defs_end()</code>	2120
29.481.3.13	<code>defs_end() const</code>	2120
29.481.3.14	<code>compositeBox() const</code>	2120
29.481.3.15	<code>computeBasis(double param, std::vector< double > &basisValues, std::vector< double > &basisDerivs) const</code>	2120
29.481.3.16	<code>deform(const std::vector< double > &vec, int vdim=0)</code>	2121
29.481.3.17	<code>derivCurve(int ider) const</code>	2121
29.481.3.18	<code>dimension() const</code>	2121
29.481.3.19	<code>directionCone() const</code>	2121
29.481.3.20	<code>endparam() const</code>	2121
29.481.3.21	<code>equalBdWeights(bool at_start)</code>	2122
29.481.3.22	<code>geometryCurve()</code>	2122
29.481.3.23	<code>getElementaryCurve()</code>	2122
29.481.3.24	<code>getWeights(std::vector< double > &weights) const</code>	2122
29.481.3.25	<code>gridEvaluator(std::vector< double > &points, const std::vector< double > &par) const</code>	2122
29.481.3.26	<code>insertKnot(double apar)</code>	2122
29.481.3.27	<code>insertKnot(const std::vector< double > &new_knots)</code>	2122
29.481.3.28	<code>instanceType() const</code>	2123
29.481.3.29	<code>interpolate(Interpolator &interpolator, int num_points, int dim, const double *param_start, const double *data_start)</code>	2123
29.481.3.30	<code>AxisRotational(Point &centre, Point &axis, Point &vec, double &angle)</code>	2123
29.481.3.31	<code>Degenerate(double degenerate_epsilon)</code>	2123
29.481.3.32	<code>ElementaryCurve()</code>	2124

29.481.3.33	InPlane(const Point &loc, const Point &axis, double eps, Point &normal) const	2124
29.481.3.34	InPlane(const Point &norm, double eps, Point &pos) const	2124
29.481.3.35	Linear(Point &dir, double tol)	2124
29.481.3.36	KnotsBegin()	2124
29.481.3.37	KnotsBegin() const	2124
29.481.3.38	KnotsEnd()	2125
29.481.3.39	KnotsEnd() const	2125
29.481.3.40	Length(double tol)	2125
29.481.3.41	MakeBernsteinKnots()	2125
29.481.3.42	MakeKnotEndRegular()	2125
29.481.3.43	MakeKnotStartRegular()	2125
29.481.3.44	NextSegmentVal(double par, bool forward, double tol) const	2125
29.481.3.45	NumCoefs() const	2126
29.481.3.46	NumElem() const	2126
29.481.3.47	Order() const	2126
29.481.3.48	Point(Point &pt, double tpar) const	2126
29.481.3.49	Point(std::vector< Point > &pts, double tpar, int derivs, bool from_right=true) const	2127
29.481.3.50	RaiseOrder(int i=1)	2127
29.481.3.51	Rational() const	2127
29.481.3.52	Coefs_begin()	2127
29.481.3.53	Coefs_begin() const	2128
29.481.3.54	Coefs_end()	2128
29.481.3.55	Coefs_end() const	2128
29.481.3.56	Read(std::istream &is)	2128
29.481.3.57	RemoveKnot(double tpar)	2128
29.481.3.58	ReplaceEndPoint(Point pnt, bool at_start)	2129
29.481.3.59	PresentAsRational()	2129
29.481.3.60	ReverseParameterDirection(bool switchparam=false)	2129
29.481.3.61	SetBdWeight(double wgt, bool at_start)	2129
29.481.3.62	SetElementaryCurve(shared_ptr< ElementaryCurve > elcurve)	2129

29.481.3.63	GetParameterInterval(double t1, double t2)	2129
29.481.3.64	Split(std::vector< double > ¶m, double fuzzy=DEFAULT_PARAMETER_EPSILON) const	2130
29.481.3.65	Split(double param, double fuzzy=DEFAULT_PARAMETER_EPSILON) const	2130
29.481.3.66	Startparam() const	2130
29.481.3.67	SubCurve(double from_par, double to_par, double fuzzy=DEFAULT_PARAMETER_EPSILON) const	2130
29.481.3.68	Swap(SplineCurve &other)	2131
29.481.3.69	TranslateCurve(const Point &dir)	2131
29.481.3.70	TranslateSwapCurve(const Point &dir, double sgn, int pdir)	2131
29.481.3.71	UpdateCoefsFromRcoefs()	2131
29.481.3.72	Write(std::ostream &os) const	2131
29.482.0	So::SplineCurveInt Class Reference	2131
29.482.1	Detailed Description	2133
29.482.2	Constructor & Destructor Documentation	2133
29.482.2.1	SplineCurveInt(shared_ptr< ParamCurve > curve)	2133
29.482.2.2	SplineCurveInt(shared_ptr< ParamCurve > curve, ParamGeomInt *parent)	2133
29.482.2.3	~SplineCurveInt()	2134
29.482.3	Member Function Documentation	2134
29.482.3.1	checkPeriodicity(int pardir=0) const	2134
29.482.3.2	getCriticalValsAndKnots(int pardir) const	2134
29.482.3.3	getInnerKnotVals(int pardir, bool sort=false) const	2134
29.482.3.4	getMesh()	2135
29.482.3.5	getMeshSize(int dir)	2135
29.482.3.6	getOptimizedConeAngle(Point &axis1, Point &axis2)	2135
29.482.3.7	getRotatedBox(std::vector< Point > &axis) const	2135
29.482.3.8	getSpline()	2136
29.482.3.9	hasCriticalValsOrKnots(int pardir) const	2136
29.482.3.10	hasInnerKnots(int pardir) const	2136
29.482.3.11	isSpline()	2136
29.482.3.12	isNotIntervalFuzzy(double &t, double tol) const	2137

29.482.3.13	MakeIntObject(shared_ptr< ParamCurve > curve)	2137
29.482.3.14	NextSegmentVal(double par, bool forward, double tol) const	2137
29.482.3.15	ParamFromMesh(int dir, int idx)	2137
29.482.4	Member Data Documentation	2138
29.482.4.1	ispcv_	2138
29.483	SplineCurvePropertySheet Class Reference	2138
29.483.1	Detailed Description	2139
29.483.2	Constructor & Destructor Documentation	2139
29.483.2.1	SplineCurvePropertySheet(Go::CurveTesselator *tess, gvCurvePaintable *pable)	2139
29.483.3	Member Function Documentation	2139
29.483.3.1	apply	2139
29.483.3.2	createSheet(QWidget *parent, gvObserver *obs)	2139
29.484	Go::SplineInterpolator Class Reference	2140
29.484.1	Detailed Description	2141
29.484.2	Member Enumeration Documentation	2141
29.484.2.1	CondType	2141
29.484.3	Constructor & Destructor Documentation	2142
29.484.3.1	SplineInterpolator()	2142
29.484.3.2	~SplineInterpolator()	2142
29.484.4	Member Function Documentation	2142
29.484.4.1	basis()	2142
29.484.4.2	getCondType()	2142
29.484.4.3	Interpolate(const std::vector< double > ¶ms, const std::vector< double > &points, const std::vector< int > &tangent_index, const std::vector< double > &tangent_points, int order, std::vector< double > &coefs)	2142
29.484.4.4	Interpolate(const std::vector< double > ¶ms, const std::vector< double > &points, const std::vector< int > &tangent_index, const std::vector< double > &tangent_points, std::vector< double > &coefs)	2143
29.484.4.5	Interpolate(int num_points, int dimension, const double *param_start, const double *data_start, std::vector< double > &coefs)	2143
29.484.4.6	makeBasis(const std::vector< double > ¶ms, const std::vector< int > &tangent_index, int order)	2143
29.484.4.7	setBasis(const BsplineBasis &basis)	2143

29.484.4.8	<code>setEndTangents(shared_ptr< Point > &start_tangent, shared_ptr< Point > &end_tangent)</code>	2143
29.484.4.9	<code>setFreeConditions()</code>	2144
29.484.4.10	<code>setHermiteConditions(const Point &start_tangent, const Point &end_tangent)</code>	2144
29.484.4.11	<code>setNaturalConditions()</code>	2144
29.484.4.12	<code>setNaturalEndCondition()</code>	2144
29.484.4.13	<code>setNaturalStartCondition()</code>	2144
29.485.0	<code>Go::SplineSurface Class Reference</code>	2145
29.485.1	Detailed Description	2150
29.485.2	Member Typedef Documentation	2151
29.485.2.1	<code>IDmatrix</code>	2151
29.485.2.2	<code>2Dvector</code>	2151
29.485.3	Member Enumeration Documentation	2151
29.485.3.1	<code>NormalConeMethod</code>	2151
29.485.4	Constructor & Destructor Documentation	2151
29.485.4.1	<code>SplineSurface()</code>	2151
29.485.4.2	<code>SplineSurface(int number1, int number2, int order1, int order2, RandomIterator1 knot1start, RandomIterator2 knot2start, RandomIterator3 coefsstart, int dim, bool rational=false)</code>	2151
29.485.4.3	<code>SplineSurface(const BsplineBasis &basis_u, const BsplineBasis &basis_v, RandomIterator coefsstart, int dim, bool rational=false)</code>	2152
29.485.4.4	<code>~SplineSurface()</code>	2152
29.485.5	Member Function Documentation	2152
29.485.5.1	<code>add(const SplineSurface *other, double tol=1.0e-10)</code>	2152
29.485.5.2	<code>allBoundaryLoops(double degenerate_epsilon=DEFAULT_SPACE_EPSILON) const</code>	2153
29.485.5.3	<code>appendSurface(ParamSurface *sf, int join_dir, int continuity, double &dist, bool repar=true)</code>	2153
29.485.5.4	<code>appendSurface(ParamSurface *sf, int join_dir, bool repar=true)</code>	2153
29.485.5.5	<code>area(double tol) const</code>	2154
29.485.5.6	<code>asSplineSurface()</code>	2154
29.485.5.7	<code>basis(int i) const</code>	2154
29.485.5.8	<code>basis_u() const</code>	2154

29.485.5.9	basis_u()	2155
29.485.5.10	basis_v() const	2155
29.485.5.11	basis_v()	2155
29.485.5.12	boundaryIndex(Point ¶m_pt1, Point ¶m_pt2) const	2155
29.485.5.13	boundingBox() const	2156
29.485.5.14	checkElementarySurface()	2156
29.485.5.15	classType()	2156
29.485.5.16	done() const	2156
29.485.5.17	closestBoundaryPoint(const Point &pt, double &clo_u, double &clo_v, Point &clo_pt, double &clo_dist, double epsilon, const RectDomain *rd=NULL, double *seed=0) const	2156
29.485.5.18	closestInDomain(double u, double v) const	2156
29.485.5.19	closestPoint(const Point &pt, double &clo_u, double &clo_v, Point &clo_pt, double &clo_dist, double epsilon, const RectDomain *domain_of_interest=NULL, double *seed=0) const	2156
29.485.5.20	defs_begin()	2157
29.485.5.21	defs_begin() const	2157
29.485.5.22	defs_end()	2157
29.485.5.23	defs_end() const	2158
29.485.5.24	compositeBox() const	2158
29.485.5.25	computeBasis(double param[], std::vector< double > &basisValues, std::vector< double > &basisDerivs_u, std::vector< double > &basisDerivs_v, bool evaluate_from_right=true) const	2158
29.485.5.26	computeBasis(const std::vector< double >::const_iterator &bas_vals_u, const std::vector< double >::const_iterator &bas_vals_v, int left_u, int left_v, std::vector< double > &basisValues, std::vector< double > &basisDerivs_u, std::vector< double > &basisDerivs_v) const	2158
29.485.5.27	computeBasis(double param_u, double param_v, BasisPtsSf &result) const	2159
29.485.5.28	computeBasis(double param_u, double param_v, BasisDerivsSf &result, bool evaluate_from_right=true) const	2159
29.485.5.29	computeBasis(double param_u, double param_v, BasisDerivsSf2 &result, bool evaluate_from_right=true) const	2159
29.485.5.30	computeBasisGrid(const Dvector ¶m_u, const Dvector ¶m_v, Dmatrix &basisValues) const	2159
29.485.5.31	computeBasisGrid(const Dvector ¶m_u, const Dvector ¶m_v, std::vector< BasisPtsSf > &result) const	2160

29.485.5.32	computeBasisGrid(const Dvector ¶m_u, const Dvector ¶m_v, Dmatrix &basisValues, Dmatrix &basisDerivs_u, Dmatrix &basisDerivs_v, bool evaluate_← _from_right=true) const	2160
29.485.5.33	computeBasisGrid(const Dvector ¶m_u, const Dvector ¶m_v, std::← :vector< BasisDerivsSf > &result, bool evaluate_from_right=true) const	2160
29.485.5.34	computeBasisGrid(const Dvector ¶m_u, const Dvector ¶m_v, std::← :vector< BasisDerivsSf2 > &result, bool evaluate_from_right=true) const	2160
29.485.5.35	ConstParamCurve(double parameter, bool paddir_is_u) const	2160
29.485.5.36	ConstParamCurve(double parameter, bool paddir_is_u, SplineCurve *&cv, ← SplineCurve *&crosscv) const	2161
29.485.5.37	ConstParamCurves(double parameter, bool paddir_is_u) const	2161
29.485.5.38	ContainingDomain() const	2161
29.485.5.39	Ctrl_begin()	2162
29.485.5.40	Ctrl_begin() const	2162
29.485.5.41	Ctrl_end()	2162
29.485.5.42	Ctrl_end() const	2162
29.485.5.43	Deform(const std::vector< double > &vec, int vdim=0)	2162
29.485.5.44	DerivSurface(int ider1, int ider2) const	2162
29.485.5.45	Dimension() const	2163
29.485.5.46	EdgeCurve(int ccw_edge_number) const	2163
29.485.5.47	ElementBoundaryStatus(int elem_ix, double eps)	2163
29.485.5.48	ElementOnBoundary(int elem_ix, double eps)	2164
29.485.5.49	Endparam(int idx) const	2164
29.485.5.50	Endparam_u() const	2164
29.485.5.51	Endparam_v() const	2164
29.485.5.52	EvalGrid(int num_u, int num_v, double umin, double umax, double vmin, double ← vmax, std::vector< double > &points, double nodata_val=-9999) const	2165
29.485.5.53	GetBoundaryIdx(Point &pt1, Point &pt2, double epsilon, int &bdindex, double ← &par1, double &par2, double knot_tol=1e-05) const	2165
29.485.5.54	GetBoundaryIdx(Point &pt1, double epsilon, int &bdindex, double knot_tol=1e-05) ← const	2165
29.485.5.55	GetBoundaryInfo(Point &pt1, Point &pt2, double epsilon, SplineCurve *&cv, ← SplineCurve *&crosscv, double knot_tol=1e-05) const	2166
29.485.5.56	GetBoundaryInfo(double par1, double par2, int bdindex, SplineCurve *&cv, ← SplineCurve *&crosscv, double knot_tol=1e-05) const	2166

29.485.5.57	<code>getConstParamCurves(const std::vector< double > &params_u, const std::vector< double > &params_v, std::vector< shared_ptr< SplineCurve > > &curves_u, std::vector< shared_ptr< SplineCurve > > &curves_v)</code>	2166
29.485.5.58	<code>getCornerPoints(std::vector< std::pair< Point, Point > > &corners) const</code>	2167
29.485.5.59	<code>DegenerateCorners(std::vector< Point > &deg_corners, double tol) const</code>	2167
29.485.5.60	<code>ElementarySurface()</code>	2167
29.485.5.61	<code>getElementBdParCvs(int elem_ix, double elem_par[])</code>	2167
29.485.5.62	<code>getWeights(std::vector< double > &weights) const</code>	2167
29.485.5.63	<code>GridEvaluator(int num_u, int num_v, std::vector< double > &points, std::vector< double > &normals, std::vector< double > &param_u, std::vector< double > &param_v, bool normalize=true) const</code>	2167
29.485.5.64	<code>GridEvaluator(int num_u, int num_v, std::vector< double > &points, std::vector< double > &param_u, std::vector< double > &param_v) const</code>	2168
29.485.5.65	<code>GridEvaluator(int num_u, int num_v, std::vector< double > &points, std::vector< double > &param_u, std::vector< double > &param_v, double start_u, double end_u, double start_v, double end_v) const</code>	2168
29.485.5.66	<code>GridEvaluator(std::vector< double > &points, const std::vector< double > &param_u, const std::vector< double > &param_v) const</code>	2169
29.485.5.67	<code>GridEvaluator(const std::vector< double > &params_u, const std::vector< double > &params_v, std::vector< double > &points, std::vector< double > &derivs_u, std::vector< double > &derivs_v, bool evaluate_from_right=true) const</code>	2169
29.485.5.68	<code>Domain(double u, double v, double eps=1.0e-4) const</code>	2169
29.485.5.69	<code>Domain2(double u, double v, double eps=1.0e-4) const</code>	2169
29.485.5.70	<code>insertKnot_u(double apar)</code>	2169
29.485.5.71	<code>insertKnot_u(const std::vector< double > &new_knots)</code>	2170
29.485.5.72	<code>insertKnot_v(double apar)</code>	2170
29.485.5.73	<code>insertKnot_v(const std::vector< double > &new_knots)</code>	2170
29.485.5.74	<code>InstanceType() const</code>	2170
29.485.5.75	<code>Interpolate(Interpolator &interpolator1, Interpolator &interpolator2, int num_points1, int num_points2, int dim, const double *param1_start, const double *param2_start, const double *data_start)</code>	2170
29.485.5.76	<code>AxisRotational(Point &centre, Point &axis, Point &vec, double &angle)</code>	2171
29.485.5.77	<code>Degenerate(bool &b, bool &r, bool &t, bool &l, double tolerance) const</code>	2171
29.485.5.78	<code>ElementarySurface()</code>	2171
29.485.5.79	<code>Planar(Point &normal, double tol)</code>	2171
29.485.5.80	<code>Spline() const</code>	2172

29.485.5.84	<code>KnottedSpan(int pdir, int iknot) const</code>	2172
29.485.5.85	<code>MakeBernsteinKnotsU()</code>	2172
29.485.5.86	<code>MakeBernsteinKnotsV()</code>	2172
29.485.5.87	<code>MakeSurfaceKRegular()</code>	2172
29.485.5.88	<code>MirrorSurface(const Point &pos, const Point &norm) const</code>	2172
29.485.5.89	<code>NextSegmentVal(int dir, double par, bool forward, double tol) const</code>	2172
29.485.5.90	<code>Normal(Point &n, double upar, double vpar) const</code>	2173
29.485.5.91	<code>Normal() const</code>	2173
29.485.5.92	<code>NormalCone(NormalConeMethod method) const</code>	2173
29.485.5.93	<code>NormalCone() const</code>	2173
29.485.5.94	<code>NormalSurface() const</code>	2174
29.485.5.95	<code>NumberOfPatches_u() const</code>	2174
29.485.5.96	<code>NumberOfPatches_v() const</code>	2174
29.485.5.97	<code>NumCoefs_u() const</code>	2174
29.485.5.98	<code>NumCoefs_v() const</code>	2174
29.485.5.99	<code>NumElem() const</code>	2175
29.485.5.100	<code>NumElem(int pdir) const</code>	2175
29.485.5.101	<code>OnBoundary(double u, double v, double eps=1.0e-4) const</code>	2175
29.485.5.102	<code>Order_u() const</code>	2175
29.485.5.103	<code>Order_v() const</code>	2175
29.485.5.104	<code>OuterBoundaryLoop(double degenerate_epsilon=DEFAULT_SPACE_EPSILON) const</code>	2175
29.485.5.105	<code>ParameterDomain() const</code>	2176
29.485.5.106	<code>Point(Point &pt, double upar, double vpar) const</code>	2176
29.485.5.107	<code>Point(std::vector< Point > &pts, double upar, double vpar, int derivs, bool u_from_right=true, bool v_from_right=true, double resolution=1.0e-12) const</code>	2176
29.485.5.108	<code>PointsGrid(int m1, int m2, int derivs, const double *basisvals1, const double *basisvals2, const int *knotint1, const int *knotint2, double *result, double *normals=0) const</code>	2177
29.485.5.109	<code>RaiseOrder(int raise_u, int raise_v)</code>	2177
29.485.5.110	<code>Rational() const</code>	2177
29.485.5.111	<code>Refs_begin()</code>	2177

29.485.5.109	<code>coefs_begin() const</code>	2178
29.485.5.110	<code>coefs_end()</code>	2178
29.485.5.111	<code>coefs_end() const</code>	2178
29.485.5.112	<code>read(std::istream &is)</code>	2178
29.485.5.113	<code>removeKnot_u(double tpar)</code>	2178
29.485.5.114	<code>removeKnot_v(double tpar)</code>	2179
29.485.5.115	<code>replaceBoundaryCurve(int bd_nmb, shared_ptr< SplineCurve > bd_crv, bool unify=true)</code>	2179
29.485.5.116	<code>replaceCoefficient(int ix, Point coef)</code>	2179
29.485.5.117	<code>representAsRational()</code>	2179
29.485.5.118	<code>reverseParameterDirection(bool direction_is_u)</code>	2179
29.485.5.119	<code>setAvBdWeight(double wgt, int pardir, bool at_start)</code>	2179
29.485.5.120	<code>ElementarySurface(shared_ptr< ElementarySurface > elsurf)</code>	2180
29.485.5.121	<code>ParameterDomain(double u1, double u2, double v1, double v2)</code>	2180
29.485.5.122	<code>partparam(int idx) const</code>	2180
29.485.5.123	<code>partparam_u() const</code>	2180
29.485.5.124	<code>partparam_v() const</code>	2180
29.485.5.125	<code>Surface(double from_upar, double from_vpar, double to_upar, double to_vpar, double fuzzy=DEFAULT_PARAMETER_EPSILON) const</code>	2180
29.485.5.126	<code>Surfaces(double from_upar, double from_vpar, double to_upar, double to_vpar, double fuzzy=DEFAULT_PARAMETER_EPSILON) const</code>	2181
29.485.5.127	<code>wrap(SplineSurface &other)</code>	2181
29.485.5.128	<code>wrapParameterDirection()</code>	2181
29.485.5.129	<code>wrapCone(bool pardir_is_u) const</code>	2181
29.485.5.130	<code>wrapOrientation()</code>	2182
29.485.5.131	<code>write(std::ostream &os) const</code>	2182
29.486	Go::SplineSurfaceInt Class Reference	2182
29.486.1	Detailed Description	2184
29.486.2	Constructor & Destructor Documentation	2184
29.486.2.1	<code>SplineSurfaceInt(shared_ptr< ParamSurface > surf)</code>	2184
29.486.2.2	<code>SplineSurfaceInt(shared_ptr< ParamSurface > surf, ParamGeomInt *parent)</code>	2185
29.486.2.3	<code>~SplineSurfaceInt()</code>	2185

29.486.3 Member Function Documentation	2185
29.486.3.1 canImplicitize()	2185
29.486.3.2 checkPeriodicity(int pdir) const	2185
29.486.3.3 constParamCurve(double parameter, bool pdir_is_u) const	2186
29.486.3.4 directionCone() const	2186
29.486.3.5 getBoundaryObjects(std::vector< shared_ptr< BoundaryGeomInt > > &bd_objs)	2186
29.486.3.6 getCriticalValsAndKnots(int pdir) const	2186
29.486.3.7 getImplicit(double tol, double &tol2, AlgObj3DInt &alg_obj_3d_int)	2187
29.486.3.8 getInnerKnotVals(int pdir, bool sort=false) const	2187
29.486.3.9 getMesh()	2187
29.486.3.10 getMeshSize(int dir)	2187
29.486.3.11 getNormalSurface() const	2188
29.486.3.12 getOptimizedConeAngle(Point &axis1, Point &axis2)	2188
29.486.3.13 getRotatedBox(std::vector< Point > &axis) const	2188
29.486.3.14 getSplineSurface()	2189
29.486.3.15 hasCriticalValsOrKnots(int pdir) const	2189
29.486.3.16 hasInnerKnots(int pdir) const	2189
29.486.3.17 implicitize(double tol)	2189
29.486.3.18 IsoParametric(ParamCurveInt *curve, int dir, double par, double ptol, double tol)	2190
29.486.3.19 Simple()	2190
29.486.3.20 Spline()	2190
29.486.3.21 knotIntervalFuzzy(double &u, double &v, double utol, double vtol) const	2190
29.486.3.22 makeIntCurve(shared_ptr< ParamCurve > crv, ParamGeomInt *parent)	2190
29.486.3.23 makeIntObject(shared_ptr< ParamSurface > surf)	2191
29.486.3.24 nextSegmentVal(int dir, double par, bool forward, double tol) const	2191
29.486.3.25 paramFromMesh(int dir, int idx)	2191
29.486.3.26 reducedDirectionCone(bool reduce_at_bd[4], double epsge) const	2192
29.486.3.27 setImplicitDeg()	2192
29.486.3.28 splineSurface() const	2192
29.486.3.29 splitAtG0(double angtol, std::vector< shared_ptr< ParamSurfaceInt > > &subG1)	2192

29.486.4	Member Data Documentation	2192
29.486.4.1	normalsf_	2192
29.486.4.2	spfsf_	2193
29.487	SplineVolume Class Reference	2193
29.487.1	Detailed Description	2197
29.487.2	Member Typedef Documentation	2197
29.487.2.1	Dmatrix	2197
29.487.2.2	Dvector	2197
29.487.3	Constructor & Destructor Documentation	2198
29.487.3.1	SplineVolume()	2198
29.487.3.2	SplineVolume(int number1, int number2, int number3, int order1, int order2, int order3, RandomIterator1 knot1start, RandomIterator2 knot2start, RandomIterator3 knot3start, RandomIterator4 coefsstart, int dim, bool rational=false)	2198
29.487.3.3	SplineVolume(const BsplineBasis &basis_u, const BsplineBasis &basis_v, const BsplineBasis &basis_w, RandomIterator coefsstart, int dim, bool rational=false)	2199
29.487.3.4	~SplineVolume()	2199
29.487.4	Member Function Documentation	2199
29.487.4.1	add(const SplineVolume *other, double tol=1.0e-10)	2199
29.487.4.2	appendVolume(SplineVolume *vol, int join_dir, int cont, bool repara=true)	2200
29.487.4.3	asSplineVolume()	2200
29.487.4.4	basis(int pdir) const	2200
29.487.4.5	boundingBox() const	2200
29.487.4.6	checkDegeneracy(double tol, int is_degenerate[]) const	2200
29.487.4.7	classType()	2201
29.487.4.8	clone() const	2201
29.487.4.9	closestCorner(const Point &pt, double &upar, double &vpar, double &wpar, Point &corner, double &dist) const	2201
29.487.4.10	closestPoint(const Point &pt, double &clo_u, double &clo_v, double &clo_w, Point &clo_pt, double &clo_dist, double epsilon, double *seed=0) const	2201
29.487.4.11	coefs_begin()	2202
29.487.4.12	coefs_begin() const	2202
29.487.4.13	coefs_end()	2202
29.487.4.14	coefs_end() const	2202

29.487.4.15	<code>computeBasis(double param[], std::vector< double > &basisValues, std::vector< double > &basisDerivs_u, std::vector< double > &basisDerivs_v, std::vector< double > &basisDerivs_w, bool evaluate_from_right=true) const</code>	2202
29.487.4.16	<code>computeBasis(const std::vector< double >::const_iterator &bas_vals_u, const std::vector< double >::const_iterator &bas_vals_v, const std::vector< double >::const_iterator &bas_vals_w, int left_u, int left_v, int left_w, std::vector< double > &basisValues, std::vector< double > &basisDerivs_u, std::vector< double > &basisDerivs_v, std::vector< double > &basisDerivs_w) const</code>	2203
29.487.4.17	<code>computeBasis(double param_u, double param_v, double param_w, BasisPts &result) const</code>	2203
29.487.4.18	<code>computeBasis(double param_u, double param_v, double param_w, BasisDerivs &result, bool evaluate_from_right=true) const</code>	2203
29.487.4.19	<code>computeBasis(double param_u, double param_v, double param_w, BasisDerivs2 &result, bool evaluate_from_right=true) const</code>	2203
29.487.4.20	<code>computeBasisGrid(const Dvector &param_u, const Dvector &param_v, const Dvector &param_w, Dmatrix &basisValues) const</code>	2203
29.487.4.21	<code>computeBasisGrid(const Dvector &param_u, const Dvector &param_v, const Dvector &param_w, std::vector< BasisPts > &result) const</code>	2203
29.487.4.22	<code>computeBasisGrid(const Dvector &param_u, const Dvector &param_v, const Dvector &param_w, Dmatrix &basisValues, Dmatrix &basisDerivs_u, Dmatrix &basisDerivs_v, Dmatrix &basisDerivs_w, bool evaluate_from_right=true) const</code>	2203
29.487.4.23	<code>computeBasisGrid(const Dvector &param_u, const Dvector &param_v, const Dvector &param_w, std::vector< BasisDerivs > &result, bool evaluate_from_right=true) const</code>	2204
29.487.4.24	<code>computeBasisGrid(const Dvector &param_u, const Dvector &param_v, const Dvector &param_w, std::vector< BasisDerivs2 > &result, bool evaluate_from_right=true) const</code>	2204
29.487.4.25	<code>constParamSurface(double parameter, int pardir) const</code>	2204
29.487.4.26	<code>trl_begin()</code>	2204
29.487.4.27	<code>trl_begin() const</code>	2204
29.487.4.28	<code>trl_end()</code>	2205
29.487.4.29	<code>trl_end() const</code>	2205
29.487.4.30	<code>transform(const std::vector< double > &vec, int vdim=0)</code>	2205
29.487.4.31	<code>derivVolume(int ider1, int ider2, int ider3) const</code>	2205
29.487.4.32	<code>dimension() const</code>	2205
29.487.4.33	<code>ndparam(int i) const</code>	2206
29.487.4.34	<code>getAllBoundarySurfaces() const</code>	2207
29.487.4.35	<code>setBoundarySurface(int idx, bool do_clear=false) const</code>	2207

29.487.4.36	GetBoundarySurfaces(bool do_clear=false) const	2207
29.487.4.37	GetElementBdPar(int elem_ix, double elem_par[]) const	2207
29.487.4.38	GetElementBdSfs(int elem_ix, double elem_par[]) const	2207
29.487.4.39	GetWeights(std::vector< double > &weights) const	2207
29.487.4.40	GridEvaluator(int num_u, int num_v, int num_w, std::vector< double > &points, std::vector< double > &der_u, std::vector< double > &der_v, std::vector< double > &der_w, std::vector< double > ¶m_u, std::vector< double > ¶m_v, std::vector< double > ¶m_w, bool evaluate_from_right=true) const	2207
29.487.4.41	GridEvaluator(const std::vector< double > ¶m_u, const std::vector< double > ¶m_v, const std::vector< double > ¶m_w, std::vector< double > &points, std::vector< double > &der_u, std::vector< double > &der_v, std::vector< double > &der_w, bool evaluate_from_right=true) const	2208
29.487.4.42	GridEvaluator(int num_u, int num_v, int num_w, std::vector< double > &points, std::vector< double > ¶m_u, std::vector< double > ¶m_v, std::vector< double > ¶m_w) const	2208
29.487.4.43	GridEvaluator(const std::vector< double > ¶m_u, const std::vector< double > ¶m_v, const std::vector< double > ¶m_w, std::vector< double > &points) const	2209
29.487.4.44	InsertKnot(int padir, double apar)	2209
29.487.4.45	InsertKnot(int padir, const std::vector< double > &new_knots)	2209
29.487.4.46	InstanceType() const	2210
29.487.4.47	IsDegenerate(int which_sf, int &type, bool &b, bool &r, bool &t, bool &l, double tol) const	2210
29.487.4.48	IsLeftHanded()	2210
29.487.4.49	IsSpline() const	2210
29.487.4.50	JoinSpan(int padir, int iknot) const	2210
29.487.4.51	MakeBernsteinKnots(int padir)	2210
29.487.4.52	NextSegmentVal(int dir, double par, bool forward, double tol) const	2211
29.487.4.53	NumberOfPatches(int padir) const	2211
29.487.4.54	NumCoefs(int padir) const	2211
29.487.4.55	NumElem() const	2212
29.487.4.56	NumElem(int padir) const	2212
29.487.4.57	Order(int padir) const	2212
29.487.4.58	ParameterSpan() const	2212
29.487.4.59	Point(Point &pt, double upar, double vpar, double wpar) const	2212

29.487.4.60	Point(std::vector< Point > &pts, double upar, double vpar, double wpar, int derivs, bool u_from_right=true, bool v_from_right=true, bool w_from_right=true, double resolution=1.0e-12) const	2213
29.487.4.61	PointsGrid(int numu, int numv, int numw, std::vector< double >::iterator basisvals_u, std::vector< double >::iterator basisvals_v, std::vector< double >::iterator basisvals_w, std::vector< int >::iterator knotinter_u, std::vector< int >::iterator knotinter_v, std::vector< int >::iterator knotinter_w, int derivs, std::vector< double > &points) const	2213
29.487.4.62	RaiseOrder(int raise_u, int raise_v, int raise_w)	2213
29.487.4.63	Rational() const	2214
29.487.4.64	Coefs_begin()	2214
29.487.4.65	Coefs_begin() const	2214
29.487.4.66	Coefs_end()	2214
29.487.4.67	Coefs_end() const	2214
29.487.4.68	Read(std::istream &is)	2214
29.487.4.69	RemoveKnot(int pardir, double tpar)	2215
29.487.4.70	ReplaceCoefficient(int ix, Point coef)	2215
29.487.4.71	ReverseParameterDirection(int pardir)	2215
29.487.4.72	Scale(double fac)	2215
29.487.4.73	SetParameterDomain(double u1, double u2, double v1, double v2, double w1, double w2)	2215
29.487.4.74	Split(std::vector< double > ¶m, int pardir, double fuzzy=DEFAULT_PARAMETER_EPSILON) const	2216
29.487.4.75	Startparam(int i) const	2216
29.487.4.76	SubVolume(double from_upar, double from_vpar, double from_wpar, double to_upar, double to_vpar, double to_wpar, double fuzzy=DEFAULT_PARAMETER_EPSILON) const	2216
29.487.4.77	Swap(SplineVolume &other)	2216
29.487.4.78	SwapParameterDirection(int pardir1, int pardir2)	2216
29.487.4.79	TangentCone(int pardir) const	2217
29.487.4.80	Translate(const Point &vec)	2217
29.487.4.81	VolumePeriodicity(int pardir, double epsilon) const	2217
29.487.4.82	Write(std::ostream &os) const	2217
29.488	NEWMAT::SPMatrix Class Reference	2218
29.488.1	Detailed Description	2220

29.488.2	Constructor & Destructor Documentation	2220
29.488.2.1	SPMatrix(const BaseMatrix *bm1x, const BaseMatrix *bm2x)	2220
29.488.2.2	~SPMatrix()	2220
29.488.3	Member Function Documentation	2220
29.488.3.1	BandWidth() const	2220
29.488.3.2	Evaluate(MatrixType mt=MatrixTypeUnSp)	2220
29.488.4	Friends And Related Function Documentation	2220
29.488.4.1	BaseMatrix	2220
29.488.4.2	GeneralMatrix	2220
29.488.4.3	GenericMatrix	2221
29.488.4.4	SP	2221
29.489	NEWMAT::StackedMatrix Class Reference	2221
29.489.1	Detailed Description	2223
29.489.2	Constructor & Destructor Documentation	2223
29.489.2.1	StackedMatrix(const BaseMatrix *bm1x, const BaseMatrix *bm2x)	2223
29.489.2.2	~StackedMatrix()	2223
29.489.3	Member Function Documentation	2223
29.489.3.1	Evaluate(MatrixType mt=MatrixTypeUnSp)	2223
29.489.4	Friends And Related Function Documentation	2223
29.489.4.1	BaseMatrix	2223
29.489.4.2	GeneralMatrix	2223
29.489.4.3	GenericMatrix	2223
29.490	Go::Streamable Class Reference	2224
29.490.1	Detailed Description	2224
29.490.2	Constructor & Destructor Documentation	2225
29.490.2.1	~Streamable()	2225
29.490.3	Member Function Documentation	2225
29.490.3.1	read(std::istream &is)=0	2225
29.490.3.2	write(std::ostream &os) const =0	2226
29.491	NEWMAT::SubMatrixDimensionException Class Reference	2226

29.491.1	Detailed Description	2228
29.491.2	Constructor & Destructor Documentation	2228
29.491.2.1	SubMatrixDimensionException()	2228
29.491.3	Member Data Documentation	2228
29.491.3.1	Select	2228
29.492	Bo::boxStructuring::SubSurfaceBoundingBox Class Reference	2228
29.492.1	Detailed Description	2229
29.492.2	Constructor & Destructor Documentation	2229
29.492.2.1	SubSurfaceBoundingBox(shared_ptr< SurfaceData > surface_data, int pos_u, int pos_v, BoundingBox box, shared_ptr< RectDomain > par_domain)	2229
29.492.3	Member Function Documentation	2229
29.492.3.1	add_polygon_corners(double par_u, double par_v)	2229
29.492.3.2	box() const	2229
29.492.3.3	has_polygon() const	2229
29.492.3.4	inside() const	2230
29.492.3.5	inside(double par_u, double par_v) const	2230
29.492.3.6	par_domain() const	2230
29.492.3.7	pos_u() const	2230
29.492.3.8	pos_v() const	2230
29.492.3.9	remove_polygon()	2230
29.492.3.10	setInside(bool inside)	2230
29.492.3.11	size_polygon() const	2231
29.492.3.12	surface_data() const	2231
29.493	IEWMAT::SubtractedMatrix Class Reference	2231
29.493.1	Detailed Description	2232
29.493.2	Constructor & Destructor Documentation	2233
29.493.2.1	~SubtractedMatrix()	2233
29.493.3	Member Function Documentation	2233
29.493.3.1	Evaluate(MatrixType mt=MatrixTypeUnSp)	2233
29.493.4	Friends And Related Function Documentation	2233
29.493.4.1	BaseMatrix	2233

29.493.4.2	GeneralMatrix	2233
29.493.4.3	GenericMatrix	2233
29.494	Go::LRBSplineUtils::support_compare Struct Reference	2233
29.494.1	Detailed Description	2234
29.494.2	Member Function Documentation	2234
29.494.2.1	operator()(const LRBSpline2D *b1, const LRBSpline2D *b2) const	2234
29.495	Go::SurfaceAssembly Class Reference	2234
29.495.1	Detailed Description	2234
29.495.2	Constructor & Destructor Documentation	2235
29.495.2.1	SurfaceAssembly(shared_ptr< ParamSurfaceInt > surf, std::vector< std::pair< double, int > > u_div, std::vector< std::pair< double, int > > v_div, std::vector< RectDomain > sing_domain, double rel_par_res)	2235
29.495.2.2	~SurfaceAssembly()	2235
29.495.3	Member Function Documentation	2235
29.495.3.1	doTouch(int idx1, int idx2)	2235
29.495.3.2	getNextAssembly(shared_ptr< ParamSurfaceInt > &assembly, int &idx, bool &potential_sing)	2235
29.495.3.3	getNextSubSurface(shared_ptr< ParamSurfaceInt > &sub_sf, int &idx, int &sing_idx)	2235
29.495.3.4	getNmbSubSurface()	2235
29.495.3.5	getSubSurfaceIndex(shared_ptr< ParamSurfaceInt > sub_srf, bool &at_end)	2235
29.495.3.6	getUdiv()	2235
29.495.3.7	getVdiv()	2235
29.495.3.8	isInFirstAssembly(shared_ptr< ParamSurfaceInt > sub_srf)	2235
29.495.3.9	isInPrevAssembly(int idx1, int idx2)	2235
29.495.3.10	setAssemblyIndex()	2235
29.495.3.11	setSubIndex(int idx=0)	2235
29.495.3.12	subSfNeighbour(int idx1, int idx2)	2236
29.495.3.13	touchAtSingularity(int idx1, int idx2)	2236
29.496	Go::boxStructuring::SurfaceData Class Reference	2236
29.496.1	Detailed Description	2236
29.496.2	Constructor & Destructor Documentation	2237

29.496.2.1	SurfaceData(shared_ptr< ParamSurface > surface)	2237
29.496.3	Member Function Documentation	2237
29.496.3.1	add_inside_point(Point pt)	2237
29.496.3.2	index() const	2237
29.496.3.3	inside_points() const	2237
29.496.3.4	segs_u() const	2237
29.496.3.5	segs_v() const	2237
29.496.3.6	setIndex(int index)	2237
29.496.3.7	setSegments(int segs_u, int segs_v)	2238
29.496.3.8	setSurfaceCopies(int nmb_copies)	2238
29.496.3.9	surface(int idx) const	2238
29.497	Go::SurfaceModel Class Reference	2238
29.497.1	Detailed Description	2242
29.497.2	Constructor & Destructor Documentation	2242
29.497.2.1	SurfaceModel(double approxtol, double gap, double neighbour, double kink, double bend, std::vector< shared_ptr< ftSurface > > &faces, bool adjacency_set=false)	2242
29.497.2.2	SurfaceModel(std::vector< shared_ptr< ftSurface > > &faces, double space_epsilon, double kink=0.01, bool adjacency_set=false)	2242
29.497.2.3	SurfaceModel(double approxtol, double gap, double neighbour, double kink, double bend, std::vector< shared_ptr< ParamSurface > > &surfaces)	2243
29.497.2.4	SurfaceModel(double approxtol, double gap, double neighbour, double kink, double bend)	2243
29.497.2.5	SurfaceModel(const SurfaceModel &sm)	2243
29.497.2.6	~SurfaceModel()	2243
29.497.3	Member Function Documentation	2243
29.497.3.1	addSegment(ftCurve &cv, ftEdgeBase *edge, ftCurveType ty)	2243
29.497.3.2	allFaces() const	2243
29.497.3.3	allSplines() const	2244
29.497.3.4	append(shared_ptr< ftSurface > face, bool set_twin=true, bool adjacency_set=false, bool remove_twins=false, int idx=-1)	2244
29.497.3.5	append(std::vector< shared_ptr< ftSurface > > faces, bool adjacency_set=false, bool set_twin=true)	2244

29.497.3.6	append(shared_ptr< SurfaceModel > anotherModel)	2244
29.497.3.7	approxFaceSet(double &error, int degree=3)	2244
29.497.3.8	asSurfaceModel()	2244
29.497.3.9	booleanIntersect(const ftPlane &plane)	2245
29.497.3.10	boundingBox()	2245
29.497.3.11	boundingBox(int idx) const	2245
29.497.3.12	buildTopology(int first_idx=0, bool set_twin_face_info=true)	2245
29.497.3.13	checkShellTopology()	2246
29.497.3.14	clone() const	2246
29.497.3.15	closestPoint(Point &pnt, Point &clo_pnt, int &idx, double clo_par[], double &dist)	2246
29.497.3.16	closestPoint(const Point &point)	2246
29.497.3.17	closestPoint(const ftPoint &point)	2246
29.497.3.18	curvature(int idx, double *par) const	2247
29.497.3.19	intersect(shared_ptr< SplineSurface > sf)	2247
29.497.3.20	enforceCoLinearCoefs()	2247
29.497.3.21	evaluate(int idx, double par[], Point &pnt) const	2247
29.497.3.22	evaluate(int idx, double par[], int nder, std::vector< Point > &der) const	2248
29.497.3.23	extremalPoint(Point &dir, Point &ext_pnt, int &idx, double ext_par[])	2248
29.497.3.24	facesInPlane(Point &pnt, Point &axis)	2248
29.497.3.25	fetchAsSharedPtr(ftFaceBase *face) const	2248
29.497.3.26	fetchSamplePoints(double density, std::vector< SamplePointData > &sample_← points) const	2249
29.497.3.27	getAllVertices(std::vector< shared_ptr< Vertex > > &vertices) const	2249
29.497.3.28	getApproximationTol() const	2249
29.497.3.29	getBody()	2249
29.497.3.30	getBoundary(int whichbound)	2249
29.497.3.31	getBoundaryEdges() const	2250
29.497.3.32	getBoundaryEdges(int boundary_idx) const	2250
29.497.3.33	getBoundaryVertices(std::vector< shared_ptr< Vertex > > &vertices) const	2250
29.497.3.34	getCell(int i) const	2250
29.497.3.35	getConnectedModels() const	2250

29.497.3.36	getCorners(std::vector< ftEdge * > &corners)	2251
29.497.3.37	getFace(int idx) const	2251
29.497.3.38	getG1Disconts()	2251
29.497.3.39	getGaps()	2251
29.497.3.40	getGaps(std::vector< ftEdge * > &gaps)	2251
29.497.3.41	getInconsistentFacePairs(std::vector< std::pair< ftFaceBase *, ftFaceBase * > > &faces)	2251
29.497.3.42	getIndex(shared_ptr< ftSurface > face) const	2251
29.497.3.43	getIndex(ftSurface *face) const	2252
29.497.3.44	getIndex(ParamSurface *surf) const	2252
29.497.3.45	getKinks()	2252
29.497.3.46	getKinks(std::vector< ftEdge * > &kinks)	2252
29.497.3.47	getOverlappingEdges(double tol, std::vector< std::pair< shared_ptr< ftEdgeBase >, shared_ptr< ftEdgeBase > > > &edges)	2252
29.497.3.48	getOverlappingFaces(double tol, std::vector< std::pair< ftSurface *, ftSurface * > > &faces)	2253
29.497.3.49	getSplineSurface(int idx) const	2253
29.497.3.50	getSurface(int idx) const	2253
29.497.3.51	getSurface2(int index) const	2253
29.497.3.52	getUniqueInnerEdges() const	2254
29.497.3.53	hit(const Point &point, const Point &dir, ftPoint &result)	2254
29.497.3.54	initializeCelldiv()	2254
29.497.3.55	intersect(const ftLine &line)	2254
29.497.3.56	intersect(const ftLine &line, ftCurve &int_curves, std::vector< ftPoint > &int_points)	2255
29.497.3.57	intersect(const ftPlane &plane)	2255
29.497.3.58	intersect(const ftLine &line, std::vector< bool > &represent_segment)	2255
29.497.3.59	intersect(shared_ptr< SplineCurve > crv, std::vector< bool > &represent_← segment)	2256
29.497.3.60	intersect_plane(const ftPlane &plane)	2256
29.497.3.61	AxisRotational(Point ¢re, Point &axis, Point &vec, double &angle, double &min_ang)	2256
29.497.3.62	Closed() const	2256
29.497.3.63	CornerToCorner() const	2257

29.497.3.64	Degenerate(int idx) const	2257
29.497.3.65	Degenerate(int idx, bool &b, bool &t, bool &l, bool &r) const	2257
29.497.3.66	LinearSwept(Point &pnt, Point &axis, double &len)	2257
29.497.3.67	LimitVolume(double xmin, double xmax, double ymin, double ymax, double zmin, double zmax)	2258
29.497.3.68	MakeCommonSplineSpaces()	2258
29.497.3.69	MakeCornerToCorner()	2258
29.497.3.70	MergeFaces(ftSurface *face1, int pdir1, double parval1, bool atstart1, ftSurface *face2, int pdir2, double parval2, bool atstart2, std::pair< Point, Point > co_↔par1, std::pair< Point, Point > co_par2, std::vector< Point > &seam_joints)	2258
29.497.3.71	MergeSeamCrvFaces(ftSurface *face1, ftSurface *face2, std::vector< Point > &seam_joints)	2258
29.497.3.72	MergeSeamFaces(ftSurface *face1, ftSurface *face2, int pdir, std::vector< Point > &seam_joints)	2258
29.497.3.73	mbBoundaries() const	2258
29.497.3.74	mbEntities() const	2259
29.497.3.75	PointWithinLimits(const ftPoint &point)	2259
29.497.3.76	PointWithinLimits(const Point &point)	2259
29.497.3.77	RegularizeTwin(ftSurface *face, std::vector< shared_ptr< ftSurface > > &twinset)	2259
29.497.3.78	RemoveFace(shared_ptr< ftSurface > face)	2260
29.497.3.79	ReplaceRegularSurface(ftSurface *face, bool only_corner=false)	2260
29.497.3.80	ReplaceRegularSurfaces()	2260
29.497.3.81	SetBoundaryCurves()	2260
29.497.3.82	SetTolerances(double approxtol, double gap, double kink)	2260
29.497.3.83	SetTolerances(double approxtol, double gap, double neighbour, double kink, double bend)	2260
29.497.3.84	SetTopology()	2261
29.497.3.85	SetTwinFaceInfo()	2261
29.497.3.86	SetVertexIdentity()	2261
29.497.3.87	SimplifyShell()	2261
29.497.3.88	SimplifyTrimLoops(double &max_dist)	2261
29.497.3.89	SplitSurfaceModel(std::vector< shared_ptr< ftSurface > > &faces, Body *model2, std::vector< std::vector< shared_ptr< ParamSurface > > > &result)	2261

29.497.3.90	SplitSurfaceModels(shared_ptr< SurfaceModel > &model2)	2262
29.497.3.91	SwapFaces(int idx1, int idx2)	2262
29.497.3.92	Resellate(std::vector< shared_ptr< GeneralMesh > > &meshes) const	2262
29.497.3.93	Resellate(int uv_res, std::vector< shared_ptr< GeneralMesh > > &meshes) const	2262
29.497.3.94	Resellate(int resolution[], std::vector< shared_ptr< GeneralMesh > > &meshes) const	2262
29.497.3.95	Resellate(double density, std::vector< shared_ptr< GeneralMesh > > &meshes) const	2263
29.497.3.96	Resellate(const std::vector< shared_ptr< ftFaceBase > > &faces, int uv_res, std::vector< shared_ptr< GeneralMesh > > &meshes) const	2263
29.497.3.97	Resellate(const std::vector< shared_ptr< ftFaceBase > > &faces, int resolu- tion[], std::vector< shared_ptr< GeneralMesh > > &meshes) const	2263
29.497.3.98	Resellate(const std::vector< shared_ptr< ftFaceBase > > &faces, double den- sity, std::vector< shared_ptr< GeneralMesh > > &meshes) const	2264
29.497.3.99	ReselatedCtrPolygon(std::vector< shared_ptr< LineCloud > > &ctr_pol) const	2264
29.497.3.100	ReselatedCtrPolygon(const std::vector< shared_ptr< ftFaceBase > > &faces, std::vector< shared_ptr< LineCloud > > &ctr_pol) const	2264
29.497.3.101	Triangulate(double density) const	2265
29.497.3.102	IntersectWithPlane(const ftPlane &plane)	2265
29.497.3.103	IsIn(int idx)	2265
29.497.3.104	IsIn()	2265
29.497.3.105	UpdateFaceTopology(shared_ptr< ftSurface > face)	2265
29.497.4	Member Data Documentation	2266
29.497.4.1	approx_tol_	2266
29.497.4.2	boundary_curves_	2266
29.497.4.3	celldiv_	2266
29.497.4.4	face_checked_	2266
29.497.4.5	faces_	2266
29.497.4.6	highest_face_checked_	2266
29.497.4.7	inconsistent_orientation_	2266
29.497.4.8	limit_box_	2266
29.497.4.9	tol2d_	2267

29.498.0	Go::SurfaceOfLinearExtrusion Class Reference	2267
29.498.1	Detailed Description	2269
29.498.2	Constructor & Destructor Documentation	2270
29.498.2.1	SurfaceOfLinearExtrusion()	2270
29.498.2.2	SurfaceOfLinearExtrusion(shared_ptr< SplineCurve > curve, Point axis_dir, bool isSwapped=false)	2270
29.498.2.3	~SurfaceOfLinearExtrusion()	2270
29.498.3	Member Function Documentation	2270
29.498.3.1	allBoundaryLoops(double degenerate_epsilon=DEFAULT_SPACE_EPSILON) const	2270
29.498.3.2	area(double tol) const	2270
29.498.3.3	boundingBox() const	2271
29.498.3.4	classType()	2271
29.498.3.5	clone() const	2271
29.498.3.6	closestBoundaryPoint(const Point &pt, double &clo_u, double &clo_v, Point &clo_pt, double &clo_dist, double epsilon, const RectDomain *rd=NULL, double *seed=0) const	2271
29.498.3.7	closestInDomain(double u, double v) const	2271
29.498.3.8	closestPoint(const Point &pt, double &clo_u, double &clo_v, Point &clo_pt, double &clo_dist, double epsilon, const RectDomain *domain_of_interest=NULL, double *seed=0) const	2271
29.498.3.9	constParamCurves(double parameter, bool pardir_is_u) const	2272
29.498.3.10	containingDomain() const	2272
29.498.3.11	createSplineSurface() const	2272
29.498.3.12	dimension() const	2273
29.498.3.13	geometrySurface() const	2273
29.498.3.14	getAxisDir() const	2273
29.498.3.15	getBoundaryInfo(Point &pt1, Point &pt2, double epsilon, SplineCurve *&cv, SplineCurve *&crosscv, double knot_tol=1e-05) const	2273
29.498.3.16	getCornerPoints(std::vector< std::pair< Point, Point > > &corners) const	2273
29.498.3.17	getCurve() const	2274
29.498.3.18	getDegenerateCorners(std::vector< Point > °_corners, double tol) const	2274
29.498.3.19	inDomain(double u, double v, double eps=1.0e-4) const	2274

29.498.3.20	Domain2(double u, double v, double eps=1.0e-4) const	2274
29.498.3.21	InstanceType() const	2274
29.498.3.22	Bounded() const	2274
29.498.3.23	Swapped() const	2274
29.498.3.24	NextSegmentVal(int dir, double par, bool forward, double tol) const	2274
29.498.3.25	Normal(Point &n, double upar, double vpar) const	2275
29.498.3.26	NormalCone() const	2275
29.498.3.27	Boundary(double u, double v, double eps=1.0e-4) const	2275
29.498.3.28	OuterBoundaryLoop(double degenerate_epsilon=DEFAULT_SPACE_EPSILON) const	2275
29.498.3.29	ParameterDomain() const	2276
29.498.3.30	Point(Point &pt, double upar, double vpar) const	2276
29.498.3.31	Point(std::vector< Point > &pts, double upar, double vpar, int derivs, bool u_← from_right=true, bool v_from_right=true, double resolution=1.0e-12) const	2276
29.498.3.32	Read(std::istream &is)	2277
29.498.3.33	ReverseParameterDirection(bool direction_is_u)	2277
29.498.3.34	SetParameterBounds(double from_upar, double from_vpar, double to_upar, dou- ble to_vpar)	2277
29.498.3.35	SubSurface(double from_upar, double from_vpar, double to_upar, double to_vpar, double fuzzy=DEFAULT_PARAMETER_EPSILON) const	2277
29.498.3.36	SubSurfaces(double from_upar, double from_vpar, double to_upar, double to_← vpar, double fuzzy=DEFAULT_PARAMETER_EPSILON) const	2278
29.498.3.37	SwapParameterDirection()	2278
29.498.3.38	TangentCone(bool pardir_is_u) const	2278
29.498.3.39	TurnOrientation()	2278
29.498.3.40	Write(std::ostream &os) const	2279
29.499.0	Go::SurfaceOfRevolution Class Reference	2279
29.499.1	Detailed Description	2282
29.499.2	Constructor & Destructor Documentation	2282
29.499.2.1	SurfaceOfRevolution()	2282
29.499.2.2	SurfaceOfRevolution(Point location, Point axis_dir, shared_ptr< SplineCurve > curve, bool isSwapped=false)	2282
29.499.2.3	~SurfaceOfRevolution()	2282

29.499.3	Member Function Documentation	2282
29.499.3.1	allBoundaryLoops(double degenerate_epsilon=DEFAULT_SPACE_EPSILON) const	2282
29.499.3.2	area(double tol) const	2283
29.499.3.3	boundingBox() const	2283
29.499.3.4	classType()	2283
29.499.3.5	clone() const	2283
29.499.3.6	closestBoundaryPoint(const Point &pt, double &clo_u, double &clo_v, Point &clo_pt, double &clo_dist, double epsilon, const RectDomain *rd=NULL, double *seed=0) const	2283
29.499.3.7	closestInDomain(double u, double v) const	2284
29.499.3.8	closestPoint(const Point &pt, double &clo_u, double &clo_v, Point &clo_pt, double &clo_dist, double epsilon, const RectDomain *domain_of_interest=NULL, double *seed=0) const	2284
29.499.3.9	constParamCurves(double parameter, bool pardir_is_u) const	2284
29.499.3.10	containingDomain() const	2284
29.499.3.11	createSplineSurface() const	2285
29.499.3.12	dimension() const	2285
29.499.3.13	geometrySurface() const	2285
29.499.3.14	getAxisDir() const	2285
29.499.3.15	getBoundaryInfo(Point &pt1, Point &pt2, double epsilon, SplineCurve *&cv, SplineCurve *&crosscv, double knot_tol=1e-05) const	2285
29.499.3.16	getCircle(double vpar) const	2286
29.499.3.17	getCornerPoints(std::vector< std::pair< Point, Point > > &corners) const	2286
29.499.3.18	getCurve() const	2286
29.499.3.19	getDegenerateCorners(std::vector< Point > °_corners, double tol) const	2286
29.499.3.20	getLocation() const	2286
29.499.3.21	inDomain(double u, double v, double eps=1.0e-4) const	2286
29.499.3.22	inDomain2(double u, double v, double eps=1.0e-4) const	2287
29.499.3.23	instanceType() const	2287
29.499.3.24	isSwapped() const	2287
29.499.3.25	nextSegmentVal(int dir, double par, bool forward, double tol) const	2287
29.499.3.26	normal(Point &n, double upar, double vpar) const	2287

29.499.3.27	NormalCone() const	2288
29.499.3.28	InBoundary(double u, double v, double eps=1.0e-4) const	2288
29.499.3.29	OuterBoundaryLoop(double degenerate_epsilon=DEFAULT_SPACE_EPSILON) const	2288
29.499.3.30	ParameterDomain() const	2288
29.499.3.31	Point(Point &pt, double upar, double vpar) const	2288
29.499.3.32	Point(std::vector< Point > &pts, double upar, double vpar, int derivs, bool u_from_right=true, bool v_from_right=true, double resolution=1.0e-12) const	2289
29.499.3.33	Read(std::istream &is)	2289
29.499.3.34	InverseParameterDirection(bool direction_is_u)	2289
29.499.3.35	SetParameterBounds(double from_upar, double from_vpar, double to_upar, double to_vpar)	2290
29.499.3.36	SubSurface(double from_upar, double from_vpar, double to_upar, double to_vpar, double fuzzy=DEFAULT_PARAMETER_EPSILON) const	2290
29.499.3.37	SubSurfaces(double from_upar, double from_vpar, double to_upar, double to_vpar, double fuzzy=DEFAULT_PARAMETER_EPSILON) const	2290
29.499.3.38	SwapParameterDirection()	2290
29.499.3.39	TangentCone(bool paddir_is_u) const	2290
29.499.3.40	TurnOrientation()	2291
29.499.3.41	Write(std::ostream &os) const	2291
29.500	Go::SurfaceOnVolume Class Reference	2291
29.500.1	Detailed Description	2295
29.500.2	Constructor & Destructor Documentation	2295
29.500.2.1	SurfaceOnVolume()	2295
29.500.2.2	SurfaceOnVolume(shared_ptr< ParamVolume > vol, shared_ptr< ParamSurface > parsurf, shared_ptr< ParamSurface > spacesurf, bool preferparameter)	2295
29.500.2.3	SurfaceOnVolume(shared_ptr< ParamVolume > vol, shared_ptr< ParamSurface > spacesurf, int constdir, double constpar, int boundary, bool swapped, int orientation=0)	2295
29.500.2.4	SurfaceOnVolume(shared_ptr< ParamVolume > vol, int constdir, double constpar, int boundary)	2296
29.500.2.5	SurfaceOnVolume(shared_ptr< ParamVolume > vol, shared_ptr< ParamSurface > spacesurf, shared_ptr< ParamSurface > parsurf, bool preferparameter, int constdir, double constpar, int boundary, bool swapped)	2296
29.500.2.6	SurfaceOnVolume(const SurfaceOnVolume &other)	2296

29.500.2.7	~SurfaceOnVolume()	2296
29.500.3	Member Function Documentation	2296
29.500.3.1	allBoundaryLoops(double degenerate_epsilon=DEFAULT_SPACE_EPSILON) const	2296
29.500.3.2	area(double tol) const	2297
29.500.3.3	asSplineSurface()	2297
29.500.3.4	boundingBox() const	2297
29.500.3.5	classType()	2297
29.500.3.6	clone() const	2297
29.500.3.7	closestBoundaryPoint(const Point &pt, double &clo_u, double &clo_v, Point &clo_pt, double &clo_dist, double epsilon, const RectDomain *rd=NULL, double *seed=0) const	2298
29.500.3.8	closestInDomain(double u, double v) const	2298
29.500.3.9	closestPoint(const Point &pt, double &clo_u, double &clo_v, Point &clo_pt, double &clo_dist, double epsilon, const RectDomain *domain_of_interest=NULL, double *seed=0) const	2298
29.500.3.10	compositeBox() const	2298
29.500.3.11	constParamCurves(double parameter, bool pardir_is_u) const	2299
29.500.3.12	containingDomain() const	2299
29.500.3.13	dimension() const	2299
29.500.3.14	ElementBoundaryStatus(int elem_ix, double eps)	2299
29.500.3.15	ElementOnBoundary(int elem_ix, double eps)	2300
29.500.3.16	getBoundaryInfo(Point &pt1, Point &pt2, double epsilon, SplineCurve *&cv, SplineCurve *&crosscv, double knot_tol=1e-05) const	2300
29.500.3.17	getConstDir() const	2301
29.500.3.18	getConstVal() const	2301
29.500.3.19	getCornerPoints(std::vector< std::pair< Point, Point > > &corners) const	2301
29.500.3.20	getDegenerateCorners(std::vector< Point > °_corners, double tol) const	2301
29.500.3.21	getVolume() const	2301
29.500.3.22	getVolume()	2301
29.500.3.23	Domain(double u, double v, double eps=1.0e-4) const	2302
29.500.3.24	Domain2(double u, double v, double eps=1.0e-4) const	2302
29.500.3.25	instanceType() const	2302

29.500.3.26	Degenerate(bool &b, bool &r, bool &t, bool &l, double tolerance) const	2302
29.500.3.27	IsoTrimmed(double tol) const	2302
29.500.3.28	Linear(Point &dir1, Point &dir2, double tol)	2303
29.500.3.29	NextSegmentVal(int dir, double par, bool forward, double tol) const	2303
29.500.3.30	Normal(Point &n, double upar, double vpar) const	2303
29.500.3.31	NormalCone() const	2303
29.500.3.32	OnBoundary(double u, double v, double eps=1.0e-4) const	2304
29.500.3.33	OuterBoundaryLoop(double degenerate_epsilon=DEFAULT_SPACE_EPSILON) const	2304
29.500.3.34	ParameterDomain() const	2304
29.500.3.35	ParameterSurface() const	2304
29.500.3.36	ParameterSurface()	2304
29.500.3.37	ParPref() const	2304
29.500.3.38	Point(Point &pt, double upar, double vpar) const	2304
29.500.3.39	Point(std::vector< Point > &pts, double upar, double vpar, int derivs, bool u_← from_right=true, bool v_from_right=true, double resolution=1.0e-12) const	2305
29.500.3.40	Read(std::istream &is)	2305
29.500.3.41	ReverseParameterDirection(bool direction_is_u)	2305
29.500.3.42	SetSpaceSurface(shared_ptr< ParamSurface > spacesurf)	2306
29.500.3.43	SetVolume(shared_ptr< ParamVolume > volume)	2306
29.500.3.44	SpaceSurface() const	2306
29.500.3.45	SpaceSurface()	2306
29.500.3.46	SubSurfaces(double from_upar, double from_vpar, double to_upar, double to_← vpar, double fuzzy=DEFAULT_PARAMETER_EPSILON) const	2306
29.500.3.47	SwapParameterDirection()	2307
29.500.3.48	TangentCone(bool padir_is_u) const	2307
29.500.3.49	TurnOrientation()	2307
29.500.3.50	UnsetParamSurf()	2307
29.500.3.51	VolumeParameter(double u_par, double v_par) const	2307
29.500.3.52	WhichBoundary(double tol, int &orientation, bool &swap) const	2307
29.500.3.53	Write(std::ostream &os) const	2308
29.500	SurfaceResolutionSheet Class Reference	2308

29.501.1	Detailed Description	2309
29.501.2	Constructor & Destructor Documentation	2309
29.501.2.1	SurfaceResolutionSheet(int ures=20, int vres=20)	2309
29.501.3	Member Function Documentation	2309
29.501.3.1	createSheet(QWidget *parent, gvObserver *obs)	2309
29.501.3.2	bk	2309
29.501.3.3	return_value	2309
29.502	Go::SweepSurfaceCreator Class Reference	2309
29.502.1	Detailed Description	2310
29.502.2	Constructor & Destructor Documentation	2310
29.502.2.1	~SweepSurfaceCreator()	2310
29.502.3	Member Function Documentation	2310
29.502.3.1	linearSweptSurface(const SplineCurve &curv1, const SplineCurve &curv2, const Point &pt)	2310
29.502.3.2	rotationalSweptSurface(const SplineCurve &curve, double angle, const Point &pt, const Point &axis)	2310
29.503	Go::SweepVolumeCreator Class Reference	2311
29.503.1	Detailed Description	2311
29.503.2	Constructor & Destructor Documentation	2311
29.503.2.1	~SweepVolumeCreator()	2311
29.503.3	Member Function Documentation	2311
29.503.3.1	linearSweptVolume(const SplineSurface &surface, const SplineCurve &curve, const Point &pt)	2311
29.503.3.2	rotationalSweptVolume(const SplineSurface &surface, double angle, const Point &pt, const Point &axis)	2312
29.504	NEWMAT::SymmetricBandMatrix Class Reference	2312
29.504.1	Detailed Description	2314
29.504.2	Constructor & Destructor Documentation	2314
29.504.2.1	SymmetricBandMatrix()	2314
29.504.2.2	~SymmetricBandMatrix()	2315
29.504.2.3	SymmetricBandMatrix(int n, int lb)	2315
29.504.2.4	SymmetricBandMatrix(const BaseMatrix &)	2315

29.504.2.5SymmetricBandMatrix(const SymmetricBandMatrix &gm)	2315
29.504.3Member Function Documentation	2315
29.504.3.1BandWidth() const	2315
29.504.3.2element(int, int)	2315
29.504.3.3element(int, int) const	2315
29.504.3.4GetCol(MatrixRowCol &)	2315
29.504.3.5GetCol(MatrixColX &)	2315
29.504.3.6GetRow(MatrixRowCol &)	2315
29.504.3.7LogDeterminant() const	2316
29.504.3.8MakeSolver()	2316
29.504.3.9Maximum() const	2316
29.504.3.10MaximumAbsoluteValue() const	2316
29.504.3.11Minimum() const	2316
29.504.3.12MinimumAbsoluteValue() const	2316
29.504.3.13operator()(int, int)	2316
29.504.3.14operator()(int, int) const	2316
29.504.3.15operator=(const BaseMatrix &)	2316
29.504.3.16operator=(Real f)	2316
29.504.3.17operator=(const SymmetricBandMatrix &m)	2317
29.504.3.18ReSize(int, int)	2317
29.504.3.19ReSize(const GeneralMatrix &A)	2317
29.504.3.20ReSizeForAdd(const GeneralMatrix &A, const GeneralMatrix &B)	2317
29.504.3.21ReSizeForSP(const GeneralMatrix &A, const GeneralMatrix &B)	2317
29.504.3.22RestoreCol(MatrixRowCol &)	2317
29.504.3.23RestoreCol(MatrixColX &)	2317
29.504.3.24SameStorageType(const GeneralMatrix &A) const	2317
29.504.3.25SetParameters(const GeneralMatrix *)	2318
29.504.3.26Sum() const	2318
29.504.3.27SumAbsoluteValue() const	2318
29.504.3.28SumSquare() const	2318

29.504.3.2	Trace() const	2318
29.504.3.3	Transpose(TransposedMatrix *, MatrixType)	2318
29.504.3.3	Type() const	2318
29.504.4	Member Data Documentation	2319
29.504.4.1	lower	2319
29.505	NEWMAT::SymmetricEigenAnalysis Class Reference	2319
29.505.1	Detailed Description	2319
29.505.2	Constructor & Destructor Documentation	2319
29.505.2.1	SymmetricEigenAnalysis(const SymmetricMatrix &)	2319
29.506	NEWMAT::SymmetricMatrix Class Reference	2320
29.506.1	Detailed Description	2321
29.506.2	Constructor & Destructor Documentation	2321
29.506.2.1	SymmetricMatrix()	2321
29.506.2.2	~SymmetricMatrix()	2321
29.506.2.3	SymmetricMatrix(ArrayLengthSpecifier)	2321
29.506.2.4	SymmetricMatrix(const BaseMatrix &)	2321
29.506.2.5	SymmetricMatrix(const SymmetricMatrix &gm)	2321
29.506.3	Member Function Documentation	2322
29.506.3.1	element(int, int)	2322
29.506.3.2	element(int, int) const	2322
29.506.3.3	GetCol(MatrixRowCol &)	2322
29.506.3.4	GetCol(MatrixColX &)	2322
29.506.3.5	GetRow(MatrixRowCol &)	2322
29.506.3.6	operator()(int, int)	2322
29.506.3.7	operator()(int, int) const	2322
29.506.3.8	operator=(const BaseMatrix &)	2322
29.506.3.9	operator=(Real f)	2322
29.506.3.10	operator=(const SymmetricMatrix &m)	2322
29.506.3.11	ReSize(int)	2322
29.506.3.12	ReSize(const GeneralMatrix &A)	2322

29.506.3.1B	RestoreCol(MatrixRowCol &)	2323
29.506.3.1A	RestoreCol(MatrixColX &)	2323
29.506.3.1S	Sum() const	2323
29.506.3.1G	SumAbsoluteValue() const	2323
29.506.3.1W	SumSquare() const	2323
29.506.3.1B	Trace() const	2323
29.506.3.1T	Transpose(TransposedMatrix *, MatrixType)	2323
29.506.3.2	Type() const	2324
29.507	Go::Tesselator Class Reference	2324
29.507.1	Detailed Description	2325
29.507.2	Constructor & Destructor Documentation	2325
29.507.2.1	~Tesselator()	2325
29.507.3	Member Function Documentation	2325
29.507.3.1	tesselate()=0	2325
29.508	TestClass Class Reference	2325
29.508.1	Detailed Description	2325
29.508.2	Constructor & Destructor Documentation	2325
29.508.2.1	TestClass()	2325
29.508.3	Member Function Documentation	2326
29.508.3.1	Sum()	2326
29.509	Go::TestInDomain Class Reference	2326
29.509.1	Detailed Description	2326
29.509.2	Member Typedef Documentation	2326
29.509.2.1	argument_type	2326
29.509.2.2	result_type	2326
29.509.3	Constructor & Destructor Documentation	2327
29.509.3.1	TestInDomain(const ParamObjectInt *obj1, const ParamObjectInt *obj2, shared_ptr< IntersectionPoint > ref_point)	2327
29.509.4	Member Function Documentation	2327
29.509.4.1	operator()(const shared_ptr< IntersectionLink > &l) const	2327
29.510	Time_lapse Class Reference	2327

29.510.1	Detailed Description	2327
29.510.2	Constructor & Destructor Documentation	2327
29.510.2.1	time_lapse()	2327
29.510.2.2	~time_lapse()	2327
29.510.3	Go::Torus Class Reference	2328
29.511.1	Detailed Description	2331
29.511.2	Constructor & Destructor Documentation	2331
29.511.2.1	Torus()	2331
29.511.2.2	Torus(double major_radius, double minor_radius, Point location, Point z_axis, Point x_axis, bool select_outer=true, bool isSwapped=false)	2331
29.511.2.3	~Torus()	2331
29.511.3	Member Function Documentation	2331
29.511.3.1	allBoundaryLoops(double degenerate_epsilon=DEFAULT_SPACE_EPSILON) const	2331
29.511.3.2	boundingBox() const	2332
29.511.3.3	classType()	2332
29.511.3.4	clone() const	2332
29.511.3.5	closestBoundaryPoint(const Point &pt, double &clo_u, double &clo_v, Point &clo_pt, double &clo_dist, double epsilon, const RectDomain *rd=NULL, double *seed=0) const	2332
29.511.3.6	closestPoint(const Point &pt, double &clo_u, double &clo_v, Point &clo_pt, double &clo_dist, double epsilon, const RectDomain *domain_of_interest=NULL, double *seed=0) const	2332
29.511.3.7	constParamCurves(double parameter, bool pardir_is_u) const	2333
29.511.3.8	createSplineSurface() const	2333
29.511.3.9	dimension() const	2333
29.511.3.10	geometrySurface() const	2334
29.511.3.11	getBoundaryInfo(Point &pt1, Point &pt2, double epsilon, SplineCurve *&cv, SplineCurve *&crosscv, double knot_tol=1e-05) const	2334
29.511.3.12	getCoordinateAxes(Point &x_axis, Point &y_axis, Point &z_axis) const	2334
29.511.3.13	getDegenerateCorners(std::vector< Point > °_corners, double tol) const	2334
29.511.3.14	getLocation() const	2334
29.511.3.15	getMajorCircle(double vpar) const	2334

29.511.3.16	GetMajorRadius() const	2335
29.511.3.17	GetMinorCircle(double upar) const	2335
29.511.3.18	GetMinorRadius() const	2335
29.511.3.19	GetPhi() const	2335
29.511.3.20	GetSelectOuter() const	2335
29.511.3.21	InstanceType() const	2336
29.511.3.22	Bounded() const	2336
29.511.3.23	Closed(bool &closed_dir_u, bool &closed_dir_v) const	2336
29.511.3.24	Degenerate(bool &b, bool &r, bool &t, bool &l, double tolerance) const	2336
29.511.3.25	DegenerateTorus() const	2336
29.511.3.26	ExtSegmentVal(int dir, double par, bool forward, double tol) const	2337
29.511.3.27	Normal(Point &n, double upar, double vpar) const	2337
29.511.3.28	NormalCone() const	2337
29.511.3.29	ParameterDomain() const	2337
29.511.3.30	Point(Point &pt, double upar, double vpar) const	2337
29.511.3.31	Point(std::vector< Point > &pts, double upar, double vpar, int derivs, bool u_↔ from_right=true, bool v_from_right=true, double resolution=1.0e-12) const	2338
29.511.3.32	Read(std::istream &is)	2338
29.511.3.33	SetCoordinateAxes()	2338
29.511.3.34	SetDefaultDomain()	2338
29.511.3.35	SetDegenerateInfo()	2339
29.511.3.36	SetParameterBounds(double from_upar, double from_vpar, double to_upar, dou- ble to_vpar)	2339
29.511.3.37	SetSelectOuter(bool select_outer)	2339
29.511.3.38	SubSurface(double from_upar, double from_vpar, double to_upar, double to_vpar, double fuzzy=DEFAULT_PARAMETER_EPSILON) const	2339
29.511.3.39	SubSurfaces(double from_upar, double from_vpar, double to_upar, double to_↔ vpar, double fuzzy=DEFAULT_PARAMETER_EPSILON) const	2339
29.511.3.40	TangentCone(bool padir_is_u) const	2339
29.511.3.41	Write(std::ostream &os) const	2340
29.511.4	Member Data Documentation	2340
29.511.4.1	domain_	2340

29.511.4.2s_degenerate_torus_	2340
29.511.4.3location_	2340
29.511.4.4major_radius_	2340
29.511.4.5minor_radius_	2340
29.511.4.6orientedDomain_	2341
29.511.4.7phi_	2341
29.511.4.8select_outer_	2341
29.511.4.9x_axis_	2341
29.511.4.10y_axis_	2341
29.511.4.11z_axis_	2341
29.512.0TorVolume Class Reference	2341
29.512.1Detailed Description	2344
29.512.2Constructor & Destructor Documentation	2344
29.512.2.1TorVolume()	2344
29.512.2.2TorVolume(double major_radius, double minor_radius, Point location, Point z← _axis, Point x_axis)	2344
29.512.2.3~TorVolume()	2344
29.512.3Member Function Documentation	2345
29.512.3.1boundingBox() const	2345
29.512.3.2classType()	2345
29.512.3.3clone() const	2345
29.512.3.4closestPoint(const Point &pt, double &clo_u, double &clo_v, double &clo_w, Point &clo_pt, double &clo_dist, double epsilon, double *seed=0) const	2345
29.512.3.5dimension() const	2345
29.512.3.6geometryVolume() const	2346
29.512.3.7getAllBoundarySurfaces() const	2346
29.512.3.8instanceType() const	2346
29.512.3.9nextSegmentVal(int dir, double par, bool forward, double tol) const	2346
29.512.3.10parameterSpan() const	2346
29.512.3.11point(Point &pt, double upar, double vpar, double wpar) const	2347

29.512.3.12	Point(std::vector< Point > &pts, double upar, double vpar, double wpar, int derivs, bool u_from_right=true, bool v_from_right=true, bool w_from_right=true, double resolution=1.0e-12) const	2347
29.512.3.13	Read(std::istream &is)	2347
29.512.3.14	InverseParameterDirection(int paddir)	2348
29.512.3.15	SetParameters(double from_par, double to_par, int paddir)	2348
29.512.3.16	SwapParameterDirection(int paddir1, int paddir2)	2348
29.512.3.17	TangentCone(int paddir) const	2348
29.512.3.18	Translate(const Point &vec)	2349
29.512.3.19	UseCentreDegen()	2349
29.512.3.20	UseCornerDegen()	2349
29.512.3.21	Write(std::ostream &os) const	2349
29.513	tpEdge Class Reference	2349
29.513.1	Detailed Description	2351
29.513.2	Constructor & Destructor Documentation	2351
29.513.2.1	~tpEdge()	2351
29.513.2.2	tpEdge()	2352
29.513.3	Member Function Documentation	2352
29.513.3.1	adjacentEdges(bool at_start_of_edge, std::vector< tpEdge * > &adjacent, std::vector< bool > &at_start)	2352
29.513.3.2	boundingBox()=0	2352
29.513.3.3	checkContinuity(tpEdge *nextedge, double neighbour, double gap, double bend, double kink) const	2352
29.513.3.4	closeLoop(tpEdge *last)	2352
29.513.3.5	closestpoint(const Point &pt, double &clo_t, Point &clo_pt, double &clo_dist, double const *seed=0) const =0	2352
29.513.3.6	connectAfter(tpEdge *edge)	2352
29.513.3.7	connectTwin(tpEdge *twin, int &status)	2352
29.513.3.8	disconnectThis()	2352
29.513.3.9	disconnectTwin()	2353
29.513.3.10	entryId()=0	2353
29.513.3.11	face()=0	2353
29.513.3.12	getConnectivityInfo()	2353

29.513.3.13	hasConnectivityInfo()	2353
29.513.3.14	next()	2353
29.513.3.15	normal(double t) const =0	2353
29.513.3.16	onBoundary()	2353
29.513.3.17	point(double t) const =0	2353
29.513.3.18	prev()	2354
29.513.3.19	resetConnectivityInfo()	2354
29.513.3.20	setConnectivityInfo(shared_ptr< FaceConnectivity< tpEdge > > info)	2354
29.513.3.21	setEntryId(int id)=0	2354
29.513.3.22	split(double t)=0	2354
29.513.3.23	tangent(double t) const =0	2354
29.513.3.24	Max() const =0	2354
29.513.3.25	Min() const =0	2354
29.513.3.26	win()	2354
29.513.4	Member Data Documentation	2355
29.513.4.1	connectivity_info_	2355
29.513.4.2	next_	2355
29.513.4.3	prev_	2355
29.513.4.4	win_	2355
29.514	Go::tpFace Class Reference	2355
29.514.1	Detailed Description	2356
29.514.2	Constructor & Destructor Documentation	2356
29.514.2.1	~tpFace()	2356
29.514.3	Member Function Documentation	2356
29.514.3.1	boundingBox()=0	2356
29.514.3.2	createInitialEdges(double degenerate_epsilon=DEFAULT_SPACE_EPSILON)=0	2356
29.514.3.3	getId()=0	2356
29.514.3.4	isolateFace()	2356
29.514.3.5	normal(double u, double v) const =0	2356
29.514.3.6	point(double u, double v) const =0	2357

29.514.3.7startEdges()=0	2357
29.515Go::tpMarchPoint Class Reference	2357
29.515.1Detailed Description	2358
29.515.2Constructor & Destructor Documentation	2358
29.515.2.1tpMarchPoint(const Go::Point &p, double pa, double pa2, double d, double ca, int q)	2358
29.515.3Member Function Documentation	2358
29.515.3.1operator<(const tpMarchPoint &other) const	2358
29.515.4Member Data Documentation	2358
29.515.4.1cos_ang	2358
29.515.4.2dist	2358
29.515.4.3par	2358
29.515.4.4par2	2358
29.515.4.5pt	2358
29.515.4.6status	2359
29.516Go::tpTableEntry< edgeType > Class Template Reference	2359
29.516.1Detailed Description	2359
29.516.2Constructor & Destructor Documentation	2360
29.516.2.1tpTableEntry(edgeType *e1, edgeType *e2, const tpTopologicalInfo &info)	2360
29.516.3Member Function Documentation	2360
29.516.3.1Info()	2360
29.516.4Member Data Documentation	2360
29.516.4.1e1_	2360
29.516.4.2e2_	2360
29.516.4.3info_	2360
29.517Go::tpTolerances Struct Reference	2360
29.517.1Detailed Description	2361
29.517.2Constructor & Destructor Documentation	2361
29.517.2.1tpTolerances(double g, double n, double k, double b)	2361
29.517.2.2tpTolerances(const tpTolerances &tol)	2361
29.517.3Member Data Documentation	2361

29.517.3.1	bend	2361
29.517.3.2	gap	2361
29.517.3.3	kink	2361
29.517.3.4	neighbour	2362
29.518	tpTopologicalInfo Struct Reference	2362
29.518.1	Detailed Description	2362
29.518.2	Member Function Documentation	2362
29.518.2.1	BestStatus() const	2362
29.518.2.2	WorstStatus() const	2362
29.518.3	Member Data Documentation	2363
29.518.3.1	parameters_	2363
29.518.3.2	status_	2363
29.519	tpTopologyTable< edgeType, faceType > Class Template Reference	2363
29.519.1	Detailed Description	2364
29.519.2	Constructor & Destructor Documentation	2365
29.519.2.1	tpTopologyTable(double tol_gap, double tol_neighbour, double tol_kink, double tol_bend)	2365
29.519.2.2	~tpTopologyTable()	2365
29.519.3	Member Function Documentation	2365
29.519.3.1	addEntry(edgeType *e1, edgeType *e2, double min1, double max1, double min2, double max2, const tpTopologicalInfo &info, int status)	2365
29.519.3.2	addFaceToTable(shared_ptr< faceType > face)	2365
29.519.3.3	BoundaryLoops(std::vector< std::vector< edgeType * > > &loopvec)	2365
29.519.3.4	BoundaryLoops(std::vector< std::vector< shared_ptr< edgeType > > > &loopvec)	2365
29.519.3.5	connectedEdges() const	2366
29.519.3.6	connectTwins(edgeType *e1, edgeType *e2, double min1, double max1, double min2, double max2, int &status)	2366
29.519.3.7	constructTable(const std::vector< shared_ptr< faceType > > &faces, bool test_orientation=false)	2366
29.519.3.8	cornersAndKinks(std::vector< edgeType * > &vec)	2366
29.519.3.9	disjointObjects(std::vector< std::vector< faceType * > > &grouped_faces)	2366
29.519.3.10	edgesBoundingFace(faceType *face) const	2366

29.519.3.1	getInconsistentFaces(std::vector< std::pair< faceType *, faceType * > > &faces)	2366
29.519.3.1	getTolerances()	2367
29.519.3.1	info(int n)	2367
29.519.3.1	prepareTable(const std::vector< shared_ptr< faceType > > &faces)	2367
29.519.3.1	removeFaceFromTable(shared_ptr< faceType > face)	2367
29.519.3.1	setTable(const std::vector< shared_ptr< faceType > > &faces)	2367
29.519.3.1	setTolerances(const tpTolerances &tol)	2367
29.519.3.1	updateTableEntry(edgeType *e1, edgeType *e2)	2367
29.519.3.1	validate()	2367
29.519.4	Member Data Documentation	2368
29.519.4.1	edges_	2368
29.519.4.2	orientation_inconsist_	2368
29.519.4.3	table_	2368
29.519.4.4	tol_	2368
29.520	RBD_COMMON::Tracer Class Reference	2368
29.520.1	Detailed Description	2369
29.520.2	Constructor & Destructor Documentation	2369
29.520.2.1	Tracer(const char *)	2369
29.520.2.2	~Tracer()	2369
29.520.3	Member Function Documentation	2369
29.520.3.1	AddTrace()	2369
29.520.3.2	PrintTrace()	2369
29.520.3.3	ReName(const char *)	2369
29.520.4	Friends And Related Function Documentation	2369
29.520.4.1	BaseException	2369
29.520.5	Member Data Documentation	2370
29.520.5.1	last	2370
29.521	NEWMAT::TransposedMatrix Class Reference	2370
29.521.1	Detailed Description	2371
29.521.2	Constructor & Destructor Documentation	2371

29.521.2.1~TransposedMatrix()	2371
29.521.3Member Function Documentation	2372
29.521.3.1BandWidth() const	2372
29.521.3.2Evaluate(MatrixType mt=MatrixTypeUnSp)	2372
29.521.4Friends And Related Function Documentation	2372
29.521.4.1BaseMatrix	2372
29.522Go::Triangle Class Reference	2372
29.522.1Detailed Description	2372
29.522.2Constructor & Destructor Documentation	2373
29.522.2.1Triangle(int index)	2373
29.522.3Member Function Documentation	2373
29.522.3.1index() const	2373
29.522.3.2setIndex(int i)	2373
29.523Betriang::Triangulation Class Reference	2373
29.523.1Detailed Description	2374
29.523.2Constructor & Destructor Documentation	2374
29.523.2.1Triangulation()	2374
29.523.2.2~Triangulation()	2374
29.523.3Member Function Documentation	2374
29.523.3.1addLeadingEdge(Edge *edge)	2374
29.523.3.2checkDelaunay() const	2375
29.523.3.3cleanAll()	2375
29.523.3.4createDart()	2375
29.523.3.5createDelaunay(std::vector< Go::ttlPoint * >::iterator first, std::vector< Go::ttlPoint * >::iterator last)	2375
29.523.3.6getBoundaryEdge() const	2375
29.523.3.7getConstrainedEdges(Go::ttlPoint point) const	2375
29.523.3.8getEdges(bool skip_boundary_edges=false) const	2375
29.523.3.9getInteriorNode() const	2375
29.523.3.10getLeadingEdges() const	2375

29.523.3.1	initTwoEnclosingTriangles(std::vector< Go::ttlPoint * >::iterator first, std::vector< Go::ttlPoint * >::iterator last)	2375
29.523.3.12	Triangles() const	2376
29.523.3.13	optimizeDelaunay()	2376
29.523.3.14	printEdges(std::ofstream &os) const	2376
29.523.3.15	removeLeadingEdgeFromList(Edge *leadingEdge)	2376
29.523.3.16	removeTriangle(Edge &edge)	2376
29.523.3.17	reverse_splitTriangle(Edge &edge)	2376
29.523.3.18	splitTriangle(Edge &edge, const Go::ttlPoint &point)	2376
29.523.3.19	swapEdge(Edge &diagonal)	2376
29.523.4	Member Data Documentation	2376
29.523.4.1	leadingEdges_	2376
29.524	Med::Triangulation Class Reference	2376
29.524.1	Detailed Description	2378
29.524.2	Constructor & Destructor Documentation	2378
29.524.2.1	Triangulation()	2378
29.524.2.2	Triangulation(const Triangulation &tr)	2378
29.524.2.3	~Triangulation()	2378
29.524.3	Member Function Documentation	2378
29.524.3.1	addLeadingEdge(Edge *edge)	2378
29.524.3.2	checkDelaunay() const	2378
29.524.3.3	cleanAll()	2378
29.524.3.4	createDart()	2379
29.524.3.5	createDelaunay(std::vector< Node * >::iterator first, std::vector< Node * >::iterator last)	2379
29.524.3.6	flagNodes(bool flag) const	2379
29.524.3.7	getBoundaryEdge() const	2379
29.524.3.8	getEdges(bool skip_boundary_edges=false) const	2379
29.524.3.9	getInteriorNode() const	2379
29.524.3.10	getLeadingEdges() const	2379
29.524.3.11	getNodes() const	2379

29.524.3.11	InitTwoEnclosingTriangles(std::vector< Node * >::iterator first, std::vector< Node * >::iterator last)	2380
29.524.3.12	Triangles() const	2380
29.524.3.13	OptimizeDelaunay()	2380
29.524.3.14	PrintEdges(std::ofstream &os) const	2380
29.524.3.15	RemoveLeadingEdgeFromList(Edge *leadingEdge)	2380
29.524.3.16	RemoveTriangle(Edge &edge)	2380
29.524.3.17	Reverse_splitTriangle(Edge &edge)	2380
29.524.3.18	SplitTriangle(Edge &edge, Node &point)	2380
29.524.3.19	SwapEdge(Edge &diagonal)	2381
29.524.4	Member Data Documentation	2381
29.524.4.1	leadingEdges_	2381
29.525	Go::TrimCurve Class Reference	2381
29.525.1	Detailed Description	2382
29.525.2	Constructor & Destructor Documentation	2382
29.525.2.1	TrimCurve(CurveOnSurface *bd_crv)	2382
29.525.2.2	TrimCurve(CurveOnSurface *bd_crv, double start, double end)	2382
29.525.2.3	TrimCurve(Point startpt, Point endpt, CurveOnSurface *bd_crv)	2382
29.525.2.4	~TrimCurve()	2382
29.525.3	Member Function Documentation	2382
29.525.3.1	approximationOK(double par, const std::vector< Point > &approxpos, double tol1, double tol2)	2382
29.525.3.2	dim()	2383
29.525.3.3	end()	2383
29.525.3.4	eval(double t)	2383
29.525.3.5	eval(double t, int n, std::vector< std::vector< Point > > &der)	2383
29.525.3.6	mbCvs()	2384
29.525.3.7	start()	2384
29.526	Go::ttlPoint Class Reference	2384
29.526.1	Detailed Description	2385
29.526.2	Constructor & Destructor Documentation	2385

29.526.2.1	ttlPoint(PointIter pnt_iter, double x, double y, double z=0.0)	2385
29.526.2.2	~ttlPoint()	2385
29.526.3	Member Function Documentation	2385
29.526.3.1	pnt_iter() const	2385
29.526.3.2	x() const	2385
29.526.3.3	y() const	2385
29.526.3.4	z() const	2385
29.527	ttl::TTLtraits Struct Reference	2386
29.527.1	Detailed Description	2387
29.527.2	Member Typedef Documentation	2387
29.527.2.1	real_type	2387
29.527.3	Member Function Documentation	2387
29.527.3.1	crossProduct2d(const Dart &v1, const Dart &v2)	2387
29.527.3.2	crossProduct2d(const Dart &v, const Node &p)	2387
29.527.3.3	orient2d(const Dart &n1, const Dart &n2, const Node &p)	2388
29.527.3.4	orient2d(const Dart &n1, const Dart &n2, const Dart &p)	2388
29.527.3.5	removeBoundaryTriangle(Dart &d)	2388
29.527.3.6	reverse_splitTriangle(Dart &dart)	2388
29.527.3.7	scalarProduct2d(const Dart &v1, const Dart &v2)	2388
29.527.3.8	scalarProduct2d(const Dart &v, const Node &p)	2388
29.527.3.9	splitTriangle(Dart &dart, Node &point)	2388
29.527.3.10	swapEdge(Dart &dart)	2389
29.527.4	Member Data Documentation	2389
29.527.4.1	triang_	2389
29.528	etriang::TTLtraits Struct Reference	2389
29.528.1	Detailed Description	2391
29.528.2	Member Typedef Documentation	2391
29.528.2.1	real_type	2391
29.528.3	Member Function Documentation	2391
29.528.3.1	crossProduct2d(const Dart &v1, const Dart &v2)	2391

29.528.3.2	crossProduct2d(const Dart &v, const Go::tllPoint &p)	2391
29.528.3.3	orient2d(const Dart &n1, const Dart &n2, const Go::tllPoint &p)	2392
29.528.3.4	orient2d(const Dart &n1, const Dart &n2, const Dart &p)	2392
29.528.3.5	removeBoundaryTriangle(Dart &d)	2392
29.528.3.6	reverse_splitTriangle(Dart &dart)	2392
29.528.3.7	scalarProduct2d(const Dart &v1, const Dart &v2)	2392
29.528.3.8	scalarProduct2d(const Dart &v, const Go::tllPoint &p)	2392
29.528.3.9	splitTriangle(Dart &dart, Go::tllPoint &point)	2392
29.528.3.10	swapEdge(Dart &dart)	2393
29.528.4	Member Data Documentation	2393
29.528.4.1	triang_	2393
29.529	InfNodeType Class Reference	2393
29.529.1	Detailed Description	2394
29.529.2	Member Function Documentation	2394
29.529.2.1	printNode()	2394
29.529.3	Member Data Documentation	2394
29.529.3.1	ln_	2394
29.529.3.2	origine_	2394
29.530	Go::CoonsPatchGen::UnKnownError Class Reference	2394
29.530.1	Detailed Description	2394
29.531	NEWMAT::UpperBandMatrix Class Reference	2395
29.531.1	Detailed Description	2397
29.531.2	Constructor & Destructor Documentation	2397
29.531.2.1	UpperBandMatrix()	2397
29.531.2.2	~UpperBandMatrix()	2397
29.531.2.3	UpperBandMatrix(int n, int ubw)	2397
29.531.2.4	UpperBandMatrix(const BaseMatrix &)	2397
29.531.2.5	UpperBandMatrix(const UpperBandMatrix &gm)	2397
29.531.3	Member Function Documentation	2397
29.531.3.1	element(int, int)	2397

29.531.3.2	element(int, int) const	2397
29.531.3.3	LogDeterminant() const	2397
29.531.3.4	MakeSolver()	2397
29.531.3.5	operator()(int, int)	2398
29.531.3.6	operator()(int, int) const	2398
29.531.3.7	operator=(const BaseMatrix &)	2398
29.531.3.8	operator=(Real f)	2398
29.531.3.9	operator=(const UpperBandMatrix &m)	2398
29.531.3.10	ReSize(int, int, int)	2398
29.531.3.11	ReSize(int n, int ubw)	2398
29.531.3.12	ReSize(const GeneralMatrix &A)	2398
29.531.3.13	Solver(MatrixColX &, const MatrixColX &)	2398
29.531.3.14	Type() const	2398
29.532	NEWMAT::UpperTriangularMatrix Class Reference	2399
29.532.1	Detailed Description	2400
29.532.2	Constructor & Destructor Documentation	2400
29.532.2.1	UpperTriangularMatrix()	2400
29.532.2.2	~UpperTriangularMatrix()	2400
29.532.2.3	UpperTriangularMatrix(ArrayLengthSpecifier)	2401
29.532.2.4	UpperTriangularMatrix(const BaseMatrix &)	2401
29.532.2.5	UpperTriangularMatrix(const UpperTriangularMatrix &gm)	2401
29.532.3	Member Function Documentation	2401
29.532.3.1	BandWidth() const	2401
29.532.3.2	element(int, int)	2401
29.532.3.3	element(int, int) const	2401
29.532.3.4	GetCol(MatrixRowCol &)	2401
29.532.3.5	GetCol(MatrixColX &)	2401
29.532.3.6	GetRow(MatrixRowCol &)	2401
29.532.3.7	LogDeterminant() const	2401
29.532.3.8	MakeSolver()	2401

29.532.3.9	NextRow(MatrixRowCol &)	2402
29.532.3.10	operator()(int, int)	2402
29.532.3.11	operator()(int, int) const	2402
29.532.3.12	operator=(const BaseMatrix &)	2402
29.532.3.13	operator=(const UpperTriangularMatrix &m)	2402
29.532.3.14	operator=(Real f)	2402
29.532.3.15	ReSize(int)	2402
29.532.3.16	ReSize(const GeneralMatrix &A)	2402
29.532.3.17	RestoreCol(MatrixRowCol &)	2402
29.532.3.18	RestoreCol(MatrixColX &c)	2402
29.532.3.19	Solver(MatrixColX &, const MatrixColX &)	2402
29.532.3.20	Trace() const	2403
29.532.3.21	Type() const	2403
29.533	vector3t< T > Class Template Reference	2403
29.533.1	Detailed Description	2405
29.533.2	Constructor & Destructor Documentation	2405
29.533.2.1	vector3t()	2405
29.533.2.2	vector3t(const T a, const T b, const T c)	2405
29.533.2.3	vector3t(const float *a)	2405
29.533.2.4	vector3t(const double *a)	2405
29.533.2.5	~vector3t(void)	2405
29.533.3	Member Function Documentation	2406
29.533.3.1	clamp(const vector3t &v0, const vector3t &v1)	2406
29.533.3.2	conv(const vector3t &v)	2406
29.533.3.3	dequal(const vector3t< T > &v) const	2406
29.533.3.4	dequal2(const vector3t< T > &v) const	2406
29.533.3.5	dequal3(const vector3t< T > &v) const	2406
29.533.3.6	dequal_2d(const vector3t< T > &v) const	2406
29.533.3.7	length(void) const	2406
29.533.3.8	length_squared(void) const	2406

29.533.3.9	<code>max(const vector3t &v)</code>	2406
29.533.3.10	<code>max_coo(void) const</code>	2406
29.533.3.11	<code>min(const vector3t &v)</code>	2407
29.533.3.12	<code>min_coo(void) const</code>	2407
29.533.3.13	<code>normalize(void)</code>	2407
29.533.3.14	<code>normalized(void) const</code>	2407
29.533.3.15	<code>operator!=(const vector3t &v) const</code>	2407
29.533.3.16	<code>operator*(const vector3t &v) const</code>	2407
29.533.3.17	<code>operator*=(const double x)</code>	2407
29.533.3.18	<code>operator+(const vector3t &v) const</code>	2407
29.533.3.19	<code>operator+=(const vector3t &v)</code>	2407
29.533.3.20	<code>operator+=(const T &d)</code>	2407
29.533.3.21	<code>operator-(const vector3t &v) const</code>	2408
29.533.3.22	<code>operator-(const T &d) const</code>	2408
29.533.3.23	<code>operator-(void) const</code>	2408
29.533.3.24	<code>operator==(const vector3t &v)</code>	2408
29.533.3.25	<code>operator==(const T &d)</code>	2408
29.533.3.26	<code>operator/(const vector3t &v) const</code>	2408
29.533.3.27	<code>operator/=(const double x)</code>	2408
29.533.3.28	<code>operator<(const vector3t &v) const</code>	2408
29.533.3.29	<code>operator==(const vector3t &v) const</code>	2408
29.533.3.30	<code>operator>(const vector3t &v) const</code>	2408
29.533.3.31	<code>print(void) const</code>	2409
29.533.3.32	<code>print2(void) const</code>	2409
29.533.3.33	<code>print3(void) const</code>	2409
29.533.3.34	<code>raw(void) const</code>	2409
29.533.3.35	<code>reciprocal(void) const</code>	2409
29.533.3.36	<code>scale(const vector3t &mi, const vector3t &ma, const vector3t &new_mi, const vector3t &new_ma)</code>	2409
29.533.3.37	<code>tate_xy(const double cosa, const double sina)</code>	2409
29.533.3.38	<code>tate_xz(const double cosa, const double sina)</code>	2409

29.533.3.39	state_yz(const double cosa, const double sina)	2409
29.533.3.40	setx(const T a)	2409
29.533.3.41	sety(const T a)	2410
29.533.3.42	setz(const T a)	2410
29.533.3.43	(void) const	2410
29.533.3.44	(void)	2410
29.533.3.45	(void) const	2410
29.533.3.46	(void)	2410
29.533.3.47	(void) const	2410
29.533.3.48	(void)	2410
29.533.4	Friends And Related Function Documentation	2410
29.533.4.1	conv2	2410
29.533.4.2	cosangle	2410
29.533.4.3	operator*	2411
29.533.4.4	operator+	2411
29.533.4.5	operator+	2411
29.533.5	Member Data Documentation	2411
29.533.5.1	coo	2411
29.534	NEWMAT::VectorException Class Reference	2411
29.534.1	Detailed Description	2412
29.534.2	Constructor & Destructor Documentation	2412
29.534.2.1	VectorException()	2412
29.534.2.2	VectorException(const GeneralMatrix &A)	2412
29.534.3	Member Data Documentation	2413
29.534.3.1	Select	2413
29.535	Go::Vertex Class Reference	2413
29.535.1	Detailed Description	2414
29.535.2	Constructor & Destructor Documentation	2415
29.535.2.1	Vertex(Point vertex_point)	2415
29.535.2.2	Vertex(Point vertex_point, std::vector< ftEdge * > edges)	2415

29.535.2.3	Vertex(Point vertex_point, ftEdge *edges)	2415
29.535.2.4	Vertex(ftEdge *edge, bool at_start)	2415
29.535.2.5	~Vertex()	2415
29.535.3	Member Function Documentation	2415
29.535.3.1	addEdge(ftEdge *edge)	2415
29.535.3.2	allEdges() const	2415
29.535.3.3	averageVertexPos()	2415
29.535.3.4	checkVertexTopology()	2415
29.535.3.5	connectedToSameVertex(Vertex *other) const	2416
29.535.3.6	disconnectTwin(ftEdge *edge)	2416
29.535.3.7	faces() const	2416
29.535.3.8	faces(Body *bd) const	2416
29.535.3.9	freeEdges()	2416
29.535.3.10	getBodies()	2416
29.535.3.11	getCommonEdge(Vertex *other) const	2416
29.535.3.12	getCommonEdgeInFace(Vertex *other, ftSurface *face) const	2416
29.535.3.13	getCommonFaces(Vertex *other) const	2416
29.535.3.14	getCommonVertex(Vertex *other) const	2416
29.535.3.15	getDist(shared_ptr< Vertex > other_point)	2417
29.535.3.16	getEdge(int idx)	2417
29.535.3.17	getEdgeDiscontinuities(std::vector< std::pair< ftEdge *, ftEdge * > > &gaps, double tol, std::vector< std::pair< ftEdge *, ftEdge * > > &kinks, double angtol, double angmax) const	2417
29.535.3.18	getFaceEdges(ftSurface *face) const	2417
29.535.3.19	getFacePar(ftSurface *face)	2417
29.535.3.20	getFaces()	2417
29.535.3.21	getFaces(Body *bd)	2417
29.535.3.22	getNextVertex(ftSurface *face) const	2417
29.535.3.23	getVertexPoint()	2417
29.535.3.24	hasEdge(ftEdge *edge) const	2418
29.535.3.25	hasEdgeSingle(ftEdge *edge) const	2418

29.535.3.26	IsFace(ftSurface *face) const	2418
29.535.3.27	BoundaryVertex() const	2418
29.535.3.28	CornerInFace(ftSurface *face, double tol)	2418
29.535.3.29	InVertex(shared_ptr< Vertex > other)	2418
29.535.3.30	MeetInVertex(ftEdge *e1, ftEdge *e2) const	2418
29.535.3.31	mbUniqueEdges()	2418
29.535.3.32	mbUniqueEdges(Body *bd)	2418
29.535.3.33	moveEdge(ftEdge *edge)	2418
29.535.3.34	Organize()	2419
29.535.3.35	sameEdge(Vertex *other) const	2419
29.535.3.36	sameFace(Vertex *other) const	2419
29.535.3.37	sameUnderlyingSurface(Vertex *other) const	2419
29.535.3.38	VertexPoint(Point vertex_point)	2419
29.535.3.39	UniqueEdges()	2419
29.535.3.40	UniqueEdges(Body *bd)	2419
29.536	VolBoundaryCondition Class Reference	2420
29.536.1	Detailed Description	2421
29.536.2	Constructor & Destructor Documentation	2421
29.536.2.1	VolBoundaryCondition(int face_nmb, BdConditionType type, BdCondFuncor *fbd, std::vector< std::pair< double, double > > &domain, VolSolution *solution)	2421
29.536.2.2	VolBoundaryCondition(int face_nmb, BdConditionType type, const Point &const_val, std::vector< std::pair< double, double > > &domain, VolSolution *solution)	2421
29.536.2.3	~VolBoundaryCondition()	2421
29.536.3	Member Function Documentation	2421
29.536.3.1	faceNumber() const	2421
29.536.3.2	getBasisFunctions(int index_of_Gauss_point1, int index_of_Gauss_point2, int &const_dir, vector< double > &basisValues, vector< double > &basisDerivs_u, vector< double > &basisDerivs_v) const	2421
29.536.3.3	getBdCoefficients(std::vector< std::pair< int, Point > > &coefs)	2421
29.536.3.4	getBdCoefficients(std::vector< std::pair< int, Point > > &coefs_bd, std::vector< std::pair< int, Point > > &coefs_inner)	2421
29.536.3.5	getCoefficientsEnumeration(std::vector< int > &local_enumeration)	2421

29.536.3.6	getCoefficientsEnumeration(std::vector< int > &local_enumeration_bd, std::vector< int > &local_enumeration_bd2)	2421
29.536.3.7	getSplineApproximation() const	2422
29.536.3.8	getTolerances() const	2422
29.536.3.9	update()	2422
29.536.3.10	updateBoundaryValue(BdCondFuncor *fbd)	2422
29.537	Go::VolPointBdCond Class Reference	2422
29.537.1	Detailed Description	2423
29.537.2	Constructor & Destructor Documentation	2423
29.537.2.1	VolPointBdCond(int face_nmb, double param[], Point &condition_value)	2423
29.537.2.2	~VolPointBdCond()	2423
29.537.3	Member Function Documentation	2423
29.537.3.1	faceNumber() const	2423
29.537.3.2	getCoefficientsEnumeration(std::vector< int > &local_enumeration) const	2423
29.537.3.3	getConditionValue() const	2423
29.537.3.4	getInterpolationFactors(std::vector< std::pair< int, double > > &factors) const	2423
29.537.3.5	getParam() const	2423
29.538	Go::VolSolution Class Reference	2424
29.538.1	Detailed Description	2425
29.538.2	Constructor & Destructor Documentation	2426
29.538.2.1	VolSolution(IsogeometricVolBlock *parent, shared_ptr< SplineVolume > sol_vol)	2426
29.538.2.2	~VolSolution()	2426
29.538.3	Member Function Documentation	2426
29.538.3.1	addBoundaryCondition(int face_nmb, BdConditionType type, Go::Point &const_val, std::vector< std::pair< double, double > > &domain)	2426
29.538.3.2	addBoundaryCondition(int face_nmb, BdConditionType type, BdCondFuncor *fbd, std::vector< std::pair< double, double > > &domain)	2426
29.538.3.3	addDirichletPointBdCond(double param[], Point &condition_value)	2426
29.538.3.4	asVolSolution()	2426
29.538.3.5	basis(int paddir) const	2426
29.538.3.6	degree(int paddir) const	2426
29.538.3.7	dimension() const	2426

29.538.3.8	distinctKnots(int paddir) const	2426
29.538.3.9	erasePreEvaluatedBasisFunctions()	2426
29.538.3.10	getBasisFunctions(int index_of_Gauss_point1, int index_of_Gauss_point2, int index_of_Gauss_point3, vector< double > &basisValues, vector< double > &basisDerivs_u, vector< double > &basisDerivs_v, vector< double > &basisDerivs_w) const	2427
29.538.3.11	getBasisFunctions(double param1, double param2, double param3, vector< double > &basisValues, vector< double > &basisDerivs_u, vector< double > &basisDerivs_v, vector< double > &basisDerivs_w) const	2427
29.538.3.12	getBasisFunctionValues(int basis_func_id_u, int basis_func_id_v, int basis_func_id_w, std::vector< int > &index_of_Gauss_points1, std::vector< int > &index_of_Gauss_points2, std::vector< int > &index_of_Gauss_points3, std::vector< double > &basisValues, std::vector< double > &basisDerivs_u, std::vector< double > &basisDerivs_v, std::vector< double > &basisDerivs_w) const	2427
29.538.3.13	getBasisFunctionValues(int basis_func_id_u, int basis_func_id_v, int basis_func_id_w, int knot_ind_u, int knot_ind_v, int knot_ind_w, std::vector< int > &index_of_Gauss_points1, std::vector< int > &index_of_Gauss_points2, std::vector< int > &index_of_Gauss_points3, std::vector< double > &basisValues, std::vector< double > &basisDerivs_u, std::vector< double > &basisDerivs_v, std::vector< double > &basisDerivs_w) const	2427
29.538.3.14	getBoundaryCoefficients(int boundary, std::vector< int > &enumeration) const	2427
29.538.3.15	getBoundaryCoefficients(int boundary, std::vector< int > &enumeration_bd, std::vector< int > &enumeration_bd2) const	2427
29.538.3.16	getBoundaryCondition(int index) const	2427
29.538.3.17	getFaceBoundaryConditions(int face_number, std::vector< shared_ptr< VolBoundaryCondition > > &bd_cond) const	2427
29.538.3.18	getFaceBoundaryConditions(std::vector< shared_ptr< VolBoundaryCondition > > &bd_cond) const	2427
29.538.3.19	getFacePointBdConditions(int face_number, std::vector< shared_ptr< VolPointBdCond > > &bd_cond) const	2427
29.538.3.20	getGaussParameter(int index_of_Gauss_point1, int index_of_Gauss_point2, int const_dir, double &par1, double &par2) const	2427
29.538.3.21	getGeometryVolume() const	2427
29.538.3.22	getJacobian(std::vector< int > &index_of_Gauss_point) const	2427
29.538.3.23	getMatchingCoefficients(BlockSolution *other, std::vector< std::pair< int, int > > &enumeration, int match_pos=0) const	2428
29.538.3.24	getNmbOfBoundaryConditions() const	2428
29.538.3.25	getNmbOfPointBdConditions() const	2428
29.538.3.26	getPointBdCond(std::vector< shared_ptr< VolPointBdCond > > &bd_cond) const	2428

29.538.3.27	GetPointBdCondition(int index) const	2428
29.538.3.28	GetSolutionVolume() const	2428
29.538.3.29	GetTolerances() const	2428
29.538.3.30	IncreaseDegree(int new_degree, int paddir)	2428
29.538.3.31	InsertKnots(const std::vector< int > &knot_intervals, int paddir)	2428
29.538.3.32	InsertKnots(const std::vector< double > &knots, int paddir)	2428
29.538.3.33	Knots(int paddir) const	2428
29.538.3.34	MakeMatchingSplineSpace(BlockSolution *other)	2428
29.538.3.35	MatchingSplineSpace(BlockSolution *other) const	2429
29.538.3.36	NbCoefs() const	2429
29.538.3.37	NbCoefs(int paddir) const	2429
29.538.3.38	PerformPreEvaluation(std::vector< std::vector< double > > &Gauss_par)	2429
29.538.3.39	RaiseToGeometry(int paddir)	2429
29.538.3.40	SetMinimumDegree(int degree)	2429
29.538.3.41	SetSolutionCoefficients(const std::vector< double > &coefs)	2429
29.538.3.42	UpdateConditions()	2429
29.538.3.43	ValuesInGaussPoint(const std::vector< int > &index_of_Gauss_point, std::vector< Point > &derivs) const	2429
29.539	Go::VolumeAdjacency Class Reference	2429
29.539.1	Detailed Description	2430
29.539.2	Constructor & Destructor Documentation	2430
29.539.2.1	VolumeAdjacency(double gap, double neighbour)	2430
29.539.2.2	~VolumeAdjacency()	2430
29.539.3	Member Function Documentation	2430
29.539.3.1	setAdjacency(std::vector< shared_ptr< Body > > &solids)	2430
29.539.3.2	setAdjacency(std::vector< shared_ptr< Body > > &solids, int new_solid_pos)	2430
29.540	Go::VolumeAdjacencyInfo Struct Reference	2431
29.540.1	Detailed Description	2432
29.540.2	Constructor & Destructor Documentation	2432
29.540.2.1	VolumeAdjacencyInfo()	2432
29.540.3	Member Data Documentation	2432

29.540.3.1adjacency_found_	2432
29.540.3.2bd_idx_1_	2432
29.540.3.3bd_idx_2_	2432
29.540.3.4corner_adjacency_	2432
29.540.3.5corner_failed_	2432
29.540.3.6edg_idx_1_	2433
29.540.3.7edg_idx_2_	2433
29.540.3.8same_dir_order_	2433
29.540.3.9same_orient_edge_	2433
29.540.3.10same_orient_u_	2433
29.540.3.11same_orient_v_	2433
29.540. Go::VolumeModel Class Reference	2434
29.541. 1Detailed Description	2436
29.541. 2Constructor & Destructor Documentation	2436
29.541.2.1VolumeModel(std::vector< shared_ptr< ftVolume > > &volumes, double gap, double neighbour, double kink, double bend, bool adjacency_set=false)	2436
29.541.2.2VolumeModel(std::vector< shared_ptr< ftVolume > > &volumes, double gap, double kink)	2437
29.541.2.3VolumeModel(double gap, double neighbour, double kink, double bend)	2437
29.541.2.4VolumeModel(const VolumeModel &vm)	2437
29.541.2.5~VolumeModel()	2437
29.541. 3Member Function Documentation	2437
29.541.3.1allSplines() const	2437
29.541.3.2append(shared_ptr< ftVolume > volume)	2437
29.541.3.3append(std::vector< shared_ptr< ftVolume > > volumes)	2437
29.541.3.4append(shared_ptr< VolumeModel > anotherModel)	2437
29.541.3.5averageCorrespondingCoefs()	2437
29.541.3.6boundingBox()	2438
29.541.3.7boundingBox(int idx) const	2438
29.541.3.8buildTopology()	2438
29.541.3.9buildTopology(shared_ptr< ftVolume > body)	2438

29.541.3.16	checkModelTopology()	2438
29.541.3.17	clone() const	2438
29.541.3.18	ClosestPoint(Point &pnt, Point &clo_pnt, int &idx, double clo_par[], double &dist)	2438
29.541.3.19	Curvature(int idx, double *par) const	2438
29.541.3.20	evaluate(int idx, double par[], Point &pnt) const	2439
29.541.3.21	evaluate(int idx, double par[], int nder, std::vector< Point > &der) const	2439
29.541.3.22	ExtremalPoint(Point &dir, Point &clo_pnt, int &idx, double ext_par[])	2439
29.541.3.23	FetchAsSharedPtr(Body *body) const	2439
29.541.3.24	GetAllVertices(std::vector< shared_ptr< Vertex > > &vertices) const	2439
29.541.3.25	GetApproximationTol() const	2439
29.541.3.26	GetBody(int idx) const	2440
29.541.3.27	GetBoundaryFaces() const	2440
29.541.3.28	GetBoundaryFaces(int boundary_idx) const	2440
29.541.3.29	GetConnectedModels()	2440
29.541.3.30	GetIndex(shared_ptr< ftVolume > body) const	2440
29.541.3.31	GetIndex(ftVolume *body) const	2440
29.541.3.32	GetOuterBoundary(int idx) const	2440
29.541.3.33	GetRadialEdges(std::vector< shared_ptr< EdgeVertex > > &rad_edges) const	2440
29.541.3.34	GetSplineVolume(int idx) const	2441
29.541.3.35	GetUniqueInnerFaces() const	2441
29.541.3.36	GetVolume(int idx) const	2441
29.541.3.37	Intersect(const ftLine &line)	2441
29.541.3.38	Intersect_plane(const ftPlane &plane)	2441
29.541.3.39	IsCornerToCorner(double tol=DEFAULT_SPACE_EPSILON) const	2441
29.541.3.40	IsDegenerate(int idx) const	2441
29.541.3.41	MakeCommonSplineSpaces()	2441
29.541.3.42	MakeCornerToCorner(double tol=DEFAULT_SPACE_EPSILON)	2441
29.541.3.43	mbBoundaries() const	2442
29.541.3.44	mbEntities() const	2442
29.541.3.45	RegularizeBdShells()	2442

29.541.3.40	moveSolid(shared_ptr< ftVolume > vol)	2442
29.541.3.41	replaceNonRegVolumes(int degree=3, int split_mode=1)	2442
29.541.3.42	setBoundarySfs()	2442
29.541.3.43	setVertexIdentity()	2442
29.541.3.44	tessellate(std::vector< shared_ptr< GeneralMesh > > &meshes) const	2442
29.541.3.45	tessellate(int resolution[], std::vector< shared_ptr< GeneralMesh > > &meshes) const	2442
29.541.3.46	tessellate(double density, std::vector< shared_ptr< GeneralMesh > > &meshes) const	2443
29.541.3.47	tessellatedCtrPolygon(std::vector< shared_ptr< LineCloud > > &ctr_pol) const	2443
29.541.3.48	turn(int idx)	2443
29.541.3.49	turn()	2443
29.541.3.50	uniqueNonRadialEdges(std::vector< shared_ptr< ftEdge > > &edges) const	2443
29.542	Go::VolumeModelFileHandler Class Reference	2444
29.542.1	Detailed Description	2444
29.542.2	Constructor & Destructor Documentation	2445
29.542.2.1	VolumeModelFileHandler()	2445
29.542.2.2	~VolumeModelFileHandler()	2445
29.542.3	Member Function Documentation	2445
29.542.3.1	readVolume(const char *filein, int id=-1)	2445
29.542.3.2	readVolumeModel(const char *filein)	2445
29.542.3.3	writeVolume(const shared_ptr< ftVolume > &body, std::ostream &os, int body← _id=-1, bool faces=true)	2445
29.542.3.4	writeVolumeModel(VolumeModel &vol_model, std::ostream &os)	2445
29.543	Go::VolumeParameterCurve Class Reference	2445
29.543.1	Detailed Description	2446
29.543.2	Constructor & Destructor Documentation	2446
29.543.2.1	VolumeParameterCurve(shared_ptr< ParamVolume > vol, shared_ptr< ParamCurve > crv)	2446
29.543.2.2	VolumeParameterCurve(shared_ptr< ParamVolume > vol, shared_ptr< ParamCurve > crv, shared_ptr< Point > par1, shared_ptr< Point > par2)	2447
29.543.2.3	~VolumeParameterCurve()	2447
29.543.3	Member Function Documentation	2447

29.543.3.1	<code>approximationOK(double par, Point approxpos, double tol1, double tol2) const</code>	2447
29.543.3.2	<code>dim() const</code>	2447
29.543.3.3	<code>end() const</code>	2448
29.543.3.4	<code>eval(double t) const</code>	2448
29.543.3.5	<code>eval(double t, int n, Point der[]) const</code>	2448
29.543.3.6	<code>start() const</code>	2448
29.544	<code>Go::VolumeSpaceCurve Class Reference</code>	2449
29.544.1	Detailed Description	2450
29.544.2	Constructor & Destructor Documentation	2450
29.544.2.1	<code>VolumeSpaceCurve(shared_ptr< ParamVolume > vol, shared_ptr< ParamCurve > crv)</code>	2450
29.544.2.2	<code>~VolumeSpaceCurve()</code>	2450
29.544.3	Member Function Documentation	2450
29.544.3.1	<code>approximationOK(double par, Point approxpos, double tol1, double tol2) const</code>	2450
29.544.3.2	<code>dim() const</code>	2451
29.544.3.3	<code>end() const</code>	2451
29.544.3.4	<code>eval(double t) const</code>	2451
29.544.3.5	<code>eval(double t, int n, Point der[]) const</code>	2451
29.544.3.6	<code>start() const</code>	2452
29.545	<code>Go::Zero_Parameter_Span_Error Class Reference</code>	2452
29.545.1	Detailed Description	2452

30 File Documentation	2453
30.1 compositemodel/include/GoTools/compositemodel/AdaptSurface.h File Reference	2453
30.2 compositemodel/include/GoTools/compositemodel/Body.h File Reference	2454
30.3 compositemodel/include/GoTools/compositemodel/CellDivision.h File Reference	2455
30.4 compositemodel/include/GoTools/compositemodel/cmUtils.h File Reference	2457
30.5 compositemodel/include/GoTools/compositemodel/CompleteEdgeNet.h File Reference	2458
30.6 compositemodel/include/GoTools/compositemodel/CompositeCurve.h File Reference	2459
30.7 compositemodel/include/GoTools/compositemodel/compositemodel-doxymain.h File Reference	2460
30.8 compositemodel/include/GoTools/compositemodel/CompositeModel.h File Reference	2460
30.9 compositemodel/include/GoTools/compositemodel/CompositeModelFactory.h File Reference	2461
30.10 compositemodel/include/GoTools/compositemodel/CompositeModelFileHandler.h File Reference	2462
30.11 compositemodel/include/GoTools/compositemodel/CurveModel.h File Reference	2463
30.12 compositemodel/include/GoTools/compositemodel/EdgeVertex.h File Reference	2464
30.13 compositemodel/include/GoTools/compositemodel/EvalOffsetSurface.h File Reference	2465
30.14 compositemodel/include/GoTools/compositemodel/examples_compositemodel_doxygen.h File Reference	2465
30.15 compositemodel/include/GoTools/compositemodel/FaceUtilities.h File Reference	2465
30.16 compositemodel/include/GoTools/compositemodel/ftChartSurface.h File Reference	2467
30.17 compositemodel/include/GoTools/compositemodel/ftCurve.h File Reference	2467
30.18 compositemodel/include/GoTools/compositemodel/ftEdge.h File Reference	2469
30.19 compositemodel/include/GoTools/compositemodel/ftEdgeBase.h File Reference	2470
30.20 compositemodel/include/GoTools/compositemodel/ftFaceBase.h File Reference	2471
30.21 compositemodel/include/GoTools/compositemodel/ftLine.h File Reference	2471
30.22 compositemodel/include/GoTools/compositemodel/ftMessage.h File Reference	2473
30.23 compositemodel/include/GoTools/compositemodel/ftPlanarGraph.h File Reference	2474
30.24 compositemodel/include/GoTools/compositemodel/ftPlane.h File Reference	2476
30.25 compositemodel/include/GoTools/compositemodel/ftPoint.h File Reference	2477
30.26 compositemodel/include/GoTools/compositemodel/ftPointSet.h File Reference	2478
30.27 compositemodel/include/GoTools/compositemodel/ftSmoothSurf.h File Reference	2479
30.28 compositemodel/include/GoTools/compositemodel/ftSSfEdge.h File Reference	2479
30.29 compositemodel/include/GoTools/compositemodel/ftSurface.h File Reference	2480

30.30	compositemodel/include/GoTools/compositemodel/ftSurfaceSet.h File Reference	2481
30.31	compositemodel/include/GoTools/compositemodel/ftSurfaceSetPoint.h File Reference	2483
30.32	compositemodel/include/GoTools/compositemodel/HermiteApprEvalSurf.h File Reference	2484
30.33	compositemodel/include/GoTools/compositemodel/IntResultsCompCv.h File Reference	2484
30.34	compositemodel/include/GoTools/compositemodel/IntResultsModel.h File Reference	2485
30.35	compositemodel/include/GoTools/compositemodel/IntResultsSfModel.h File Reference	2487
30.36	compositemodel/include/GoTools/compositemodel/Loop.h File Reference	2487
30.37	compositemodel/include/GoTools/compositemodel/OffsetSurfaceUtils.h File Reference	2488
30.37.1	Enumeration Type Documentation	2489
30.37.1.1	OffsetSurfaceStatus	2489
30.38	compositemodel/include/GoTools/compositemodel/Path.h File Reference	2489
30.39	compositemodel/include/GoTools/compositemodel/PointOnEdge.h File Reference	2490
30.40	compositemodel/include/GoTools/compositemodel/PointSetApp.h File Reference	2491
30.41	compositemodel/include/GoTools/compositemodel/RegularizeFace.h File Reference	2491
30.42	compositemodel/include/GoTools/compositemodel/RegularizeFaceSet.h File Reference	2492
30.43	compositemodel/include/GoTools/compositemodel/RegularizeUtils.h File Reference	2493
30.44	compositemodel/include/GoTools/compositemodel/SplitModelUtils.h File Reference	2494
30.45	compositemodel/include/GoTools/compositemodel/SurfaceModel.h File Reference	2495
30.46	compositemodel/include/GoTools/compositemodel/SurfaceModelUtils.h File Reference	2496
30.47	compositemodel/include/GoTools/compositemodel/ttlDart.h File Reference	2497
30.48	compositemodel/include/GoTools/compositemodel/ttlPoint.h File Reference	2498
30.49	compositemodel/include/GoTools/compositemodel/ttlTraits.h File Reference	2499
30.50	compositemodel/include/GoTools/compositemodel/ttlTriang.h File Reference	2500
30.51	compositemodel/include/GoTools/compositemodel/Vertex.h File Reference	2501
30.52	doc/html/2dpoly__for__s2m_8h.js File Reference	2502
30.53	doc/html/_adapt_surface_8h.js File Reference	2502
30.53.1	Variable Documentation	2502
30.53.1.1	_adapt_surface_8h	2502
30.54	doc/html/_array_8h.js File Reference	2502
30.54.1	Variable Documentation	2502

30.54.1.1 _array_8h	2502
30.55doc/html/ _bary_coord_system_8h.js File Reference	2503
30.55.1 Variable Documentation	2503
30.55.1.1 _bary_coord_system_8h	2503
30.56doc/html/ _bd_condition_type_8h.js File Reference	2503
30.56.1 Variable Documentation	2503
30.56.1.1 _bd_condition_type_8h	2503
30.57doc/html/ _bernstein_multi_8h.js File Reference	2504
30.57.1 Variable Documentation	2504
30.57.1.1 _bernstein_multi_8h	2504
30.58doc/html/ _bernstein_poly_8h.js File Reference	2504
30.58.1 Variable Documentation	2504
30.58.1.1 _bernstein_poly_8h	2504
30.59doc/html/ _bernstein_tetrahedral_poly_8h.js File Reference	2505
30.59.1 Variable Documentation	2505
30.59.1.1 _bernstein_tetrahedral_poly_8h	2505
30.60doc/html/ _bernstein_triangular_poly_8h.js File Reference	2505
30.60.1 Variable Documentation	2505
30.60.1.1 _bernstein_triangular_poly_8h	2505
30.61doc/html/ _bernstein_utils_8h.js File Reference	2506
30.61.1 Variable Documentation	2506
30.61.1.1 _bernstein_utils_8h	2506
30.62doc/html/ _bezier_triangle_8h.js File Reference	2506
30.62.1 Variable Documentation	2506
30.62.1.1 _bezier_triangle_8h	2506
30.63doc/html/ _bounded_utils_8h.js File Reference	2507
30.63.1 Variable Documentation	2507
30.63.1.1 _bounded_utils_8h	2507
30.64doc/html/ _bounding_box_8h.js File Reference	2507
30.64.1 Variable Documentation	2507

30.64.1.1 _bounding_box_8h	2507
30.65doc/html/ _bspline_basis_8h.js File Reference	2507
30.65.1 Variable Documentation	2507
30.65.1.1 _bspline_basis_8h	2507
30.66doc/html/ _class_type_8h.js File Reference	2508
30.66.1 Variable Documentation	2508
30.66.1.1 _class_type_8h	2508
30.67doc/html/ _closest_point_8h.js File Reference	2508
30.67.1 Variable Documentation	2508
30.67.1.1 _closest_point_8h	2508
30.68doc/html/ _closest_point_utils_8h.js File Reference	2508
30.68.1 Variable Documentation	2509
30.68.1.1 _closest_point_utils_8h	2509
30.69doc/html/ _coincidence_8h.js File Reference	2509
30.69.1 Variable Documentation	2509
30.69.1.1 _coincidence_8h	2509
30.70doc/html/ _composite_model_8h.js File Reference	2510
30.70.1 Variable Documentation	2510
30.70.1.1 _composite_model_8h	2510
30.71doc/html/ _constraint_definitions_8h.js File Reference	2510
30.71.1 Variable Documentation	2510
30.71.1.1 _constraint_definitions_8h	2510
30.72doc/html/ _coons_patch_gen_8h.js File Reference	2511
30.72.1 Variable Documentation	2511
30.72.1.1 _coons_patch_gen_8h	2511
30.73doc/html/ _coons_patch_volume_gen_8h.js File Reference	2511
30.73.1 Variable Documentation	2511
30.73.1.1 _coons_patch_volume_gen_8h	2511
30.74doc/html/ _creators_offset_utils_8h.js File Reference	2512
30.74.1 Variable Documentation	2512

30.74.1.1 _creators_offset_utils_8h	2512
30.75doc/html/ _creators_utils_8h.js File Reference	2512
30.75.1 Variable Documentation	2512
30.75.1.1 _creators_utils_8h	2512
30.76doc/html/ _curvature_8h.js File Reference	2512
30.76.1 Variable Documentation	2513
30.76.1.1 _curvature_8h	2513
30.77doc/html/ _curvature_analysis_8h.js File Reference	2513
30.77.1 Variable Documentation	2513
30.77.1.1 _curvature_analysis_8h	2513
30.78doc/html/ _curvature_utils_8h.js File Reference	2513
30.78.1 Variable Documentation	2514
30.78.1.1 _curvature_utils_8h	2514
30.79doc/html/ _curve_creators_8h.js File Reference	2514
30.79.1 Variable Documentation	2514
30.79.1.1 _curve_creators_8h	2514
30.80doc/html/ _curve_interpolator_8h.js File Reference	2514
30.80.1 Variable Documentation	2515
30.80.1.1 _curve_interpolator_8h	2515
30.81doc/html/ _curve_loop_8h.js File Reference	2515
30.81.1 Variable Documentation	2515
30.81.1.1 _curve_loop_8h	2515
30.82doc/html/ _direction2_d_8h.js File Reference	2515
30.82.1 Variable Documentation	2515
30.82.1.1 _direction2_d_8h	2515
30.83doc/html/ _face_utilities_8h.js File Reference	2516
30.83.1 Variable Documentation	2516
30.83.1.1 _face_utilities_8h	2516
30.84doc/html/ _factory_8h.js File Reference	2516
30.84.1 Variable Documentation	2516

30.84.1.1 _factory_8h	2516
30.85doc/html/ _gap_removal_8h.js File Reference	2516
30.85.1 Variable Documentation	2517
30.85.1.1 _gap_removal_8h	2517
30.86doc/html/ _gap_removal_volume_8h.js File Reference	2517
30.86.1 Variable Documentation	2517
30.86.1.1 _gap_removal_volume_8h	2517
30.87doc/html/ _general_function_minimizer_8h.js File Reference	2517
30.87.1 Variable Documentation	2518
30.87.1.1 _general_function_minimizer_8h	2518
30.88doc/html/ _general_function_minimizer__implementation_8h.js File Reference	2518
30.88.1 Variable Documentation	2518
30.88.1.1 _general_function_minimizer__implementation_8h	2518
30.89doc/html/ _geom_object_8h.js File Reference	2518
30.89.1 Variable Documentation	2519
30.89.1.1 _geom_object_8h	2519
30.90doc/html/ _geometry_tools_8h.js File Reference	2519
30.90.1 Variable Documentation	2519
30.90.1.1 _geometry_tools_8h	2519
30.91doc/html/ _go_intersections_8h.js File Reference	2519
30.91.1 Variable Documentation	2519
30.91.1.1 _go_intersections_8h	2519
30.92doc/html/ _go_read_write_8cpp.js File Reference	2520
30.92.1 Variable Documentation	2520
30.92.1.1 _go_read_write_8cpp	2520
30.93doc/html/ _go_read_write_8h.js File Reference	2520
30.93.1 Variable Documentation	2520
30.93.1.1 _go_read_write_8h	2520
30.94doc/html/ _go_tools__version_8h.js File Reference	2521
30.94.1 Variable Documentation	2521

30.94.1.1 _go_tools__version_8h	2521
30.95doc/html/ _hahns_surface_gen_8h.js File Reference	2521
30.95.1 Variable Documentation	2521
30.95.1.1 _hahns_surface_gen_8h	2521
30.96doc/html/ _he_triang_8h.js File Reference	2521
30.96.1 Variable Documentation	2522
30.96.1.1 _he_triang_8h	2522
30.97doc/html/ _i_g_e_sconverter_8h.js File Reference	2522
30.97.1 Variable Documentation	2522
30.97.1.1 _i_g_e_sconverter_8h	2522
30.98doc/html/ _implicit_utils_8h.js File Reference	2523
30.98.1 Variable Documentation	2523
30.98.1.1 _implicit_utils_8h	2523
30.99doc/html/ _int_results_model_8h.js File Reference	2523
30.99.1 Variable Documentation	2523
30.99.1.1 _int_results_model_8h	2523
30.100doc/html/ _integrate_8h.js File Reference	2524
30.100.1 Variable Documentation	2524
30.100.1.1 _integrate_8h	2524
30.101doc/html/ _integration_8h.js File Reference	2524
30.101.1 Variable Documentation	2524
30.101.1.1 _integration_8h	2524
30.102doc/html/ _intersection_curve_8h.js File Reference	2524
30.102.1 Variable Documentation	2525
30.102.1.1 _intersection_curve_8h	2525
30.103doc/html/ _intersection_interface_8h.js File Reference	2525
30.103.1 Variable Documentation	2525
30.103.1.1 _intersection_interface_8h	2525
30.104doc/html/ _intersection_point_utils_8h.js File Reference	2526
30.104.1 Variable Documentation	2526

30.104.1.1_intersection_point_utils_8h	2526
30.104.doc/html/_intersection_pool_utils_8h.js File Reference	2526
30.105.Variable Documentation	2527
30.105.1.1_intersection_pool_utils_8h	2527
30.105.doc/html/_intersection_utils_8h.js File Reference	2527
30.106.Variable Documentation	2527
30.106.1.1_intersection_utils_8h	2527
30.106.doc/html/_l_r_approx_app_8h.js File Reference	2528
30.107.Variable Documentation	2528
30.107.1.1_l_r_approx_app_8h	2528
30.107.doc/html/_l_r_b_spline2_d_8h.js File Reference	2528
30.108.Variable Documentation	2528
30.108.1.1_l_r_b_spline2_d_8h	2528
30.108.doc/html/_l_r_b_spline2_d_utils_8h.js File Reference	2528
30.109.Variable Documentation	2529
30.109.1.1_l_r_b_spline2_d_utils_8h	2529
30.109.doc/html/_l_r_benchmark_utils_8h.js File Reference	2529
30.110.Variable Documentation	2529
30.110.1.1_l_r_benchmark_utils_8h	2529
30.110.doc/html/_l_r_spline_eval_grid_8h.js File Reference	2529
30.111.Variable Documentation	2529
30.111.1.1_l_r_spline_eval_grid_8h	2529
30.111.doc/html/_l_r_spline_m_b_a_8h.js File Reference	2530
30.112.Variable Documentation	2530
30.112.1.1_l_r_spline_m_b_a_8h	2530
30.112.doc/html/_l_r_spline_plot_utils_8h.js File Reference	2530
30.113.Variable Documentation	2530
30.113.1.1_l_r_spline_plot_utils_8h	2530
30.113.doc/html/_l_r_spline_utils_8h.js File Reference	2530
30.114.Variable Documentation	2531

30.114.1.1_l_r_spline_utils_8h	2531
30.114.doc/html/_l_r_surf_smooth_l_s_8h.js File Reference	2531
30.115.Variable Documentation	2531
30.115.1.1_l_r_surf_smooth_l_s_8h	2531
30.115.doc/html/_l_r_surf_stitch_8h.js File Reference	2532
30.116.Variable Documentation	2532
30.116.1.1_l_r_surf_stitch_8h	2532
30.116.doc/html/_l_u_decomp_8h.js File Reference	2532
30.117.Variable Documentation	2532
30.117.1.1_l_u_decomp_8h	2532
30.117.doc/html/_l_u_decomp__implementation_8h.js File Reference	2533
30.118.Variable Documentation	2533
30.118.1.1_l_u_decomp__implementation_8h	2533
30.118.doc/html/_lin_dep_utils_8h.js File Reference	2533
30.119.Variable Documentation	2533
30.119.1.1_lin_dep_utils_8h	2533
30.119.doc/html/_link_type_8h.js File Reference	2534
30.120.Variable Documentation	2534
30.120.1.1_link_type_8h	2534
30.120.doc/html/_loft_surface_creator_8h.js File Reference	2534
30.121.Variable Documentation	2534
30.121.1.1_loft_surface_creator_8h	2534
30.121.doc/html/_loft_volume_creator_8h.js File Reference	2534
30.122.Variable Documentation	2535
30.122.1.1_loft_volume_creator_8h	2535
30.122.doc/html/_loop_utils_8h.js File Reference	2535
30.123.Variable Documentation	2535
30.123.1.1_loop_utils_8h	2535
30.123.doc/html/_matrix_x_d_8h.js File Reference	2535
30.124.Variable Documentation	2536

30.124.1.1_matrix_x_d_8h	2536
30.124.doc/html/_mesh2_d_8h.js File Reference	2536
30.125.Variable Documentation	2536
30.125.1.1_mesh2_d_8h	2536
30.125.doc/html/_mesh2_d_utils_8h.js File Reference	2536
30.126.Variable Documentation	2537
30.126.1.1_mesh2_d_utils_8h	2537
30.126.doc/html/_modify_surf_8h.js File Reference	2537
30.127.Variable Documentation	2537
30.127.1.1_modify_surf_8h	2537
30.127.doc/html/_offset_surface_utils_8h.js File Reference	2537
30.128.Variable Documentation	2538
30.128.1.1_offset_surface_utils_8h	2538
30.128.doc/html/_param_surface_8h.js File Reference	2538
30.129.Variable Documentation	2538
30.129.1.1_param_surface_8h	2538
30.129.doc/html/_path_8h.js File Reference	2539
30.130.Variable Documentation	2539
30.130.1.1_path_8h	2539
30.130.doc/html/_point_8h.js File Reference	2539
30.131.Variable Documentation	2539
30.131.1.1_point_8h	2539
30.131.doc/html/_point_cloud_8h.js File Reference	2540
30.132.Variable Documentation	2540
30.132.1.1_point_cloud_8h	2540
30.132.doc/html/_point_sequence_8h.js File Reference	2540
30.133.Variable Documentation	2540
30.133.1.1_point_sequence_8h	2540
30.133.doc/html/_point_set_app_8h.js File Reference	2541
30.134.Variable Documentation	2541

30.134.1.1_point_set_app_8h	2541
30.134.doc/html/_pr_dijkstra_8h.js File Reference	2541
30.135.Variable Documentation	2541
30.135.1.1_pr_dijkstra_8h	2541
30.135.doc/html/_pr_geodesics_8h.js File Reference	2541
30.136.Variable Documentation	2542
30.136.1.1_pr_geodesics_8h	2542
30.136.doc/html/_pr_interface_8h.js File Reference	2542
30.137.Variable Documentation	2542
30.137.1.1_pr_interface_8h	2542
30.137.doc/html/_pr_multi_dijkstra_8h.js File Reference	2543
30.138.Variable Documentation	2543
30.138.1.1_pr_multi_dijkstra_8h	2543
30.138.doc/html/_pr_param_util_8h.js File Reference	2543
30.139.Variable Documentation	2543
30.139.1.1_pr_param_util_8h	2543
30.139.doc/html/_pr_parametrize_bdy_8h.js File Reference	2544
30.140.Variable Documentation	2544
30.140.1.1_pr_parametrize_bdy_8h	2544
30.140.doc/html/_pr_parametrize_int_8h.js File Reference	2544
30.141.Variable Documentation	2544
30.141.1.1_pr_parametrize_int_8h	2544
30.141.doc/html/_pr_path_triangle_seq_8h.js File Reference	2545
30.142.Variable Documentation	2545
30.142.1.1_pr_path_triangle_seq_8h	2545
30.142.doc/html/_pr_texture_8h.js File Reference	2545
30.143.Variable Documentation	2545
30.143.1.1_pr_texture_8h	2545
30.143.doc/html/_pr_wavelet_util_8h.js File Reference	2545
30.144.Variable Documentation	2545

30.144.1.1_pr_wavelet_util_8h	2545
30.144.doc/html/_quality_utils_8h.js File Reference	2546
30.145.Variable Documentation	2546
30.145.1.1_quality_utils_8h	2546
30.145.doc/html/_rational_8h.js File Reference	2546
30.146.Variable Documentation	2546
30.146.1.1_rational_8h	2546
30.146.doc/html/_registration_utils_8h.js File Reference	2547
30.147.Variable Documentation	2547
30.147.1.1_registration_utils_8h	2547
30.147.doc/html/_regularize_utils_8h.js File Reference	2547
30.148.Variable Documentation	2547
30.148.1.1_regularize_utils_8h	2547
30.148.doc/html/_s_i_s_lconversion_8h.js File Reference	2548
30.149.Variable Documentation	2548
30.149.1.1_s_i_s_lconversion_8h	2548
30.149.doc/html/_singular_8h.js File Reference	2548
30.150.Variable Documentation	2549
30.150.1.1_singular_8h	2549
30.150.doc/html/_singularity_classification_8h.js File Reference	2549
30.151.Variable Documentation	2549
30.151.1.1_singularity_classification_8h	2549
30.151.doc/html/_singularity_type_8h.js File Reference	2549
30.152.Variable Documentation	2550
30.152.1.1_singularity_type_8h	2550
30.152.doc/html/_smooth_volume_8h.js File Reference	2550
30.153.Variable Documentation	2550
30.153.1.1_smooth_volume_8h	2550
30.153.doc/html/_spline_debug_utils_8h.js File Reference	2550
30.154.Variable Documentation	2551

30.154.1.1_spline_debug_utils_8h	2551
30.154 doc/html/_spline_utils_8h.js File Reference	2551
30.155. Variable Documentation	2551
30.155.1.1_spline_utils_8h	2551
30.155 doc/html/_spline_volume_8h.js File Reference	2552
30.156. Variable Documentation	2552
30.156.1.1_spline_volume_8h	2552
30.156 doc/html/_split_model_utils_8h.js File Reference	2552
30.157. Variable Documentation	2552
30.157.1.1_split_model_utils_8h	2552
30.157 doc/html/_stream_utils_8h.js File Reference	2552
30.158. Variable Documentation	2553
30.158.1.1_stream_utils_8h	2553
30.158 doc/html/_streamable_8h.js File Reference	2553
30.159. Variable Documentation	2553
30.159.1.1_streamable_8h	2553
30.159 doc/html/_subdivision_classification_8h.js File Reference	2553
30.160. Variable Documentation	2554
30.160.1.1_subdivision_classification_8h	2554
30.160 doc/html/_surface_creators_8h.js File Reference	2554
30.161. Variable Documentation	2554
30.161.1.1_surface_creators_8h	2554
30.161 doc/html/_surface_interpolator_8h.js File Reference	2555
30.162. Variable Documentation	2555
30.162.1.1_surface_interpolator_8h	2555
30.162 doc/html/_surface_model_utils_8h.js File Reference	2555
30.163. Variable Documentation	2555
30.163.1.1_surface_model_utils_8h	2555
30.163 doc/html/_surface_on_volume_tools_8h.js File Reference	2555
30.164. Variable Documentation	2556

30.164.1.1_surface_on_volume_tools_8h	2556
30.164.doc/html/_surface_tools_8h.js File Reference	2556
30.165.Variable Documentation	2556
30.165.1.1_surface_tools_8h	2556
30.165.doc/html/_tesselator_utils_8h.js File Reference	2556
30.166.Variable Documentation	2557
30.166.1.1_tesselator_utils_8h	2557
30.166.doc/html/_utils_8h.js File Reference	2557
30.167.Variable Documentation	2557
30.167.1.1_utils_8h	2557
30.167.doc/html/_values_8h.js File Reference	2557
30.168.Variable Documentation	2558
30.168.1.1_values_8h	2558
30.168.doc/html/_volume_interpolator_8h.js File Reference	2558
30.169.Variable Documentation	2558
30.169.1.1_volume_interpolator_8h	2558
30.169.doc/html/_volume_model_creator_8h.js File Reference	2558
30.170.Variable Documentation	2558
30.170.1.1_volume_model_creator_8h	2558
30.170.doc/html/_volume_model_file_handler_8h.js File Reference	2559
30.171.Variable Documentation	2559
30.171.1.1_volume_model_file_handler_8h	2559
30.171.doc/html/_volume_tools_8h.js File Reference	2559
30.172.Variable Documentation	2559
30.172.1.1_volume_tools_8h	2559
30.172.doc/html/_volumes_8h.js File Reference	2560
30.173.Variable Documentation	2560
30.173.1.1_volumes_8h	2560
30.173.doc/html/annotated_dup.js File Reference	2560
30.174.Variable Documentation	2560

30.174.1.1annotated_dup	2560
30.174.doc/html/aux2_8cpp.js File Reference	2560
30.175.1.Variable Documentation	2561
30.175.1.1aux2_8cpp	2561
30.175.doc/html/aux2_8h.js File Reference	2561
30.176.1.Variable Documentation	2561
30.176.1.1aux2_8h	2561
30.176.doc/html/bandmat_8cpp.js File Reference	2561
30.177.1.Variable Documentation	2561
30.177.1.1bandmat_8cpp	2561
30.177.doc/html/binom_8h.js File Reference	2562
30.178.1.Variable Documentation	2562
30.178.1.1binom_8h	2562
30.178.doc/html/boolean_8h.js File Reference	2562
30.179.1.Variable Documentation	2562
30.179.1.1boolean_8h	2562
30.179.doc/html/brent__minimize_8h.js File Reference	2562
30.180.1.Variable Documentation	2563
30.180.1.1brent__minimize_8h	2563
30.180.doc/html/checks_8h.js File Reference	2563
30.181.1.Variable Documentation	2563
30.181.1.1checks_8h	2563
30.181.doc/html/cholesky_8cpp.js File Reference	2563
30.182.1.Variable Documentation	2564
30.182.1.1cholesky_8cpp	2564
30.182.doc/html/class_control_word.js File Reference	2564
30.183.1.Variable Documentation	2564
30.183.1.1class_control_word	2564
30.183.doc/html/class_curve_resolution_sheet.js File Reference	2564
30.184.1.Variable Documentation	2565

30.184.1.1class_curve_resolution_sheet	2565
30.185doc/html/class_data_handler.js File Reference	2565
30.185.1.Variable Documentation	2565
30.185.1.1class_data_handler	2565
30.186doc/html/class_data_handler_vol_and_l_r.js File Reference	2565
30.186.1.Variable Documentation	2566
30.186.1.1class_data_handler_vol_and_l_r	2566
30.187doc/html/class_default_data_handler.js File Reference	2566
30.187.1.Variable Documentation	2566
30.187.1.1class_default_data_handler	2566
30.188doc/html/class_dijkstra.js File Reference	2566
30.188.1.Variable Documentation	2567
30.188.1.1class_dijkstra	2567
30.189doc/html/class_edge_type.js File Reference	2567
30.189.1.Variable Documentation	2567
30.189.1.1class_edge_type	2567
30.190doc/html/class_g_a_r_c_h11___l_l.js File Reference	2567
30.190.1.Variable Documentation	2568
30.190.1.1class_g_a_r_c_h11___l_l	2568
30.191doc/html/class_go_1_1_adapt_curve.js File Reference	2568
30.191.1.Variable Documentation	2568
30.191.1.1class_go_1_1_adapt_curve	2568
30.192doc/html/class_go_1_1_alg_obj2_d_int.js File Reference	2568
30.192.1.Variable Documentation	2569
30.192.1.1class_go_1_1_alg_obj2_d_int	2569
30.193doc/html/class_go_1_1_alg_obj3_d_int.js File Reference	2569
30.193.1.Variable Documentation	2569
30.193.1.1class_go_1_1_alg_obj3_d_int	2569
30.194doc/html/class_go_1_1_alg_object_int.js File Reference	2569
30.194.1.Variable Documentation	2570

30.194.1.1class_go_1_1_alg_object_int	2570
30.194.doc/html/class_go_1_1_approx_crv_to_seqs.js File Reference	2570
30.195.1Variable Documentation	2570
30.195.1.1class_go_1_1_approx_crv_to_seqs	2570
30.195.doc/html/class_go_1_1_approx_curve.js File Reference	2570
30.196.1Variable Documentation	2571
30.196.1.1class_go_1_1_approx_curve	2571
30.196.doc/html/class_go_1_1_approx_surf.js File Reference	2571
30.197.1Variable Documentation	2571
30.197.1.1class_go_1_1_approx_surf	2571
30.197.doc/html/class_go_1_1_array.js File Reference	2571
30.198.1Variable Documentation	2572
30.198.1.1class_go_1_1_array	2572
30.198.doc/html/class_go_1_1_bary_coord_system.js File Reference	2572
30.199.1Variable Documentation	2572
30.199.1.1class_go_1_1_bary_coord_system	2572
30.199.doc/html/class_go_1_1_bary_coord_system_triangle3_d.js File Reference	2572
30.200.1Variable Documentation	2572
30.200.1.1class_go_1_1_bary_coord_system_triangle3_d	2572
30.200.doc/html/class_go_1_1_bd_cond_funcnor.js File Reference	2573
30.201.1Variable Documentation	2573
30.201.1.1class_go_1_1_bd_cond_funcnor	2573
30.201.doc/html/class_go_1_1_bernstein_multi.js File Reference	2573
30.202.1Variable Documentation	2573
30.202.1.1class_go_1_1_bernstein_multi	2573
30.202.doc/html/class_go_1_1_bernstein_poly.js File Reference	2573
30.203.1Variable Documentation	2573
30.203.1.1class_go_1_1_bernstein_poly	2573
30.203.doc/html/class_go_1_1_bernstein_tetrahedral_poly.js File Reference	2574
30.204.1Variable Documentation	2574

30.204.1.1class_go_1_1_bernstein_tetrahedral_poly	2574
30.204.doc/html/class_go_1_1_bernstein_triangular_poly.js File Reference	2574
30.205.1.Variable Documentation	2575
30.205.1.1class_go_1_1_bernstein_triangular_poly	2575
30.205.doc/html/class_go_1_1_bezier_triangle.js File Reference	2575
30.206.1.Variable Documentation	2575
30.206.1.1class_go_1_1_bezier_triangle	2575
30.206.doc/html/class_go_1_1_binomial.js File Reference	2576
30.207.1.Variable Documentation	2576
30.207.1.1class_go_1_1_binomial	2576
30.207.doc/html/class_go_1_1_block_boundary_condition.js File Reference	2576
30.208.1.Variable Documentation	2576
30.208.1.1class_go_1_1_block_boundary_condition	2576
30.208.doc/html/class_go_1_1_block_point_bd_cond.js File Reference	2577
30.209.1.Variable Documentation	2577
30.209.1.1class_go_1_1_block_point_bd_cond	2577
30.209.doc/html/class_go_1_1_block_solution.js File Reference	2577
30.210.1.Variable Documentation	2577
30.210.1.1class_go_1_1_block_solution	2577
30.210.doc/html/class_go_1_1_body.js File Reference	2578
30.211.1.Variable Documentation	2578
30.211.1.1class_go_1_1_body	2578
30.211.doc/html/class_go_1_1_bounded_curve.js File Reference	2579
30.212.1.Variable Documentation	2579
30.212.1.1class_go_1_1_bounded_curve	2579
30.212.doc/html/class_go_1_1_bounded_surface.js File Reference	2579
30.213.1.Variable Documentation	2579
30.213.1.1class_go_1_1_bounded_surface	2579
30.213.doc/html/class_go_1_1_bounding_box.js File Reference	2579
30.214.1.Variable Documentation	2579

30.214.1.1class_go_1_1_bounding_box	2579
30.215doc/html/class_go_1_1_bspline_basis.js File Reference	2580
30.215.1Variable Documentation	2580
30.215.1.1class_go_1_1_bspline_basis	2580
30.216doc/html/class_go_1_1_c_p_unlock.js File Reference	2580
30.216.1Variable Documentation	2580
30.216.1.1class_go_1_1_c_p_unlock	2580
30.217doc/html/class_go_1_1_cell_division.js File Reference	2580
30.217.1Variable Documentation	2581
30.217.1.1class_go_1_1_cell_division	2581
30.218doc/html/class_go_1_1_circle.js File Reference	2581
30.218.1Variable Documentation	2581
30.218.1.1class_go_1_1_circle	2581
30.219doc/html/class_go_1_1_closest_point_calculator.js File Reference	2581
30.219.1Variable Documentation	2581
30.219.1.1class_go_1_1_closest_point_calculator	2581
30.220doc/html/class_go_1_1_complete_edge_net.js File Reference	2582
30.220.1Variable Documentation	2582
30.220.1.1class_go_1_1_complete_edge_net	2582
30.221doc/html/class_go_1_1_complexity_info.js File Reference	2582
30.221.1Variable Documentation	2582
30.221.1.1class_go_1_1_complexity_info	2582
30.222doc/html/class_go_1_1_composite_box.js File Reference	2583
30.222.1Variable Documentation	2583
30.222.1.1class_go_1_1_composite_box	2583
30.223doc/html/class_go_1_1_composite_curve.js File Reference	2583
30.223.1Variable Documentation	2583
30.223.1.1class_go_1_1_composite_curve	2583
30.224doc/html/class_go_1_1_composite_model.js File Reference	2583
30.224.1Variable Documentation	2584

30.224.1.1class_go_1_1_composite_model	2584
30.224.doc/html/class_go_1_1_composite_model_factory.js File Reference	2584
30.225.1Variable Documentation	2584
30.225.1.1class_go_1_1_composite_model_factory	2584
30.225.doc/html/class_go_1_1_composite_model_file_handler.js File Reference	2585
30.226.1Variable Documentation	2585
30.226.1.1class_go_1_1_composite_model_file_handler	2585
30.226.doc/html/class_go_1_1_composite_surface.js File Reference	2585
30.227.1Variable Documentation	2585
30.227.1.1class_go_1_1_composite_surface	2585
30.227.doc/html/class_go_1_1_concrete_creator.js File Reference	2586
30.228.1Variable Documentation	2586
30.228.1.1class_go_1_1_concrete_creator	2586
30.228.doc/html/class_go_1_1_cone.js File Reference	2586
30.229.1Variable Documentation	2586
30.229.1.1class_go_1_1_cone	2586
30.229.doc/html/class_go_1_1_cone_volume.js File Reference	2586
30.230.1Variable Documentation	2587
30.230.1.1class_go_1_1_cone_volume	2587
30.230.doc/html/class_go_1_1_connection_functor.js File Reference	2587
30.231.1Variable Documentation	2587
30.231.1.1class_go_1_1_connection_functor	2587
30.231.doc/html/class_go_1_1_coordinate_system.js File Reference	2588
30.232.1Variable Documentation	2588
30.232.1.1class_go_1_1_coordinate_system	2588
30.232.doc/html/class_go_1_1_creator.js File Reference	2588
30.233.1Variable Documentation	2588
30.233.1.1class_go_1_1_creator	2588
30.233.doc/html/class_go_1_1_cross_tan_off_dist.js File Reference	2589
30.234.1Variable Documentation	2589

30.234.1.1class_go_1_1_cross_tan_off_dist	2589
30.235doc/html/class_go_1_1_cross_tangent_offset.js File Reference	2589
30.235.1.Variable Documentation	2589
30.235.1.1class_go_1_1_cross_tangent_offset	2589
30.236doc/html/class_go_1_1_crosses_value.js File Reference	2590
30.236.1.Variable Documentation	2590
30.236.1.1class_go_1_1_crosses_value	2590
30.237doc/html/class_go_1_1_curve_bounded_domain.js File Reference	2590
30.237.1.Variable Documentation	2590
30.237.1.1class_go_1_1_curve_bounded_domain	2590
30.238doc/html/class_go_1_1_curve_loop.js File Reference	2591
30.238.1.Variable Documentation	2591
30.238.1.1class_go_1_1_curve_loop	2591
30.239doc/html/class_go_1_1_curve_model.js File Reference	2591
30.239.1.Variable Documentation	2591
30.239.1.1class_go_1_1_curve_model	2591
30.240doc/html/class_go_1_1_curve_on_surface.js File Reference	2592
30.240.1.Variable Documentation	2592
30.240.1.1class_go_1_1_curve_on_surface	2592
30.241doc/html/class_go_1_1_curve_on_volume.js File Reference	2592
30.241.1.Variable Documentation	2592
30.241.1.1class_go_1_1_curve_on_volume	2592
30.242doc/html/class_go_1_1_curve_tesselator.js File Reference	2593
30.242.1.Variable Documentation	2593
30.242.1.1class_go_1_1_curve_tesselator	2593
30.243doc/html/class_go_1_1_cv_cv_intersector.js File Reference	2593
30.243.1.Variable Documentation	2593
30.243.1.1class_go_1_1_cv_cv_intersector	2593
30.244doc/html/class_go_1_1_cv_pt_intersector.js File Reference	2594
30.244.1.Variable Documentation	2594

30.244.1.1class_go_1_1_cv_pt_intersector	2594
30.244doc/html/class_go_1_1_cylinder.js File Reference	2594
30.245.1Variable Documentation	2594
30.245.1.1class_go_1_1_cylinder	2594
30.245doc/html/class_go_1_1_cylinder_int.js File Reference	2594
30.246.1Variable Documentation	2595
30.246.1.1class_go_1_1_cylinder_int	2595
30.246doc/html/class_go_1_1_cylinder_volume.js File Reference	2595
30.247.1Variable Documentation	2595
30.247.1.1class_go_1_1_cylinder_volume	2595
30.247doc/html/class_go_1_1_degenerated_intersection_curve.js File Reference	2596
30.248.1Variable Documentation	2596
30.248.1.1class_go_1_1_degenerated_intersection_curve	2596
30.248doc/html/class_go_1_1_direction_cone.js File Reference	2596
30.249.1Variable Documentation	2596
30.249.1.1class_go_1_1_direction_cone	2596
30.249doc/html/class_go_1_1_disc.js File Reference	2597
30.250.1Variable Documentation	2597
30.250.1.1class_go_1_1_disc	2597
30.250doc/html/class_go_1_1_domain.js File Reference	2597
30.251.1Variable Documentation	2597
30.251.1.1class_go_1_1_domain	2597
30.251doc/html/class_go_1_1_edge_vertex.js File Reference	2597
30.252.1Variable Documentation	2598
30.252.1.1class_go_1_1_edge_vertex	2598
30.252doc/html/class_go_1_1_element2_d.js File Reference	2598
30.253.1Variable Documentation	2598
30.253.1.1class_go_1_1_element2_d	2598
30.253doc/html/class_go_1_1_elementary_curve.js File Reference	2598
30.254.1Variable Documentation	2599

30.254.1.1class_go_1_1_elementary_curve	2599
30.254.doc/html/class_go_1_1_elementary_surface.js File Reference	2599
30.255.1Variable Documentation	2599
30.255.1.1class_go_1_1_elementary_surface	2599
30.255.doc/html/class_go_1_1_elementary_volume.js File Reference	2600
30.256.1Variable Documentation	2600
30.256.1.1class_go_1_1_elementary_volume	2600
30.256.doc/html/class_go_1_1_ellipse.js File Reference	2600
30.257.1Variable Documentation	2600
30.257.1.1class_go_1_1_ellipse	2600
30.257.doc/html/class_go_1_1_eval_curve.js File Reference	2601
30.258.1Variable Documentation	2601
30.258.1.1class_go_1_1_eval_curve	2601
30.258.doc/html/class_go_1_1_eval_curve_set.js File Reference	2601
30.259.1Variable Documentation	2601
30.259.1.1class_go_1_1_eval_curve_set	2601
30.259.doc/html/class_go_1_1_eval_functor_curve.js File Reference	2602
30.260.1Variable Documentation	2602
30.260.1.1class_go_1_1_eval_functor_curve	2602
30.260.doc/html/class_go_1_1_eval_functor_surface.js File Reference	2602
30.261.1Variable Documentation	2602
30.261.1.1class_go_1_1_eval_functor_surface	2602
30.261.doc/html/class_go_1_1_eval_offset_surface.js File Reference	2603
30.262.1Variable Documentation	2603
30.262.1.1class_go_1_1_eval_offset_surface	2603
30.262.doc/html/class_go_1_1_eval_param_curve.js File Reference	2603
30.263.1Variable Documentation	2603
30.263.1.1class_go_1_1_eval_param_curve	2603
30.263.doc/html/class_go_1_1_eval_surface.js File Reference	2604
30.264.1Variable Documentation	2604

30.264.1.1class_go_1_1_eval_surface	2604
30.264doc/html/class_go_1_1_face_adjacency.js File Reference	2604
30.265.1Variable Documentation	2604
30.265.1.1class_go_1_1_face_adjacency	2604
30.265doc/html/class_go_1_1_face_connectivity_utils.js File Reference	2605
30.266.1Variable Documentation	2605
30.266.1.1class_go_1_1_face_connectivity_utils	2605
30.266doc/html/class_go_1_1_face_set_quality.js File Reference	2605
30.267.1Variable Documentation	2605
30.267.1.1class_go_1_1_face_set_quality	2605
30.267doc/html/class_go_1_1_face_set_repair.js File Reference	2605
30.268.1Variable Documentation	2606
30.268.1.1class_go_1_1_face_set_repair	2606
30.268doc/html/class_go_1_1_factory.js File Reference	2606
30.269.1Variable Documentation	2606
30.269.1.1class_go_1_1_factory	2606
30.269doc/html/class_go_1_1_fun2_fun.js File Reference	2606
30.270.1Variable Documentation	2607
30.270.1.1class_go_1_1_fun2_fun	2607
30.270doc/html/class_go_1_1_function_minimizer.js File Reference	2607
30.271.1Variable Documentation	2607
30.271.1.1class_go_1_1_function_minimizer	2607
30.271doc/html/class_go_1_1_general_mesh.js File Reference	2607
30.272.1Variable Documentation	2608
30.272.1.1class_go_1_1_general_mesh	2608
30.272doc/html/class_go_1_1_generic_tri_mesh.js File Reference	2608
30.273.1Variable Documentation	2608
30.273.1.1class_go_1_1_generic_tri_mesh	2608
30.273doc/html/class_go_1_1_geo_tol.js File Reference	2609
30.274.1Variable Documentation	2609

30.274.1.1class_go_1_1_geo_tol	2609
30.274doc/html/class_go_1_1_geom_object.js File Reference	2609
30.275.1Variable Documentation	2609
30.275.1.1class_go_1_1_geom_object	2609
30.275doc/html/class_go_1_1_geom_object_int.js File Reference	2610
30.276.1Variable Documentation	2610
30.276.1.1class_go_1_1_geom_object_int	2610
30.276doc/html/class_go_1_1_go_tools.js File Reference	2610
30.277.1Variable Documentation	2610
30.277.1.1class_go_1_1_go_tools	2610
30.277doc/html/class_go_1_1_gpu_matrix.js File Reference	2610
30.278.1Variable Documentation	2611
30.278.1.1class_go_1_1_gpu_matrix	2611
30.278doc/html/class_go_1_1_hermite_app_c.js File Reference	2611
30.279.1Variable Documentation	2611
30.279.1.1class_go_1_1_hermite_app_c	2611
30.279doc/html/class_go_1_1_hermite_app_s.js File Reference	2611
30.280.1Variable Documentation	2612
30.280.1.1class_go_1_1_hermite_app_s	2612
30.280doc/html/class_go_1_1_hermite_appr_eval_surf.js File Reference	2612
30.281.1Variable Documentation	2612
30.281.1.1class_go_1_1_hermite_appr_eval_surf	2612
30.281doc/html/class_go_1_1_hermite_grid1_d.js File Reference	2612
30.282.1Variable Documentation	2613
30.282.1.1class_go_1_1_hermite_grid1_d	2613
30.282doc/html/class_go_1_1_hermite_grid1_d_multi.js File Reference	2613
30.283.1Variable Documentation	2613
30.283.1.1class_go_1_1_hermite_grid1_d_multi	2613
30.283doc/html/class_go_1_1_hermite_grid2_d.js File Reference	2613
30.284.1Variable Documentation	2614

30.284.1.1class_go_1_1_hermite_grid2_d	2614
30.284.doc/html/class_go_1_1_hermite_interpolator.js File Reference	2614
30.285.1.Variable Documentation	2614
30.285.1.1class_go_1_1_hermite_interpolator	2614
30.285.doc/html/class_go_1_1_hyperbola.js File Reference	2614
30.286.1.Variable Documentation	2615
30.286.1.1class_go_1_1_hyperbola	2615
30.286.doc/html/class_go_1_1_i_g_e_sconverter.js File Reference	2615
30.287.1.Variable Documentation	2615
30.287.1.1class_go_1_1_i_g_e_sconverter	2615
30.287.doc/html/class_go_1_1_identity.js File Reference	2615
30.288.1.Variable Documentation	2616
30.288.1.1class_go_1_1_identity	2616
30.288.doc/html/class_go_1_1_implicitize_curve_algo.js File Reference	2616
30.289.1.Variable Documentation	2616
30.289.1.1class_go_1_1_implicitize_curve_algo	2616
30.289.doc/html/class_go_1_1_implicitize_curve_and_vector_algo.js File Reference	2616
30.290.1.Variable Documentation	2617
30.290.1.1class_go_1_1_implicitize_curve_and_vector_algo	2617
30.290.doc/html/class_go_1_1_implicitize_point_cloud_algo.js File Reference	2617
30.291.1.Variable Documentation	2617
30.291.1.1class_go_1_1_implicitize_point_cloud_algo	2617
30.291.doc/html/class_go_1_1_implicitize_surface_algo.js File Reference	2618
30.292.1.Variable Documentation	2618
30.292.1.1class_go_1_1_implicitize_surface_algo	2618
30.292.doc/html/class_go_1_1_index_mesh2_d_iterator.js File Reference	2618
30.293.1.Variable Documentation	2618
30.293.1.1class_go_1_1_index_mesh2_d_iterator	2618
30.293.doc/html/class_go_1_1_int_crv_evaluator.js File Reference	2619
30.294.1.Variable Documentation	2619

30.294.1.1class_go_1_1_int_crv_evaluator	2619
30.294.doc/html/class_go_1_1_int_results_comp_cv.js File Reference	2619
30.295.1.Variable Documentation	2619
30.295.1.1class_go_1_1_int_results_comp_cv	2619
30.295.doc/html/class_go_1_1_int_results_model.js File Reference	2620
30.296.1.Variable Documentation	2620
30.296.1.1class_go_1_1_int_results_model	2620
30.296.doc/html/class_go_1_1_int_results_sf_model.js File Reference	2620
30.297.1.Variable Documentation	2620
30.297.1.1class_go_1_1_int_results_sf_model	2620
30.297.doc/html/class_go_1_1_interpolated_intersection_curve.js File Reference	2621
30.298.1.Variable Documentation	2621
30.298.1.1class_go_1_1_interpolated_intersection_curve	2621
30.298.doc/html/class_go_1_1_interpolator.js File Reference	2622
30.299.1.Variable Documentation	2622
30.299.1.1class_go_1_1_interpolator	2622
30.299.doc/html/class_go_1_1_intersection_curve.js File Reference	2622
30.300.1.Variable Documentation	2622
30.300.1.1class_go_1_1_intersection_curve	2622
30.300.doc/html/class_go_1_1_intersection_link.js File Reference	2623
30.301.1.Variable Documentation	2623
30.301.1.1class_go_1_1_intersection_link	2623
30.301.doc/html/class_go_1_1_intersection_point.js File Reference	2623
30.302.1.Variable Documentation	2623
30.302.1.1class_go_1_1_intersection_point	2623
30.302.doc/html/class_go_1_1_intersection_pool.js File Reference	2623
30.303.1.Variable Documentation	2624
30.303.1.1class_go_1_1_intersection_pool	2624
30.303.doc/html/class_go_1_1_intersector.js File Reference	2624
30.304.1.Variable Documentation	2624

30.304.1.1class_go_1_1_intersector	2624
30.305doc/html/class_go_1_1_intersector2_obj.js File Reference	2624
30.305.1Variable Documentation	2624
30.305.1.1class_go_1_1_intersector2_obj	2624
30.306doc/html/class_go_1_1_intersector_alg_par.js File Reference	2624
30.306.1Variable Documentation	2625
30.306.1.1class_go_1_1_intersector_alg_par	2625
30.307doc/html/class_go_1_1_intersector_func_const.js File Reference	2625
30.307.1Variable Documentation	2625
30.307.1.1class_go_1_1_intersector_func_const	2625
30.308doc/html/class_go_1_1_isogeometric_block.js File Reference	2626
30.308.1Variable Documentation	2626
30.308.1.1class_go_1_1_isogeometric_block	2626
30.309doc/html/class_go_1_1_isogeometric_model.js File Reference	2627
30.309.1Variable Documentation	2627
30.309.1.1class_go_1_1_isogeometric_model	2627
30.310doc/html/class_go_1_1_isogeometric_sf_block.js File Reference	2627
30.310.1Variable Documentation	2627
30.310.1.1class_go_1_1_isogeometric_sf_block	2627
30.311doc/html/class_go_1_1_isogeometric_sf_model.js File Reference	2628
30.311.1Variable Documentation	2628
30.311.1.1class_go_1_1_isogeometric_sf_model	2628
30.312doc/html/class_go_1_1_isogeometric_vol_block.js File Reference	2628
30.312.1Variable Documentation	2628
30.312.1.1class_go_1_1_isogeometric_vol_block	2628
30.313doc/html/class_go_1_1_isogeometric_vol_model.js File Reference	2628
30.313.1Variable Documentation	2629
30.313.1.1class_go_1_1_isogeometric_vol_model	2629
30.314doc/html/class_go_1_1_isoparametric_intersection_curve.js File Reference	2629
30.314.1Variable Documentation	2629

30.314.1.1class_go_1_1_isoparametric_intersection_curve	2629
30.315doc/html/class_go_1_1_l_r_b_spline2_d.js File Reference	2630
30.315.1.Variable Documentation	2630
30.315.1.1class_go_1_1_l_r_b_spline2_d	2630
30.316doc/html/class_go_1_1_l_r_spline_eval_grid.js File Reference	2630
30.316.1.Variable Documentation	2630
30.316.1.1class_go_1_1_l_r_spline_eval_grid	2630
30.317doc/html/class_go_1_1_l_r_spline_surface.js File Reference	2631
30.317.1.Variable Documentation	2631
30.317.1.1class_go_1_1_l_r_spline_surface	2631
30.318doc/html/class_go_1_1_l_r_surf_approx.js File Reference	2631
30.318.1.Variable Documentation	2631
30.318.1.1class_go_1_1_l_r_surf_approx	2631
30.319doc/html/class_go_1_1_l_r_surf_smooth_l_s.js File Reference	2632
30.319.1.Variable Documentation	2632
30.319.1.1class_go_1_1_l_r_surf_smooth_l_s	2632
30.320doc/html/class_go_1_1_lift_curve.js File Reference	2632
30.320.1.Variable Documentation	2632
30.320.1.1class_go_1_1_lift_curve	2632
30.321doc/html/class_go_1_1_line.js File Reference	2633
30.321.1.Variable Documentation	2633
30.321.1.1class_go_1_1_line	2633
30.322doc/html/class_go_1_1_line2_d_int.js File Reference	2633
30.322.1.Variable Documentation	2633
30.322.1.1class_go_1_1_line2_d_int	2633
30.323doc/html/class_go_1_1_line_cloud.js File Reference	2633
30.323.1.Variable Documentation	2634
30.323.1.1class_go_1_1_line_cloud	2634
30.324doc/html/class_go_1_1_line_cloud_tesselator.js File Reference	2634
30.324.1.Variable Documentation	2634

30.324.1.1class_go_1_1_line_cloud_tesselator	2634
30.325doc/html/class_go_1_1_line_strip.js File Reference	2634
30.325.1.Variable Documentation	2635
30.325.1.1class_go_1_1_line_strip	2635
30.326doc/html/class_go_1_1_locked_dir_dist_func.js File Reference	2635
30.326.1.Variable Documentation	2635
30.326.1.1class_go_1_1_locked_dir_dist_func	2635
30.327doc/html/class_go_1_1_loop.js File Reference	2635
30.327.1.Variable Documentation	2636
30.327.1.1class_go_1_1_loop	2636
30.328doc/html/class_go_1_1_march_point.js File Reference	2636
30.328.1.Variable Documentation	2636
30.328.1.1class_go_1_1_march_point	2636
30.329doc/html/class_go_1_1_matrix_x_d.js File Reference	2636
30.329.1.Variable Documentation	2637
30.329.1.1class_go_1_1_matrix_x_d	2637
30.330doc/html/class_go_1_1_mesh2_d.js File Reference	2637
30.330.1.Variable Documentation	2637
30.330.1.1class_go_1_1_mesh2_d	2637
30.331doc/html/class_go_1_1_mesh2_d_iterator.js File Reference	2637
30.331.1.Variable Documentation	2638
30.331.1.1class_go_1_1_mesh2_d_iterator	2638
30.332doc/html/class_go_1_1_model_quality.js File Reference	2638
30.332.1.Variable Documentation	2638
30.332.1.1class_go_1_1_model_quality	2638
30.333doc/html/class_go_1_1_model_repair.js File Reference	2638
30.333.1.Variable Documentation	2638
30.333.1.1class_go_1_1_model_repair	2638
30.334doc/html/class_go_1_1_non_evaluable_intersection_curve.js File Reference	2639
30.334.1.Variable Documentation	2639

30.334.1.1class_go_1_1_non_evaluable_intersection_curve	2639
30.335doc/html/class_go_1_1_noop_tesselator.js File Reference	2639
30.335.1Variable Documentation	2639
30.335.1.1class_go_1_1_noop_tesselator	2639
30.336doc/html/class_go_1_1_object_header.js File Reference	2640
30.336.1Variable Documentation	2640
30.336.1.1class_go_1_1_object_header	2640
30.337doc/html/class_go_1_1_par0_func_intersector.js File Reference	2640
30.337.1Variable Documentation	2640
30.337.1.1class_go_1_1_par0_func_intersector	2640
30.338doc/html/class_go_1_1_par1_func_intersector.js File Reference	2641
30.338.1Variable Documentation	2641
30.338.1.1class_go_1_1_par1_func_intersector	2641
30.339doc/html/class_go_1_1_par2_func_intersector.js File Reference	2641
30.339.1Variable Documentation	2641
30.339.1.1class_go_1_1_par2_func_intersector	2641
30.340doc/html/class_go_1_1_parabola.js File Reference	2642
30.340.1Variable Documentation	2642
30.340.1.1class_go_1_1_parabola	2642
30.341doc/html/class_go_1_1_parallelepiped.js File Reference	2642
30.341.1Variable Documentation	2642
30.341.1.1class_go_1_1_parallelepiped	2642
30.342doc/html/class_go_1_1_param0_function_int.js File Reference	2643
30.342.1Variable Documentation	2643
30.342.1.1class_go_1_1_param0_function_int	2643
30.343doc/html/class_go_1_1_param1_function_int.js File Reference	2643
30.343.1Variable Documentation	2643
30.343.1.1class_go_1_1_param1_function_int	2643
30.344doc/html/class_go_1_1_param2_function_int.js File Reference	2643
30.344.1Variable Documentation	2643

30.344.1.1class_go_1_1_param2_function_int	2643
30.345doc/html/class_go_1_1_param_curve.js File Reference	2643
30.345.1.Variable Documentation	2644
30.345.1.1class_go_1_1_param_curve	2644
30.346doc/html/class_go_1_1_param_curve_int.js File Reference	2644
30.346.1.Variable Documentation	2644
30.346.1.1class_go_1_1_param_curve_int	2644
30.347doc/html/class_go_1_1_param_function_int.js File Reference	2644
30.347.1.Variable Documentation	2644
30.347.1.1class_go_1_1_param_function_int	2644
30.348doc/html/class_go_1_1_param_geom_int.js File Reference	2645
30.348.1.Variable Documentation	2645
30.348.1.1class_go_1_1_param_geom_int	2645
30.349doc/html/class_go_1_1_param_object_int.js File Reference	2645
30.349.1.Variable Documentation	2645
30.349.1.1class_go_1_1_param_object_int	2645
30.350doc/html/class_go_1_1_param_point_int.js File Reference	2646
30.350.1.Variable Documentation	2646
30.350.1.1class_go_1_1_param_point_int	2646
30.351doc/html/class_go_1_1_param_surface.js File Reference	2646
30.351.1.Variable Documentation	2646
30.351.1.1class_go_1_1_param_surface	2646
30.352doc/html/class_go_1_1_param_surface_int.js File Reference	2646
30.352.1.Variable Documentation	2646
30.352.1.1class_go_1_1_param_surface_int	2646
30.353doc/html/class_go_1_1_param_volume.js File Reference	2646
30.353.1.Variable Documentation	2647
30.353.1.1class_go_1_1_param_volume	2647
30.354doc/html/class_go_1_1_parametric_surface_tesselator.js File Reference	2647
30.354.1.Variable Documentation	2647

30.354.1.1class_go_1_1_parametric_surface_tesselator	2647
30.355doc/html/class_go_1_1_plane.js File Reference	2648
30.355.1.Variable Documentation	2648
30.355.1.1class_go_1_1_plane	2648
30.356doc/html/class_go_1_1_plane_int.js File Reference	2648
30.356.1.Variable Documentation	2648
30.356.1.1class_go_1_1_plane_int	2648
30.357doc/html/class_go_1_1_point.js File Reference	2648
30.357.1.Variable Documentation	2649
30.357.1.1class_go_1_1_point	2649
30.358doc/html/class_go_1_1_point_cloud.js File Reference	2649
30.358.1.Variable Documentation	2649
30.358.1.1class_go_1_1_point_cloud	2649
30.359doc/html/class_go_1_1_point_on_curve.js File Reference	2649
30.359.1.Variable Documentation	2650
30.359.1.1class_go_1_1_point_on_curve	2650
30.360doc/html/class_go_1_1_point_on_edge.js File Reference	2650
30.360.1.Variable Documentation	2650
30.360.1.1class_go_1_1_point_on_edge	2650
30.361doc/html/class_go_1_1_point_sequence.js File Reference	2650
30.361.1.Variable Documentation	2651
30.361.1.1class_go_1_1_point_sequence	2651
30.362doc/html/class_go_1_1_project_curve.js File Reference	2651
30.362.1.Variable Documentation	2651
30.362.1.1class_go_1_1_project_curve	2651
30.363doc/html/class_go_1_1_project_curve_and_cross_tan.js File Reference	2651
30.363.1.Variable Documentation	2652
30.363.1.1class_go_1_1_project_curve_and_cross_tan	2652
30.364doc/html/class_go_1_1_project_intersection_curve.js File Reference	2652
30.364.1.Variable Documentation	2652

30.364.1.1class_go_1_1_project_intersection_curve	2652
30.364doc/html/class_go_1_1_pt_pt_intersector.js File Reference	2652
30.365.1Variable Documentation	2653
30.365.1.1class_go_1_1_pt_pt_intersector	2653
30.365doc/html/class_go_1_1_quad_mesh.js File Reference	2653
30.366.1Variable Documentation	2653
30.366.1.1class_go_1_1_quad_mesh	2653
30.366doc/html/class_go_1_1_quality_results.js File Reference	2654
30.367.1Variable Documentation	2654
30.367.1.1class_go_1_1_quality_results	2654
30.367doc/html/class_go_1_1_rational.js File Reference	2654
30.368.1Variable Documentation	2654
30.368.1.1class_go_1_1_rational	2654
30.368doc/html/class_go_1_1_rect_domain.js File Reference	2655
30.369.1Variable Documentation	2655
30.369.1.1class_go_1_1_rect_domain	2655
30.369doc/html/class_go_1_1_rect_grid.js File Reference	2655
30.370.1Variable Documentation	2655
30.370.1.1class_go_1_1_rect_grid	2655
30.370doc/html/class_go_1_1_rect_grid_tesselator.js File Reference	2656
30.371.1Variable Documentation	2656
30.371.1.1class_go_1_1_rect_grid_tesselator	2656
30.371doc/html/class_go_1_1_rectangular_surface_tesselator.js File Reference	2656
30.372.1Variable Documentation	2656
30.372.1.1class_go_1_1_rectangular_surface_tesselator	2656
30.372doc/html/class_go_1_1_rectangular_volume_tesselator.js File Reference	2657
30.373.1Variable Documentation	2657
30.373.1.1class_go_1_1_rectangular_volume_tesselator	2657
30.373doc/html/class_go_1_1_registrator.js File Reference	2657
30.374.1Variable Documentation	2657

30.374.1.1class_go_1_1_registrator	2657
30.374doc/html/class_go_1_1_regular_mesh.js File Reference	2658
30.375.1Variable Documentation	2658
30.375.1.1class_go_1_1_regular_mesh	2658
30.375doc/html/class_go_1_1_regular_vol_mesh.js File Reference	2658
30.376.1Variable Documentation	2658
30.376.1.1class_go_1_1_regular_vol_mesh	2658
30.376doc/html/class_go_1_1_regularize_face.js File Reference	2659
30.377.1Variable Documentation	2659
30.377.1.1class_go_1_1_regularize_face	2659
30.377doc/html/class_go_1_1_regularize_face_set.js File Reference	2659
30.378.1Variable Documentation	2659
30.378.1.1class_go_1_1_regularize_face_set	2659
30.378doc/html/class_go_1_1_rotated_box.js File Reference	2660
30.379.1Variable Documentation	2660
30.379.1.1class_go_1_1_rotated_box	2660
30.379doc/html/class_go_1_1_scratch_vect.js File Reference	2660
30.380.1Variable Documentation	2660
30.380.1.1class_go_1_1_scratch_vect	2660
30.380doc/html/class_go_1_1_sf_boundary_condition.js File Reference	2661
30.381.1Variable Documentation	2661
30.381.1.1class_go_1_1_sf_boundary_condition	2661
30.381doc/html/class_go_1_1_sf_cv_intersector.js File Reference	2661
30.382.1Variable Documentation	2661
30.382.1.1class_go_1_1_sf_cv_intersector	2661
30.382doc/html/class_go_1_1_sf_point_bd_cond.js File Reference	2662
30.383.1Variable Documentation	2662
30.383.1.1class_go_1_1_sf_point_bd_cond	2662
30.383doc/html/class_go_1_1_sf_pt_intersector.js File Reference	2662
30.384.1Variable Documentation	2663

30.384.1.1class_go_1_1_sf_pt_intersector	2663
30.385doc/html/class_go_1_1_sf_self_intersector.js File Reference	2663
30.385.1.Variable Documentation	2663
30.385.1.1class_go_1_1_sf_self_intersector	2663
30.386doc/html/class_go_1_1_sf_sf_intersector.js File Reference	2664
30.386.1.Variable Documentation	2664
30.386.1.1class_go_1_1_sf_sf_intersector	2664
30.387doc/html/class_go_1_1_sf_solution.js File Reference	2664
30.387.1.Variable Documentation	2664
30.387.1.1class_go_1_1_sf_solution	2664
30.388doc/html/class_go_1_1_singularity_info.js File Reference	2664
30.388.1.Variable Documentation	2665
30.388.1.1class_go_1_1_singularity_info	2665
30.389doc/html/class_go_1_1_smooth_curve.js File Reference	2665
30.389.1.Variable Documentation	2665
30.389.1.1class_go_1_1_smooth_curve	2665
30.390doc/html/class_go_1_1_smooth_curve_set.js File Reference	2666
30.390.1.Variable Documentation	2666
30.390.1.1class_go_1_1_smooth_curve_set	2666
30.391doc/html/class_go_1_1_smooth_surf.js File Reference	2666
30.391.1.Variable Documentation	2666
30.391.1.1class_go_1_1_smooth_surf	2666
30.392doc/html/class_go_1_1_smooth_surf_set.js File Reference	2666
30.392.1.Variable Documentation	2667
30.392.1.1class_go_1_1_smooth_surf_set	2667
30.393doc/html/class_go_1_1_smooth_transition.js File Reference	2667
30.393.1.Variable Documentation	2667
30.393.1.1class_go_1_1_smooth_transition	2667
30.394doc/html/class_go_1_1_smooth_volume.js File Reference	2668
30.394.1.Variable Documentation	2668

30.394.1.1class_go_1_1_smooth_volume	2668
30.394.doc/html/class_go_1_1_solve_b_c_g.js File Reference	2668
30.395.1.Variable Documentation	2668
30.395.1.1class_go_1_1_solve_b_c_g	2668
30.395.doc/html/class_go_1_1_solve_c_g.js File Reference	2669
30.396.1.Variable Documentation	2669
30.396.1.1class_go_1_1_solve_c_g	2669
30.396.doc/html/class_go_1_1_solve_c_g_c_o.js File Reference	2669
30.397.1.Variable Documentation	2669
30.397.1.1class_go_1_1_solve_c_g_c_o	2669
30.397.doc/html/class_go_1_1_space_int_crv.js File Reference	2670
30.398.1.Variable Documentation	2670
30.398.1.1class_go_1_1_space_int_crv	2670
30.398.doc/html/class_go_1_1_sphere.js File Reference	2670
30.399.1.Variable Documentation	2670
30.399.1.1class_go_1_1_sphere	2670
30.399.doc/html/class_go_1_1_sphere_int.js File Reference	2670
30.400.1.Variable Documentation	2671
30.400.1.1class_go_1_1_sphere_int	2671
30.400.doc/html/class_go_1_1_sphere_volume.js File Reference	2671
30.401.1.Variable Documentation	2671
30.401.1.1class_go_1_1_sphere_volume	2671
30.401.doc/html/class_go_1_1_spline1_function_int.js File Reference	2672
30.402.1.Variable Documentation	2672
30.402.1.1class_go_1_1_spline1_function_int	2672
30.402.doc/html/class_go_1_1_spline2_function_int.js File Reference	2672
30.403.1.Variable Documentation	2672
30.403.1.1class_go_1_1_spline2_function_int	2672
30.403.doc/html/class_go_1_1_spline_approximator.js File Reference	2673
30.404.1.Variable Documentation	2673

30.404.1.1class_go_1_1_spline_approximator	2673
30.405doc/html/class_go_1_1_spline_curve.js File Reference	2673
30.405.1.Variable Documentation	2674
30.405.1.1class_go_1_1_spline_curve	2674
30.406doc/html/class_go_1_1_spline_curve_int.js File Reference	2674
30.406.1.Variable Documentation	2674
30.406.1.1class_go_1_1_spline_curve_int	2674
30.407doc/html/class_go_1_1_spline_interpolator.js File Reference	2674
30.407.1.Variable Documentation	2675
30.407.1.1class_go_1_1_spline_interpolator	2675
30.408doc/html/class_go_1_1_spline_surface.js File Reference	2675
30.408.1.Variable Documentation	2675
30.408.1.1class_go_1_1_spline_surface	2675
30.409doc/html/class_go_1_1_spline_surface_int.js File Reference	2676
30.409.1.Variable Documentation	2676
30.409.1.1class_go_1_1_spline_surface_int	2676
30.410doc/html/class_go_1_1_spline_volume.js File Reference	2676
30.410.1.Variable Documentation	2676
30.410.1.1class_go_1_1_spline_volume	2676
30.411doc/html/class_go_1_1_streamable.js File Reference	2676
30.411.1.Variable Documentation	2676
30.411.1.1class_go_1_1_streamable	2676
30.412doc/html/class_go_1_1_surface_assembly.js File Reference	2677
30.412.1.Variable Documentation	2677
30.412.1.1class_go_1_1_surface_assembly	2677
30.413doc/html/class_go_1_1_surface_model.js File Reference	2677
30.413.1.Variable Documentation	2677
30.413.1.1class_go_1_1_surface_model	2677
30.414doc/html/class_go_1_1_surface_of_linear_extrusion.js File Reference	2677
30.414.1.Variable Documentation	2678

30.414.1.1class_go_1_1_surface_of_linear_extrusion	2678
30.415doc/html/class_go_1_1_surface_of_revolution.js File Reference	2678
30.415.1.Variable Documentation	2678
30.415.1.1class_go_1_1_surface_of_revolution	2678
30.416doc/html/class_go_1_1_surface_on_volume.js File Reference	2678
30.416.1.Variable Documentation	2678
30.416.1.1class_go_1_1_surface_on_volume	2678
30.417doc/html/class_go_1_1_sweep_surface_creator.js File Reference	2678
30.417.1.Variable Documentation	2679
30.417.1.1class_go_1_1_sweep_surface_creator	2679
30.418doc/html/class_go_1_1_sweep_volume_creator.js File Reference	2679
30.418.1.Variable Documentation	2679
30.418.1.1class_go_1_1_sweep_volume_creator	2679
30.419doc/html/class_go_1_1_tesselator.js File Reference	2679
30.419.1.Variable Documentation	2679
30.419.1.1class_go_1_1_tesselator	2679
30.420doc/html/class_go_1_1_test_in_domain.js File Reference	2680
30.420.1.Variable Documentation	2680
30.420.1.1class_go_1_1_test_in_domain	2680
30.421doc/html/class_go_1_1_torus.js File Reference	2680
30.421.1.Variable Documentation	2680
30.421.1.1class_go_1_1_torus	2680
30.422doc/html/class_go_1_1_torus_volume.js File Reference	2680
30.422.1.Variable Documentation	2681
30.422.1.1class_go_1_1_torus_volume	2681
30.423doc/html/class_go_1_1_triangle.js File Reference	2681
30.423.1.Variable Documentation	2681
30.423.1.1class_go_1_1_triangle	2681
30.424doc/html/class_go_1_1_trim_curve.js File Reference	2682
30.424.1.Variable Documentation	2682

30.424.1.1class_go_1_1_trim_curve	2682
30.425doc/html/class_go_1_1_vertex.js File Reference	2682
30.425.1.Variable Documentation	2682
30.425.1.1class_go_1_1_vertex	2682
30.426doc/html/class_go_1_1_vol_boundary_condition.js File Reference	2682
30.426.1.Variable Documentation	2683
30.426.1.1class_go_1_1_vol_boundary_condition	2683
30.427doc/html/class_go_1_1_vol_point_bd_cond.js File Reference	2683
30.427.1.Variable Documentation	2683
30.427.1.1class_go_1_1_vol_point_bd_cond	2683
30.428doc/html/class_go_1_1_vol_solution.js File Reference	2684
30.428.1.Variable Documentation	2684
30.428.1.1class_go_1_1_vol_solution	2684
30.429doc/html/class_go_1_1_volume_adjacency.js File Reference	2684
30.429.1.Variable Documentation	2684
30.429.1.1class_go_1_1_volume_adjacency	2684
30.430doc/html/class_go_1_1_volume_model.js File Reference	2684
30.430.1.Variable Documentation	2684
30.430.1.1class_go_1_1_volume_model	2684
30.431doc/html/class_go_1_1_volume_model_file_handler.js File Reference	2685
30.431.1.Variable Documentation	2685
30.431.1.1class_go_1_1_volume_model_file_handler	2685
30.432doc/html/class_go_1_1_volume_parameter_curve.js File Reference	2685
30.432.1.Variable Documentation	2685
30.432.1.1class_go_1_1_volume_parameter_curve	2685
30.433doc/html/class_go_1_1_volume_space_curve.js File Reference	2686
30.433.1.Variable Documentation	2686
30.433.1.1class_go_1_1_volume_space_curve	2686
30.434doc/html/class_go_1_1box_structuring_1_1_bounding_box_structure.js File Reference	2686
30.434.1.Variable Documentation	2686

30.434.1.1class_go_1_1box_structuring_1_1_bounding_box_structure	2686
30.435doc/html/class_go_1_1box_structuring_1_1_sub_surface_bounding_box.js File Reference	2687
30.435.1Variable Documentation	2687
30.435.1.1class_go_1_1box_structuring_1_1_sub_surface_bounding_box	2687
30.436doc/html/class_go_1_1box_structuring_1_1_surface_data.js File Reference	2688
30.436.1Variable Documentation	2688
30.436.1.1class_go_1_1box_structuring_1_1_surface_data	2688
30.437doc/html/class_go_1_1ft_cell.js File Reference	2688
30.437.1Variable Documentation	2688
30.437.1.1class_go_1_1ft_cell	2688
30.438doc/html/class_go_1_1ft_cell_info.js File Reference	2689
30.438.1Variable Documentation	2689
30.438.1.1class_go_1_1ft_cell_info	2689
30.439doc/html/class_go_1_1ft_chart_surface.js File Reference	2689
30.439.1Variable Documentation	2689
30.439.1.1class_go_1_1ft_chart_surface	2689
30.440doc/html/class_go_1_1ft_curve.js File Reference	2689
30.440.1Variable Documentation	2689
30.440.1.1class_go_1_1ft_curve	2689
30.441doc/html/class_go_1_1ft_curve_segment.js File Reference	2690
30.441.1Variable Documentation	2690
30.441.1.1class_go_1_1ft_curve_segment	2690
30.442doc/html/class_go_1_1ft_edge.js File Reference	2690
30.442.1Variable Documentation	2690
30.442.1.1class_go_1_1ft_edge	2690
30.443doc/html/class_go_1_1ft_edge_base.js File Reference	2690
30.443.1Variable Documentation	2690
30.443.1.1class_go_1_1ft_edge_base	2690
30.444doc/html/class_go_1_1ft_face_base.js File Reference	2690
30.444.1Variable Documentation	2691

30.444.1.1class_go_1_1ft_face_base	2691
30.445doc/html/class_go_1_1ft_graph_edge.js File Reference	2691
30.445.1Variable Documentation	2691
30.445.1.1class_go_1_1ft_graph_edge	2691
30.446doc/html/class_go_1_1ft_group_geom.js File Reference	2692
30.446.1Variable Documentation	2692
30.446.1.1class_go_1_1ft_group_geom	2692
30.447doc/html/class_go_1_1ft_line.js File Reference	2692
30.447.1Variable Documentation	2692
30.447.1.1class_go_1_1ft_line	2692
30.448doc/html/class_go_1_1ft_message.js File Reference	2693
30.448.1Variable Documentation	2693
30.448.1.1class_go_1_1ft_message	2693
30.449doc/html/class_go_1_1ft_planar_graph.js File Reference	2693
30.449.1Variable Documentation	2693
30.449.1.1class_go_1_1ft_planar_graph	2693
30.450doc/html/class_go_1_1ft_plane.js File Reference	2694
30.450.1Variable Documentation	2694
30.450.1.1class_go_1_1ft_plane	2694
30.451doc/html/class_go_1_1ft_point.js File Reference	2694
30.451.1Variable Documentation	2694
30.451.1.1class_go_1_1ft_point	2694
30.452doc/html/class_go_1_1ft_point_set.js File Reference	2695
30.452.1Variable Documentation	2695
30.452.1.1class_go_1_1ft_point_set	2695
30.453doc/html/class_go_1_1ft_sf_edge.js File Reference	2695
30.453.1Variable Documentation	2695
30.453.1.1class_go_1_1ft_sf_edge	2695
30.454doc/html/class_go_1_1ft_sample_point.js File Reference	2695
30.454.1Variable Documentation	2696

30.454.1.1class_go_1_1ft_sample_point	2696
30.454doc/html/class_go_1_1ft_search_node.js File Reference	2696
30.455.1Variable Documentation	2696
30.455.1.1class_go_1_1ft_search_node	2696
30.455doc/html/class_go_1_1ft_smooth_surf.js File Reference	2696
30.456.1Variable Documentation	2696
30.456.1.1class_go_1_1ft_smooth_surf	2696
30.456doc/html/class_go_1_1ft_surface.js File Reference	2697
30.457.1Variable Documentation	2697
30.457.1.1class_go_1_1ft_surface	2697
30.457doc/html/class_go_1_1ft_surface_set.js File Reference	2697
30.458.1Variable Documentation	2697
30.458.1.1class_go_1_1ft_surface_set	2697
30.458doc/html/class_go_1_1ft_surface_set_point.js File Reference	2697
30.459.1Variable Documentation	2697
30.459.1.1class_go_1_1ft_surface_set_point	2697
30.459doc/html/class_go_1_1ft_volume.js File Reference	2698
30.460.1Variable Documentation	2698
30.460.1.1class_go_1_1ft_volume	2698
30.460doc/html/class_go_1_1tp_edge.js File Reference	2698
30.461.1Variable Documentation	2698
30.461.1.1class_go_1_1tp_edge	2698
30.461doc/html/class_go_1_1tp_face.js File Reference	2698
30.462.1Variable Documentation	2698
30.462.1.1class_go_1_1tp_face	2698
30.462doc/html/class_go_1_1tp_march_point.js File Reference	2699
30.463.1Variable Documentation	2699
30.463.1.1class_go_1_1tp_march_point	2699
30.463doc/html/class_go_1_1tp_table_entry.js File Reference	2699
30.464.1Variable Documentation	2699

30.464.1.1class_go_1_1tp_table_entry	2699
30.465doc/html/class_go_1_1tp_topology_table.js File Reference	2700
30.465.1.Variable Documentation	2700
30.465.1.1class_go_1_1tp_topology_table	2700
30.466doc/html/class_go_1_1ttl_point.js File Reference	2700
30.466.1.Variable Documentation	2700
30.466.1.1class_go_1_1ttl_point	2700
30.467doc/html/class_handle.js File Reference	2701
30.467.1.Variable Documentation	2701
30.467.1.1class_handle	2701
30.468doc/html/class_handle_id.js File Reference	2701
30.468.1.Variable Documentation	2701
30.468.1.1class_handle_id	2701
30.469doc/html/class_heap_node.js File Reference	2702
30.469.1.Variable Documentation	2702
30.469.1.1class_heap_node	2702
30.470doc/html/class_heap_node2.js File Reference	2702
30.470.1.Variable Documentation	2702
30.470.1.1class_heap_node2	2702
30.471doc/html/class_load_and_store_flag.js File Reference	2703
30.471.1.Variable Documentation	2703
30.471.1.1class_load_and_store_flag	2703
30.472doc/html/class_matrix_col.js File Reference	2703
30.472.1.Variable Documentation	2703
30.472.1.1class_matrix_col	2703
30.473doc/html/class_matrix_col_x.js File Reference	2703
30.473.1.Variable Documentation	2704
30.473.1.1class_matrix_col_x	2704
30.474doc/html/class_matrix_row.js File Reference	2704
30.474.1.Variable Documentation	2704

30.474.1.1class_matrix_row	2704
30.474.doc/html/class_matrix_row_col.js File Reference	2704
30.475.1Variable Documentation	2704
30.475.1.1class_matrix_row_col	2704
30.475.doc/html/class_model__3pe.js File Reference	2705
30.476.1Variable Documentation	2705
30.476.1.1class_model__3pe	2705
30.476.doc/html/class_multi_dijkstra.js File Reference	2705
30.477.1Variable Documentation	2705
30.477.1.1class_multi_dijkstra	2705
30.477.doc/html/class_n_e_w_m_a_t_1_1_added_matrix.js File Reference	2706
30.478.1Variable Documentation	2706
30.478.1.1class_n_e_w_m_a_t_1_1_added_matrix	2706
30.478.doc/html/class_n_e_w_m_a_t_1_1_array_length_specifier.js File Reference	2706
30.479.1Variable Documentation	2706
30.479.1.1class_n_e_w_m_a_t_1_1_array_length_specifier	2706
30.479.doc/html/class_n_e_w_m_a_t_1_1_band_l_u_matrix.js File Reference	2706
30.480.1Variable Documentation	2707
30.480.1.1class_n_e_w_m_a_t_1_1_band_l_u_matrix	2707
30.480.doc/html/class_n_e_w_m_a_t_1_1_band_matrix.js File Reference	2707
30.481.1Variable Documentation	2707
30.481.1.1class_n_e_w_m_a_t_1_1_band_matrix	2707
30.481.doc/html/class_n_e_w_m_a_t_1_1_base_matrix.js File Reference	2707
30.482.1Variable Documentation	2707
30.482.1.1class_n_e_w_m_a_t_1_1_base_matrix	2707
30.482.doc/html/class_n_e_w_m_a_t_1_1_cannot_build_exception.js File Reference	2708
30.483.1Variable Documentation	2708
30.483.1.1class_n_e_w_m_a_t_1_1_cannot_build_exception	2708
30.483.doc/html/class_n_e_w_m_a_t_1_1_coled_matrix.js File Reference	2708
30.484.1Variable Documentation	2708

30.484.1.1class_n_e_w_m_a_t_1_1_coled_matrix	2708
30.484.doc/html/class_n_e_w_m_a_t_1_1_column_vector.js File Reference	2708
30.485.1.Variable Documentation	2709
30.485.1.1class_n_e_w_m_a_t_1_1_column_vector	2709
30.485.doc/html/class_n_e_w_m_a_t_1_1_concatenated_matrix.js File Reference	2709
30.486.1.Variable Documentation	2709
30.486.1.1class_n_e_w_m_a_t_1_1_concatenated_matrix	2709
30.486.doc/html/class_n_e_w_m_a_t_1_1_convergence_exception.js File Reference	2710
30.487.1.Variable Documentation	2710
30.487.1.1class_n_e_w_m_a_t_1_1_convergence_exception	2710
30.487.doc/html/class_n_e_w_m_a_t_1_1_crout_matrix.js File Reference	2710
30.488.1.Variable Documentation	2710
30.488.1.1class_n_e_w_m_a_t_1_1_crout_matrix	2710
30.488.doc/html/class_n_e_w_m_a_t_1_1_diaged_matrix.js File Reference	2711
30.489.1.Variable Documentation	2711
30.489.1.1class_n_e_w_m_a_t_1_1_diaged_matrix	2711
30.489.doc/html/class_n_e_w_m_a_t_1_1_diagonal_matrix.js File Reference	2711
30.490.1.Variable Documentation	2711
30.490.1.1class_n_e_w_m_a_t_1_1_diagonal_matrix	2711
30.490.doc/html/class_n_e_w_m_a_t_1_1_find_maximum2.js File Reference	2711
30.491.1.Variable Documentation	2712
30.491.1.1class_n_e_w_m_a_t_1_1_find_maximum2	2712
30.491.doc/html/class_n_e_w_m_a_t_1_1_general_matrix.js File Reference	2712
30.492.1.Variable Documentation	2712
30.492.1.1class_n_e_w_m_a_t_1_1_general_matrix	2712
30.492.doc/html/class_n_e_w_m_a_t_1_1_generic_matrix.js File Reference	2712
30.493.1.Variable Documentation	2712
30.493.1.1class_n_e_w_m_a_t_1_1_generic_matrix	2712
30.493.doc/html/class_n_e_w_m_a_t_1_1_get_sub_matrix.js File Reference	2713
30.494.1.Variable Documentation	2713

30.494.1.1class_n_e_w_m_a_t_1_1_get_sub_matrix	2713
30.494doc/html/class_n_e_w_m_a_t_1_1_identity_matrix.js File Reference	2714
30.495.1Variable Documentation	2714
30.495.1.1class_n_e_w_m_a_t_1_1_identity_matrix	2714
30.495doc/html/class_n_e_w_m_a_t_1_1_incompatible_dimensions_exception.js File Reference	2714
30.496.1Variable Documentation	2715
30.496.1.1class_n_e_w_m_a_t_1_1_incompatible_dimensions_exception	2715
30.496doc/html/class_n_e_w_m_a_t_1_1_index_exception.js File Reference	2715
30.497.1Variable Documentation	2715
30.497.1.1class_n_e_w_m_a_t_1_1_index_exception	2715
30.497doc/html/class_n_e_w_m_a_t_1_1_internal_exception.js File Reference	2715
30.498.1Variable Documentation	2716
30.498.1.1class_n_e_w_m_a_t_1_1_internal_exception	2716
30.498doc/html/class_n_e_w_m_a_t_1_1_inverted_matrix.js File Reference	2716
30.499.1Variable Documentation	2716
30.499.1.1class_n_e_w_m_a_t_1_1_inverted_matrix	2716
30.499doc/html/class_n_e_w_m_a_t_1_1_k_p_matrix.js File Reference	2716
30.500.1Variable Documentation	2717
30.500.1.1class_n_e_w_m_a_t_1_1_k_p_matrix	2717
30.500doc/html/class_n_e_w_m_a_t_1_1_l_l_d_f_i.js File Reference	2717
30.501.1Variable Documentation	2717
30.501.1.1class_n_e_w_m_a_t_1_1_l_l_d_f_i	2717
30.501doc/html/class_n_e_w_m_a_t_1_1_linear_equation_solver.js File Reference	2717
30.502.1Variable Documentation	2718
30.502.1.1class_n_e_w_m_a_t_1_1_linear_equation_solver	2718
30.502doc/html/class_n_e_w_m_a_t_1_1_log_and_sign.js File Reference	2718
30.503.1Variable Documentation	2718
30.503.1.1class_n_e_w_m_a_t_1_1_log_and_sign	2718
30.503doc/html/class_n_e_w_m_a_t_1_1_lower_band_matrix.js File Reference	2718
30.504.1Variable Documentation	2719

30.504.1.1class_n_e_w_m_a_t_1_1_lower_band_matrix	2719
30.505doc/html/class_n_e_w_m_a_t_1_1_lower_triangular_matrix.js File Reference	2719
30.505.1.Variable Documentation	2719
30.505.1.1class_n_e_w_m_a_t_1_1_lower_triangular_matrix	2719
30.506doc/html/class_n_e_w_m_a_t_1_1_m_l_e____d____f_i.js File Reference	2720
30.506.1.Variable Documentation	2720
30.506.1.1class_n_e_w_m_a_t_1_1_m_l_e____d____f_i	2720
30.507doc/html/class_n_e_w_m_a_t_1_1_mated_matrix.js File Reference	2721
30.507.1.Variable Documentation	2721
30.507.1.1class_n_e_w_m_a_t_1_1_mated_matrix	2721
30.508doc/html/class_n_e_w_m_a_t_1_1_matrix.js File Reference	2721
30.508.1.Variable Documentation	2721
30.508.1.1class_n_e_w_m_a_t_1_1_matrix	2721
30.509doc/html/class_n_e_w_m_a_t_1_1_matrix_band_width.js File Reference	2722
30.509.1.Variable Documentation	2722
30.509.1.1class_n_e_w_m_a_t_1_1_matrix_band_width	2722
30.510doc/html/class_n_e_w_m_a_t_1_1_matrix_input.js File Reference	2722
30.510.1.Variable Documentation	2722
30.510.1.1class_n_e_w_m_a_t_1_1_matrix_input	2722
30.511doc/html/class_n_e_w_m_a_t_1_1_matrix_type.js File Reference	2723
30.511.1.Variable Documentation	2723
30.511.1.1class_n_e_w_m_a_t_1_1_matrix_type	2723
30.512doc/html/class_n_e_w_m_a_t_1_1_multi_radix_counter.js File Reference	2723
30.512.1.Variable Documentation	2723
30.512.1.1class_n_e_w_m_a_t_1_1_multi_radix_counter	2723
30.513doc/html/class_n_e_w_m_a_t_1_1_multiplied_matrix.js File Reference	2723
30.513.1.Variable Documentation	2724
30.513.1.1class_n_e_w_m_a_t_1_1_multiplied_matrix	2724
30.514doc/html/class_n_e_w_m_a_t_1_1_n_p_d_exception.js File Reference	2724
30.514.1.Variable Documentation	2724

30.514.1.class_n_e_w_m_a_t_1_1_n_p_d_exception	2724
30.515.doc/html/class_n_e_w_m_a_t_1_1_neg_shifted_matrix.js File Reference	2724
30.515.1.Variable Documentation	2725
30.515.1.1.class_n_e_w_m_a_t_1_1_neg_shifted_matrix	2725
30.516.doc/html/class_n_e_w_m_a_t_1_1_negated_matrix.js File Reference	2725
30.516.1.Variable Documentation	2725
30.516.1.1.class_n_e_w_m_a_t_1_1_negated_matrix	2725
30.517.doc/html/class_n_e_w_m_a_t_1_1_non_linear_least_squares.js File Reference	2725
30.517.1.Variable Documentation	2726
30.517.1.1.class_n_e_w_m_a_t_1_1_non_linear_least_squares	2726
30.518.doc/html/class_n_e_w_m_a_t_1_1_not_defined_exception.js File Reference	2726
30.518.1.Variable Documentation	2726
30.518.1.1.class_n_e_w_m_a_t_1_1_not_defined_exception	2726
30.519.doc/html/class_n_e_w_m_a_t_1_1_not_square_exception.js File Reference	2726
30.519.1.Variable Documentation	2727
30.519.1.1.class_n_e_w_m_a_t_1_1_not_square_exception	2727
30.520.doc/html/class_n_e_w_m_a_t_1_1_overflow_exception.js File Reference	2727
30.520.1.Variable Documentation	2727
30.520.1.1.class_n_e_w_m_a_t_1_1_overflow_exception	2727
30.521.doc/html/class_n_e_w_m_a_t_1_1_program_exception.js File Reference	2727
30.521.1.Variable Documentation	2728
30.521.1.1.class_n_e_w_m_a_t_1_1_program_exception	2728
30.522.doc/html/class_n_e_w_m_a_t_1_1_r1__col__i__d.js File Reference	2728
30.522.1.Variable Documentation	2728
30.522.1.1.class_n_e_w_m_a_t_1_1_r1__col__i__d	2728
30.523.doc/html/class_n_e_w_m_a_t_1_1_return_matrix_x.js File Reference	2728
30.523.1.Variable Documentation	2729
30.523.1.1.class_n_e_w_m_a_t_1_1_return_matrix_x	2729
30.524.doc/html/class_n_e_w_m_a_t_1_1_reversed_matrix.js File Reference	2729
30.524.1.Variable Documentation	2729

30.524.1.1class_n_e_w_m_a_t_1_1_reversed_matrix	2729
30.524.doc/html/class_n_e_w_m_a_t_1_1_row_vector.js File Reference	2729
30.525.1.Variable Documentation	2730
30.525.1.1class_n_e_w_m_a_t_1_1_row_vector	2730
30.525.doc/html/class_n_e_w_m_a_t_1_1_rowed_matrix.js File Reference	2730
30.526.1.Variable Documentation	2730
30.526.1.1class_n_e_w_m_a_t_1_1_rowed_matrix	2730
30.526.doc/html/class_n_e_w_m_a_t_1_1_s_p_matrix.js File Reference	2731
30.527.1.Variable Documentation	2731
30.527.1.1class_n_e_w_m_a_t_1_1_s_p_matrix	2731
30.527.doc/html/class_n_e_w_m_a_t_1_1_scaled_matrix.js File Reference	2731
30.528.1.Variable Documentation	2731
30.528.1.1class_n_e_w_m_a_t_1_1_scaled_matrix	2731
30.528.doc/html/class_n_e_w_m_a_t_1_1_shifted_matrix.js File Reference	2732
30.529.1.Variable Documentation	2732
30.529.1.1class_n_e_w_m_a_t_1_1_shifted_matrix	2732
30.529.doc/html/class_n_e_w_m_a_t_1_1_simple_int_array.js File Reference	2732
30.530.1.Variable Documentation	2732
30.530.1.1class_n_e_w_m_a_t_1_1_simple_int_array	2732
30.530.doc/html/class_n_e_w_m_a_t_1_1_singular_exception.js File Reference	2733
30.531.1.Variable Documentation	2733
30.531.1.1class_n_e_w_m_a_t_1_1_singular_exception	2733
30.531.doc/html/class_n_e_w_m_a_t_1_1_solved_matrix.js File Reference	2733
30.532.1.Variable Documentation	2733
30.532.1.1class_n_e_w_m_a_t_1_1_solved_matrix	2733
30.532.doc/html/class_n_e_w_m_a_t_1_1_stacked_matrix.js File Reference	2733
30.533.1.Variable Documentation	2734
30.533.1.1class_n_e_w_m_a_t_1_1_stacked_matrix	2734
30.533.doc/html/class_n_e_w_m_a_t_1_1_sub_matrix_dimension_exception.js File Reference	2734
30.534.1.Variable Documentation	2734

30.534.1.1class_n_e_w_m_a_t_1_1_sub_matrix_dimension_exception	2734
30.534doc/html/class_n_e_w_m_a_t_1_1_subtracted_matrix.js File Reference	2734
30.535.1Variable Documentation	2735
30.535.1.1class_n_e_w_m_a_t_1_1_subtracted_matrix	2735
30.535doc/html/class_n_e_w_m_a_t_1_1_symmetric_band_matrix.js File Reference	2735
30.536.1Variable Documentation	2735
30.536.1.1class_n_e_w_m_a_t_1_1_symmetric_band_matrix	2735
30.536doc/html/class_n_e_w_m_a_t_1_1_symmetric_eigen_analysis.js File Reference	2735
30.537.1Variable Documentation	2735
30.537.1.1class_n_e_w_m_a_t_1_1_symmetric_eigen_analysis	2735
30.537doc/html/class_n_e_w_m_a_t_1_1_symmetric_matrix.js File Reference	2736
30.538.1Variable Documentation	2736
30.538.1.1class_n_e_w_m_a_t_1_1_symmetric_matrix	2736
30.538doc/html/class_n_e_w_m_a_t_1_1_transposed_matrix.js File Reference	2736
30.539.1Variable Documentation	2737
30.539.1.1class_n_e_w_m_a_t_1_1_transposed_matrix	2737
30.539doc/html/class_n_e_w_m_a_t_1_1_upper_band_matrix.js File Reference	2737
30.540.1Variable Documentation	2737
30.540.1.1class_n_e_w_m_a_t_1_1_upper_band_matrix	2737
30.540doc/html/class_n_e_w_m_a_t_1_1_upper_triangular_matrix.js File Reference	2738
30.541.1Variable Documentation	2738
30.541.1.1class_n_e_w_m_a_t_1_1_upper_triangular_matrix	2738
30.541doc/html/class_n_e_w_m_a_t_1_1_vector_exception.js File Reference	2739
30.542.1Variable Documentation	2739
30.542.1.1class_n_e_w_m_a_t_1_1_vector_exception	2739
30.542doc/html/class_n_e_w_m_a_t_1_1_nric_matrix.js File Reference	2739
30.543.1Variable Documentation	2739
30.543.1.1class_n_e_w_m_a_t_1_1_nric_matrix	2739
30.543doc/html/class_parametric_surface_property_sheet.js File Reference	2740
30.544.1Variable Documentation	2740

30.544.1.1class_parametric_surface_property_sheet	2740
30.544doc/html/class_path_type.js File Reference	2740
30.545.1Variable Documentation	2740
30.545.1.1class_path_type	2740
30.546doc/html/class_point_cloud_property_sheet.js File Reference	2741
30.546.1Variable Documentation	2741
30.546.1.1class_point_cloud_property_sheet	2741
30.547doc/html/class_pr_bi_c_g_stab.js File Reference	2741
30.547.1Variable Documentation	2741
30.547.1.1class_pr_bi_c_g_stab	2741
30.548doc/html/class_pr_c_g.js File Reference	2742
30.548.1Variable Documentation	2742
30.548.1.1class_pr_c_g	2742
30.549doc/html/class_pr_cell_structure.js File Reference	2742
30.549.1Variable Documentation	2742
30.549.1.1class_pr_cell_structure	2742
30.550doc/html/class_pr_explicit_connectivity.js File Reference	2743
30.550.1Variable Documentation	2743
30.550.1.1class_pr_explicit_connectivity	2743
30.551doc/html/class_pr_faber___f.js File Reference	2743
30.551.1Variable Documentation	2743
30.551.1.1class_pr_faber___f	2743
30.552doc/html/class_pr_fast_unorganized___o_p.js File Reference	2744
30.552.1Variable Documentation	2744
30.552.1.1class_pr_fast_unorganized___o_p	2744
30.553doc/html/class_pr_filterbank.js File Reference	2744
30.553.1Variable Documentation	2744
30.553.1.1class_pr_filterbank	2744
30.554doc/html/class_pr_heap.js File Reference	2745
30.554.1Variable Documentation	2745

30.554.1.1class_pr_heap	2745
30.554doc/html/class_pr_level_triangulation___o_p.js File Reference	2745
30.555.1Variable Documentation	2745
30.555.1.1class_pr_level_triangulation___o_p	2745
30.555doc/html/class_pr_mat.js File Reference	2746
30.556.1Variable Documentation	2746
30.556.1.1class_pr_mat	2746
30.556doc/html/class_pr_mat_sparse.js File Reference	2746
30.557.1Variable Documentation	2747
30.557.1.1class_pr_mat_sparse	2747
30.557doc/html/class_pr_matrix.js File Reference	2747
30.558.1Variable Documentation	2747
30.558.1.1class_pr_matrix	2747
30.558doc/html/class_pr_nested_node.js File Reference	2748
30.559.1Variable Documentation	2748
30.559.1.1class_pr_nested_node	2748
30.559doc/html/class_pr_nested_triangulation.js File Reference	2748
30.560.1Variable Documentation	2748
30.560.1.1class_pr_nested_triangulation	2748
30.560doc/html/class_pr_node.js File Reference	2749
30.561.1Variable Documentation	2749
30.561.1.1class_pr_node	2749
30.561doc/html/class_pr_organized_points.js File Reference	2749
30.562.1Variable Documentation	2749
30.562.1.1class_pr_organized_points	2749
30.562doc/html/class_pr_p_g_node.js File Reference	2749
30.563.1Variable Documentation	2750
30.563.1.1class_pr_p_g_node	2750
30.563doc/html/class_pr_param_triangulation.js File Reference	2750
30.564.1Variable Documentation	2750

30.564.1.1class_pr_param_triangulation	2750
30.564doc/html/class_pr_parametrize_bdy.js File Reference	2751
30.565.1Variable Documentation	2751
30.565.1.1class_pr_parametrize_bdy	2751
30.565doc/html/class_pr_parametrize_int.js File Reference	2751
30.566.1Variable Documentation	2751
30.566.1.1class_pr_parametrize_int	2751
30.566doc/html/class_pr_parametrize_mesh.js File Reference	2752
30.567.1Variable Documentation	2752
30.567.1.1class_pr_parametrize_mesh	2752
30.567doc/html/class_pr_planar_graph___o_p.js File Reference	2753
30.568.1Variable Documentation	2753
30.568.1.1class_pr_planar_graph___o_p	2753
30.568doc/html/class_pr_prewavelet___f.js File Reference	2753
30.569.1Variable Documentation	2753
30.569.1.1class_pr_prewavelet___f	2753
30.569doc/html/class_pr_prm_e_d_d_h_l_s.js File Reference	2754
30.570.1Variable Documentation	2754
30.570.1.1class_pr_prm_e_d_d_h_l_s	2754
30.570doc/html/class_pr_prm_experimental.js File Reference	2754
30.571.1Variable Documentation	2754
30.571.1.1class_pr_prm_experimental	2754
30.571doc/html/class_pr_prm_least_square.js File Reference	2754
30.572.1Variable Documentation	2755
30.572.1.1class_pr_prm_least_square	2755
30.572doc/html/class_pr_prm_mean_value.js File Reference	2755
30.573.1Variable Documentation	2755
30.573.1.1class_pr_prm_mean_value	2755
30.573doc/html/class_pr_prm_shp_pres.js File Reference	2755
30.574.1Variable Documentation	2756

30.574.1.1class_pr_prm_shp_pres	2756
30.574.doc/html/class_pr_prm_surface.js File Reference	2756
30.575.1Variable Documentation	2756
30.575.1.1class_pr_prm_surface	2756
30.575.doc/html/class_pr_prm_sym_mean_value.js File Reference	2756
30.576.1Variable Documentation	2757
30.576.1.1class_pr_prm_sym_mean_value	2757
30.576.doc/html/class_pr_prm_uniform.js File Reference	2757
30.577.1Variable Documentation	2757
30.577.1.1class_pr_prm_uniform	2757
30.577.doc/html/class_pr_prm_wachspress.js File Reference	2757
30.578.1Variable Documentation	2758
30.578.1.1class_pr_prm_wachspress	2758
30.578.doc/html/class_pr_rectangular_grid___o_p.js File Reference	2758
30.579.1Variable Documentation	2758
30.579.1.1class_pr_rectangular_grid___o_p	2758
30.579.doc/html/class_pr_sub_triangulation.js File Reference	2758
30.580.1Variable Documentation	2758
30.580.1.1class_pr_sub_triangulation	2758
30.580.doc/html/class_pr_thin.js File Reference	2759
30.581.1Variable Documentation	2759
30.581.1.1class_pr_thin	2759
30.581.doc/html/class_pr_threshold.js File Reference	2759
30.582.1Variable Documentation	2759
30.582.1.1class_pr_threshold	2759
30.582.doc/html/class_pr_triangle.js File Reference	2760
30.583.1Variable Documentation	2760
30.583.1.1class_pr_triangle	2760
30.583.doc/html/class_pr_triangulation___n_t.js File Reference	2760
30.584.1Variable Documentation	2760

30.584.1.1class_pr_triangulation__n_t	2760
30.584doc/html/class_pr_triangulation__o_p.js File Reference	2760
30.585.1Variable Documentation	2761
30.585.1.1class_pr_triangulation__o_p	2761
30.586doc/html/class_pr_unorganized__o_p.js File Reference	2761
30.586.1Variable Documentation	2761
30.586.1.1class_pr_unorganized__o_p	2761
30.587doc/html/class_pr_vec.js File Reference	2761
30.587.1Variable Documentation	2762
30.587.1.1class_pr_vec	2762
30.588doc/html/class_print_counter.js File Reference	2762
30.588.1Variable Documentation	2762
30.588.1.1class_print_counter	2762
30.589doc/html/class_r_b_d__c_o_m_m_o_n_1_1_bad__alloc.js File Reference	2762
30.589.1Variable Documentation	2763
30.589.1.1class_r_b_d__c_o_m_m_o_n_1_1_bad__alloc	2763
30.590doc/html/class_r_b_d__c_o_m_m_o_n_1_1_base_exception.js File Reference	2763
30.590.1Variable Documentation	2763
30.590.1.1class_r_b_d__c_o_m_m_o_n_1_1_base_exception	2763
30.591doc/html/class_r_b_d__c_o_m_m_o_n_1_1_domain__error.js File Reference	2763
30.591.1Variable Documentation	2763
30.591.1.1class_r_b_d__c_o_m_m_o_n_1_1_domain__error	2763
30.592doc/html/class_r_b_d__c_o_m_m_o_n_1_1_invalid__argument.js File Reference	2764
30.592.1Variable Documentation	2764
30.592.1.1class_r_b_d__c_o_m_m_o_n_1_1_invalid__argument	2764
30.593doc/html/class_r_b_d__c_o_m_m_o_n_1_1_janitor.js File Reference	2764
30.593.1Variable Documentation	2764
30.593.1.1class_r_b_d__c_o_m_m_o_n_1_1_janitor	2764
30.594doc/html/class_r_b_d__c_o_m_m_o_n_1_1_length__error.js File Reference	2764
30.594.1Variable Documentation	2765

30.594.1.1class_r_b_d__c_o_m_m_o_n_1_1_length__error	2765
30.594doc/html/class_r_b_d__c_o_m_m_o_n_1_1_logic__error.js File Reference	2765
30.595.1Variable Documentation	2765
30.595.1.1class_r_b_d__c_o_m_m_o_n_1_1_logic__error	2765
30.595doc/html/class_r_b_d__c_o_m_m_o_n_1_1_one_dim_solve.js File Reference	2765
30.596.1Variable Documentation	2765
30.596.1.1class_r_b_d__c_o_m_m_o_n_1_1_one_dim_solve	2765
30.596doc/html/class_r_b_d__c_o_m_m_o_n_1_1_out_of__range.js File Reference	2766
30.597.1Variable Documentation	2766
30.597.1.1class_r_b_d__c_o_m_m_o_n_1_1_out_of__range	2766
30.597doc/html/class_r_b_d__c_o_m_m_o_n_1_1_overflow__error.js File Reference	2766
30.598.1Variable Documentation	2766
30.598.1.1class_r_b_d__c_o_m_m_o_n_1_1_overflow__error	2766
30.598doc/html/class_r_b_d__c_o_m_m_o_n_1_1_r1__r1.js File Reference	2766
30.599.1Variable Documentation	2767
30.599.1.1class_r_b_d__c_o_m_m_o_n_1_1_r1__r1	2767
30.599doc/html/class_r_b_d__c_o_m_m_o_n_1_1_range__error.js File Reference	2767
30.600.1Variable Documentation	2767
30.600.1.1class_r_b_d__c_o_m_m_o_n_1_1_range__error	2767
30.600doc/html/class_r_b_d__c_o_m_m_o_n_1_1_runtime__error.js File Reference	2767
30.601.1Variable Documentation	2768
30.601.1.1class_r_b_d__c_o_m_m_o_n_1_1_runtime__error	2768
30.601doc/html/class_r_b_d__c_o_m_m_o_n_1_1_solution_exception.js File Reference	2768
30.602.1Variable Documentation	2768
30.602.1.1class_r_b_d__c_o_m_m_o_n_1_1_solution_exception	2768
30.602doc/html/class_r_b_d__c_o_m_m_o_n_1_1_tracer.js File Reference	2768
30.603.1Variable Documentation	2769
30.603.1.1class_r_b_d__c_o_m_m_o_n_1_1_tracer	2769
30.603doc/html/class_rect_matrix_col.js File Reference	2769
30.604.1Variable Documentation	2769

30.604.1.1class_rect_matrix_col	2769
30.605doc/html/class_rect_matrix_diag.js File Reference	2769
30.605.1Variable Documentation	2770
30.605.1.1class_rect_matrix_diag	2770
30.606doc/html/class_rect_matrix_row.js File Reference	2770
30.606.1Variable Documentation	2770
30.606.1.1class_rect_matrix_row	2770
30.607doc/html/class_rect_matrix_row_col.js File Reference	2770
30.607.1Variable Documentation	2771
30.607.1.1class_rect_matrix_row_col	2771
30.608doc/html/class_rectangular_surface_property_sheet.js File Reference	2771
30.608.1Variable Documentation	2771
30.608.1.1class_rectangular_surface_property_sheet	2771
30.609doc/html/class_rectangular_volume_property_sheet.js File Reference	2772
30.609.1Variable Documentation	2772
30.609.1.1class_rectangular_volume_property_sheet	2772
30.610doc/html/class_spline_curve_property_sheet.js File Reference	2772
30.610.1Variable Documentation	2772
30.610.1.1class_spline_curve_property_sheet	2772
30.611doc/html/class_surface_resolution_sheet.js File Reference	2772
30.611.1Variable Documentation	2773
30.611.1.1class_surface_resolution_sheet	2773
30.612doc/html/class_test_class.js File Reference	2773
30.612.1Variable Documentation	2773
30.612.1.1class_test_class	2773
30.613doc/html/class_unf_node_type.js File Reference	2773
30.613.1Variable Documentation	2774
30.613.1.1class_unf_node_type	2774
30.614doc/html/classbool.js File Reference	2774
30.614.1Variable Documentation	2774

30.614.1.1classbool	2774
30.615doc/html/classgv_action_sheet.js File Reference	2774
30.615.1Variable Documentation	2775
30.615.1.1classgv_action_sheet	2775
30.616doc/html/classgv_application.js File Reference	2775
30.616.1Variable Documentation	2775
30.616.1.1classgv_application	2775
30.617doc/html/classgv_application_vol_and_l_r.js File Reference	2775
30.617.1Variable Documentation	2775
30.617.1.1classgv_application_vol_and_l_r	2775
30.618doc/html/classgv_camera.js File Reference	2776
30.618.1Variable Documentation	2776
30.618.1.1classgv_camera	2776
30.619doc/html/classgv_curve_paintable.js File Reference	2776
30.619.1Variable Documentation	2776
30.619.1.1classgv_curve_paintable	2776
30.620doc/html/classgv_data.js File Reference	2776
30.620.1Variable Documentation	2776
30.620.1.1classgv_data	2776
30.621doc/html/classgv_generic_tri_paintable.js File Reference	2777
30.621.1Variable Documentation	2777
30.621.1.1classgv_generic_tri_paintable	2777
30.622doc/html/classgv_generic_tri_quad_mesh.js File Reference	2777
30.622.1Variable Documentation	2777
30.622.1.1classgv_generic_tri_quad_mesh	2777
30.623doc/html/classgv_generic_tri_quad_paintable.js File Reference	2778
30.623.1Variable Documentation	2778
30.623.1.1classgv_generic_tri_quad_paintable	2778
30.624doc/html/classgv_group.js File Reference	2778
30.624.1Variable Documentation	2778

30.624.1.1classgv_group	2778
30.625doc/html/classgv_group_property_sheet.js File Reference	2779
30.625.1Variable Documentation	2779
30.625.1.1classgv_group_property_sheet	2779
30.626doc/html/classgv_line_cloud_paintable.js File Reference	2779
30.626.1Variable Documentation	2779
30.626.1.1classgv_line_cloud_paintable	2779
30.627doc/html/classgv_noop_paintable.js File Reference	2780
30.627.1Variable Documentation	2780
30.627.1.1classgv_noop_paintable	2780
30.628doc/html/classgv_object_list.js File Reference	2780
30.628.1Variable Documentation	2780
30.628.1.1classgv_object_list	2780
30.629doc/html/classgv_observer.js File Reference	2780
30.629.1Variable Documentation	2781
30.629.1.1classgv_observer	2781
30.630doc/html/classgv_paintable.js File Reference	2781
30.630.1Variable Documentation	2781
30.630.1.1classgv_paintable	2781
30.631doc/html/classgvPainter.js File Reference	2781
30.631.1Variable Documentation	2782
30.631.1.1classgvPainter	2782
30.632doc/html/classgv_parametric_surface_paintable.js File Reference	2782
30.632.1Variable Documentation	2782
30.632.1.1classgv_parametric_surface_paintable	2782
30.633doc/html/classgv_point_cloud_paintable.js File Reference	2782
30.633.1Variable Documentation	2783
30.633.1.1classgv_point_cloud_paintable	2783
30.634doc/html/classgv_property_sheet.js File Reference	2783
30.634.1Variable Documentation	2783

30.634.1.1classgv_property_sheet	2783
30.635doc/html/classgv_quads_paintable.js File Reference	2783
30.635.1Variable Documentation	2784
30.635.1.1classgv_quads_paintable	2784
30.636doc/html/classgv_rectangular_surface_paintable.js File Reference	2784
30.636.1Variable Documentation	2784
30.636.1.1classgv_rectangular_surface_paintable	2784
30.637doc/html/classgv_rectangular_volume_paintable.js File Reference	2784
30.637.1Variable Documentation	2785
30.637.1.1classgv_rectangular_volume_paintable	2785
30.638doc/html/classgv_resolution_dialog.js File Reference	2785
30.638.1Variable Documentation	2785
30.638.1.1classgv_resolution_dialog	2785
30.639doc/html/classgv_standard_mouse_handler.js File Reference	2785
30.639.1Variable Documentation	2786
30.639.1.1classgv_standard_mouse_handler	2786
30.640doc/html/classgv_texture.js File Reference	2786
30.640.1Variable Documentation	2786
30.640.1.1classgv_texture	2786
30.641doc/html/classgv_view.js File Reference	2787
30.641.1Variable Documentation	2787
30.641.1.1classgv_view	2787
30.642doc/html/classhed_1_1_dart.js File Reference	2787
30.642.1Variable Documentation	2787
30.642.1.1classhed_1_1_dart	2787
30.643doc/html/classhed_1_1_edge.js File Reference	2787
30.643.1Variable Documentation	2788
30.643.1.1classhed_1_1_edge	2788
30.644doc/html/classhed_1_1_node.js File Reference	2788
30.644.1Variable Documentation	2788

30.644.1.1classhed_1_1_node	2788
30.645doc/html/classhed_1_1_triangulation.js File Reference	2789
30.645.1.Variable Documentation	2789
30.645.1.1classhed_1_1_triangulation	2789
30.646doc/html/classhetriang_1_1_dart.js File Reference	2789
30.646.1.Variable Documentation	2790
30.646.1.1classhetriang_1_1_dart	2790
30.647doc/html/classhetriang_1_1_edge.js File Reference	2790
30.647.1.Variable Documentation	2790
30.647.1.1classhetriang_1_1_edge	2790
30.648doc/html/classhetriang_1_1_node.js File Reference	2791
30.648.1.Variable Documentation	2791
30.648.1.1classhetriang_1_1_node	2791
30.649doc/html/classhetriang_1_1_triangulation.js File Reference	2791
30.649.1.Variable Documentation	2791
30.649.1.1classhetriang_1_1_triangulation	2791
30.650doc/html/classmaterial__appearance.js File Reference	2792
30.650.1.Variable Documentation	2792
30.650.1.1classmaterial__appearance	2792
30.651doc/html/classime__lapse.js File Reference	2792
30.651.1.Variable Documentation	2792
30.651.1.1classime__lapse	2792
30.652doc/html/classvector3t.js File Reference	2792
30.652.1.Variable Documentation	2793
30.652.1.1classvector3t	2793
30.653doc/html/cm_utils_8h.js File Reference	2793
30.653.1.Variable Documentation	2793
30.653.1.1cm_utils_8h	2793
30.654doc/html/config_8h.js File Reference	2793
30.654.1.Variable Documentation	2793

30.654.1.1config_8h	2793
30.654doc/html/construct_8c.js File Reference	2794
30.655.1Variable Documentation	2794
30.655.1.1construct_8c	2794
30.655doc/html/controlw_8h.js File Reference	2794
30.656.1Variable Documentation	2794
30.656.1.1controlw_8h	2794
30.657doc/html/crvarctang_8c.js File Reference	2795
30.657.1Variable Documentation	2795
30.657.1.1crvarctang_8c	2795
30.658doc/html/crvcrvtang_8c.js File Reference	2795
30.658.1Variable Documentation	2795
30.658.1.1crvcrvtang_8c	2795
30.659doc/html/crvlintang_8c.js File Reference	2795
30.659.1Variable Documentation	2796
30.659.1.1crvlintang_8c	2796
30.660doc/html/destruct_8c.js File Reference	2796
30.660.1Variable Documentation	2796
30.660.1.1destruct_8c	2796
30.661doc/html/dir_0015c891e2eac5fd9e489e846b834700.js File Reference	2796
30.661.1Variable Documentation	2797
30.661.1.1dir_0015c891e2eac5fd9e489e846b834700	2797
30.662doc/html/dir_038b4c324d5349c9475465ffee024b1c.js File Reference	2797
30.662.1Variable Documentation	2797
30.662.1.1dir_038b4c324d5349c9475465ffee024b1c	2797
30.663doc/html/dir_04f3dbfb06d4e757244c31fc16142d05.js File Reference	2797
30.663.1Variable Documentation	2797
30.663.1.1dir_04f3dbfb06d4e757244c31fc16142d05	2797
30.664doc/html/dir_0a492bd6eb1e5e04630449d6b6b92955.js File Reference	2798
30.664.1Variable Documentation	2798

30.664.1.dir_0a492bd6eb1e5e04630449d6b6b92955	2798
30.664.doc/html/dir_0df052b1ef5b53594ca9be5a6074277d.js File Reference	2798
30.665.Variable Documentation	2798
30.665.1.dir_0df052b1ef5b53594ca9be5a6074277d	2798
30.666.doc/html/dir_15757c9e0b20f079697e78f9f294c1ee.js File Reference	2798
30.666.Variable Documentation	2799
30.666.1.dir_15757c9e0b20f079697e78f9f294c1ee	2799
30.667.doc/html/dir_1c9e4414cb2b8eb9526a3aa2069941f8.js File Reference	2799
30.667.Variable Documentation	2799
30.667.1.dir_1c9e4414cb2b8eb9526a3aa2069941f8	2799
30.668.doc/html/dir_1ce88f9178ec82ce3d3b5366984a4a76.js File Reference	2799
30.668.Variable Documentation	2799
30.668.1.dir_1ce88f9178ec82ce3d3b5366984a4a76	2799
30.669.doc/html/dir_26434c1759c56a49292c328c033e76d7.js File Reference	2800
30.669.Variable Documentation	2800
30.669.1.dir_26434c1759c56a49292c328c033e76d7	2800
30.670.doc/html/dir_27047e9e82ebbd34bcfea6195a1d237f.js File Reference	2800
30.670.Variable Documentation	2800
30.670.1.dir_27047e9e82ebbd34bcfea6195a1d237f	2800
30.671.doc/html/dir_287d50cea2dbd2d342307d58f17ae566.js File Reference	2800
30.671.Variable Documentation	2800
30.671.1.dir_287d50cea2dbd2d342307d58f17ae566	2800
30.672.doc/html/dir_28d725a1074140e3875acd99df863a9a.js File Reference	2801
30.672.Variable Documentation	2801
30.672.1.dir_28d725a1074140e3875acd99df863a9a	2801
30.673.doc/html/dir_2e9f58fed2f8cb33783712134eca6f54.js File Reference	2801
30.673.Variable Documentation	2801
30.673.1.dir_2e9f58fed2f8cb33783712134eca6f54	2801
30.674.doc/html/dir_2ea40cee675de4bd87b06c9233dde18b.js File Reference	2801
30.674.Variable Documentation	2801

30.674.1.dir_2ea40cee675de4bd87b06c9233dde18b	2801
30.675.doc/html/dir_34a54fc55cc4c76e2db47ca198785905.js File Reference	2802
30.675.Variable Documentation	2802
30.675.1.dir_34a54fc55cc4c76e2db47ca198785905	2802
30.676.doc/html/dir_34a7c6eb6b0d9060dd547ea008608408.js File Reference	2802
30.676.Variable Documentation	2802
30.676.1.dir_34a7c6eb6b0d9060dd547ea008608408	2802
30.677.doc/html/dir_3706bca45f06e8690f2585ec746205ce.js File Reference	2802
30.677.Variable Documentation	2803
30.677.1.dir_3706bca45f06e8690f2585ec746205ce	2803
30.678.doc/html/dir_37bdf26acb6f0e99095c72f4b261057d.js File Reference	2803
30.678.Variable Documentation	2803
30.678.1.dir_37bdf26acb6f0e99095c72f4b261057d	2803
30.679.doc/html/dir_3a8e8a1c81bc0d4ea45d413753d6b9d6.js File Reference	2803
30.679.Variable Documentation	2803
30.679.1.dir_3a8e8a1c81bc0d4ea45d413753d6b9d6	2803
30.680.doc/html/dir_3b29025e4f1a16d99c0ac7281d29a749.js File Reference	2804
30.680.Variable Documentation	2804
30.680.1.dir_3b29025e4f1a16d99c0ac7281d29a749	2804
30.681.doc/html/dir_3cd14e887b3be1bb0699a0f4b9d7471b.js File Reference	2804
30.681.Variable Documentation	2804
30.681.1.dir_3cd14e887b3be1bb0699a0f4b9d7471b	2804
30.682.doc/html/dir_442358e1b75f159df2b4b10f68cb6669.js File Reference	2804
30.682.Variable Documentation	2805
30.682.1.dir_442358e1b75f159df2b4b10f68cb6669	2805
30.683.doc/html/dir_44eb60e2f3e103c46810fc3a6f6f0b38.js File Reference	2805
30.683.Variable Documentation	2805
30.683.1.dir_44eb60e2f3e103c46810fc3a6f6f0b38	2805
30.684.doc/html/dir_46fb2897033f53c047d4860af5672505.js File Reference	2805
30.684.Variable Documentation	2806

30.684.1.dir_46fb2897033f53c047d4860af5672505	2806
30.685.doc/html/dir_4dc6ea3c21164f9c4718441280129d09.js File Reference	2806
30.685.Variable Documentation	2806
30.685.1.dir_4dc6ea3c21164f9c4718441280129d09	2806
30.686.doc/html/dir_4ed957b398f8a342f35e202c6dbeff82.js File Reference	2806
30.686.Variable Documentation	2807
30.686.1.dir_4ed957b398f8a342f35e202c6dbeff82	2807
30.687.doc/html/dir_4f584829c2589cd0dd901c7bd8d4b8ca.js File Reference	2807
30.687.Variable Documentation	2807
30.687.1.dir_4f584829c2589cd0dd901c7bd8d4b8ca	2807
30.688.doc/html/dir_54c0807dc9dc58401cdf98175b301107.js File Reference	2807
30.688.Variable Documentation	2808
30.688.1.dir_54c0807dc9dc58401cdf98175b301107	2808
30.689.doc/html/dir_5eaf107a44af8927c3e88d7693acb3ec.js File Reference	2808
30.689.Variable Documentation	2808
30.689.1.dir_5eaf107a44af8927c3e88d7693acb3ec	2808
30.690.doc/html/dir_61c47cbd77726dc04327162bb82f2acf.js File Reference	2808
30.690.Variable Documentation	2808
30.690.1.dir_61c47cbd77726dc04327162bb82f2acf	2808
30.691.doc/html/dir_61f4d2722e3a1702decd4dcefd169ded.js File Reference	2809
30.691.Variable Documentation	2809
30.691.1.dir_61f4d2722e3a1702decd4dcefd169ded	2809
30.692.doc/html/dir_662c58e926d5c8a2e92c3a0efd184e27.js File Reference	2809
30.692.Variable Documentation	2809
30.692.1.dir_662c58e926d5c8a2e92c3a0efd184e27	2809
30.693.doc/html/dir_69009d91722e8158a6e1821a3d198fd0.js File Reference	2809
30.693.Variable Documentation	2810
30.693.1.dir_69009d91722e8158a6e1821a3d198fd0	2810
30.694.doc/html/dir_6917c6a4538152e6852497b387e02df2.js File Reference	2810
30.694.Variable Documentation	2810

30.694.1.dir_6917c6a4538152e6852497b387e02df2	2810
30.694.doc/html/dir_6958061072da6881959e38b0a737a4c6.js File Reference	2810
30.695.Variable Documentation	2810
30.695.1.dir_6958061072da6881959e38b0a737a4c6	2810
30.695.doc/html/dir_6a9f218ac24eb7b754dc250ffd508dae.js File Reference	2811
30.696.Variable Documentation	2811
30.696.1.dir_6a9f218ac24eb7b754dc250ffd508dae	2811
30.696.doc/html/dir_764e20a7fa93cafa3199ac45b6e6b986.js File Reference	2811
30.697.Variable Documentation	2811
30.697.1.dir_764e20a7fa93cafa3199ac45b6e6b986	2811
30.697.doc/html/dir_787e29cc9b8f49ef343dc9ff9fbda368.js File Reference	2811
30.698.Variable Documentation	2812
30.698.1.dir_787e29cc9b8f49ef343dc9ff9fbda368	2812
30.698.doc/html/dir_79b3003bf1bdb319185970350806bf23.js File Reference	2812
30.699.Variable Documentation	2812
30.699.1.dir_79b3003bf1bdb319185970350806bf23	2812
30.700.doc/html/dir_7b91ef4997e28bdb148c447548446353.js File Reference	2812
30.700.Variable Documentation	2813
30.700.1.dir_7b91ef4997e28bdb148c447548446353	2813
30.700.doc/html/dir_7c72f061721a5863212d64c180702103.js File Reference	2813
30.701.Variable Documentation	2813
30.701.1.dir_7c72f061721a5863212d64c180702103	2813
30.701.doc/html/dir_7efa0b394c677aa9d932dff09cb14a2d.js File Reference	2813
30.702.Variable Documentation	2814
30.702.1.dir_7efa0b394c677aa9d932dff09cb14a2d	2814
30.702.doc/html/dir_7f10ec973083a3eaca908361800e49c7.js File Reference	2814
30.703.Variable Documentation	2814
30.703.1.dir_7f10ec973083a3eaca908361800e49c7	2814
30.703.doc/html/dir_84497794bb37d283dbd68163bed8ef05.js File Reference	2814
30.704.Variable Documentation	2815

30.704.1.dir_84497794bb37d283dbd68163bed8ef05	2815
30.705.doc/html/dir_85c45df6158cba585b9d394ec2ca8ba6.js File Reference	2815
30.705.Variable Documentation	2815
30.705.1.dir_85c45df6158cba585b9d394ec2ca8ba6	2815
30.706.doc/html/dir_892cdac2bffabe6fa625049467eefe0a.js File Reference	2815
30.706.Variable Documentation	2816
30.706.1.dir_892cdac2bffabe6fa625049467eefe0a	2816
30.707.doc/html/dir_8946de0dbd7af8713b300be94ace6b21.js File Reference	2816
30.707.Variable Documentation	2816
30.707.1.dir_8946de0dbd7af8713b300be94ace6b21	2816
30.708.doc/html/dir_8c65b503da15ddfa217ea827bbddedc7.js File Reference	2817
30.708.Variable Documentation	2817
30.708.1.dir_8c65b503da15ddfa217ea827bbddedc7	2817
30.709.doc/html/dir_8dcf46dfbb2e2ca3d570ad71ab7f5984.js File Reference	2817
30.709.Variable Documentation	2817
30.709.1.dir_8dcf46dfbb2e2ca3d570ad71ab7f5984	2817
30.710.doc/html/dir_90fbe926ef93e71fdbf5723e062c9da4.js File Reference	2817
30.710.Variable Documentation	2818
30.710.1.dir_90fbe926ef93e71fdbf5723e062c9da4	2818
30.711.doc/html/dir_981871de7611ee7a54f2d0f3790a0f82.js File Reference	2818
30.711.Variable Documentation	2818
30.711.1.dir_981871de7611ee7a54f2d0f3790a0f82	2818
30.712.doc/html/dir_99c721c896ff1c980c5b18b3dcbaff94.js File Reference	2818
30.712.Variable Documentation	2818
30.712.1.dir_99c721c896ff1c980c5b18b3dcbaff94	2818
30.713.doc/html/dir_9df88d95c92a16255dc1e6f9b3433e60.js File Reference	2819
30.713.Variable Documentation	2819
30.713.1.dir_9df88d95c92a16255dc1e6f9b3433e60	2819
30.714.doc/html/dir_9e6bb4020941d0be38bb0d27bbe7cccb.js File Reference	2819
30.714.Variable Documentation	2819

30.714.1.dir_9e6bb4020941d0be38bb0d27bbe7cccb	2819
30.715.doc/html/dir_9e7f8af3df212ec4d479463341bb72a6.js File Reference	2819
30.715.Variable Documentation	2820
30.715.1.dir_9e7f8af3df212ec4d479463341bb72a6	2820
30.716.doc/html/dir_a0fa69fab3be50a1b4c8c4ebaca5098c.js File Reference	2820
30.716.Variable Documentation	2820
30.716.1.dir_a0fa69fab3be50a1b4c8c4ebaca5098c	2820
30.717.doc/html/dir_a39cd4dcd495fe456e1f661046bd5530.js File Reference	2820
30.717.Variable Documentation	2820
30.717.1.dir_a39cd4dcd495fe456e1f661046bd5530	2820
30.718.doc/html/dir_a537b1e5a645cb4471f6768545d20c2b.js File Reference	2821
30.718.Variable Documentation	2821
30.718.1.dir_a537b1e5a645cb4471f6768545d20c2b	2821
30.719.doc/html/dir_a6ecec081e9e6ebaf06ff4a52c078f1.js File Reference	2821
30.719.Variable Documentation	2821
30.719.1.dir_a6ecec081e9e6ebaf06ff4a52c078f1	2821
30.720.doc/html/dir_a702eb1d615fc8e494327e00bfd64e2d.js File Reference	2822
30.720.Variable Documentation	2822
30.720.1.dir_a702eb1d615fc8e494327e00bfd64e2d	2822
30.721.doc/html/dir_ab08e349df88865f421593b417138732.js File Reference	2822
30.721.Variable Documentation	2822
30.721.1.dir_ab08e349df88865f421593b417138732	2822
30.722.doc/html/dir_b07f39590aec0282ad5866778d6f956e.js File Reference	2822
30.722.Variable Documentation	2823
30.722.1.dir_b07f39590aec0282ad5866778d6f956e	2823
30.723.doc/html/dir_b168ed9c8c05331b63ad454a11f635f2.js File Reference	2823
30.723.Variable Documentation	2823
30.723.1.dir_b168ed9c8c05331b63ad454a11f635f2	2823
30.724.doc/html/dir_b28ca551f00416d098dfb2d5d86cb85e.js File Reference	2823
30.724.Variable Documentation	2823

30.724.1.dir_b28ca551f00416d098dfb2d5d86cb85e	2823
30.725.doc/html/dir_b3f8b92e7ea1946157ca7a805e4d2267.js File Reference	2824
30.725.Variable Documentation	2824
30.725.1.dir_b3f8b92e7ea1946157ca7a805e4d2267	2824
30.726.doc/html/dir_bf6a253d933d2fc523d4a21708f60d86.js File Reference	2824
30.726.Variable Documentation	2824
30.726.1.dir_bf6a253d933d2fc523d4a21708f60d86	2824
30.727.doc/html/dir_ce2be4821eafb03cec507f192e45eceb.js File Reference	2824
30.727.Variable Documentation	2825
30.727.1.dir_ce2be4821eafb03cec507f192e45eceb	2825
30.728.doc/html/dir_ceb2d94ff415c9faa44cf694ce53cd4a.js File Reference	2825
30.728.Variable Documentation	2825
30.728.1.dir_ceb2d94ff415c9faa44cf694ce53cd4a	2825
30.729.doc/html/dir_cfbe617369797e2c088f7718a7b296e4.js File Reference	2825
30.729.Variable Documentation	2826
30.729.1.dir_cfbe617369797e2c088f7718a7b296e4	2826
30.730.doc/html/dir_d3b744872420378775da279872820ed2.js File Reference	2826
30.730.Variable Documentation	2826
30.730.1.dir_d3b744872420378775da279872820ed2	2826
30.731.doc/html/dir_d3d5f8ee42e1e2dc7017f3f7157ca5bc.js File Reference	2826
30.731.Variable Documentation	2827
30.731.1.dir_d3d5f8ee42e1e2dc7017f3f7157ca5bc	2827
30.732.doc/html/dir_dcc2523c85224d60248ee4ee7462cfa8.js File Reference	2827
30.732.Variable Documentation	2827
30.732.1.dir_dcc2523c85224d60248ee4ee7462cfa8	2827
30.733.doc/html/dir_e2ef5fb48cb01c9435f4dad81bf7eab.js File Reference	2827
30.733.Variable Documentation	2827
30.733.1.dir_e2ef5fb48cb01c9435f4dad81bf7eab	2827
30.734.doc/html/dir_e3e0722e0dd9c71678af19110aee8f8.js File Reference	2828
30.734.Variable Documentation	2828

30.734.1.dir_e3e0722e0dd9c71678af19110aeeec8f8	2828
30.735.doc/html/dir_e66b996f8b4be0681f8e9eb9869a3079.js File Reference	2828
30.735.Variable Documentation	2828
30.735.1.dir_e66b996f8b4be0681f8e9eb9869a3079	2828
30.736.doc/html/dir_e846bdbb82b84d132cb98df96cb07bf0.js File Reference	2828
30.736.Variable Documentation	2829
30.736.1.dir_e846bdbb82b84d132cb98df96cb07bf0	2829
30.737.doc/html/dir_e98c7b4edc7740dfc859667cd423ef3f.js File Reference	2829
30.737.Variable Documentation	2829
30.737.1.dir_e98c7b4edc7740dfc859667cd423ef3f	2829
30.738.doc/html/dir_eb531be94dc3f2da4c4acbf84e0e7dc.js File Reference	2829
30.738.Variable Documentation	2829
30.738.1.dir_eb531be94dc3f2da4c4acbf84e0e7dc	2829
30.739.doc/html/dir_f2117bb2bb20a60155ae701cb86e7351.js File Reference	2830
30.739.Variable Documentation	2830
30.739.1.dir_f2117bb2bb20a60155ae701cb86e7351	2830
30.740.doc/html/dir_f2bf6ebad7a5cab5a643ce43ec016ae9.js File Reference	2830
30.740.Variable Documentation	2830
30.740.1.dir_f2bf6ebad7a5cab5a643ce43ec016ae9	2830
30.741.doc/html/dir_f30d67bf21cfa80c94c2f56f2c05062d.js File Reference	2831
30.741.Variable Documentation	2831
30.741.1.dir_f30d67bf21cfa80c94c2f56f2c05062d	2831
30.742.doc/html/dir_ffa09c8575bbfbed340eb641d2e0178f.js File Reference	2831
30.742.Variable Documentation	2831
30.742.1.dir_ffa09c8575bbfbed340eb641d2e0178f	2831
30.743.doc/html/dynsections.js File Reference	2831
30.743.Function Documentation	2832
30.743.1.toggleFolder(id)	2832
30.743.2.toggleInherit(id)	2832
30.743.3.toggleLevel(level)	2832

30.743.1.4.toggleVisibility(linkObj)	2832
30.743.1.5.updateStripes()	2832
30.744.doc/html/errormacros_8h.js File Reference	2832
30.744.1.Variable Documentation	2832
30.744.1.1.errormacros_8h	2832
30.745.doc/html/ev_cv_off_8c.js File Reference	2833
30.745.1.Variable Documentation	2833
30.745.1.1.ev_cv_off_8c	2833
30.746.doc/html/eval_2_crv_8c.js File Reference	2833
30.746.1.Variable Documentation	2833
30.746.1.1.eval_2_crv_8c	2833
30.747.doc/html/evalcrvarc_8c.js File Reference	2833
30.747.1.Variable Documentation	2834
30.747.1.1.evalcrvarc_8c	2834
30.748.doc/html/evaluate_8cpp.js File Reference	2834
30.748.1.Variable Documentation	2834
30.748.1.1.evaluate_8cpp	2834
30.749.doc/html/example01_8cpp.js File Reference	2834
30.749.1.Variable Documentation	2835
30.749.1.1.example01_8cpp	2835
30.750.doc/html/example02_8cpp.js File Reference	2835
30.750.1.Variable Documentation	2835
30.750.1.1.example02_8cpp	2835
30.751.doc/html/example03_8cpp.js File Reference	2835
30.751.1.Variable Documentation	2835
30.751.1.1.example03_8cpp	2835
30.752.doc/html/example04_8cpp.js File Reference	2836
30.752.1.Variable Documentation	2836
30.752.1.1.example04_8cpp	2836
30.753.doc/html/example05_8cpp.js File Reference	2836

30.753. Variable Documentation	2836
30.753.1. 1example05_8cpp	2836
30.754. doc/html/example06_8cpp.js File Reference	2836
30.754. Variable Documentation	2837
30.754.1. 1example06_8cpp	2837
30.755. doc/html/example07_8cpp.js File Reference	2837
30.755. Variable Documentation	2837
30.755.1. 1example07_8cpp	2837
30.756. doc/html/example08_8cpp.js File Reference	2837
30.756. Variable Documentation	2837
30.756.1. 1example08_8cpp	2837
30.757. doc/html/example09_8cpp.js File Reference	2838
30.757. Variable Documentation	2838
30.757.1. 1example09_8cpp	2838
30.758. doc/html/example10_8cpp.js File Reference	2838
30.758. Variable Documentation	2838
30.758.1. 1example10_8cpp	2838
30.759. doc/html/example11_8cpp.js File Reference	2838
30.759. Variable Documentation	2839
30.759.1. 1example11_8cpp	2839
30.760. doc/html/example12_8cpp.js File Reference	2839
30.760. Variable Documentation	2839
30.760.1. 1example12_8cpp	2839
30.761. doc/html/example13_8cpp.js File Reference	2839
30.761. Variable Documentation	2839
30.761.1. 1example13_8cpp	2839
30.762. doc/html/example14_8cpp.js File Reference	2840
30.762. Variable Documentation	2840
30.762.1. 1example14_8cpp	2840
30.763. doc/html/example15_8cpp.js File Reference	2840

30.763. Variable Documentation	2840
30.763.1. 1example15_8cpp	2840
30.764. doc/html/example_8cpp.js File Reference	2840
30.764. Variable Documentation	2841
30.764.1. 1example_8cpp	2841
30.765. doc/html/examples.js File Reference	2841
30.765. Variable Documentation	2841
30.765.1. 1examples	2841
30.766. doc/html/extremal_pt_surf_surf_8h.js File Reference	2841
30.766. Variable Documentation	2841
30.766.1. 1extremal_pt_surf_surf_8h	2841
30.767. doc/html/fft_8cpp.js File Reference	2842
30.767. Variable Documentation	2842
30.767.1. 1fft_8cpp	2842
30.768. doc/html/files.js File Reference	2842
30.768. Variable Documentation	2842
30.768.1. 1files	2842
30.769. doc/html/ft_curve_8h.js File Reference	2843
30.769. Variable Documentation	2843
30.769.1. 1ft_curve_8h	2843
30.770. doc/html/ft_message_8h.js File Reference	2843
30.770. Variable Documentation	2844
30.770.1. 1ft_message_8h	2844
30.771. doc/html/ft_planar_graph_8h.js File Reference	2844
30.771. Variable Documentation	2844
30.771.1. 1ft_planar_graph_8h	2844
30.772. doc/html/ft_point_set_8h.js File Reference	2844
30.772. Variable Documentation	2844
30.772.1. 1ft_point_set_8h	2844
30.773. doc/html/ft_surface_set_8h.js File Reference	2845

30.773. Variable Documentation	2845
30.773.1. <code>fft_surface_set_8h</code>	2845
30.774. <code>doc/html/ft_tang_priority_8h.js</code> File Reference	2845
30.774. Variable Documentation	2845
30.774.1. <code>fft_tang_priority_8h</code>	2845
30.775. <code>doc/html/ft_volume_tools_8h.js</code> File Reference	2845
30.775. Variable Documentation	2846
30.775.1. <code>fft_volume_tools_8h</code>	2846
30.776. <code>doc/html/functions_dup.js</code> File Reference	2846
30.776. Variable Documentation	2846
30.776.1. <code>functions_dup</code>	2846
30.777. <code>doc/html/functions_func.js</code> File Reference	2847
30.777. Variable Documentation	2847
30.777.1. <code>functions_func</code>	2847
30.778. <code>doc/html/functions_vars.js</code> File Reference	2847
30.778. Variable Documentation	2848
30.778.1. <code>functions_vars</code>	2848
30.779. <code>doc/html/garch_8cpp.js</code> File Reference	2848
30.779. Variable Documentation	2848
30.779.1. <code>garch_8cpp</code>	2848
30.780. <code>doc/html/generic__graph__algorithms_8h.js</code> File Reference	2849
30.780. Variable Documentation	2849
30.780.1. <code>generic__graph__algorithms_8h</code>	2849
30.781. <code>doc/html/generic__graph__algorithms__implementation_8h.js</code> File Reference	2849
30.781. Variable Documentation	2849
30.781.1. <code>generic__graph__algorithms__implementation_8h</code>	2849
30.782. <code>doc/html/gl__aux_8cpp.js</code> File Reference	2850
30.782. Variable Documentation	2850
30.782.1. <code>gl__aux_8cpp</code>	2850
30.783. <code>doc/html/gl__aux_8h.js</code> File Reference	2850

30.783. Variable Documentation	2850
30.783.1. <code>igl__aux_8h</code>	2850
30.784. <code>doc/html/globals_defs.js</code> File Reference	2851
30.784. Variable Documentation	2851
30.784.1. <code>globals_defs</code>	2851
30.785. <code>doc/html/globals_dup.js</code> File Reference	2851
30.785. Variable Documentation	2851
30.785.1. <code>globals_dup</code>	2851
30.786. <code>doc/html/globals_func.js</code> File Reference	2852
30.786. Variable Documentation	2852
30.786.1. <code>globals_func</code>	2852
30.787. <code>doc/html/glutils_8cpp.js</code> File Reference	2853
30.787. Variable Documentation	2853
30.787.1. <code>glutils_8cpp</code>	2853
30.788. <code>doc/html/glutils_8h.js</code> File Reference	2853
30.788. Variable Documentation	2853
30.788.1. <code>glutils_8h</code>	2853
30.789. <code>doc/html/gotools-core_2include_2_go_tools_2geometry_2copyright_8h.js</code> File Reference	2854
30.789. Variable Documentation	2854
30.789.1. <code>gotools_core_2include_2_go_tools_2geometry_2copyright_8h</code>	2854
30.790. <code>doc/html/group__rbd__common.js</code> File Reference	2854
30.790. Variable Documentation	2854
30.790.1. <code>group__rbd__common</code>	2854
30.791. <code>doc/html/gv_application_8h.js</code> File Reference	2855
30.791. Variable Documentation	2855
30.791.1. <code>gv_application_8h</code>	2855
30.792. <code>doc/html/gv_parametric_surface_paintable_8h.js</code> File Reference	2855
30.792. Variable Documentation	2855
30.792.1. <code>gv_parametric_surface_paintable_8h</code>	2855
30.793. <code>doc/html/gv_standard_mouse_handler_8h.js</code> File Reference	2855

30.793. Variable Documentation	2856
30.793.1. 1gv_standard_mouse_handler_8h	2856
30.794. doc/html/gv_texture_8h.js File Reference	2856
30.794. Variable Documentation	2856
30.794.1. 1gv_texture_8h	2856
30.795. doc/html/gv_utilities_8h.js File Reference	2857
30.795. Variable Documentation	2857
30.795.1. 1gv_utilities_8h	2857
30.796. doc/html/hholder_8cpp.js File Reference	2857
30.796. Variable Documentation	2857
30.796.1. 1hholder_8cpp	2857
30.797. doc/html/hierarchy.js File Reference	2858
30.797. Variable Documentation	2858
30.797.1. 1hierarchy	2858
30.798. doc/html/hp__s1880_8c.js File Reference	2858
30.798. Variable Documentation	2858
30.798.1. 1hp__s1880_8c	2858
30.799. doc/html/igeslib__doxymain_8h.js File Reference	2858
30.799. Variable Documentation	2858
30.799.1. 1igeslib__doxymain_8h	2858
30.800. doc/html/include_8h.js File Reference	2859
30.800. Variable Documentation	2859
30.800.1. 1include_8h	2859
30.801. doc/html/intjoinper_8c.js File Reference	2859
30.801. Variable Documentation	2859
30.801.1. 1intjoinper_8c	2859
30.802. doc/html/jacobi_8cpp.js File Reference	2859
30.802. Variable Documentation	2860
30.802.1. 1jacobi_8cpp	2860
30.803. doc/html/jquery.js File Reference	2860

30.803. Function Documentation	2861
30.803.1.1b(function(){if(!b.support.reliableMarginRight){b.cssHooks.marginRight={get:↵ :function(bw, bv){var e;b.swap(bw,{display:"inline-block"}, function(){if(bv){e=↵ Z(bw,"margin-right","marginRight")}else{e=bw.style.marginRight}};return e}}})	2861
30.803.1.2aach(["height","width"], function(bv, e){b.cssHooks[e]={get:function(by, bx, bw){var bz;if(bx){if(by.offsetWidth!==0){return p(by, e, bw)}else{b.swap(by, a7, function(){bz=p(by, e, bw)}}return bz}}, set:function(bw, bx){if(bc.↵ test(bx)){bx=parseFloat(bx);if(bx >=0){return bx+"px"}}else{return bx}}})	2861
30.803.1.3aextend({cssHooks:{opacity:{get:function(bw, bv){if(bv){var e=Z(bw,"opacity","opacity");return e===""?"1":e}else{return bw.style.opacity}}}, cssNumber:{fillOpacity:true, fontWeight:true, lineHeight:true, opacity:true, orphans:true, widows:true, zIndex↵ :true, zoom:true}, cssProps:{float:"b.support.cssFloat?"cssFloat":"style↵ Float"}, style:function(bx, bw, bD, by){if(!bx bx.nodeType===3 bx.↵ nodeType===8 !bx.style){return}var bB, bC, bz=b.camelCase(bw), bv=bx.style, bE=b.cssHooks[bz];bw=b.cssProps[bz] bz;if(bD!==L){b↵ C=typeof bD;if(bC===string&&(bB=l.exec(bD)){bD=(+bB[1]+1)*+bB[2]+parse↵ Float(b.css(bx, bw));bC="number"}if(bD==null bC===number&&isNa↵ N(bD)){return}if(bC===number&&!b.cssNumber[bz]){bD+="px"}if(!bE !(↵ set in bE) (bD=bE.set(bx, bD))!==L){try{bv[bw]=bD}catch(bA){}}else{if(b↵ E &&"get" in bE &&(bB=bE.get(bx, false, by))!==L){return bB}return bv[bw]}, css:function(by, bx, bv){var bw, e;bx=b.camelCase(bx);e=b.cssHooks[bx];bx=b.↵ cssProps[bx] bx;if(bx===cssFloat){bx=float}if(e &&"get" in e &&(bw=e.get(by, true, bv))!==L){return bw}else{if(Z){return Z(by, bx)}}, swap:function(bx, bw, by){var e={};for(var bv in bw){e[bv]=bx.style[bv];bx.↵ style[bv]=bw[bv]}by.call(bx);for(bv in bw){bx.style[bv]=e[bv]}}	2861
30.803.1.4f(av.documentElement.currentStyle)	2861
30.803.1.5f(av.defaultView &&av.defaultView.getComputedStyle)	2861
30.803.1.6f(b.expr &&b.expr.filters)	2862
30.803.1.7f(!b.support.opacity)	2862
30.803.1.8p(by, bw, bv)	2862
30.803.2/variable Documentation	2862
30.803.2.1aD	2862
30.803.2.2aM	2862
30.803.2.3ap	2862
30.803.2.4aQ	2862
30.803.2.5au	2862
30.803.2.6aZ	2862
30.803.2.7b	2863
30.803.2.8bb	2871
30.803.2.9bq	2871

30.803.2.16s	2871
30.803.2.1d	2871
30.803.2.12ss	2871
30.803.2.14rCSS	2872
30.803.2.14	2872
30.803.2.15	2872
30.803.2.17	2872
30.804tools-core/doc/html/jquery.js File Reference	2872
30.804. Function Documentation	2873
30.804.1.1b(function(){if(!b.support.reliableMarginRight){b.cssHooks.marginRight={get:↵ :function(bw, bv){var e;b.swap(bw,{display:"inline-block"}, function(){if(bv){e=↵ Z(bw,"margin-right","marginRight")}else{e=bw.style.marginRight}};return e}})})) 2873	
30.804.1.2each(["height","width"], function(bv, e){b.cssHooks[e]={get:function(by, bx, bw){var bz;if(bx){if(by.offsetWidth!==0){return p(by, e, bw)}else{b.swap(by, a7, function(){bz=p(by, e, bw)}})return bz}}, set:function(bw, bx){if(bc.↵ test(bx)){bx=parseFloat(bx);if(bx >=0){return bx+"px"}}else{return bx}}}) . . . 2873	
30.804.1.3extend({cssHooks:{opacity:{get:function(bw, bv){if(bv){var e=Z(bw,"opacity","opacity");return e===""?"1":e}else{return bw.style.opacity}}}, cssNumber:{fillOpacity:true, fontWeight:true, lineHeight:true, opacity:true, orphans:true, widows:true, zIndex↵ :true, zoom:true}, cssProps:{float:"b.support.cssFloat?"cssFloat":"style↵ Float"}, style:function(bx, bw, bD, by){if(!bx ! bx.nodeType===3 ! bx.↵ nodeType===8 !bx.style){return}var bB, bC, bz=b.camelCase(bw), bv=bx.style, bE=b.cssHooks[bz];bw=b.cssProps[bz] bz;if(bD!==L){b↵ C=typeof bD;if(bC=== "string"&&(bB=l.exec(bD)){bD=(+(bB[1]+1)*bB[2])+parse↵ Float(b.css(bx, bw));bC="number"}if(bD==null ! bC=== "number"&&isNa↵ N(bD)){return}if(bC=== "number"&&!b.cssNumber[bz]){bD+="px"}if(!"bE" ! ("set"↵ in bE) (bD=bE.set(bx, bD))!==L){try{bv[bw]=bD}catch(bA){}}else{if(b↵ E &&"get" in bE &&(bB=bE.get(bx, false, by))!==L){return bB}return bv[bw]}, css:function(by, bx, bv){var bw, e;bx=b.camelCase(bx);e=b.cssHooks[bx];bx=b.↵ cssProps[bx] bx;if(bx=== "cssFloat"){bx="float"}if(e &&"get" in e &&(bw=e.get(by, true, bv))!==L){return bw}else{if(Z){return Z(by, bx)}}, swap:function(bx, bw, by){var e={};for(var bv in bw){e[bv]=bx.style[bv];bx.↵ style[bv]=bw[bv]}by.call(bx);for(bv in bw){bx.style[bv]=e[bv]}}}) . . . 2873	
30.804.1.4f(av.documentElement.currentStyle)	2873
30.804.1.5f(av.defaultView &&av.defaultView.getComputedStyle)	2873
30.804.1.6f(b.expr &&b.expr.filters)	2874
30.804.1.7f(!b.support.opacity)	2874
30.804.1.8p(by, bw, bv)	2874
30.804.2variable Documentation	2874
30.804.2.1aD	2874

30.804.2.2aM	2874
30.804.2.3ap	2874
30.804.2.4aQ	2874
30.804.2.5au	2874
30.804.2.6aZ	2874
30.804.2.7b	2875
30.804.2.8bb	2883
30.804.2.9bq	2883
30.804.2.10s	2883
30.804.2.1d	2883
30.804.2.12ss	2883
30.804.2.13urCSS	2884
30.804.2.1k4	2884
30.804.2.1l5	2884
30.804.2.17	2884
30.805oc/html/jquery_8js.js File Reference	2884
30.805.1Variable Documentation	2884
30.805.1.1jquery_8js	2884
30.806oc/html/main_8cpp.js File Reference	2885
30.806.1Variable Documentation	2885
30.806.1.1main_8cpp	2885
30.807oc/html/make3_d_8c.js File Reference	2885
30.807.1Variable Documentation	2885
30.807.1.1make3_d_8c	2885
30.808oc/html/makecvkreg_8c.js File Reference	2885
30.808.1Variable Documentation	2886
30.808.1.1makecvkreg_8c	2886
30.809oc/html/makesfkgreg_8c.js File Reference	2886
30.809.1Variable Documentation	2886
30.809.1.1makesfkgreg_8c	2886

30.810doc/html/maketracks_8c.js File Reference	2886
30.810.1.Variable Documentation	2886
30.810.1.1maketracks_8c	2886
30.811doc/html/mk_cv_cycl_8c.js File Reference	2887
30.811.1.Variable Documentation	2887
30.811.1.1mk_cv_cycl_8c	2887
30.812doc/html/modules.js File Reference	2887
30.812.1.Variable Documentation	2887
30.812.1.1modules	2887
30.813doc/html/mouse_8cpp.js File Reference	2887
30.813.1.Variable Documentation	2888
30.813.1.1mouse_8cpp	2888
30.814doc/html/mouse_8h.js File Reference	2888
30.814.1.Variable Documentation	2888
30.814.1.1mouse_8h	2888
30.815doc/html/myexcept_8cpp.js File Reference	2889
30.815.1.Variable Documentation	2889
30.815.1.1myexcept_8cpp	2889
30.816doc/html/myexcept_8h.js File Reference	2889
30.816.1.Variable Documentation	2889
30.816.1.1myexcept_8h	2889
30.817doc/html/namespace_go.js File Reference	2890
30.817.1.Variable Documentation	2890
30.817.1.1namespace_go	2890
30.818doc/html/namespace_go_1_1_coons_patch_gen.js File Reference	2890
30.818.1.Variable Documentation	2890
30.818.1.1namespace_go_1_1_coons_patch_gen	2890
30.819doc/html/namespace_go_1_1_l_r_spline_utils.js File Reference	2891
30.819.1.Variable Documentation	2891
30.819.1.1namespace_go_1_1_l_r_spline_utils	2891

30.820	doc/html/namespace_go_1_1box_structuring.js File Reference	2891
30.820	Variable Documentation	2891
30.820.1	namespace_go_1_1box_structuring	2891
30.821	doc/html/namespace_n_e_w_m_a_t.js File Reference	2891
30.821	Variable Documentation	2892
30.821.1	namespace_n_e_w_m_a_t	2892
30.822	doc/html/namespace_r_b_d___c_o_m_m_o_n.js File Reference	2892
30.822	Variable Documentation	2892
30.822.1	namespace_r_b_d___c_o_m_m_o_n	2892
30.823	doc/html/namespacehed.js File Reference	2892
30.823	Variable Documentation	2893
30.823.1	namespacehed	2893
30.824	doc/html/namespacehetriang.js File Reference	2893
30.824	Variable Documentation	2893
30.824.1	namespacehetriang	2893
30.825	doc/html/namespacemembers_dup.js File Reference	2893
30.825	Variable Documentation	2894
30.825.1	namespacemembers_dup	2894
30.826	doc/html/namespacemembers_eval.js File Reference	2894
30.826	Variable Documentation	2894
30.826.1	namespacemembers_eval	2894
30.827	doc/html/namespacemembers_func.js File Reference	2895
30.827	Variable Documentation	2895
30.827.1	namespacemembers_func	2895
30.828	doc/html/namespaces.js File Reference	2895
30.828	Variable Documentation	2896
30.828.1	namespaces	2896
30.829	doc/html/navtree.js File Reference	2896
30.829	Function Documentation	2897
30.829.1	cachedLink()	2897

30.829.1.2	createIndent(o, domNode, node, level)	2897
30.829.1.3	deleteLink()	2897
30.829.1.4	expandNode(o, node, imm, showRoot)	2897
30.829.1.5	getData(varName)	2897
30.829.1.6	getNode(o, po)	2897
30.829.1.7	getScript(scriptName, func, show)	2897
30.829.1.8	glowEffect(n, duration)	2897
30.829.1.9	gotoAnchor(anchor, aname, updateLocation)	2897
30.829.1.10	gotoNode(o, subIndex, root, hash, relpath)	2898
30.829.1.11	hashUrl()	2898
30.829.1.12	hashValue()	2898
30.829.1.13	highlightAnchor()	2898
30.829.1.14	initNavTree(toroot, relpath)	2898
30.829.1.15	localStorageSupported()	2898
30.829.1.16	navTo(o, root, hash, relpath)	2898
30.829.1.17	newNode(o, po, text, link, childrenData, lastNode)	2898
30.829.1.18	pathName()	2898
30.829.1.19	removeToInsertLater(element)	2898
30.829.1.20	selectAndHighlight(hash, n)	2899
30.829.1.21	showNode(o, node, index, hash)	2899
30.829.1.22	showRoot()	2899
30.829.1.23	showSyncOff(n, relpath)	2899
30.829.1.24	showSyncOn(n, relpath)	2899
30.829.1.25	storeLink(link)	2899
30.829.1.26	stripPath(uri)	2899
30.829.1.27	stripPath2(uri)	2899
30.829.1.28	toggleSyncButton(relpath)	2899
30.829.2	variable Documentation	2899
30.829.2.1	animationInProgress	2899
30.829.2.2	navTreeSubIndices	2900

30.830doc/html/navtreedata.js File Reference	2900
30.830.1Variable Documentation	2900
30.830.1.1NAV TREE	2900
30.830.1.2NAV TREE INDEX	2900
30.830.1.3SYNCOFFMSG	2900
30.830.1.4SYNCONMSG	2900
30.831doc/html/navtreeindex0.js File Reference	2900
30.831.1Variable Documentation	2900
30.831.1.1NAV TREE INDEX 0	2900
30.832doc/html/navtreeindex1.js File Reference	2901
30.832.1Variable Documentation	2901
30.832.1.1NAV TREE INDEX 1	2901
30.833doc/html/navtreeindex10.js File Reference	2901
30.833.1Variable Documentation	2901
30.833.1.1NAV TREE INDEX 10	2901
30.834doc/html/navtreeindex11.js File Reference	2901
30.834.1Variable Documentation	2901
30.834.1.1NAV TREE INDEX 11	2901
30.835doc/html/navtreeindex12.js File Reference	2901
30.835.1Variable Documentation	2902
30.835.1.1NAV TREE INDEX 12	2902
30.836doc/html/navtreeindex13.js File Reference	2902
30.836.1Variable Documentation	2902
30.836.1.1NAV TREE INDEX 13	2902
30.837doc/html/navtreeindex14.js File Reference	2902
30.837.1Variable Documentation	2902
30.837.1.1NAV TREE INDEX 14	2902
30.838doc/html/navtreeindex15.js File Reference	2902
30.838.1Variable Documentation	2902
30.838.1.1NAV TREE INDEX 15	2902

30.836	doc/html/navtreeindex16.js File Reference	2903
30.839	Variable Documentation	2903
30.839.1	1.NAVTREEINDEX16	2903
30.840	doc/html/navtreeindex17.js File Reference	2903
30.840	Variable Documentation	2903
30.840.1	1.NAVTREEINDEX17	2903
30.841	doc/html/navtreeindex18.js File Reference	2903
30.841	Variable Documentation	2903
30.841.1	1.NAVTREEINDEX18	2903
30.842	doc/html/navtreeindex19.js File Reference	2903
30.842	Variable Documentation	2904
30.842.1	1.NAVTREEINDEX19	2904
30.843	doc/html/navtreeindex2.js File Reference	2904
30.843	Variable Documentation	2904
30.843.1	1.NAVTREEINDEX2	2904
30.844	doc/html/navtreeindex20.js File Reference	2904
30.844	Variable Documentation	2904
30.844.1	1.NAVTREEINDEX20	2904
30.845	doc/html/navtreeindex21.js File Reference	2904
30.845	Variable Documentation	2904
30.845.1	1.NAVTREEINDEX21	2904
30.846	doc/html/navtreeindex22.js File Reference	2905
30.846	Variable Documentation	2905
30.846.1	1.NAVTREEINDEX22	2905
30.847	doc/html/navtreeindex23.js File Reference	2905
30.847	Variable Documentation	2905
30.847.1	1.NAVTREEINDEX23	2905
30.848	doc/html/navtreeindex24.js File Reference	2905
30.848	Variable Documentation	2905
30.848.1	1.NAVTREEINDEX24	2905

30.846	doc/html/navtreeindex25.js File Reference	2905
30.849	Variable Documentation	2906
30.849.1	1.NAVTREEINDEX25	2906
30.850	doc/html/navtreeindex26.js File Reference	2906
30.850	Variable Documentation	2906
30.850.1	1.NAVTREEINDEX26	2906
30.851	doc/html/navtreeindex27.js File Reference	2906
30.851	Variable Documentation	2906
30.851.1	1.NAVTREEINDEX27	2906
30.852	doc/html/navtreeindex28.js File Reference	2906
30.852	Variable Documentation	2906
30.852.1	1.NAVTREEINDEX28	2906
30.853	doc/html/navtreeindex29.js File Reference	2907
30.853	Variable Documentation	2907
30.853.1	1.NAVTREEINDEX29	2907
30.854	doc/html/navtreeindex3.js File Reference	2907
30.854	Variable Documentation	2907
30.854.1	1.NAVTREEINDEX3	2907
30.855	doc/html/navtreeindex30.js File Reference	2907
30.855	Variable Documentation	2907
30.855.1	1.NAVTREEINDEX30	2907
30.856	doc/html/navtreeindex31.js File Reference	2907
30.856	Variable Documentation	2908
30.856.1	1.NAVTREEINDEX31	2908
30.857	doc/html/navtreeindex32.js File Reference	2908
30.857	Variable Documentation	2908
30.857.1	1.NAVTREEINDEX32	2908
30.858	doc/html/navtreeindex33.js File Reference	2908
30.858	Variable Documentation	2908
30.858.1	1.NAVTREEINDEX33	2908

30.855	doc/html/navtreeindex34.js File Reference	2908
30.859	Variable Documentation	2908
30.859.1	1.NAVTREEINDEX34	2908
30.860	doc/html/navtreeindex35.js File Reference	2909
30.860	Variable Documentation	2909
30.860.1	1.NAVTREEINDEX35	2909
30.861	doc/html/navtreeindex36.js File Reference	2909
30.861	Variable Documentation	2909
30.861.1	1.NAVTREEINDEX36	2909
30.862	doc/html/navtreeindex37.js File Reference	2909
30.862	Variable Documentation	2909
30.862.1	1.NAVTREEINDEX37	2909
30.863	doc/html/navtreeindex38.js File Reference	2909
30.863	Variable Documentation	2910
30.863.1	1.NAVTREEINDEX38	2910
30.864	doc/html/navtreeindex39.js File Reference	2910
30.864	Variable Documentation	2910
30.864.1	1.NAVTREEINDEX39	2910
30.865	doc/html/navtreeindex4.js File Reference	2910
30.865	Variable Documentation	2910
30.865.1	1.NAVTREEINDEX4	2910
30.866	doc/html/navtreeindex40.js File Reference	2910
30.866	Variable Documentation	2910
30.866.1	1.NAVTREEINDEX40	2910
30.867	doc/html/navtreeindex41.js File Reference	2911
30.867	Variable Documentation	2911
30.867.1	1.NAVTREEINDEX41	2911
30.868	doc/html/navtreeindex42.js File Reference	2911
30.868	Variable Documentation	2911
30.868.1	1.NAVTREEINDEX42	2911

30.866	doc/html/navtreeindex43.js File Reference	2911
30.869	Variable Documentation	2911
30.869.1	1.NAVTREEINDEX43	2911
30.870	doc/html/navtreeindex44.js File Reference	2911
30.870	Variable Documentation	2912
30.870.1	1.NAVTREEINDEX44	2912
30.871	doc/html/navtreeindex45.js File Reference	2912
30.871	Variable Documentation	2912
30.871.1	1.NAVTREEINDEX45	2912
30.872	doc/html/navtreeindex46.js File Reference	2912
30.872	Variable Documentation	2912
30.872.1	1.NAVTREEINDEX46	2912
30.873	doc/html/navtreeindex47.js File Reference	2912
30.873	Variable Documentation	2912
30.873.1	1.NAVTREEINDEX47	2912
30.874	doc/html/navtreeindex48.js File Reference	2913
30.874	Variable Documentation	2913
30.874.1	1.NAVTREEINDEX48	2913
30.875	doc/html/navtreeindex49.js File Reference	2913
30.875	Variable Documentation	2913
30.875.1	1.NAVTREEINDEX49	2913
30.876	doc/html/navtreeindex5.js File Reference	2913
30.876	Variable Documentation	2913
30.876.1	1.NAVTREEINDEX5	2913
30.877	doc/html/navtreeindex50.js File Reference	2913
30.877	Variable Documentation	2914
30.877.1	1.NAVTREEINDEX50	2914
30.878	doc/html/navtreeindex51.js File Reference	2914
30.878	Variable Documentation	2914
30.878.1	1.NAVTREEINDEX51	2914

30.876	doc/html/navtreeindex52.js File Reference	2914
30.879	Variable Documentation	2914
30.879.1	1.NAVTREEINDEX52	2914
30.880	doc/html/navtreeindex53.js File Reference	2914
30.880	Variable Documentation	2914
30.880.1	1.NAVTREEINDEX53	2914
30.881	doc/html/navtreeindex54.js File Reference	2915
30.881	Variable Documentation	2915
30.881.1	1.NAVTREEINDEX54	2915
30.882	doc/html/navtreeindex55.js File Reference	2915
30.882	Variable Documentation	2915
30.882.1	1.NAVTREEINDEX55	2915
30.883	doc/html/navtreeindex56.js File Reference	2915
30.883	Variable Documentation	2915
30.883.1	1.NAVTREEINDEX56	2915
30.884	doc/html/navtreeindex57.js File Reference	2915
30.884	Variable Documentation	2916
30.884.1	1.NAVTREEINDEX57	2916
30.885	doc/html/navtreeindex58.js File Reference	2916
30.885	Variable Documentation	2916
30.885.1	1.NAVTREEINDEX58	2916
30.886	doc/html/navtreeindex59.js File Reference	2916
30.886	Variable Documentation	2916
30.886.1	1.NAVTREEINDEX59	2916
30.887	doc/html/navtreeindex6.js File Reference	2916
30.887	Variable Documentation	2916
30.887.1	1.NAVTREEINDEX6	2916
30.888	doc/html/navtreeindex60.js File Reference	2917
30.888	Variable Documentation	2917
30.888.1	1.NAVTREEINDEX60	2917

30.886	doc/html/navtreeindex61.js File Reference	2917
30.889	Variable Documentation	2917
30.889.1	1.NAVTREEINDEX61	2917
30.890	doc/html/navtreeindex62.js File Reference	2917
30.890	Variable Documentation	2917
30.890.1	1.NAVTREEINDEX62	2917
30.891	doc/html/navtreeindex63.js File Reference	2917
30.891	Variable Documentation	2918
30.891.1	1.NAVTREEINDEX63	2918
30.892	doc/html/navtreeindex7.js File Reference	2918
30.892	Variable Documentation	2918
30.892.1	1.NAVTREEINDEX7	2918
30.893	doc/html/navtreeindex8.js File Reference	2918
30.893	Variable Documentation	2918
30.893.1	1.NAVTREEINDEX8	2918
30.894	doc/html/navtreeindex9.js File Reference	2918
30.894	Variable Documentation	2918
30.894.1	1.NAVTREEINDEX9	2918
30.895	doc/html/newfft_8cpp.js File Reference	2919
30.895	Variable Documentation	2919
30.895.1	1.newfft_8cpp	2919
30.896	doc/html/newknots_8c.js File Reference	2919
30.896	Variable Documentation	2919
30.896.1	1.newknots_8c	2919
30.897	doc/html/newmat1_8cpp.js File Reference	2919
30.897	Variable Documentation	2920
30.897.1	1.newmat1_8cpp	2920
30.898	doc/html/newmat2_8cpp.js File Reference	2920
30.898	Variable Documentation	2920
30.898.1	1.newmat2_8cpp	2920

30.899	doc/html/newmat3_8cpp.js File Reference	2920
30.899	Variable Documentation	2920
30.899.1	newmat3_8cpp	2920
30.900	doc/html/newmat4_8cpp.js File Reference	2921
30.900	Variable Documentation	2921
30.900.1	newmat4_8cpp	2921
30.901	doc/html/newmat5_8cpp.js File Reference	2921
30.901	Variable Documentation	2921
30.901.1	newmat5_8cpp	2921
30.902	doc/html/newmat6_8cpp.js File Reference	2921
30.902	Variable Documentation	2922
30.902.1	newmat6_8cpp	2922
30.903	doc/html/newmat7_8cpp.js File Reference	2922
30.903	Variable Documentation	2922
30.903.1	newmat7_8cpp	2922
30.904	doc/html/newmat8_8cpp.js File Reference	2922
30.904	Variable Documentation	2923
30.904.1	newmat8_8cpp	2923
30.905	doc/html/newmat9_8cpp.js File Reference	2923
30.905	Variable Documentation	2923
30.905.1	newmat9_8cpp	2923
30.906	doc/html/newmat_8h.js File Reference	2923
30.906	Variable Documentation	2923
30.906.1	newmat_8h	2923
30.907	doc/html/newmatap_8h.js File Reference	2924
30.907	Variable Documentation	2924
30.907.1	newmatap_8h	2924
30.908	doc/html/newmatex_8cpp.js File Reference	2924
30.908	Variable Documentation	2924
30.908.1	newmatex_8cpp	2924

30.908	doc/html/newmatio_8h.js File Reference	2924
30.909	Variable Documentation	2924
30.909.1	newmatio_8h	2924
30.910	doc/html/newmatnl_8cpp.js File Reference	2925
30.910	Variable Documentation	2925
30.910.1	newmatnl_8cpp	2925
30.911	doc/html/newmatnl_8h.js File Reference	2925
30.911	Variable Documentation	2925
30.911.1	newmatnl_8h	2925
30.912	doc/html/newmatrc_8h.js File Reference	2925
30.912	Variable Documentation	2926
30.912.1	newmatrc_8h	2926
30.913	doc/html/newmatrm_8cpp.js File Reference	2926
30.913	Variable Documentation	2926
30.913.1	newmatrm_8cpp	2926
30.914	doc/html/newmatrm_8h.js File Reference	2926
30.914	Variable Documentation	2927
30.914.1	newmatrm_8h	2927
30.915	doc/html/nl__ex_8cpp.js File Reference	2927
30.915	Variable Documentation	2927
30.915.1	nl__ex_8cpp	2927
30.916	doc/html/orient_curves_8h.js File Reference	2927
30.916	Variable Documentation	2928
30.916.1	orient_curves_8h	2928
30.917	doc/html/pickcrvsf_8c.js File Reference	2928
30.917	Variable Documentation	2928
30.917.1	pickcrvsf_8c	2928
30.918	doc/html/pocrvtang_8c.js File Reference	2928
30.918	Variable Documentation	2928
30.918.1	pocrvtang_8c	2928

30.918	doc/html/precisio_8h.js File Reference	2929
30.919	Variable Documentation	2929
30.919.1	1precisio_8h	2929
30.920	doc/html/randomnoise_8h.js File Reference	2929
30.920	Variable Documentation	2929
30.920.1	1randomnoise_8h	2929
30.921	doc/html/raster_8h.js File Reference	2929
30.921	Variable Documentation	2930
30.921.1	1raster_8h	2930
30.922	doc/html/refine__all_8c.js File Reference	2930
30.922	Variable Documentation	2930
30.922.1	1refine__all_8c	2930
30.923	doc/html/resize.js File Reference	2930
30.923	Function Documentation	2931
30.923.1	1initResizable()	2931
30.923.1	2readCookie(cookie)	2931
30.923.1	3resizeHeight()	2931
30.923.1	4resizeWidth()	2931
30.923.1	5restoreWidth(navWidth)	2931
30.923.1	6writeCookie(cookie, val, expiration)	2931
30.923.2	Variable Documentation	2931
30.923.2	1content	2931
30.923.2	2cookie_namespace	2931
30.923.2	3header	2932
30.923.2	4navtree	2932
30.923.2	5sidenav	2932
30.924	doc/html/s1001_8c.js File Reference	2932
30.924	Variable Documentation	2932
30.924.1	1s1001_8c	2932
30.925	doc/html/s1011_8c.js File Reference	2932

30.925. Variable Documentation	2933
30.925.1. <code>ts1011_8c</code>	2933
30.926. <code>doc/html/s1012_8c.js</code> File Reference	2933
30.926. Variable Documentation	2933
30.926.1. <code>ts1012_8c</code>	2933
30.927. <code>doc/html/s1013_8c.js</code> File Reference	2933
30.927. Variable Documentation	2933
30.927.1. <code>ts1013_8c</code>	2933
30.928. <code>doc/html/s1014_8c.js</code> File Reference	2934
30.928. Variable Documentation	2934
30.928.1. <code>ts1014_8c</code>	2934
30.929. <code>doc/html/s1015_8c.js</code> File Reference	2934
30.929. Variable Documentation	2934
30.929.1. <code>ts1015_8c</code>	2934
30.930. <code>doc/html/s1016_8c.js</code> File Reference	2934
30.930. Variable Documentation	2935
30.930.1. <code>ts1016_8c</code>	2935
30.931. <code>doc/html/s1017_8c.js</code> File Reference	2935
30.931. Variable Documentation	2935
30.931.1. <code>ts1017_8c</code>	2935
30.932. <code>doc/html/s1018_8c.js</code> File Reference	2935
30.932. Variable Documentation	2935
30.932.1. <code>ts1018_8c</code>	2935
30.933. <code>doc/html/s1021_8c.js</code> File Reference	2936
30.933. Variable Documentation	2936
30.933.1. <code>ts1021_8c</code>	2936
30.934. <code>doc/html/s1022_8c.js</code> File Reference	2936
30.934. Variable Documentation	2936
30.934.1. <code>ts1022_8c</code>	2936
30.935. <code>doc/html/s1023_8c.js</code> File Reference	2936

30.935. Variable Documentation	2937
30.935.1.1s1023_8c	2937
30.936. doc/html/s1024_8c.js File Reference	2937
30.936. Variable Documentation	2937
30.936.1.1s1024_8c	2937
30.937. doc/html/s1025_8c.js File Reference	2937
30.937. Variable Documentation	2937
30.937.1.1s1025_8c	2937
30.938. doc/html/s1119_8c.js File Reference	2938
30.938. Variable Documentation	2938
30.938.1.1s1119_8c	2938
30.939. doc/html/s1161_8c.js File Reference	2938
30.939. Variable Documentation	2938
30.939.1.1s1161_8c	2938
30.940. doc/html/s1162_8c.js File Reference	2938
30.940. Variable Documentation	2939
30.940.1.1s1162_8c	2939
30.941. doc/html/s1172_8c.js File Reference	2939
30.941. Variable Documentation	2939
30.941.1.1s1172_8c	2939
30.942. doc/html/s1173_8c.js File Reference	2939
30.942. Variable Documentation	2939
30.942.1.1s1173_8c	2939
30.943. doc/html/s1174_8c.js File Reference	2940
30.943. Variable Documentation	2940
30.943.1.1s1174_8c	2940
30.944. doc/html/s1190_8c.js File Reference	2940
30.944. Variable Documentation	2940
30.944.1.1s1190_8c	2940
30.945. doc/html/s1192_8c.js File Reference	2940

30.945. Variable Documentation	2941
30.945.1.1s1192_8c	2941
30.946. doc/html/s1219_8c.js File Reference	2941
30.946. Variable Documentation	2941
30.946.1.1s1219_8c	2941
30.947. doc/html/s1220_8c.js File Reference	2941
30.947. Variable Documentation	2941
30.947.1.1s1220_8c	2941
30.948. doc/html/s1221_8c.js File Reference	2942
30.948. Variable Documentation	2942
30.948.1.1s1221_8c	2942
30.949. doc/html/s1222_8c.js File Reference	2942
30.949. Variable Documentation	2942
30.949.1.1s1222_8c	2942
30.950. doc/html/s1223_8c.js File Reference	2942
30.950. Variable Documentation	2943
30.950.1.1s1223_8c	2943
30.951. doc/html/s1224_8c.js File Reference	2943
30.951. Variable Documentation	2943
30.951.1.1s1224_8c	2943
30.952. doc/html/s1225_8c.js File Reference	2943
30.952. Variable Documentation	2943
30.952.1.1s1225_8c	2943
30.953. doc/html/s1226_8c.js File Reference	2944
30.953. Variable Documentation	2944
30.953.1.1s1226_8c	2944
30.954. doc/html/s1227_8c.js File Reference	2944
30.954. Variable Documentation	2944
30.954.1.1s1227_8c	2944
30.955. doc/html/s1231_8c.js File Reference	2944

30.955. Variable Documentation	2945
30.955.1. 1s1231_8c	2945
30.956 doc/html/s1232_8c.js File Reference	2945
30.956. Variable Documentation	2945
30.956.1. 1s1232_8c	2945
30.957 doc/html/s1233_8c.js File Reference	2945
30.957. Variable Documentation	2945
30.957.1. 1s1233_8c	2945
30.958 doc/html/s1235_8c.js File Reference	2946
30.958. Variable Documentation	2946
30.958.1. 1s1235_8c	2946
30.959 doc/html/s1236_8c.js File Reference	2946
30.959. Variable Documentation	2946
30.959.1. 1s1236_8c	2946
30.960 doc/html/s1237_8c.js File Reference	2946
30.960. Variable Documentation	2947
30.960.1. 1s1237_8c	2947
30.961 doc/html/s1238_8c.js File Reference	2947
30.961. Variable Documentation	2947
30.961.1. 1s1238_8c	2947
30.962 doc/html/s1239_8c.js File Reference	2947
30.962. Variable Documentation	2947
30.962.1. 1s1239_8c	2947
30.963 doc/html/s1240_8c.js File Reference	2948
30.963. Variable Documentation	2948
30.963.1. 1s1240_8c	2948
30.964 doc/html/s1241_8c.js File Reference	2948
30.964. Variable Documentation	2948
30.964.1. 1s1241_8c	2948
30.965 doc/html/s1243_8c.js File Reference	2948

30.965. Variable Documentation	2949
30.965.1.1s1243_8c	2949
30.966. doc/html/s1244_8c.js File Reference	2949
30.966. Variable Documentation	2949
30.966.1.1s1244_8c	2949
30.967. doc/html/s1245_8c.js File Reference	2949
30.967. Variable Documentation	2949
30.967.1.1s1245_8c	2949
30.968. doc/html/s1251_8c.js File Reference	2950
30.968. Variable Documentation	2950
30.968.1.1s1251_8c	2950
30.969. doc/html/s1252_8c.js File Reference	2950
30.969. Variable Documentation	2950
30.969.1.1s1252_8c	2950
30.970. doc/html/s1301_8c.js File Reference	2950
30.970. Variable Documentation	2951
30.970.1.1s1301_8c	2951
30.971. doc/html/s1302_8c.js File Reference	2951
30.971. Variable Documentation	2951
30.971.1.1s1302_8c	2951
30.972. doc/html/s1303_8c.js File Reference	2951
30.972. Variable Documentation	2951
30.972.1.1s1303_8c	2951
30.973. doc/html/s1304_8c.js File Reference	2952
30.973. Variable Documentation	2952
30.973.1.1s1304_8c	2952
30.974. doc/html/s1305_8c.js File Reference	2952
30.974. Variable Documentation	2952
30.974.1.1s1305_8c	2952
30.975. doc/html/s1306_8c.js File Reference	2952

30.975. Variable Documentation	2953
30.975.1.1s1306_8c	2953
30.976. doc/html/s1307_8c.js File Reference	2953
30.976. Variable Documentation	2953
30.976.1.1s1307_8c	2953
30.977. doc/html/s1308_8c.js File Reference	2953
30.977. Variable Documentation	2953
30.977.1.1s1308_8c	2953
30.978. doc/html/s1309_8c.js File Reference	2954
30.978. Variable Documentation	2954
30.978.1.1s1309_8c	2954
30.979. doc/html/s1310_8c.js File Reference	2954
30.979. Variable Documentation	2954
30.979.1.1s1310_8c	2954
30.980. doc/html/s1311_8c.js File Reference	2954
30.980. Variable Documentation	2955
30.980.1.1s1311_8c	2955
30.981. doc/html/s1312_8c.js File Reference	2955
30.981. Variable Documentation	2955
30.981.1.1s1312_8c	2955
30.982. doc/html/s1313_8c.js File Reference	2955
30.982. Variable Documentation	2955
30.982.1.1s1313_8c	2955
30.983. doc/html/s1314_8c.js File Reference	2956
30.983. Variable Documentation	2956
30.983.1.1s1314_8c	2956
30.984. doc/html/s1315_8c.js File Reference	2956
30.984. Variable Documentation	2956
30.984.1.1s1315_8c	2956
30.985. doc/html/s1316_8c.js File Reference	2956

30.985. Variable Documentation	2957
30.985.1.1s1316_8c	2957
30.986. doc/html/s1317_8c.js File Reference	2957
30.986. Variable Documentation	2957
30.986.1.1s1317_8c	2957
30.987. doc/html/s1318_8c.js File Reference	2957
30.987. Variable Documentation	2957
30.987.1.1s1318_8c	2957
30.988. doc/html/s1319_8c.js File Reference	2958
30.988. Variable Documentation	2958
30.988.1.1s1319_8c	2958
30.989. doc/html/s1320_8c.js File Reference	2958
30.989. Variable Documentation	2958
30.989.1.1s1320_8c	2958
30.990. doc/html/s1321_8c.js File Reference	2958
30.990. Variable Documentation	2959
30.990.1.1s1321_8c	2959
30.991. doc/html/s1322_8c.js File Reference	2959
30.991. Variable Documentation	2959
30.991.1.1s1322_8c	2959
30.992. doc/html/s1323_8c.js File Reference	2959
30.992. Variable Documentation	2959
30.992.1.1s1323_8c	2959
30.993. doc/html/s1324_8c.js File Reference	2960
30.993. Variable Documentation	2960
30.993.1.1s1324_8c	2960
30.994. doc/html/s1325_8c.js File Reference	2960
30.994. Variable Documentation	2960
30.994.1.1s1325_8c	2960
30.995. doc/html/s1327_8c.js File Reference	2960

30.995. Variable Documentation	2961
30.995.1. s1327_8c	2961
30.996. doc/html/s1328_8c.js File Reference	2961
30.996. Variable Documentation	2961
30.996.1. s1328_8c	2961
30.997. doc/html/s1329_8c.js File Reference	2961
30.997. Variable Documentation	2961
30.997.1. s1329_8c	2961
30.998. doc/html/s1330_8c.js File Reference	2962
30.998. Variable Documentation	2962
30.998.1. s1330_8c	2962
30.999. doc/html/s1331_8c.js File Reference	2962
30.999. Variable Documentation	2962
30.999.1. s1331_8c	2962
30.1000. doc/html/s1332_8c.js File Reference	2962
30.1000. Variable Documentation	2963
30.1000.1. s1332_8c	2963
30.1001. doc/html/s1333_8c.js File Reference	2963
30.1001. Variable Documentation	2963
30.1001.1. s1333_8c	2963
30.1002. doc/html/s1333count_8c.js File Reference	2963
30.1002. Variable Documentation	2963
30.1002.1. s1333count_8c	2963
30.1003. doc/html/s1333cycli_8c.js File Reference	2964
30.1003. Variable Documentation	2964
30.1003.1. s1333cycli_8c	2964
30.1004. doc/html/s1334_8c.js File Reference	2964
30.1004. Variable Documentation	2964
30.1004.1. s1334_8c	2964
30.1005. doc/html/s1340_8c.js File Reference	2964

30.1005	Variable Documentation	2965
30.1005.1	s1340_8c	2965
30.1006	html/s1341_8c.js File Reference	2965
30.1006	Variable Documentation	2965
30.1006.1	s1341_8c	2965
30.1007	html/s1342_8c.js File Reference	2965
30.1007	Variable Documentation	2965
30.1007.1	s1342_8c	2965
30.1008	html/s1343_8c.js File Reference	2966
30.1008	Variable Documentation	2966
30.1008.1	s1343_8c	2966
30.1009	html/s1345_8c.js File Reference	2966
30.1009	Variable Documentation	2966
30.1009.1	s1345_8c	2966
30.1010	html/s1346_8c.js File Reference	2966
30.1010	Variable Documentation	2967
30.1010.1	s1346_8c	2967
30.1011	html/s1347_8c.js File Reference	2967
30.1011	Variable Documentation	2967
30.1011.1	s1347_8c	2967
30.1012	html/s1348_8c.js File Reference	2967
30.1012	Variable Documentation	2967
30.1012.1	s1348_8c	2967
30.1013	html/s1349_8c.js File Reference	2968
30.1013	Variable Documentation	2968
30.1013.1	s1349_8c	2968
30.1014	html/s1350_8c.js File Reference	2968
30.1014	Variable Documentation	2968
30.1014.1	s1350_8c	2968
30.1015	html/s1351_8c.js File Reference	2968

30.1015	Variable Documentation	2969
30.1015.1	s1351_8c	2969
30.1016	html/s1352_8c.js File Reference	2969
30.1016	Variable Documentation	2969
30.1016.1	s1352_8c	2969
30.1017	html/s1353_8c.js File Reference	2969
30.1017	Variable Documentation	2969
30.1017.1	s1353_8c	2969
30.1018	html/s1354_8c.js File Reference	2970
30.1018	Variable Documentation	2970
30.1018.1	s1354_8c	2970
30.1019	html/s1355_8c.js File Reference	2970
30.1019	Variable Documentation	2970
30.1019.1	s1355_8c	2970
30.1020	html/s1356_8c.js File Reference	2970
30.1020	Variable Documentation	2971
30.1020.1	s1356_8c	2971
30.1021	html/s1357_8c.js File Reference	2971
30.1021	Variable Documentation	2971
30.1021.1	s1357_8c	2971
30.1022	html/s1358_8c.js File Reference	2971
30.1022	Variable Documentation	2971
30.1022.1	s1358_8c	2971
30.1023	html/s1359_8c.js File Reference	2972
30.1023	Variable Documentation	2972
30.1023.1	s1359_8c	2972
30.1024	html/s1360_8c.js File Reference	2972
30.1024	Variable Documentation	2972
30.1024.1	s1360_8c	2972
30.1025	html/s1361_8c.js File Reference	2972

30.1025	Variable Documentation	2973
30.1025.1	s1361_8c	2973
30.1026	html/s1362_8c.js File Reference	2973
30.1026	Variable Documentation	2973
30.1026.1	s1362_8c	2973
30.1027	html/s1363_8c.js File Reference	2973
30.1027	Variable Documentation	2973
30.1027.1	s1363_8c	2973
30.1028	html/s1364_8c.js File Reference	2974
30.1028	Variable Documentation	2974
30.1028.1	s1364_8c	2974
30.1029	html/s1365_8c.js File Reference	2974
30.1029	Variable Documentation	2974
30.1029.1	s1365_8c	2974
30.1030	html/s1366_8c.js File Reference	2974
30.1030	Variable Documentation	2975
30.1030.1	s1366_8c	2975
30.1031	html/s1367_8c.js File Reference	2975
30.1031	Variable Documentation	2975
30.1031.1	s1367_8c	2975
30.1032	html/s1369_8c.js File Reference	2975
30.1032	Variable Documentation	2975
30.1032.1	s1369_8c	2975
30.1033	html/s1370_8c.js File Reference	2976
30.1033	Variable Documentation	2976
30.1033.1	s1370_8c	2976
30.1034	html/s1371_8c.js File Reference	2976
30.1034	Variable Documentation	2976
30.1034.1	s1371_8c	2976
30.1035	html/s1372_8c.js File Reference	2976

30.1035	Variable Documentation	2977
30.1035.1	s1372_8c	2977
30.1036	30.1036/html/s1373_8c.js File Reference	2977
30.1036	Variable Documentation	2977
30.1036.1	s1373_8c	2977
30.1037	30.1037/html/s1374_8c.js File Reference	2977
30.1037	Variable Documentation	2977
30.1037.1	s1374_8c	2977
30.1038	30.1038/html/s1375_8c.js File Reference	2978
30.1038	Variable Documentation	2978
30.1038.1	s1375_8c	2978
30.1039	30.1039/html/s1376_8c.js File Reference	2978
30.1039	Variable Documentation	2978
30.1039.1	s1376_8c	2978
30.1040	30.1040/html/s1377_8c.js File Reference	2978
30.1040	Variable Documentation	2979
30.1040.1	s1377_8c	2979
30.1041	30.1041/html/s1378_8c.js File Reference	2979
30.1041	Variable Documentation	2979
30.1041.1	s1378_8c	2979
30.1042	30.1042/html/s1379_8c.js File Reference	2979
30.1042	Variable Documentation	2979
30.1042.1	s1379_8c	2979
30.1043	30.1043/html/s1380_8c.js File Reference	2980
30.1043	Variable Documentation	2980
30.1043.1	s1380_8c	2980
30.1044	30.1044/html/s1381_8c.js File Reference	2980
30.1044	Variable Documentation	2980
30.1044.1	s1381_8c	2980
30.1045	30.1045/html/s1382_8c.js File Reference	2980

30.1045	Variable Documentation	2981
30.1045.1	s1382_8c	2981
30.1046	html/s1383_8c.js File Reference	2981
30.1046	Variable Documentation	2981
30.1046.1	s1383_8c	2981
30.1047	html/s1384_8c.js File Reference	2981
30.1047	Variable Documentation	2981
30.1047.1	s1384_8c	2981
30.1048	html/s1385_8c.js File Reference	2982
30.1048	Variable Documentation	2982
30.1048.1	s1385_8c	2982
30.1049	html/s1386_8c.js File Reference	2982
30.1049	Variable Documentation	2982
30.1049.1	s1386_8c	2982
30.1050	html/s1387_8c.js File Reference	2982
30.1050	Variable Documentation	2983
30.1050.1	s1387_8c	2983
30.1051	html/s1388_8c.js File Reference	2983
30.1051	Variable Documentation	2983
30.1051.1	s1388_8c	2983
30.1052	html/s1389_8c.js File Reference	2983
30.1052	Variable Documentation	2983
30.1052.1	s1389_8c	2983
30.1053	html/s1390_8c.js File Reference	2984
30.1053	Variable Documentation	2984
30.1053.1	s1390_8c	2984
30.1054	html/s1391_8c.js File Reference	2984
30.1054	Variable Documentation	2984
30.1054.1	s1391_8c	2984
30.1055	html/s1393_8c.js File Reference	2984

30.1055	Variable Documentation	2985
30.1055.1	s1393_8c	2985
30.1056	html/s1399_8c.js File Reference	2985
30.1056	Variable Documentation	2985
30.1056.1	s1399_8c	2985
30.1057	html/s1401_8c.js File Reference	2985
30.1057	Variable Documentation	2985
30.1057.1	s1401_8c	2985
30.1058	html/s1421_8c.js File Reference	2986
30.1058	Variable Documentation	2986
30.1058.1	s1421_8c	2986
30.1059	html/s1422_8c.js File Reference	2986
30.1059	Variable Documentation	2986
30.1059.1	s1422_8c	2986
30.1060	html/s1424_8c.js File Reference	2986
30.1060	Variable Documentation	2987
30.1060.1	s1424_8c	2987
30.1061	html/s1425_8c.js File Reference	2987
30.1061	Variable Documentation	2987
30.1061.1	s1425_8c	2987
30.1062	html/s1435_8c.js File Reference	2987
30.1062	Variable Documentation	2987
30.1062.1	s1435_8c	2987
30.1063	html/s1436_8c.js File Reference	2988
30.1063	Variable Documentation	2988
30.1063.1	s1436_8c	2988
30.1064	html/s1437_8c.js File Reference	2988
30.1064	Variable Documentation	2988
30.1064.1	s1437_8c	2988
30.1065	html/s1438_8c.js File Reference	2988

30.1065	Variable Documentation	2989
30.1065.1	s1438_8c	2989
30.1066	html/s1439_8c.js File Reference	2989
30.1066	Variable Documentation	2989
30.1066.1	s1439_8c	2989
30.1067	html/s1440_8c.js File Reference	2989
30.1067	Variable Documentation	2989
30.1067.1	s1440_8c	2989
30.1068	html/s1450_8c.js File Reference	2990
30.1068	Variable Documentation	2990
30.1068.1	s1450_8c	2990
30.1069	html/s1451_8c.js File Reference	2990
30.1069	Variable Documentation	2990
30.1069.1	s1451_8c	2990
30.1070	html/s1452_8c.js File Reference	2990
30.1070	Variable Documentation	2991
30.1070.1	s1452_8c	2991
30.1071	html/s1500_8c.js File Reference	2991
30.1071	Variable Documentation	2991
30.1071.1	s1500_8c	2991
30.1072	html/s1501_8c.js File Reference	2991
30.1072	Variable Documentation	2991
30.1072.1	s1501_8c	2991
30.1073	html/s1502_8c.js File Reference	2992
30.1073	Variable Documentation	2992
30.1073.1	s1502_8c	2992
30.1074	html/s1503_8c.js File Reference	2992
30.1074	Variable Documentation	2992
30.1074.1	s1503_8c	2992
30.1075	html/s1504_8c.js File Reference	2992

30.1075	Variable Documentation	2993
30.1075.1	s1504_8c	2993
30.1076	html/s1505_8c.js File Reference	2993
30.1076	Variable Documentation	2993
30.1076.1	s1505_8c	2993
30.1077	html/s1506_8c.js File Reference	2993
30.1077	Variable Documentation	2993
30.1077.1	s1506_8c	2993
30.1078	html/s1507_8c.js File Reference	2994
30.1078	Variable Documentation	2994
30.1078.1	s1507_8c	2994
30.1079	html/s1508_8c.js File Reference	2994
30.1079	Variable Documentation	2994
30.1079.1	s1508_8c	2994
30.1080	html/s1510_8c.js File Reference	2994
30.1080	Variable Documentation	2995
30.1080.1	s1510_8c	2995
30.1081	html/s1511_8c.js File Reference	2995
30.1081	Variable Documentation	2995
30.1081.1	s1511_8c	2995
30.1082	html/s1512_8c.js File Reference	2995
30.1082	Variable Documentation	2995
30.1082.1	s1512_8c	2995
30.1083	html/s1513_8c.js File Reference	2996
30.1083	Variable Documentation	2996
30.1083.1	s1513_8c	2996
30.1084	html/s1514_8c.js File Reference	2996
30.1084	Variable Documentation	2996
30.1084.1	s1514_8c	2996
30.1085	html/s1515_8c.js File Reference	2996

30.1085	Variable Documentation	2997
30.1085.1	s1515_8c	2997
30.1086	30.1086c/html/s1516_8c.js File Reference	2997
30.1086	Variable Documentation	2997
30.1086.1	s1516_8c	2997
30.1087	30.1087c/html/s1517_8c.js File Reference	2997
30.1087	Variable Documentation	2997
30.1087.1	s1517_8c	2997
30.1088	30.1088c/html/s1518_8c.js File Reference	2998
30.1088	Variable Documentation	2998
30.1088.1	s1518_8c	2998
30.1089	30.1089c/html/s1520_8c.js File Reference	2998
30.1089	Variable Documentation	2998
30.1089.1	s1520_8c	2998
30.1090	30.1090c/html/s1521_8c.js File Reference	2998
30.1090	Variable Documentation	2999
30.1090.1	s1521_8c	2999
30.1091	30.1091c/html/s1522_8c.js File Reference	2999
30.1091	Variable Documentation	2999
30.1091.1	s1522_8c	2999
30.1092	30.1092c/html/s1528_8c.js File Reference	2999
30.1092	Variable Documentation	2999
30.1092.1	s1528_8c	2999
30.1093	30.1093c/html/s1529_8c.js File Reference	3000
30.1093	Variable Documentation	3000
30.1093.1	s1529_8c	3000
30.1094	30.1094c/html/s1530_8c.js File Reference	3000
30.1094	Variable Documentation	3000
30.1094.1	s1530_8c	3000
30.1095	30.1095c/html/s1531_8c.js File Reference	3000

30.1095	Variable Documentation	3001
30.1095.1	s1531_8c	3001
30.1096	c/html/s1534_8c.js File Reference	3001
30.1096	Variable Documentation	3001
30.1096.1	s1534_8c	3001
30.1097	c/html/s1535_8c.js File Reference	3001
30.1097	Variable Documentation	3001
30.1097.1	s1535_8c	3001
30.1098	c/html/s1536_8c.js File Reference	3002
30.1098	Variable Documentation	3002
30.1098.1	s1536_8c	3002
30.1099	c/html/s1537_8c.js File Reference	3002
30.1099	Variable Documentation	3002
30.1099.1	s1537_8c	3002
30.1100	c/html/s1538_8c.js File Reference	3002
30.1100	Variable Documentation	3003
30.1100.1	s1538_8c	3003
30.1101	c/html/s1539_8c.js File Reference	3003
30.1101	Variable Documentation	3003
30.1101.1	s1539_8c	3003
30.1102	c/html/s1540_8c.js File Reference	3003
30.1102	Variable Documentation	3003
30.1102.1	s1540_8c	3003
30.1103	c/html/s1541_8c.js File Reference	3004
30.1103	Variable Documentation	3004
30.1103.1	s1541_8c	3004
30.1104	c/html/s1542_8c.js File Reference	3004
30.1104	Variable Documentation	3004
30.1104.1	s1542_8c	3004
30.1105	c/html/s1600_8c.js File Reference	3004

30.1105	Variable Documentation	3005
30.1105.1	s1600_8c	3005
30.1106	html/s1601_8c.js File Reference	3005
30.1106	Variable Documentation	3005
30.1106.1	s1601_8c	3005
30.1107	html/s1602_8c.js File Reference	3005
30.1107	Variable Documentation	3005
30.1107.1	s1602_8c	3005
30.1108	html/s1603_8c.js File Reference	3006
30.1108	Variable Documentation	3006
30.1108.1	s1603_8c	3006
30.1109	html/s1604_8c.js File Reference	3006
30.1109	Variable Documentation	3006
30.1109.1	s1604_8c	3006
30.1110	html/s1605_8c.js File Reference	3006
30.1110	Variable Documentation	3007
30.1110.1	s1605_8c	3007
30.1111	html/s1606_8c.js File Reference	3007
30.1111	Variable Documentation	3007
30.1111.1	s1606_8c	3007
30.1112	html/s1607_8c.js File Reference	3007
30.1112	Variable Documentation	3007
30.1112.1	s1607_8c	3007
30.1113	html/s1608_8c.js File Reference	3008
30.1113	Variable Documentation	3008
30.1113.1	s1608_8c	3008
30.1114	html/s1609_8c.js File Reference	3008
30.1114	Variable Documentation	3008
30.1114.1	s1609_8c	3008
30.1115	html/s1611_8c.js File Reference	3008

30.1115	Variable Documentation	3009
30.1115.1	s1611_8c	3009
30.1116	c/html/s1612_8c.js File Reference	3009
30.1116	Variable Documentation	3009
30.1116.1	s1612_8c	3009
30.1117	c/html/s1613_8c.js File Reference	3009
30.1117	Variable Documentation	3009
30.1117.1	s1613_8c	3009
30.1118	c/html/s1613bez_8c.js File Reference	3010
30.1118	Variable Documentation	3010
30.1118.1	s1613bez_8c	3010
30.1119	c/html/s1614_8c.js File Reference	3010
30.1119	Variable Documentation	3010
30.1119.1	s1614_8c	3010
30.1120	c/html/s1615_8c.js File Reference	3010
30.1120	Variable Documentation	3011
30.1120.1	s1615_8c	3011
30.1121	c/html/s1616_8c.js File Reference	3011
30.1121	Variable Documentation	3011
30.1121.1	s1616_8c	3011
30.1122	c/html/s1617_8c.js File Reference	3011
30.1122	Variable Documentation	3011
30.1122.1	s1617_8c	3011
30.1123	c/html/s1618_8c.js File Reference	3012
30.1123	Variable Documentation	3012
30.1123.1	s1618_8c	3012
30.1124	c/html/s1619_8c.js File Reference	3012
30.1124	Variable Documentation	3012
30.1124.1	s1619_8c	3012
30.1125	c/html/s1620_8c.js File Reference	3012

30.1125	Variable Documentation	3013
30.1125.1	s1620_8c	3013
30.1126	html/s1630_8c.js File Reference	3013
30.1126	Variable Documentation	3013
30.1126.1	s1630_8c	3013
30.1127	html/s1631_8c.js File Reference	3013
30.1127	Variable Documentation	3013
30.1127.1	s1631_8c	3013
30.1128	html/s1700_8c.js File Reference	3014
30.1128	Variable Documentation	3014
30.1128.1	s1700_8c	3014
30.1129	html/s1701_8c.js File Reference	3014
30.1129	Variable Documentation	3014
30.1129.1	s1701_8c	3014
30.1130	html/s1705_8c.js File Reference	3014
30.1130	Variable Documentation	3015
30.1130.1	s1705_8c	3015
30.1131	html/s1706_8c.js File Reference	3015
30.1131	Variable Documentation	3015
30.1131.1	s1706_8c	3015
30.1132	html/s1707_8c.js File Reference	3015
30.1132	Variable Documentation	3015
30.1132.1	s1707_8c	3015
30.1133	html/s1708_8c.js File Reference	3016
30.1133	Variable Documentation	3016
30.1133.1	s1708_8c	3016
30.1134	html/s1710_8c.js File Reference	3016
30.1134	Variable Documentation	3016
30.1134.1	s1710_8c	3016
30.1135	html/s1711_8c.js File Reference	3016

30.1135	Variable Documentation	3017
30.1135.1	s1711_8c	3017
30.1136	30c/html/s1712_8c.js File Reference	3017
30.1136	Variable Documentation	3017
30.1136.1	s1712_8c	3017
30.1137	30c/html/s1713_8c.js File Reference	3017
30.1137	Variable Documentation	3017
30.1137.1	s1713_8c	3017
30.1138	30c/html/s1714_8c.js File Reference	3018
30.1138	Variable Documentation	3018
30.1138.1	s1714_8c	3018
30.1139	30c/html/s1715_8c.js File Reference	3018
30.1139	Variable Documentation	3018
30.1139.1	s1715_8c	3018
30.1140	30c/html/s1716_8c.js File Reference	3018
30.1140	Variable Documentation	3019
30.1140.1	s1716_8c	3019
30.1141	30c/html/s1720_8c.js File Reference	3019
30.1141	Variable Documentation	3019
30.1141.1	s1720_8c	3019
30.1142	30c/html/s1730_8c.js File Reference	3019
30.1142	Variable Documentation	3019
30.1142.1	s1730_8c	3019
30.1143	30c/html/s1731_8c.js File Reference	3020
30.1143	Variable Documentation	3020
30.1143.1	s1731_8c	3020
30.1144	30c/html/s1732_8c.js File Reference	3020
30.1144	Variable Documentation	3020
30.1144.1	s1732_8c	3020
30.1145	30c/html/s1733_8c.js File Reference	3020

30.1145	Variable Documentation	3021
30.1145.1	s1733_8c	3021
30.1146	c/html/s1741_8c.js File Reference	3021
30.1146	Variable Documentation	3021
30.1146.1	s1741_8c	3021
30.1147	c/html/s1750_8c.js File Reference	3021
30.1147	Variable Documentation	3021
30.1147.1	s1750_8c	3021
30.1148	c/html/s1753_8c.js File Reference	3022
30.1148	Variable Documentation	3022
30.1148.1	s1753_8c	3022
30.1149	c/html/s1754_8c.js File Reference	3022
30.1149	Variable Documentation	3022
30.1149.1	s1754_8c	3022
30.1150	c/html/s1755_8c.js File Reference	3022
30.1150	Variable Documentation	3023
30.1150.1	s1755_8c	3023
30.1151	c/html/s17702d_8c.js File Reference	3023
30.1151	Variable Documentation	3023
30.1151.1	s17702d_8c	3023
30.1152	c/html/s1770_8c.js File Reference	3023
30.1152	Variable Documentation	3024
30.1152.1	s1770_8c	3024
30.1153	c/html/s1771_8c.js File Reference	3024
30.1153	Variable Documentation	3024
30.1153.1	s1771_8c	3024
30.1154	c/html/s1772_8c.js File Reference	3024
30.1154	Variable Documentation	3025
30.1154.1	s1772_8c	3025
30.1155	c/html/s1773_8c.js File Reference	3025

30.1155	Variable Documentation	3025
30.1155.1	s1773_8c	3025
30.1156	html/s1774_8c.js File Reference	3025
30.1156	Variable Documentation	3026
30.1156.1	s1774_8c	3026
30.1157	html/s1775_8c.js File Reference	3026
30.1157	Variable Documentation	3026
30.1157.1	s1775_8c	3026
30.1158	html/s1780_8c.js File Reference	3026
30.1158	Variable Documentation	3026
30.1158.1	s1780_8c	3026
30.1159	html/s1785_8c.js File Reference	3027
30.1159	Variable Documentation	3027
30.1159.1	s1785_8c	3027
30.1160	html/s1786_8c.js File Reference	3027
30.1160	Variable Documentation	3027
30.1160.1	s1786_8c	3027
30.1161	html/s1787_8c.js File Reference	3027
30.1161	Variable Documentation	3028
30.1161.1	s1787_8c	3028
30.1162	html/s1788_8c.js File Reference	3028
30.1162	Variable Documentation	3028
30.1162.1	s1788_8c	3028
30.1163	html/s1789_8c.js File Reference	3028
30.1163	Variable Documentation	3028
30.1163.1	s1789_8c	3028
30.1164	html/s1790_8c.js File Reference	3029
30.1164	Variable Documentation	3029
30.1164.1	s1790_8c	3029
30.1165	html/s1791_8c.js File Reference	3029

30.1165	Variable Documentation	3029
30.1165.1	s1791_8c	3029
30.1166	html/s1792_8c.js File Reference	3029
30.1166	Variable Documentation	3030
30.1166.1	s1792_8c	3030
30.1167	html/s1795_8c.js File Reference	3030
30.1167	Variable Documentation	3030
30.1167.1	s1795_8c	3030
30.1168	html/s1796_8c.js File Reference	3030
30.1168	Variable Documentation	3030
30.1168.1	s1796_8c	3030
30.1169	html/s1797_8c.js File Reference	3031
30.1169	Variable Documentation	3031
30.1169.1	s1797_8c	3031
30.1170	html/s1830_8c.js File Reference	3031
30.1170	Variable Documentation	3031
30.1170.1	s1830_8c	3031
30.1171	html/s1834_8c.js File Reference	3031
30.1171	Variable Documentation	3032
30.1171.1	s1834_8c	3032
30.1172	html/s1839_8c.js File Reference	3032
30.1172	Variable Documentation	3032
30.1172.1	s1839_8c	3032
30.1173	html/s1840_8c.js File Reference	3032
30.1173	Variable Documentation	3032
30.1173.1	s1840_8c	3032
30.1174	html/s1850_8c.js File Reference	3033
30.1174	Variable Documentation	3033
30.1174.1	s1850_8c	3033
30.1175	html/s1851_8c.js File Reference	3033

30.1175	Variable Documentation	3033
30.1175.1	s1851_8c	3033
30.1176	html/s1852_8c.js File Reference	3033
30.1176	Variable Documentation	3034
30.1176.1	s1852_8c	3034
30.1177	html/s1853_8c.js File Reference	3034
30.1177	Variable Documentation	3034
30.1177.1	s1853_8c	3034
30.1178	html/s1854_8c.js File Reference	3034
30.1178	Variable Documentation	3034
30.1178.1	s1854_8c	3034
30.1179	html/s1855_8c.js File Reference	3035
30.1179	Variable Documentation	3035
30.1179.1	s1855_8c	3035
30.1180	html/s1856_8c.js File Reference	3035
30.1180	Variable Documentation	3035
30.1180.1	s1856_8c	3035
30.1181	html/s1857_8c.js File Reference	3035
30.1181	Variable Documentation	3036
30.1181.1	s1857_8c	3036
30.1182	html/s1858_8c.js File Reference	3036
30.1182	Variable Documentation	3036
30.1182.1	s1858_8c	3036
30.1183	html/s1859_8c.js File Reference	3036
30.1183	Variable Documentation	3036
30.1183.1	s1859_8c	3036
30.1184	html/s1860_8c.js File Reference	3037
30.1184	Variable Documentation	3037
30.1184.1	s1860_8c	3037
30.1185	html/s1870_8c.js File Reference	3037

30.1185	Variable Documentation	3037
30.1185.1	s1870_8c	3037
30.1186	c/html/s1871_8c.js File Reference	3037
30.1186	Variable Documentation	3038
30.1186.1	s1871_8c	3038
30.1187	c/html/s1880_8c.js File Reference	3038
30.1187	Variable Documentation	3038
30.1187.1	s1880_8c	3038
30.1188	c/html/s1890_8c.js File Reference	3038
30.1188	Variable Documentation	3038
30.1188.1	s1890_8c	3038
30.1189	c/html/s1891_8c.js File Reference	3039
30.1189	Variable Documentation	3039
30.1189.1	s1891_8c	3039
30.1190	c/html/s1893_8c.js File Reference	3039
30.1190	Variable Documentation	3039
30.1190.1	s1893_8c	3039
30.1191	c/html/s1894_8c.js File Reference	3039
30.1191	Variable Documentation	3040
30.1191.1	s1894_8c	3040
30.1192	c/html/s1896_8c.js File Reference	3040
30.1192	Variable Documentation	3040
30.1192.1	s1896_8c	3040
30.1193	c/html/s1897_8c.js File Reference	3040
30.1193	Variable Documentation	3040
30.1193.1	s1897_8c	3040
30.1194	c/html/s1900_8c.js File Reference	3041
30.1194	Variable Documentation	3041
30.1194.1	s1900_8c	3041
30.1195	c/html/s1901_8c.js File Reference	3041

30.1195	Variable Documentation	3041
30.1195.1	s1901_8c	3041
30.1196	06c/html/s1902_8c.js File Reference	3041
30.1196	Variable Documentation	3042
30.1196.1	s1902_8c	3042
30.1197	07c/html/s1903_8c.js File Reference	3042
30.1197	Variable Documentation	3042
30.1197.1	s1903_8c	3042
30.1198	08c/html/s1904_8c.js File Reference	3042
30.1198	Variable Documentation	3042
30.1198.1	s1904_8c	3042
30.1199	09c/html/s1905_8c.js File Reference	3043
30.1199	Variable Documentation	3043
30.1199.1	s1905_8c	3043
30.1200	0c/html/s1906_8c.js File Reference	3043
30.1200	Variable Documentation	3043
30.1200.1	s1906_8c	3043
30.1201	0c/html/s1907_8c.js File Reference	3043
30.1201	Variable Documentation	3044
30.1201.1	s1907_8c	3044
30.1202	0c/html/s1908_8c.js File Reference	3044
30.1202	Variable Documentation	3044
30.1202.1	s1908_8c	3044
30.1203	0c/html/s1909_8c.js File Reference	3044
30.1203	Variable Documentation	3044
30.1203.1	s1909_8c	3044
30.1204	0c/html/s1910_8c.js File Reference	3045
30.1204	Variable Documentation	3045
30.1204.1	s1910_8c	3045
30.1205	0c/html/s1911_8c.js File Reference	3045

30.1205	Variable Documentation	3045
30.1205.1	s1911_8c	3045
30.1206	cc/html/s1912_8c.js File Reference	3045
30.1206	Variable Documentation	3046
30.1206.1	s1912_8c	3046
30.1207	cc/html/s1916_8c.js File Reference	3046
30.1207	Variable Documentation	3046
30.1207.1	s1916_8c	3046
30.1208	cc/html/s1917_8c.js File Reference	3046
30.1208	Variable Documentation	3046
30.1208.1	s1917_8c	3046
30.1209	cc/html/s1918_8c.js File Reference	3047
30.1209	Variable Documentation	3047
30.1209.1	s1918_8c	3047
30.1210	cc/html/s1919_8c.js File Reference	3047
30.1210	Variable Documentation	3047
30.1210.1	s1919_8c	3047
30.1211	cc/html/s1920_8c.js File Reference	3047
30.1211	Variable Documentation	3048
30.1211.1	s1920_8c	3048
30.1212	cc/html/s1921_8c.js File Reference	3048
30.1212	Variable Documentation	3048
30.1212.1	s1921_8c	3048
30.1213	cc/html/s1924_8c.js File Reference	3048
30.1213	Variable Documentation	3048
30.1213.1	s1924_8c	3048
30.1214	cc/html/s1925_8c.js File Reference	3049
30.1214	Variable Documentation	3049
30.1214.1	s1925_8c	3049
30.1215	cc/html/s1926_8c.js File Reference	3049

30.1215	Variable Documentation	3049
30.1215.1	s1926_8c	3049
30.1216	c/html/s1927_8c.js File Reference	3049
30.1216	Variable Documentation	3050
30.1216.1	s1927_8c	3050
30.1217	c/html/s1930_8c.js File Reference	3050
30.1217	Variable Documentation	3050
30.1217.1	s1930_8c	3050
30.1218	c/html/s1931_8c.js File Reference	3050
30.1218	Variable Documentation	3050
30.1218.1	s1931_8c	3050
30.1219	c/html/s1931unit_8c.js File Reference	3051
30.1219	Variable Documentation	3051
30.1219.1	s1931unit_8c	3051
30.1220	c/html/s1932_8c.js File Reference	3051
30.1220	Variable Documentation	3051
30.1220.1	s1932_8c	3051
30.1221	c/html/s1933_8c.js File Reference	3051
30.1221	Variable Documentation	3052
30.1221.1	s1933_8c	3052
30.1222	c/html/s1934_8c.js File Reference	3052
30.1222	Variable Documentation	3052
30.1222.1	s1934_8c	3052
30.1223	c/html/s1935_8c.js File Reference	3052
30.1223	Variable Documentation	3052
30.1223.1	s1935_8c	3052
30.1224	c/html/s1936_8c.js File Reference	3053
30.1224	Variable Documentation	3053
30.1224.1	s1936_8c	3053
30.1225	c/html/s1937_8c.js File Reference	3053

30.1225	Variable Documentation	3053
30.1225.1	s1937_8c	3053
30.1226	30c/html/s1938_8c.js File Reference	3053
30.1226	Variable Documentation	3054
30.1226.1	s1938_8c	3054
30.1227	30c/html/s1940_8c.js File Reference	3054
30.1227	Variable Documentation	3054
30.1227.1	s1940_8c	3054
30.1228	30c/html/s1941_8c.js File Reference	3054
30.1228	Variable Documentation	3054
30.1228.1	s1941_8c	3054
30.1229	30c/html/s1942_8c.js File Reference	3055
30.1229	Variable Documentation	3055
30.1229.1	s1942_8c	3055
30.1230	30c/html/s1943_8c.js File Reference	3055
30.1230	Variable Documentation	3055
30.1230.1	s1943_8c	3055
30.1231	30c/html/s1944_8c.js File Reference	3055
30.1231	Variable Documentation	3056
30.1231.1	s1944_8c	3056
30.1232	30c/html/s1945_8c.js File Reference	3056
30.1232	Variable Documentation	3056
30.1232.1	s1945_8c	3056
30.1233	30c/html/s1946_8c.js File Reference	3056
30.1233	Variable Documentation	3056
30.1233.1	s1946_8c	3056
30.1234	30c/html/s1947_8c.js File Reference	3057
30.1234	Variable Documentation	3057
30.1234.1	s1947_8c	3057
30.1235	30c/html/s1948_8c.js File Reference	3057

30.1235	Variable Documentation	3057
30.1235.1	s1948_8c	3057
30.1236	30c/html/s1949_8c.js File Reference	3057
30.1236	Variable Documentation	3058
30.1236.1	s1949_8c	3058
30.1237	30c/html/s1950_8c.js File Reference	3058
30.1237	Variable Documentation	3058
30.1237.1	s1950_8c	3058
30.1238	30c/html/s1951_8c.js File Reference	3058
30.1238	Variable Documentation	3058
30.1238.1	s1951_8c	3058
30.1239	30c/html/s1953_8c.js File Reference	3059
30.1239	Variable Documentation	3059
30.1239.1	s1953_8c	3059
30.1240	30c/html/s1954_8c.js File Reference	3059
30.1240	Variable Documentation	3059
30.1240.1	s1954_8c	3059
30.1241	30c/html/s1955_8c.js File Reference	3059
30.1241	Variable Documentation	3060
30.1241.1	s1955_8c	3060
30.1242	30c/html/s1956_8c.js File Reference	3060
30.1242	Variable Documentation	3060
30.1242.1	s1956_8c	3060
30.1243	30c/html/s1957_8c.js File Reference	3060
30.1243	Variable Documentation	3060
30.1243.1	s1957_8c	3060
30.1244	30c/html/s1958_8c.js File Reference	3061
30.1244	Variable Documentation	3061
30.1244.1	s1958_8c	3061
30.1245	30c/html/s1959_8c.js File Reference	3061

30.1245	Variable Documentation	3061
30.1245.1	s1959_8c	3061
30.1246	html/s1960_8c.js File Reference	3061
30.1246	Variable Documentation	3062
30.1246.1	s1960_8c	3062
30.1247	html/s1961_8c.js File Reference	3062
30.1247	Variable Documentation	3062
30.1247.1	s1961_8c	3062
30.1248	html/s1962_8c.js File Reference	3062
30.1248	Variable Documentation	3062
30.1248.1	s1962_8c	3062
30.1249	html/s1963_8c.js File Reference	3063
30.1249	Variable Documentation	3063
30.1249.1	s1963_8c	3063
30.1250	html/s1965_8c.js File Reference	3063
30.1250	Variable Documentation	3063
30.1250.1	s1965_8c	3063
30.1251	html/s1966_8c.js File Reference	3063
30.1251	Variable Documentation	3064
30.1251.1	s1966_8c	3064
30.1252	html/s1967_8c.js File Reference	3064
30.1252	Variable Documentation	3064
30.1252.1	s1967_8c	3064
30.1253	html/s1968_8c.js File Reference	3064
30.1253	Variable Documentation	3064
30.1253.1	s1968_8c	3064
30.1254	html/s1986_8c.js File Reference	3065
30.1254	Variable Documentation	3065
30.1254.1	s1986_8c	3065
30.1255	html/s1987_8c.js File Reference	3065

30.1255	Variable Documentation	3065
30.1255.1	s1987_8c	3065
30.1256	c/html/s1988_8c.js File Reference	3065
30.1256	Variable Documentation	3066
30.1256.1	s1988_8c	3066
30.1257	c/html/s1989_8c.js File Reference	3066
30.1257	Variable Documentation	3066
30.1257.1	s1989_8c	3066
30.1258	c/html/s1990_8c.js File Reference	3066
30.1258	Variable Documentation	3066
30.1258.1	s1990_8c	3066
30.1259	c/html/s1991_8c.js File Reference	3067
30.1259	Variable Documentation	3067
30.1259.1	s1991_8c	3067
30.1260	c/html/s1992_8c.js File Reference	3067
30.1260	Variable Documentation	3067
30.1260.1	s1992_8c	3067
30.1261	c/html/s1993_8c.js File Reference	3067
30.1261	Variable Documentation	3068
30.1261.1	s1993_8c	3068
30.1262	c/html/s1994_8c.js File Reference	3068
30.1262	Variable Documentation	3068
30.1262.1	s1994_8c	3068
30.1263	c/html/s2500_8c.js File Reference	3068
30.1263	Variable Documentation	3068
30.1263.1	s2500_8c	3068
30.1264	c/html/s2501_8c.js File Reference	3069
30.1264	Variable Documentation	3069
30.1264.1	s2501_8c	3069
30.1265	c/html/s2502_8c.js File Reference	3069

30.1265	Variable Documentation	3069
30.1265.1	s 2502_8c	3069
30.1266	c/html/s 2503_8c.js File Reference	3069
30.1266	Variable Documentation	3070
30.1266.1	s 2503_8c	3070
30.1267	c/html/s 2504_8c.js File Reference	3070
30.1267	Variable Documentation	3070
30.1267.1	s 2504_8c	3070
30.1268	c/html/s 2505_8c.js File Reference	3070
30.1268	Variable Documentation	3070
30.1268.1	s 2505_8c	3070
30.1269	c/html/s 2506_8c.js File Reference	3071
30.1269	Variable Documentation	3071
30.1269.1	s 2506_8c	3071
30.1270	c/html/s 2507_8c.js File Reference	3071
30.1270	Variable Documentation	3071
30.1270.1	s 2507_8c	3071
30.1271	c/html/s 2508_8c.js File Reference	3071
30.1271	Variable Documentation	3072
30.1271.1	s 2508_8c	3072
30.1272	c/html/s 2509_8c.js File Reference	3072
30.1272	Variable Documentation	3072
30.1272.1	s 2509_8c	3072
30.1273	c/html/s 2510_8c.js File Reference	3072
30.1273	Variable Documentation	3072
30.1273.1	s 2510_8c	3072
30.1274	c/html/s 2511_8c.js File Reference	3073
30.1274	Variable Documentation	3073
30.1274.1	s 2511_8c	3073
30.1275	c/html/s 2512_8c.js File Reference	3073

30.1275	Variable Documentation	3073
30.1275.1	s2512_8c	3073
30.1276	html/s2513_8c.js File Reference	3073
30.1276	Variable Documentation	3074
30.1276.1	s2513_8c	3074
30.1277	html/s2514_8c.js File Reference	3074
30.1277	Variable Documentation	3074
30.1277.1	s2514_8c	3074
30.1278	html/s2515_8c.js File Reference	3074
30.1278	Variable Documentation	3074
30.1278.1	s2515_8c	3074
30.1279	html/s2516_8c.js File Reference	3075
30.1279	Variable Documentation	3075
30.1279.1	s2516_8c	3075
30.1280	html/s2532_8c.js File Reference	3075
30.1280	Variable Documentation	3075
30.1280.1	s2532_8c	3075
30.1281	html/s2533_8c.js File Reference	3075
30.1281	Variable Documentation	3076
30.1281.1	s2533_8c	3076
30.1282	html/s2534_8c.js File Reference	3076
30.1282	Variable Documentation	3076
30.1282.1	s2534_8c	3076
30.1283	html/s2535_8c.js File Reference	3076
30.1283	Variable Documentation	3076
30.1283.1	s2535_8c	3076
30.1284	html/s2536_8c.js File Reference	3077
30.1284	Variable Documentation	3077
30.1284.1	s2536_8c	3077
30.1285	html/s2540_8c.js File Reference	3077

30.1285	Variable Documentation	3077
30.1285.1	s2540_8c	3077
30.1286	c/html/s2541_8c.js File Reference	3077
30.1286	Variable Documentation	3078
30.1286.1	s2541_8c	3078
30.1287	c/html/s2542_8c.js File Reference	3078
30.1287	Variable Documentation	3078
30.1287.1	s2542_8c	3078
30.1288	c/html/s2543_8c.js File Reference	3078
30.1288	Variable Documentation	3078
30.1288.1	s2543_8c	3078
30.1289	c/html/s2544_8c.js File Reference	3079
30.1289	Variable Documentation	3079
30.1289.1	s2544_8c	3079
30.1290	c/html/s2545_8c.js File Reference	3079
30.1290	Variable Documentation	3079
30.1290.1	s2545_8c	3079
30.1291	c/html/s2550_8c.js File Reference	3079
30.1291	Variable Documentation	3080
30.1291.1	s2550_8c	3080
30.1292	c/html/s2551_8c.js File Reference	3080
30.1292	Variable Documentation	3080
30.1292.1	s2551_8c	3080
30.1293	c/html/s2553_8c.js File Reference	3080
30.1293	Variable Documentation	3080
30.1293.1	s2553_8c	3080
30.1294	c/html/s2554_8c.js File Reference	3081
30.1294	Variable Documentation	3081
30.1294.1	s2554_8c	3081
30.1295	c/html/s2555_8c.js File Reference	3081

30.1295	Variable Documentation	3081
30.1295.1	s2555_8c	3081
30.1296	c/html/s2556_8c.js File Reference	3081
30.1296	Variable Documentation	3082
30.1296.1	s2556_8c	3082
30.1297	c/html/s2557_8c.js File Reference	3082
30.1297	Variable Documentation	3082
30.1297.1	s2557_8c	3082
30.1298	c/html/s2558_8c.js File Reference	3082
30.1298	Variable Documentation	3082
30.1298.1	s2558_8c	3082
30.1299	c/html/s2559_8c.js File Reference	3083
30.1299	Variable Documentation	3083
30.1299.1	s2559_8c	3083
30.1300	c/html/s2560_8c.js File Reference	3083
30.1300	Variable Documentation	3083
30.1300.1	s2560_8c	3083
30.1301	c/html/s2561_8c.js File Reference	3083
30.1301	Variable Documentation	3084
30.1301.1	s2561_8c	3084
30.1302	c/html/s2562_8c.js File Reference	3084
30.1302	Variable Documentation	3084
30.1302.1	s2562_8c	3084
30.1303	c/html/s6addcurve_8c.js File Reference	3084
30.1303	Variable Documentation	3084
30.1303.1	s6addcurve_8c	3084
30.1304	c/html/s6affdist_8c.js File Reference	3085
30.1304	Variable Documentation	3085
30.1304.1	s6affdist_8c	3085
30.1305	c/html/s6ang_8c.js File Reference	3085

30.1305	Variable Documentation	3085
30.1305.1	s6ang_8c	3085
30.1306	html/s6angle_8c.js File Reference	3085
30.1306	Variable Documentation	3086
30.1306.1	s6angle_8c	3086
30.1307	html/s6castelja_8c.js File Reference	3086
30.1307	Variable Documentation	3086
30.1307.1	s6castelja_8c	3086
30.1308	html/s6chpar_8c.js File Reference	3086
30.1308	Variable Documentation	3086
30.1308.1	s6chpar_8c	3086
30.1309	html/s6crss_8c.js File Reference	3087
30.1309	Variable Documentation	3087
30.1309.1	s6crss_8c	3087
30.1310	html/s6crvature_8c.js File Reference	3087
30.1310	Variable Documentation	3087
30.1310.1	s6crvature_8c	3087
30.1311	html/s6crvcheck_8c.js File Reference	3087
30.1311	Variable Documentation	3088
30.1311.1	s6crvcheck_8c	3088
30.1312	html/s6curvrad_8c.js File Reference	3088
30.1312	Variable Documentation	3088
30.1312.1	s6curvrad_8c	3088
30.1313	html/s6decomp_8c.js File Reference	3088
30.1313	Variable Documentation	3088
30.1313.1	s6decomp_8c	3088
30.1314	html/s6degnorm_8c.js File Reference	3089
30.1314	Variable Documentation	3089
30.1314.1	s6degnorm_8c	3089
30.1315	html/s6dertopt_8c.js File Reference	3089

30.1315	Variable Documentation	3089
30.1315.1	<code>s6derto_8c</code>	3089
30.1316	<code>html/s6diff_8c.js</code> File Reference	3089
30.1316	Variable Documentation	3090
30.1316.1	<code>s6diff_8c</code>	3090
30.1317	<code>html/s6dist_8c.js</code> File Reference	3090
30.1317	Variable Documentation	3090
30.1317.1	<code>s6dist_8c</code>	3090
30.1318	<code>html/s6dline_8c.js</code> File Reference	3090
30.1318	Variable Documentation	3090
30.1318.1	<code>s6dline_8c</code>	3090
30.1319	<code>html/s6dplane_8c.js</code> File Reference	3091
30.1319	Variable Documentation	3091
30.1319.1	<code>s6dplane_8c</code>	3091
30.1320	<code>html/s6drawseq_8c.js</code> File Reference	3091
30.1320	Variable Documentation	3091
30.1320.1	<code>s6drawseq_8c</code>	3091
30.1321	<code>html/s6equal_8c.js</code> File Reference	3091
30.1321	Variable Documentation	3092
30.1321.1	<code>s6equal_8c</code>	3092
30.1322	<code>html/s6err_8c.js</code> File Reference	3092
30.1322	Variable Documentation	3092
30.1322.1	<code>s6err_8c</code>	3092
30.1323	<code>html/s6existbox_8c.js</code> File Reference	3092
30.1323	Variable Documentation	3092
30.1323.1	<code>s6existbox_8c</code>	3092
30.1324	<code>html/s6findfac_8c.js</code> File Reference	3093
30.1324	Variable Documentation	3093
30.1324.1	<code>s6findfac_8c</code>	3093
30.1325	<code>html/s6fndintv_8c.js</code> File Reference	3093

30.1325	Variable Documentation	3093
30.1325.1	s6fndintv_8c	3093
30.1326	../html/s6herm_8c.js File Reference	3093
30.1326	Variable Documentation	3094
30.1326.1	s6herm_8c	3094
30.1327	../html/s6herm__bez_8c.js File Reference	3094
30.1327	Variable Documentation	3094
30.1327.1	s6herm__bez_8c	3094
30.1328	../html/s6idcon_8c.js File Reference	3094
30.1328	Variable Documentation	3094
30.1328.1	s6idcon_8c	3094
30.1329	../html/s6idcpt_8c.js File Reference	3095
30.1329	Variable Documentation	3095
30.1329.1	s6idcpt_8c	3095
30.1330	../html/s6idedg_8c.js File Reference	3095
30.1330	Variable Documentation	3095
30.1330.1	s6idedg_8c	3095
30.1331	../html/s6identify_8c.js File Reference	3095
30.1331	Variable Documentation	3096
30.1331.1	s6identify_8c	3096
30.1332	../html/s6idget_8c.js File Reference	3096
30.1332	Variable Documentation	3096
30.1332.1	s6idget_8c	3096
30.1333	../html/s6idint_8c.js File Reference	3096
30.1333	Variable Documentation	3096
30.1333.1	s6idint_8c	3096
30.1334	../html/s6idklist_8c.js File Reference	3097
30.1334	Variable Documentation	3097
30.1334.1	s6idklist_8c	3097
30.1335	../html/s6idkpt_8c.js File Reference	3097

30.1335	Variable Documentation	3097
30.1335.1	s6idkpt_8c	3097
30.1336	c/html/s6idlis_8c.js File Reference	3097
30.1336	Variable Documentation	3098
30.1336.1	s6idlis_8c	3098
30.1337	c/html/s6idnpt_8c.js File Reference	3098
30.1337	Variable Documentation	3098
30.1337.1	s6idnpt_8c	3098
30.1338	c/html/s6idput_8c.js File Reference	3098
30.1338	Variable Documentation	3098
30.1338.1	s6idput_8c	3098
30.1339	c/html/s6inv4_8c.js File Reference	3099
30.1339	Variable Documentation	3099
30.1339.1	s6inv4_8c	3099
30.1340	c/html/s6invert_8c.js File Reference	3099
30.1340	Variable Documentation	3099
30.1340.1	s6invert_8c	3099
30.1341	c/html/s6knotmult_8c.js File Reference	3099
30.1341	Variable Documentation	3100
30.1341.1	s6knotmult_8c	3100
30.1342	c/html/s6length_8c.js File Reference	3100
30.1342	Variable Documentation	3100
30.1342.1	s6length_8c	3100
30.1343	c/html/s6line_8c.js File Reference	3100
30.1343	Variable Documentation	3100
30.1343.1	s6line_8c	3100
30.1344	c/html/s6lprj_8c.js File Reference	3101
30.1344	Variable Documentation	3101
30.1344.1	s6lprj_8c	3101
30.1345	c/html/s6lufacp_8c.js File Reference	3101

30.1345	Variable Documentation	3101
30.1345.1	s6lufacp_8c	3101
30.1346	/html/s6lusolp_8c.js File Reference	3101
30.1346	Variable Documentation	3102
30.1346.1	s6lusolp_8c	3102
30.1347	/html/s6metric_8c.js File Reference	3102
30.1347	Variable Documentation	3102
30.1347.1	s6metric_8c	3102
30.1348	/html/s6move_8c.js File Reference	3102
30.1348	Variable Documentation	3102
30.1348.1	s6move_8c	3102
30.1349	/html/s6mulvec_8c.js File Reference	3103
30.1349	Variable Documentation	3103
30.1349.1	s6mulvec_8c	3103
30.1350	/html/s6mvec_8c.js File Reference	3103
30.1350	Variable Documentation	3103
30.1350.1	s6mvec_8c	3103
30.1351	/html/s6newbox_8c.js File Reference	3103
30.1351	Variable Documentation	3104
30.1351.1	s6newbox_8c	3104
30.1352	/html/s6norm_8c.js File Reference	3104
30.1352	Variable Documentation	3104
30.1352.1	s6norm_8c	3104
30.1353	/html/s6ratder_8c.js File Reference	3104
30.1353	Variable Documentation	3104
30.1353.1	s6ratder_8c	3104
30.1354	/html/s6rotax_8c.js File Reference	3105
30.1354	Variable Documentation	3105
30.1354.1	s6rotax_8c	3105
30.1355	/html/s6rotmat_8c.js File Reference	3105

30.1355	Variable Documentation	3105
30.1355.1	s6rotmat_8c	3105
30.1356	html/s6schoen_8c.js File Reference	3105
30.1356	Variable Documentation	3106
30.1356.1	s6schoen_8c	3106
30.1357	html/s6scpr_8c.js File Reference	3106
30.1357	Variable Documentation	3106
30.1357.1	s6scpr_8c	3106
30.1358	html/s6sortpar_8c.js File Reference	3106
30.1358	Variable Documentation	3106
30.1358.1	s6sortpar_8c	3106
30.1359	html/s6sratder_8c.js File Reference	3107
30.1359	Variable Documentation	3107
30.1359.1	s6sratder_8c	3107
30.1360	html/s6strider_8c.js File Reference	3107
30.1360	Variable Documentation	3107
30.1360.1	s6strider_8c	3107
30.1361	html/s6takunion_8c.js File Reference	3107
30.1361	Variable Documentation	3108
30.1361.1	s6takunion_8c	3108
30.1362	html/s6twonorm_8c.js File Reference	3108
30.1362	Variable Documentation	3108
30.1362.1	s6twonorm_8c	3108
30.1363	html/s9adsimp_8c.js File Reference	3108
30.1363	Variable Documentation	3108
30.1363.1	s9adsimp_8c	3108
30.1364	html/s9adstep_8c.js File Reference	3109
30.1364	Variable Documentation	3109
30.1364.1	s9adstep_8c	3109
30.1365	html/s9boundimp_8c.js File Reference	3109

30.1365	Variable Documentation	3109
30.1365.1	s9boundimp_8c	3109
30.1366	66c/html/s9boundit_8c.js File Reference	3109
30.1366	Variable Documentation	3110
30.1366.1	s9boundit_8c	3110
30.1367	67c/html/s9clipimp_8c.js File Reference	3110
30.1367	Variable Documentation	3110
30.1367.1	s9clipimp_8c	3110
30.1368	68c/html/s9clipit_8c.js File Reference	3110
30.1368	Variable Documentation	3110
30.1368.1	s9clipit_8c	3110
30.1369	69c/html/s9conmarch_8c.js File Reference	3111
30.1369	Variable Documentation	3111
30.1369.1	s9conmarch_8c	3111
30.1370	70c/html/s9iterate_8c.js File Reference	3111
30.1370	Variable Documentation	3111
30.1370.1	s9iterate_8c	3111
30.1371	71c/html/s9iterimp_8c.js File Reference	3111
30.1371	Variable Documentation	3112
30.1371.1	s9iterimp_8c	3112
30.1372	72c/html/s9smplknot_8c.js File Reference	3112
30.1372	Variable Documentation	3112
30.1372.1	s9smplknot_8c	3112
30.1373	73c/html/s9surmarch_8c.js File Reference	3112
30.1373	Variable Documentation	3112
30.1373.1	s9surmarch_8c	3112
30.1374	74c/html/search/all_0.js File Reference	3113
30.1374	Variable Documentation	3113
30.1374.1	searchData	3113
30.1375	75c/html/search/all_1.js File Reference	3113

30.1375	Variable Documentation	3113
30.1375.1	searchData	3113
30.1376	../html/search/all_10.js File Reference	3114
30.1376	Variable Documentation	3114
30.1376.1	const	3114
30.1376.1	l2	3116
30.1376.1	i3	3116
30.1376.1	idx	3116
30.1376.1	sum	3116
30.1376.1	searchData	3116
30.1377	../html/search/all_11.js File Reference	3116
30.1377	Variable Documentation	3116
30.1377.1	j\$	3116
30.1377.1	searchData	3117
30.1378	../html/search/all_12.js File Reference	3117
30.1378	Variable Documentation	3117
30.1378.1	dpp	3117
30.1378.1	searchData	3117
30.1379	../html/search/all_13.js File Reference	3117
30.1379	Variable Documentation	3118
30.1379.1	h1	3118
30.1379.1	searchData	3118
30.1380	../html/search/all_14.js File Reference	3118
30.1380	Variable Documentation	3118
30.1380.1	d	3118
30.1380.1	searchData	3119
30.1381	../html/search/all_15.js File Reference	3119
30.1381	Variable Documentation	3119
30.1381.1	dmp	3119
30.1381.1	l2	3119

30.1381.1	const	3119
30.1381.1	gt	3121
30.1381.1	l5	3121
30.1381.1	l6	3121
30.1381.1	searchData	3122
30.1382	../html/search/all_16.js File Reference	3123
30.1382	Variable Documentation	3123
30.1382.1	searchData	3123
30.1383	../html/search/all_17.js File Reference	3123
30.1383	Variable Documentation	3123
30.1383.1	gt	3123
30.1383.1	searchData	3124
30.1384	../html/search/all_18.js File Reference	3124
30.1384	Variable Documentation	3124
30.1384.1	dpp	3124
30.1384.1	searchData	3124
30.1385	../html/search/all_19.js File Reference	3125
30.1385	Variable Documentation	3125
30.1385.1	dpp	3125
30.1385.1	searchData	3125
30.1386	../html/search/all_1a.js File Reference	3125
30.1386	Variable Documentation	3125
30.1386.1	dpp	3125
30.1386.1	searchData	3126
30.1387	../html/search/all_1b.js File Reference	3126
30.1387	Variable Documentation	3126
30.1387.1	js	3126
30.1387.1	searchData	3126
30.1388	../html/search/all_1c.js File Reference	3127
30.1388	Variable Documentation	3127

30.1388.1	searchData	3127
30.1388	doc/html/search/all_2.js File Reference	3127
30.1389	Variable Documentation	3127
30.1389.1	depsge	3127
30.1389.1	constdir	3127
30.1389.1	app	3127
30.1389.1	dim	3127
30.1389.1	is	3128
30.1389.1	repar	3128
30.1389.1	searchData	3128
30.1389	doc/html/search/all_3.js File Reference	3129
30.1390	Variable Documentation	3129
30.1390.1	js	3129
30.1390.1	searchData	3129
30.1391	doc/html/search/all_4.js File Reference	3129
30.1391	Variable Documentation	3130
30.1391.1	amp	3130
30.1391.1	basisDerivs_u	3130
30.1391.1	basisDerivs_v	3130
30.1391.1	basisValues	3130
30.1391.1	is	3130
30.1391.1	const	3130
30.1391.1	opos1	3130
30.1391.1	app	3130
30.1391.1	degree	3130
30.1391.1	di0t	3130
30.1391.1	evaluate_from_right	3131
30.1391.1	ops2	3131
30.1391.1	gt3	3131
30.1391.1	init_basis	3131

30.1391.1j5	3131
30.1391.1j6	3131
30.1391.1mb_cvs	3131
30.1391.1p8cv	3131
30.1391.1p9su	3131
30.1391.1searchData	3131
30.1391.1second_order	3132
30.1391.1t2	3132
30.1392c/html/search/all_5.js File Reference	3132
30.1392Variable Documentation	3132
30.1392.1dpp	3132
30.1392.1searchData	3132
30.1393c/html/search/all_6.js File Reference	3133
30.1393Variable Documentation	3133
30.1393.1amp	3133
30.1393.1e	3133
30.1393.1const	3133
30.1393.1gt	3134
30.1393.1j5	3134
30.1393.1j6	3134
30.1393.1rder	3134
30.1393.1searchData	3135
30.1394c/html/search/all_7.js File Reference	3135
30.1394Variable Documentation	3135
30.1394.1gt	3135
30.1394.1searchData	3136
30.1395c/html/search/all_8.js File Reference	3136
30.1395Variable Documentation	3136
30.1395.1amp	3136
30.1395.1dpp	3136

30.1395.1 ~~3~~DoubleBuffer 3137

30.1395.1 ~~4~~ 3137

30.1395.1 ~~5~~ighting 3137

30.1395.1 ~~6~~ormalize 3137

30.1395.1 ~~7~~earchData 3137

30.1395.1 ~~8~~mooth 3137

30.1395.1 ~~9~~exfile 3137

30.1395.1 ~~10~~ature_mode 3137

30.1395.1 ~~11~~o_sided 3137

30.1395.1 ~~12~~ 3138

30.1395.1 ~~13~~ 3138

30.1395.1 ~~14~~ 3138

30.1395.1 ~~15~~ 3138

30.1395.1 ~~16~~cale 3138

30.1395.1 ~~17~~ans 3138

30.1395.1 ~~18~~ 3138

30.1395.1 ~~19~~ 3138

30.1395.1 ~~20~~cale 3138

30.1395.1 ~~21~~ans 3139

30.1395.1 ~~22~~cale 3139

30.1395.1 ~~23~~ans 3139

30.1396 ~~6~~c/html/search/all_9.js File Reference 3139

30.1396 Variable Documentation 3139

30.1396.1 ~~d~~ims 3139

30.1396.1 ~~e~~ 3139

30.1396.1 ~~i~~itpars_v 3139

30.1396.1 ~~l~~ 3140

30.1396.1 ~~m~~ 3140

30.1396.1 ~~n~~ 3140

30.1396.1 ~~o~~ 3140

30.1396.1 ~~r~~ 3140

30.1396.1	param_v	3140
30.1396.1	searchData	3140
30.1396.1	tolerance1	3140
30.1396.1	tolerance2	3140
30.1397	c/html/search/all_a.js File Reference	3141
30.1397	Variable Documentation	3141
30.1397.1	searchData	3141
30.1398	c/html/search/all_b.js File Reference	3141
30.1398	Variable Documentation	3141
30.1398.1	dpp	3141
30.1398.1	searchData	3141
30.1399	c/html/search/all_c.js File Reference	3142
30.1399	Variable Documentation	3142
30.1399.1	dpp	3142
30.1399.1	searchData	3142
30.1400	c/html/search/all_d.js File Reference	3143
30.1400	Variable Documentation	3143
30.1400.1	closest_dist	3143
30.1400.1	dpp	3143
30.1400.1	epsge	3143
30.1400.1	init_mba	3143
30.1400.1	mba_level	3144
30.1400.1	repar	3144
30.1400.1	searchData	3144
30.1401	c/html/search/all_e.js File Reference	3145
30.1401	Variable Documentation	3145
30.1401.1	dpp	3145
30.1401.1	searchData	3145
30.1402	c/html/search/all_f.js File Reference	3146
30.1402	Variable Documentation	3146

30.1402.1H	3146
30.1402.1SearchData	3146
30.1403c/html/search/classes_0.js File Reference	3146
30.1403Variable Documentation	3146
30.1403.1double	3146
30.1403.1gt	3146
30.1403.1SearchData	3146
30.1404c/html/search/classes_1.js File Reference	3147
30.1404Variable Documentation	3147
30.1404.1gt	3147
30.1404.1SearchData	3147
30.1405c/html/search/classes_10.js File Reference	3147
30.1405Variable Documentation	3147
30.1405.1SearchData	3147
30.1406c/html/search/classes_11.js File Reference	3148
30.1406Variable Documentation	3148
30.1406.1SearchData	3148
30.1407c/html/search/classes_12.js File Reference	3148
30.1407Variable Documentation	3148
30.1407.1SearchData	3148
30.1408c/html/search/classes_13.js File Reference	3148
30.1408Variable Documentation	3148
30.1408.1SearchData	3148
30.1409c/html/search/classes_14.js File Reference	3149
30.1409Variable Documentation	3149
30.1409.1SearchData	3149
30.1410c/html/search/classes_15.js File Reference	3149
30.1410Variable Documentation	3149
30.1410.1SearchData	3149
30.1411c/html/search/classes_16.js File Reference	3150

30.1411	Variable Documentation	3150
30.1411.1	searchData	3150
30.1412	doc/html/search/classes_2.js File Reference	3150
30.1412	Variable Documentation	3150
30.1412.1	searchData	3150
30.1413	doc/html/search/classes_3.js File Reference	3150
30.1413	Variable Documentation	3151
30.1413.1	searchData	3151
30.1414	doc/html/search/classes_4.js File Reference	3151
30.1414	Variable Documentation	3151
30.1414.1	searchData	3151
30.1415	doc/html/search/classes_5.js File Reference	3152
30.1415	Variable Documentation	3152
30.1415.1	gt	3152
30.1415.1	searchData	3152
30.1416	doc/html/search/classes_6.js File Reference	3152
30.1416	Variable Documentation	3152
30.1416.1	gt	3152
30.1416.1	searchData	3153
30.1417	doc/html/search/classes_7.js File Reference	3153
30.1417	Variable Documentation	3153
30.1417.1	gt	3153
30.1417.1	searchData	3153
30.1418	doc/html/search/classes_8.js File Reference	3153
30.1418	Variable Documentation	3154
30.1418.1	searchData	3154
30.1419	doc/html/search/classes_9.js File Reference	3154
30.1419	Variable Documentation	3154
30.1419.1	searchData	3154
30.1420	doc/html/search/classes_a.js File Reference	3154

30.1420	Variable Documentation	3154
30.1420.1	searchData	3154
30.1420	c/html/search/classes_b.js File Reference	3155
30.1421	Variable Documentation	3155
30.1421.1	searchData	3155
30.1421	c/html/search/classes_c.js File Reference	3155
30.1422	Variable Documentation	3155
30.1422.1	double	3155
30.1422.1	g	3156
30.1422.1	searchData	3156
30.1422	c/html/search/classes_d.js File Reference	3156
30.1423	Variable Documentation	3156
30.1423.1	searchData	3156
30.1423	c/html/search/classes_e.js File Reference	3157
30.1424	Variable Documentation	3157
30.1424.1	searchData	3157
30.1424	c/html/search/classes_f.js File Reference	3157
30.1425	Variable Documentation	3157
30.1425.1	gt	3157
30.1425.1	searchData	3158
30.1425	c/html/search/defines_0.js File Reference	3158
30.1426	Variable Documentation	3158
30.1426.1	searchData	3158
30.1426	c/html/search/defines_1.js File Reference	3159
30.1427	Variable Documentation	3159
30.1427.1	h	3159
30.1427.1	searchData	3159
30.1427	c/html/search/defines_10.js File Reference	3159
30.1428	Variable Documentation	3159
30.1428.1	searchData	3159

30.1428	c/html/search/defines_11.js File Reference	3160
30.1429	Variable Documentation	3160
30.1429.1	dpp	3160
30.1429.1	searchData	3160
30.1430	c/html/search/defines_2.js File Reference	3160
30.1430	Variable Documentation	3160
30.1430.1	searchData	3160
30.1431	c/html/search/defines_3.js File Reference	3161
30.1431	Variable Documentation	3161
30.1431.1	d	3161
30.1431.1	searchData	3161
30.1432	c/html/search/defines_4.js File Reference	3161
30.1432	Variable Documentation	3161
30.1432.1	d	3161
30.1432.1	searchData	3162
30.1433	c/html/search/defines_5.js File Reference	3162
30.1433	Variable Documentation	3162
30.1433.1	searchData	3162
30.1434	c/html/search/defines_6.js File Reference	3162
30.1434	Variable Documentation	3163
30.1434.1	searchData	3163
30.1435	c/html/search/defines_7.js File Reference	3163
30.1435	Variable Documentation	3163
30.1435.1	H	3163
30.1435.1	searchData	3163
30.1436	c/html/search/defines_8.js File Reference	3163
30.1436	Variable Documentation	3164
30.1436.1	searchData	3164
30.1437	c/html/search/defines_9.js File Reference	3164
30.1437	Variable Documentation	3164

30.1437.1d	3164
30.1437.1searchData	3164
30.1438c/html/search/defines_a.js File Reference	3164
30.1438Variable Documentation	3165
30.1438.1d	3165
30.1438.1searchData	3165
30.1439c/html/search/defines_b.js File Reference	3165
30.1439Variable Documentation	3165
30.1439.1searchData	3165
30.1440c/html/search/defines_c.js File Reference	3166
30.1440Variable Documentation	3166
30.1440.1H	3166
30.1440.1searchData	3166
30.1441c/html/search/defines_d.js File Reference	3166
30.1441Variable Documentation	3166
30.1441.1H	3166
30.1441.1searchData	3166
30.1442c/html/search/defines_e.js File Reference	3167
30.1442Variable Documentation	3167
30.1442.1d	3167
30.1442.1searchData	3167
30.1443c/html/search/defines_f.js File Reference	3167
30.1443Variable Documentation	3167
30.1443.1H	3167
30.1443.1searchData	3167
30.1444c/html/search/enums_0.js File Reference	3168
30.1444Variable Documentation	3168
30.1444.1searchData	3168
30.1445c/html/search/enums_1.js File Reference	3168
30.1445Variable Documentation	3168

30.1445.1 searchData	3168
30.1446c/html/search/enums_2.js File Reference	3168
30.1446Variable Documentation	3169
30.1446.1 searchData	3169
30.1447c/html/search/enums_3.js File Reference	3169
30.1447Variable Documentation	3169
30.1447.1 searchData	3169
30.1448c/html/search/enums_4.js File Reference	3169
30.1448Variable Documentation	3170
30.1448.1 searchData	3170
30.1449c/html/search/enums_5.js File Reference	3170
30.1449Variable Documentation	3170
30.1449.1 searchData	3170
30.1450c/html/search/enums_6.js File Reference	3170
30.1450Variable Documentation	3171
30.1450.1 searchData	3171
30.1451c/html/search/enums_7.js File Reference	3171
30.1451Variable Documentation	3171
30.1451.1 searchData	3171
30.1452c/html/search/enums_8.js File Reference	3171
30.1452Variable Documentation	3172
30.1452.1 searchData	3172
30.1453c/html/search/enums_9.js File Reference	3172
30.1453Variable Documentation	3172
30.1453.1 searchData	3172
30.1454c/html/search/enums_a.js File Reference	3172
30.1454Variable Documentation	3173
30.1454.1 searchData	3173
30.1455c/html/search/enums_b.js File Reference	3173
30.1455Variable Documentation	3173

30.1455.1	searchData	3173
30.1456c/html/search/enums_c.js	File Reference	3173
30.1456	Variable Documentation	3174
30.1456.1	searchData	3174
30.1457c/html/search/enums_d.js	File Reference	3174
30.1457	Variable Documentation	3174
30.1457.1	searchData	3174
30.1458c/html/search/enums_e.js	File Reference	3174
30.1458	Variable Documentation	3175
30.1458.1	searchData	3175
30.1459c/html/search/enums_f.js	File Reference	3175
30.1459	Variable Documentation	3175
30.1459.1	searchData	3175
30.1460c/html/search/enumvalues_0.js	File Reference	3175
30.1460	Variable Documentation	3175
30.1460.1	searchData	3175
30.1461c/html/search/enumvalues_1.js	File Reference	3176
30.1461	Variable Documentation	3176
30.1461.1	searchData	3176
30.1462c/html/search/enumvalues_10.js	File Reference	3176
30.1462	Variable Documentation	3176
30.1462.1	searchData	3176
30.1463c/html/search/enumvalues_11.js	File Reference	3176
30.1463	Variable Documentation	3177
30.1463.1	searchData	3177
30.1464c/html/search/enumvalues_12.js	File Reference	3177
30.1464	Variable Documentation	3177
30.1464.1	searchData	3177
30.1465c/html/search/enumvalues_13.js	File Reference	3177
30.1465	Variable Documentation	3177

30.1465.1 searchData	3177
30.1466c/html/search/enumvalues_14.js File Reference	3178
30.1466Variable Documentation	3178
30.1466.1 gt	3178
30.1466.1 searchData	3178
30.1467c/html/search/enumvalues_15.js File Reference	3178
30.1467Variable Documentation	3178
30.1467.1 searchData	3178
30.1468c/html/search/enumvalues_16.js File Reference	3179
30.1468Variable Documentation	3179
30.1468.1 searchData	3179
30.1469c/html/search/enumvalues_17.js File Reference	3179
30.1469Variable Documentation	3179
30.1469.1 searchData	3179
30.1470c/html/search/enumvalues_18.js File Reference	3179
30.1470Variable Documentation	3180
30.1470.1 searchData	3180
30.1471c/html/search/enumvalues_2.js File Reference	3180
30.1471Variable Documentation	3180
30.1471.1 searchData	3180
30.1472c/html/search/enumvalues_3.js File Reference	3180
30.1472Variable Documentation	3181
30.1472.1 searchData	3181
30.1473c/html/search/enumvalues_4.js File Reference	3181
30.1473Variable Documentation	3182
30.1473.1 searchData	3182
30.1474c/html/search/enumvalues_5.js File Reference	3182
30.1474Variable Documentation	3182
30.1474.1 searchData	3182
30.1475c/html/search/enumvalues_6.js File Reference	3182

30.1475	Variable Documentation	3183
30.1475.1	searchData	3183
30.1476	c/html/search/enumvalues_7.js File Reference	3183
30.1476	Variable Documentation	3183
30.1476.1	searchData	3183
30.1477	c/html/search/enumvalues_8.js File Reference	3183
30.1477	Variable Documentation	3184
30.1477.1	searchData	3184
30.1478	c/html/search/enumvalues_9.js File Reference	3184
30.1478	Variable Documentation	3184
30.1478.1	searchData	3184
30.1479	c/html/search/enumvalues_a.js File Reference	3185
30.1479	Variable Documentation	3185
30.1479.1	searchData	3185
30.1480	c/html/search/enumvalues_b.js File Reference	3185
30.1480	Variable Documentation	3185
30.1480.1	searchData	3185
30.1481	c/html/search/enumvalues_c.js File Reference	3186
30.1481	Variable Documentation	3186
30.1481.1	searchData	3186
30.1482	c/html/search/enumvalues_d.js File Reference	3187
30.1482	Variable Documentation	3187
30.1482.1	searchData	3187
30.1483	c/html/search/enumvalues_e.js File Reference	3187
30.1483	Variable Documentation	3188
30.1483.1	searchData	3188
30.1484	c/html/search/enumvalues_f.js File Reference	3188
30.1484	Variable Documentation	3188
30.1484.1	searchData	3188
30.1485	c/html/search/files_0.js File Reference	3189

30.1485	Variable Documentation	3189
30.1485.1	searchData	3189
30.1486	c/html/search/files_1.js File Reference	3189
30.1486	Variable Documentation	3189
30.1486.1	searchData	3189
30.1487	c/html/search/files_10.js File Reference	3190
30.1487	Variable Documentation	3190
30.1487.1	searchData	3190
30.1488	c/html/search/files_11.js File Reference	3190
30.1488	Variable Documentation	3190
30.1488.1	searchData	3190
30.1489	c/html/search/files_12.js File Reference	3191
30.1489	Variable Documentation	3191
30.1489.1	searchData	3191
30.1490	c/html/search/files_13.js File Reference	3191
30.1490	Variable Documentation	3191
30.1490.1	searchData	3191
30.1491	c/html/search/files_14.js File Reference	3191
30.1491	Variable Documentation	3191
30.1491.1	searchData	3191
30.1492	c/html/search/files_15.js File Reference	3192
30.1492	Variable Documentation	3192
30.1492.1	searchData	3192
30.1493	c/html/search/files_2.js File Reference	3192
30.1493	Variable Documentation	3192
30.1493.1	searchData	3192
30.1494	c/html/search/files_3.js File Reference	3193
30.1494	Variable Documentation	3193
30.1494.1	searchData	3193
30.1495	c/html/search/files_4.js File Reference	3193

30.1495	Variable Documentation	3194
30.1495.1	searchData	3194
30.1496	c/html/search/files_5.js File Reference	3194
30.1496	Variable Documentation	3194
30.1496.1	searchData	3194
30.1497	c/html/search/files_6.js File Reference	3194
30.1497	Variable Documentation	3194
30.1497.1	searchData	3194
30.1498	c/html/search/files_7.js File Reference	3195
30.1498	Variable Documentation	3195
30.1498.1	searchData	3195
30.1499	c/html/search/files_8.js File Reference	3195
30.1499	Variable Documentation	3195
30.1499.1	searchData	3195
30.1500	c/html/search/files_9.js File Reference	3195
30.1500	Variable Documentation	3196
30.1500.1	searchData	3196
30.1501	c/html/search/files_a.js File Reference	3196
30.1501	Variable Documentation	3196
30.1501.1	searchData	3196
30.1502	c/html/search/files_b.js File Reference	3196
30.1502	Variable Documentation	3196
30.1502.1	searchData	3196
30.1503	c/html/search/files_c.js File Reference	3196
30.1503	Variable Documentation	3197
30.1503.1	searchData	3197
30.1504	c/html/search/files_d.js File Reference	3197
30.1504	Variable Documentation	3197
30.1504.1	searchData	3197
30.1505	c/html/search/files_e.js File Reference	3198

30.1505	Variable Documentation	3198
30.1505.1	searchData	3198
30.1506	c/html/search/files_f.js File Reference	3198
30.1506	Variable Documentation	3198
30.1506.1	searchData	3198
30.1507	c/html/search/functions_0.js File Reference	3198
30.1507	Variable Documentation	3199
30.1507.1	searchData	3199
30.1508	c/html/search/functions_1.js File Reference	3199
30.1508	Variable Documentation	3199
30.1508.1	depsge	3199
30.1508.1	constdir	3199
30.1508.1	dim	3199
30.1508.1	H	3199
30.1508.1	separ	3200
30.1508.1	searchData	3200
30.1509	c/html/search/functions_10.js File Reference	3200
30.1509	Variable Documentation	3200
30.1509.1	H	3200
30.1509.1	searchData	3201
30.1510	c/html/search/functions_11.js File Reference	3201
30.1510	Variable Documentation	3201
30.1510.1	dpp	3201
30.1510.1	searchData	3201
30.1511	c/html/search/functions_12.js File Reference	3201
30.1511	Variable Documentation	3201
30.1511.1	dpp	3201
30.1511.1	searchData	3202
30.1512	c/html/search/functions_13.js File Reference	3203
30.1512	Variable Documentation	3203

30.1512.1H	3203
30.1512.1SearchData	3203
30.1513c/html/search/functions_14.js File Reference	3204
30.1513Variable Documentation	3204
30.1513.1amp	3204
30.1513.1const	3204
30.1513.1of	3206
30.1513.1H	3206
30.1513.1J5	3206
30.1513.1SearchData	3206
30.1514c/html/search/functions_15.js File Reference	3207
30.1514Variable Documentation	3207
30.1514.1SearchData	3207
30.1515c/html/search/functions_16.js File Reference	3207
30.1515Variable Documentation	3207
30.1515.1SearchData	3207
30.1516c/html/search/functions_17.js File Reference	3207
30.1516Variable Documentation	3207
30.1516.1dpp	3207
30.1516.1SearchData	3208
30.1517c/html/search/functions_18.js File Reference	3209
30.1517Variable Documentation	3209
30.1517.1SearchData	3209
30.1518c/html/search/functions_19.js File Reference	3209
30.1518Variable Documentation	3210
30.1518.1SearchData	3210
30.1519c/html/search/functions_1a.js File Reference	3210
30.1519Variable Documentation	3210
30.1519.1SearchData	3210
30.1520c/html/search/functions_1b.js File Reference	3211

30.1520	Variable Documentation	3211
30.1520.1	searchData	3211
30.1520	c/html/search/functions_2.js File Reference	3211
30.1521	Variable Documentation	3211
30.1521.1	js	3211
30.1521.1	searchData	3211
30.1521	c/html/search/functions_3.js File Reference	3212
30.1522	Variable Documentation	3212
30.1522.1	amp	3212
30.1522.1	basisDerivs_u	3212
30.1522.1	basisDerivs_v	3212
30.1522.1	basisValues	3212
30.1522.1	const	3212
30.1522.1	cp1	3213
30.1522.1	opp	3213
30.1522.1	degree	3213
30.1522.1	dist	3214
30.1522.1	evaluate_from_right	3214
30.1522.1	gpos2	3214
30.1522.1	gt2	3215
30.1522.1	h3	3215
30.1522.1	init_basis	3215
30.1522.1	h5	3215
30.1522.1	h6b_cvs	3215
30.1522.1	h7_cv	3215
30.1522.1	h8su	3215
30.1522.1	searchData	3215
30.1522.1	second_order	3216
30.1522.1	h1	3217
30.1522	c/html/search/functions_4.js File Reference	3217

30.1523	Variable Documentation	3217
30.1523.1	dpp	3217
30.1523.1	searchData	3217
30.1524	c/html/search/functions_5.js File Reference	3218
30.1524	Variable Documentation	3218
30.1524.1	amp	3218
30.1524.1	const	3218
30.1524.1	q	3219
30.1524.1	k	3219
30.1524.1	l5	3219
30.1524.1	oder	3219
30.1524.1	searchData	3219
30.1525	c/html/search/functions_6.js File Reference	3220
30.1525	Variable Documentation	3220
30.1525.1	dpp	3220
30.1525.1	searchData	3220
30.1526	c/html/search/functions_7.js File Reference	3221
30.1526	Variable Documentation	3222
30.1526.1	amp	3222
30.1526.1	app	3222
30.1526.1	doubleBuffer	3222
30.1526.1	k	3222
30.1526.1	ighting	3222
30.1526.1	ormalize	3222
30.1526.1	searchData	3222
30.1526.1	smooth	3222
30.1526.1	exfile	3222
30.1526.1	ature_mode	3223
30.1526.1	wo_sided	3223
30.1526.1	k2	3223

30.1526.1x3	3223
30.1526.1x4	3223
30.1526.1x5	3223
30.1526.1x6	3223
30.1526.1x7	3223
30.1526.1x8	3223
30.1526.1x9	3223
30.1526.1x0	3224
30.1526.1x1	3224
30.1526.1x2	3224
30.1526.1x3	3224
30.1527c/html/search/functions_8.js File Reference	3224
30.1527Variable Documentation	3224
30.1527.1H	3224
30.1527.1SearchData	3224
30.1528c/html/search/functions_9.js File Reference	3224
30.1528Variable Documentation	3225
30.1528.1j\$	3225
30.1528.1SearchData	3225
30.1529c/html/search/functions_a.js File Reference	3225
30.1529Variable Documentation	3226
30.1529.1dpp	3226
30.1529.1SearchData	3226
30.1530c/html/search/functions_b.js File Reference	3226
30.1530Variable Documentation	3226
30.1530.1dpp	3226
30.1530.1SearchData	3226
30.1531c/html/search/functions_c.js File Reference	3227
30.1531Variable Documentation	3227
30.1531.1closest_dist	3227

30.1531.1	opsge	3228
30.1531.1	i3	3228
30.1531.1	init_mba	3228
30.1531.1	mba_level	3228
30.1531.1	repar	3228
30.1531.1	searchData	3229
30.1531	c/html/search/functions_d.js File Reference	3229
30.1532	Variable Documentation	3229
30.1532.1	dpp	3229
30.1532.1	searchData	3230
30.1532	c/html/search/functions_e.js File Reference	3230
30.1533	Variable Documentation	3230
30.1533.1	H	3230
30.1533.1	searchData	3230
30.1534	c/html/search/functions_f.js File Reference	3231
30.1534	Variable Documentation	3231
30.1534.1	const	3231
30.1534.1	i2	3231
30.1534.1	i3	3231
30.1534.1	idx	3231
30.1534.1	sum	3232
30.1534.1	searchData	3232
30.1535	c/html/search/groups_0.js File Reference	3232
30.1535	Variable Documentation	3232
30.1535.1	searchData	3232
30.1536	c/html/search/namespaces_0.js File Reference	3232
30.1536	Variable Documentation	3232
30.1536.1	searchData	3232
30.1537	c/html/search/namespaces_1.js File Reference	3233
30.1537	Variable Documentation	3233

30.1537.1 searchData	3233
30.1538 c/html/search/namespaces_2.js File Reference	3233
30.1538 Variable Documentation	3233
30.1538.1 searchData	3233
30.1539 c/html/search/namespaces_3.js File Reference	3233
30.1539 Variable Documentation	3234
30.1539.1 searchData	3234
30.1540 c/html/search/namespaces_4.js File Reference	3234
30.1540 Variable Documentation	3234
30.1540.1 searchData	3234
30.1541 c/html/search/namespaces_5.js File Reference	3234
30.1541 Variable Documentation	3234
30.1541.1 searchData	3234
30.1542 c/html/search/pages_0.js File Reference	3235
30.1542 Variable Documentation	3235
30.1542.1 searchData	3235
30.1543 c/html/search/pages_1.js File Reference	3235
30.1543 Variable Documentation	3235
30.1543.1 searchData	3235
30.1544 c/html/search/pages_2.js File Reference	3235
30.1544 Variable Documentation	3236
30.1544.1 searchData	3236
30.1545 c/html/search/pages_3.js File Reference	3236
30.1545 Variable Documentation	3236
30.1545.1 searchData	3236
30.1546 c/html/search/pages_4.js File Reference	3236
30.1546 Variable Documentation	3237
30.1546.1 searchData	3237
30.1547 c/html/search/related_0.js File Reference	3237
30.1547 Variable Documentation	3237

30.1547.1 searchData	3237
30.1548c/html/search/related_1.js File Reference	3237
30.1548Variable Documentation	3238
30.1548.1 searchData	3238
30.1549c/html/search/related_2.js File Reference	3238
30.1549Variable Documentation	3238
30.1549.1 searchData	3238
30.1550c/html/search/related_3.js File Reference	3239
30.1550Variable Documentation	3239
30.1550.1 searchData	3239
30.1551c/html/search/related_4.js File Reference	3240
30.1551Variable Documentation	3240
30.1551.1 searchData	3240
30.1552c/html/search/related_5.js File Reference	3240
30.1552Variable Documentation	3240
30.1552.1 searchData	3240
30.1553c/html/search/related_6.js File Reference	3241
30.1553Variable Documentation	3241
30.1553.1 searchData	3241
30.1554c/html/search/related_7.js File Reference	3241
30.1554Variable Documentation	3242
30.1554.1 searchData	3242
30.1555c/html/search/related_8.js File Reference	3242
30.1555Variable Documentation	3242
30.1555.1 searchData	3242
30.1556c/html/search/related_9.js File Reference	3242
30.1556Variable Documentation	3243
30.1556.1 searchData	3243
30.1557c/html/search/related_a.js File Reference	3243
30.1557Variable Documentation	3243

30.1557.1 <code>searchData</code>	3243
30.1558c/html/search/related_b.js File Reference	3243
30.1558Variable Documentation	3244
30.1558.1 <code>Hit</code>	3244
30.1558.1 <code>searchData</code>	3244
30.1559c/html/search/related_c.js File Reference	3244
30.1559Variable Documentation	3244
30.1559.1 <code>searchData</code>	3244
30.1560c/html/search/related_d.js File Reference	3245
30.1560Variable Documentation	3245
30.1560.1 <code>searchData</code>	3245
30.1561c/html/search/related_e.js File Reference	3245
30.1561Variable Documentation	3246
30.1561.1 <code>searchData</code>	3246
30.1562c/html/search/related_f.js File Reference	3246
30.1562Variable Documentation	3246
30.1562.1 <code>searchData</code>	3246
30.1563c/html/search/search.js File Reference	3246
30.1563Function Documentation	3247
30.1563.1 <code>convertTold(search)</code>	3247
30.1563.1 <code>createResults()</code>	3247
30.1563.1 <code>getXPos(item)</code>	3247
30.1563.1 <code>getYPos(item)</code>	3247
30.1563.1 <code>init_search()</code>	3247
30.1563.1 <code>SearchBox(name, resultsPath, inFrame, label)</code>	3247
30.1563.1 <code>SearchResults(name)</code>	3247
30.1563.1 <code>setClassAttr(elem, attr)</code>	3247
30.1563.1 <code>setKeyActions(elem, action)</code>	3247
30.1564c/html/search/searchdata.js File Reference	3248
30.1564Variable Documentation	3248

30.1564.1 indexSectionLabels	3248
30.1564.1 indexSectionNames	3248
30.1564.1 indexSectionsWithContent	3249
30.1565c/html/search/typedefs_0.js File Reference	3249
30.1565 Variable Documentation	3249
30.1565.1 searchData	3249
30.1566c/html/search/typedefs_1.js File Reference	3249
30.1566 Variable Documentation	3250
30.1566.1 searchData	3250
30.1567c/html/search/typedefs_2.js File Reference	3250
30.1567 Variable Documentation	3250
30.1567.1 searchData	3250
30.1568c/html/search/typedefs_3.js File Reference	3250
30.1568 Variable Documentation	3251
30.1568.1 gt	3251
30.1568.1 searchData	3251
30.1569c/html/search/typedefs_4.js File Reference	3251
30.1569 Variable Documentation	3251
30.1569.1 searchData	3251
30.1570c/html/search/typedefs_5.js File Reference	3251
30.1570 Variable Documentation	3252
30.1570.1 d	3252
30.1570.1 searchData	3252
30.1571c/html/search/typedefs_6.js File Reference	3252
30.1571 Variable Documentation	3252
30.1571.1 searchData	3252
30.1572c/html/search/typedefs_7.js File Reference	3252
30.1572 Variable Documentation	3253
30.1572.1 searchData	3253
30.1573c/html/search/typedefs_8.js File Reference	3253

30.1573	Variable Documentation	3253
30.1573.1	searchData	3253
30.1574	c/html/search/typedefs_9.js File Reference	3253
30.1574	Variable Documentation	3253
30.1574.1	searchData	3253
30.1575	c/html/search/typedefs_a.js File Reference	3254
30.1575	Variable Documentation	3254
30.1575.1	searchData	3254
30.1576	c/html/search/typedefs_b.js File Reference	3254
30.1576	Variable Documentation	3254
30.1576.1	searchData	3254
30.1577	c/html/search/typedefs_c.js File Reference	3254
30.1577	Variable Documentation	3255
30.1577.1	gt	3255
30.1577.1	searchData	3255
30.1578	c/html/search/typedefs_d.js File Reference	3255
30.1578	Variable Documentation	3255
30.1578.1	searchData	3255
30.1579	c/html/search/typedefs_e.js File Reference	3255
30.1579	Variable Documentation	3256
30.1579.1	searchData	3256
30.1580	c/html/search/typedefs_f.js File Reference	3256
30.1580	Variable Documentation	3256
30.1580.1	gt	3256
30.1580.1	searchData	3256
30.1581	c/html/search/variables_0.js File Reference	3257
30.1581	Variable Documentation	3257
30.1581.1	dpp	3257
30.1581.1	searchData	3257
30.1582	c/html/search/variables_1.js File Reference	3257

30.1582	Variable Documentation	3257
30.1582.1	searchData	3257
30.1583	c/html/search/variables_10.js File Reference	3258
30.1583	Variable Documentation	3258
30.1583.1	searchData	3258
30.1584	c/html/search/variables_11.js File Reference	3258
30.1584	Variable Documentation	3258
30.1584.1	searchData	3258
30.1585	c/html/search/variables_12.js File Reference	3259
30.1585	Variable Documentation	3259
30.1585.1	searchData	3259
30.1586	c/html/search/variables_13.js File Reference	3259
30.1586	Variable Documentation	3259
30.1586.1	searchData	3259
30.1587	c/html/search/variables_14.js File Reference	3260
30.1587	Variable Documentation	3260
30.1587.1	searchData	3260
30.1588	c/html/search/variables_15.js File Reference	3260
30.1588	Variable Documentation	3261
30.1588.1	searchData	3261
30.1589	c/html/search/variables_16.js File Reference	3261
30.1589	Variable Documentation	3262
30.1589.1	searchData	3262
30.1590	c/html/search/variables_17.js File Reference	3262
30.1590	Variable Documentation	3262
30.1590.1	searchData	3262
30.1591	c/html/search/variables_18.js File Reference	3262
30.1591	Variable Documentation	3263
30.1591.1	searchData	3263
30.1592	c/html/search/variables_19.js File Reference	3263

30.1592	Variable Documentation	3263
30.1592.1	searchData	3263
30.1593	c/html/search/variables_2.js File Reference	3263
30.1593	Variable Documentation	3263
30.1593.1	searchData	3263
30.1594	c/html/search/variables_3.js File Reference	3264
30.1594	Variable Documentation	3264
30.1594.1	searchData	3264
30.1595	c/html/search/variables_4.js File Reference	3264
30.1595	Variable Documentation	3264
30.1595.1	searchData	3264
30.1596	c/html/search/variables_5.js File Reference	3264
30.1596	Variable Documentation	3265
30.1596.1	searchData	3265
30.1597	c/html/search/variables_6.js File Reference	3265
30.1597	Variable Documentation	3266
30.1597.1	searchData	3266
30.1598	c/html/search/variables_7.js File Reference	3266
30.1598	Variable Documentation	3267
30.1598.1	searchData	3267
30.1599	c/html/search/variables_8.js File Reference	3267
30.1599	Variable Documentation	3267
30.1599.1	searchData	3267
30.1600	c/html/search/variables_9.js File Reference	3267
30.1600	Variable Documentation	3267
30.1600.1	searchData	3267
30.1601	c/html/search/variables_a.js File Reference	3268
30.1601	Variable Documentation	3268
30.1601.1	searchData	3268
30.1602	c/html/search/variables_b.js File Reference	3268

30.1602	Variable Documentation	3268
30.1602.1	dpp	3268
30.1602.1	searchData	3269
30.1603	30.1602c/html/search/variables_c.js File Reference	3269
30.1603	Variable Documentation	3269
30.1603.1	dpp	3269
30.1603.1	searchData	3269
30.1604	30.1603c/html/search/variables_d.js File Reference	3269
30.1604	Variable Documentation	3270
30.1604.1	searchData	3270
30.1605	30.1604c/html/search/variables_e.js File Reference	3270
30.1605	Variable Documentation	3270
30.1605.1	searchData	3270
30.1606	30.1605c/html/search/variables_f.js File Reference	3270
30.1606	Variable Documentation	3271
30.1606.1	dpp	3271
30.1606.1	searchData	3271
30.1607	30.1606c/html/sh1260_8c.js File Reference	3271
30.1607	Variable Documentation	3271
30.1607.1	sh1260_8c	3271
30.1608	30.1607c/html/sh1261_8c.js File Reference	3271
30.1608	Variable Documentation	3271
30.1608.1	sh1261_8c	3271
30.1609	30.1608c/html/sh1262_8c.js File Reference	3272
30.1609	Variable Documentation	3272
30.1609.1	sh1262_8c	3272
30.1610	30.1609c/html/sh1263_8c.js File Reference	3272
30.1610	Variable Documentation	3272
30.1610.1	sh1263_8c	3272
30.1611	30.1610c/html/sh1365_8c.js File Reference	3272

30.1611	Variable Documentation	3273
30.1611.1	sh1365_8c	3273
30.1612	../html/sh1369_8c.js File Reference	3273
30.1612	Variable Documentation	3273
30.1612.1	sh1369_8c	3273
30.1613	../html/sh1371_8c.js File Reference	3273
30.1613	Variable Documentation	3273
30.1613.1	sh1371_8c	3273
30.1614	../html/sh1372_8c.js File Reference	3274
30.1614	Variable Documentation	3274
30.1614.1	sh1372_8c	3274
30.1615	../html/sh1373_8c.js File Reference	3274
30.1615	Variable Documentation	3274
30.1615.1	sh1373_8c	3274
30.1616	../html/sh1374_8c.js File Reference	3274
30.1616	Variable Documentation	3275
30.1616.1	sh1374_8c	3275
30.1617	../html/sh1375_8c.js File Reference	3275
30.1617	Variable Documentation	3275
30.1617.1	sh1375_8c	3275
30.1618	../html/sh1460_8c.js File Reference	3275
30.1618	Variable Documentation	3275
30.1618.1	sh1460_8c	3275
30.1619	../html/sh1461_8c.js File Reference	3276
30.1619	Variable Documentation	3276
30.1619.1	sh1461_8c	3276
30.1620	../html/sh1462_8c.js File Reference	3276
30.1620	Variable Documentation	3276
30.1620.1	sh1462_8c	3276
30.1621	../html/sh1463_8c.js File Reference	3276

30.1621	Variable Documentation	3277
30.1621.1	sh1463_8c	3277
30.1622	../html/sh1464_8c.js File Reference	3277
30.1622	Variable Documentation	3277
30.1622.1	sh1464_8c	3277
30.1623	../html/sh1465_8c.js File Reference	3277
30.1623	Variable Documentation	3278
30.1623.1	sh1465_8c	3278
30.1624	../html/sh1466_8c.js File Reference	3278
30.1624	Variable Documentation	3278
30.1624.1	sh1466_8c	3278
30.1625	../html/sh1467_8c.js File Reference	3278
30.1625	Variable Documentation	3278
30.1625.1	sh1467_8c	3278
30.1626	../html/sh1502_8c.js File Reference	3279
30.1626	Variable Documentation	3279
30.1626.1	sh1502_8c	3279
30.1627	../html/sh1503_8c.js File Reference	3279
30.1627	Variable Documentation	3279
30.1627.1	sh1503_8c	3279
30.1628	../html/sh1510_8c.js File Reference	3279
30.1628	Variable Documentation	3280
30.1628.1	sh1510_8c	3280
30.1629	../html/sh1511_8c.js File Reference	3280
30.1629	Variable Documentation	3280
30.1629.1	sh1511_8c	3280
30.1630	../html/sh1761_8c.js File Reference	3280
30.1630	Variable Documentation	3280
30.1630.1	sh1761_8c	3280
30.1631	../html/sh1762_8c.js File Reference	3281

30.1631	Variable Documentation	3281
30.1631.1	sh1762_8c	3281
30.1632	30c/html/sh1779_8c.js File Reference	3281
30.1632	Variable Documentation	3281
30.1632.1	sh1779_8c	3281
30.1633	30c/html/sh1779__at_8c.js File Reference	3281
30.1633	Variable Documentation	3282
30.1633.1	sh1779__at_8c	3282
30.1634	30c/html/sh1780_8c.js File Reference	3282
30.1634	Variable Documentation	3282
30.1634.1	sh1780_8c	3282
30.1635	30c/html/sh1780__at_8c.js File Reference	3282
30.1635	Variable Documentation	3282
30.1635.1	sh1780__at_8c	3282
30.1636	30c/html/sh1781_8c.js File Reference	3283
30.1636	Variable Documentation	3283
30.1636.1	sh1781_8c	3283
30.1637	30c/html/sh1781__at_8c.js File Reference	3283
30.1637	Variable Documentation	3283
30.1637.1	sh1781__at_8c	3283
30.1638	30c/html/sh1782_8c.js File Reference	3283
30.1638	Variable Documentation	3284
30.1638.1	sh1782_8c	3284
30.1639	30c/html/sh1783_8c.js File Reference	3284
30.1639	Variable Documentation	3284
30.1639.1	sh1783_8c	3284
30.1640	30c/html/sh1784_8c.js File Reference	3284
30.1640	Variable Documentation	3284
30.1640.1	sh1784_8c	3284
30.1641	30c/html/sh1786_8c.js File Reference	3285

30.1641	Variable Documentation	3285
30.1641.1	sh1786_8c	3285
30.1642	../html/sh1787_8c.js File Reference	3285
30.1642	Variable Documentation	3285
30.1642.1	sh1787_8c	3285
30.1643	../html/sh1790_8c.js File Reference	3285
30.1643	Variable Documentation	3286
30.1643.1	sh1790_8c	3286
30.1644	../html/sh1830_8c.js File Reference	3286
30.1644	Variable Documentation	3286
30.1644.1	sh1830_8c	3286
30.1645	../html/sh1831_8c.js File Reference	3286
30.1645	Variable Documentation	3286
30.1645.1	sh1831_8c	3286
30.1646	../html/sh1834_8c.js File Reference	3287
30.1646	Variable Documentation	3287
30.1646.1	sh1834_8c	3287
30.1647	../html/sh1839_8c.js File Reference	3287
30.1647	Variable Documentation	3287
30.1647.1	sh1839_8c	3287
30.1648	../html/sh1850_8c.js File Reference	3287
30.1648	Variable Documentation	3288
30.1648.1	sh1850_8c	3288
30.1649	../html/sh1851_8c.js File Reference	3288
30.1649	Variable Documentation	3288
30.1649.1	sh1851_8c	3288
30.1650	../html/sh1852_8c.js File Reference	3288
30.1650	Variable Documentation	3288
30.1650.1	sh1852_8c	3288
30.1651	../html/sh1853_8c.js File Reference	3289

30.1651	Variable Documentation	3289
30.1651.1	sh1853_8c	3289
30.1652	30c/html/sh1854_8c.js File Reference	3289
30.1652	Variable Documentation	3289
30.1652.1	sh1854_8c	3289
30.1653	30c/html/sh1855_8c.js File Reference	3289
30.1653	Variable Documentation	3290
30.1653.1	sh1855_8c	3290
30.1654	30c/html/sh1856_8c.js File Reference	3290
30.1654	Variable Documentation	3290
30.1654.1	sh1856_8c	3290
30.1655	30c/html/sh1857_8c.js File Reference	3290
30.1655	Variable Documentation	3290
30.1655.1	sh1857_8c	3290
30.1656	30c/html/sh1858_8c.js File Reference	3291
30.1656	Variable Documentation	3291
30.1656.1	sh1858_8c	3291
30.1657	30c/html/sh1859_8c.js File Reference	3291
30.1657	Variable Documentation	3291
30.1657.1	sh1859_8c	3291
30.1658	30c/html/sh1860_8c.js File Reference	3291
30.1658	Variable Documentation	3292
30.1658.1	sh1860_8c	3292
30.1659	30c/html/sh1870_8c.js File Reference	3292
30.1659	Variable Documentation	3292
30.1659.1	sh1870_8c	3292
30.1660	30c/html/sh1871_8c.js File Reference	3292
30.1660	Variable Documentation	3292
30.1660.1	sh1871_8c	3292
30.1661	30c/html/sh1922_8c.js File Reference	3293

30.1661	Variable Documentation	3293
30.1661.1	sh1922_8c	3293
30.1662	../html/sh1923_8c.js File Reference	3293
30.1662	Variable Documentation	3293
30.1662.1	sh1923_8c	3293
30.1663	../html/sh1924_8c.js File Reference	3293
30.1663	Variable Documentation	3294
30.1663.1	sh1924_8c	3294
30.1664	../html/sh1925_8c.js File Reference	3294
30.1664	Variable Documentation	3294
30.1664.1	sh1925_8c	3294
30.1665	../html/sh1926_8c.js File Reference	3294
30.1665	Variable Documentation	3294
30.1665.1	sh1926_8c	3294
30.1666	../html/sh1927_8c.js File Reference	3295
30.1666	Variable Documentation	3295
30.1666.1	sh1927_8c	3295
30.1667	../html/sh1928_8c.js File Reference	3295
30.1667	Variable Documentation	3295
30.1667.1	sh1928_8c	3295
30.1668	../html/sh1929_8c.js File Reference	3295
30.1668	Variable Documentation	3296
30.1668.1	sh1929_8c	3296
30.1669	../html/sh1930_8c.js File Reference	3296
30.1669	Variable Documentation	3296
30.1669.1	sh1930_8c	3296
30.1670	../html/sh1992_8c.js File Reference	3296
30.1670	Variable Documentation	3296
30.1670.1	sh1992_8c	3296
30.1671	../html/sh1993_8c.js File Reference	3297

30.1671	Variable Documentation	3297
30.1671.1	sh1993_8c	3297
30.1672	../html/sh1994_8c.js File Reference	3297
30.1672	Variable Documentation	3297
30.1672.1	sh1994_8c	3297
30.1673	../html/sh6clvert_8c.js File Reference	3297
30.1673	Variable Documentation	3298
30.1673.1	sh6clvert_8c	3298
30.1674	../html/sh6comedg_8c.js File Reference	3298
30.1674	Variable Documentation	3298
30.1674.1	sh6comedg_8c	3298
30.1675	../html/sh6connect_8c.js File Reference	3298
30.1675	Variable Documentation	3298
30.1675.1	sh6connect_8c	3298
30.1676	../html/sh6count_8c.js File Reference	3299
30.1676	Variable Documentation	3299
30.1676.1	sh6count_8c	3299
30.1677	../html/sh6cvvert_8c.js File Reference	3299
30.1677	Variable Documentation	3299
30.1677.1	sh6cvvert_8c	3299
30.1678	../html/sh6degen_8c.js File Reference	3299
30.1678	Variable Documentation	3300
30.1678.1	sh6degen_8c	3300
30.1679	../html/sh6disconn_8c.js File Reference	3300
30.1679	Variable Documentation	3300
30.1679.1	sh6disconn_8c	3300
30.1680	../html/sh6edgpnt_8c.js File Reference	3300
30.1680	Variable Documentation	3300
30.1680.1	sh6edgpnt_8c	3300
30.1681	../html/sh6edgred_8c.js File Reference	3301

30.1681	Variable Documentation	3301
30.1681.1	sh6edgred_8c	3301
30.1682	c/html/sh6evalint_8c.js File Reference	3301
30.1682	Variable Documentation	3301
30.1682.1	sh6evalint_8c	3301
30.1683	c/html/sh6floop_8c.js File Reference	3301
30.1683	Variable Documentation	3302
30.1683.1	sh6floop_8c	3302
30.1684	c/html/sh6fndsplt_8c.js File Reference	3302
30.1684	Variable Documentation	3302
30.1684.1	sh6fndsplt_8c	3302
30.1685	c/html/sh6getgeom_8c.js File Reference	3302
30.1685	Variable Documentation	3302
30.1685.1	sh6getgeom_8c	3302
30.1686	c/html/sh6getlist_8c.js File Reference	3303
30.1686	Variable Documentation	3303
30.1686.1	sh6getlist_8c	3303
30.1687	c/html/sh6getmain_8c.js File Reference	3303
30.1687	Variable Documentation	3303
30.1687.1	sh6getmain_8c	3303
30.1688	c/html/sh6getnbrs_8c.js File Reference	3303
30.1688	Variable Documentation	3304
30.1688.1	sh6getnbrs_8c	3304
30.1689	c/html/sh6getnext_8c.js File Reference	3304
30.1689	Variable Documentation	3304
30.1689.1	sh6getnext_8c	3304
30.1690	c/html/sh6getthr_8c.js File Reference	3304
30.1690	Variable Documentation	3304
30.1690.1	sh6getthr_8c	3304
30.1691	c/html/sh6getprev_8c.js File Reference	3305

30.1691	Variable Documentation	3305
30.1691.1	sh6getprev_8c	3305
30.1692	../html/sh6gettop_8c.js File Reference	3305
30.1692	Variable Documentation	3305
30.1692.1	sh6gettop_8c	3305
30.1693	../html/sh6idaledg_8c.js File Reference	3305
30.1693	Variable Documentation	3306
30.1693.1	sh6idaledg_8c	3306
30.1694	../html/sh6idcon_8c.js File Reference	3306
30.1694	Variable Documentation	3306
30.1694.1	sh6idcon_8c	3306
30.1695	../html/sh6idfcros_8c.js File Reference	3306
30.1695	Variable Documentation	3306
30.1695.1	sh6idfcros_8c	3306
30.1696	../html/sh6idget_8c.js File Reference	3307
30.1696	Variable Documentation	3307
30.1696.1	sh6idget_8c	3307
30.1697	../html/sh6idkpt_8c.js File Reference	3307
30.1697	Variable Documentation	3307
30.1697.1	sh6idkpt_8c	3307
30.1698	../html/sh6idlis_8c.js File Reference	3307
30.1698	Variable Documentation	3308
30.1698.1	sh6idlis_8c	3308
30.1699	../html/sh6idnpt_8c.js File Reference	3308
30.1699	Variable Documentation	3308
30.1699.1	sh6idnpt_8c	3308
30.1700	../html/sh6idnwun_8c.js File Reference	3308
30.1700	Variable Documentation	3308
30.1700.1	sh6idnwun_8c	3308
30.1701	../html/sh6idput_8c.js File Reference	3309

30.1701	Variable Documentation	3309
30.1701.1	sh6idput_8c	3309
30.1702	doc/html/sh6idrcros_8c.js File Reference	3309
30.1702	Variable Documentation	3309
30.1702.1	sh6idrcros_8c	3309
30.1703	doc/html/sh6idsplit_8c.js File Reference	3309
30.1703	Variable Documentation	3310
30.1703.1	sh6idsplit_8c	3310
30.1704	doc/html/sh6idunite_8c.js File Reference	3310
30.1704	Variable Documentation	3310
30.1704.1	sh6idunite_8c	3310
30.1705	doc/html/sh6insert_8c.js File Reference	3310
30.1705	Variable Documentation	3310
30.1705.1	sh6insert_8c	3310
30.1706	doc/html/sh6inspnt_8c.js File Reference	3311
30.1706	Variable Documentation	3311
30.1706.1	sh6inspnt_8c	3311
30.1707	doc/html/sh6iscnect_8c.js File Reference	3311
30.1707	Variable Documentation	3311
30.1707.1	sh6iscnect_8c	3311
30.1708	doc/html/sh6ishelp_8c.js File Reference	3311
30.1708	Variable Documentation	3312
30.1708.1	sh6ishelp_8c	3312
30.1709	doc/html/sh6isinsid_8c.js File Reference	3312
30.1709	Variable Documentation	3312
30.1709.1	sh6isinsid_8c	3312
30.1710	doc/html/sh6ismain_8c.js File Reference	3312
30.1710	Variable Documentation	3312
30.1710.1	sh6ismain_8c	3312
30.1711	doc/html/sh6nmbhelp_8c.js File Reference	3313

30.1711	Variable Documentation	3313
30.1711.1	sh6nmbhelp_8c	3313
30.1712	10c/html/sh6nmbmain_8c.js File Reference	3313
30.1712	Variable Documentation	3313
30.1712.1	sh6nmbmain_8c	3313
30.1713	10c/html/sh6ptobj_8c.js File Reference	3313
30.1713	Variable Documentation	3314
30.1713.1	sh6ptobj_8c	3314
30.1714	10c/html/sh6ptouchp_8c.js File Reference	3314
30.1714	Variable Documentation	3314
30.1714.1	sh6ptouchp_8c	3314
30.1715	10c/html/sh6putsing_8c.js File Reference	3314
30.1715	Variable Documentation	3314
30.1715.1	sh6putsing_8c	3314
30.1716	10c/html/sh6red_8c.js File Reference	3315
30.1716	Variable Documentation	3315
30.1716.1	sh6red_8c	3315
30.1717	10c/html/sh6remcon_8c.js File Reference	3315
30.1717	Variable Documentation	3315
30.1717.1	sh6remcon_8c	3315
30.1718	10c/html/sh6rempt_8c.js File Reference	3315
30.1718	Variable Documentation	3316
30.1718.1	sh6rempt_8c	3316
30.1719	10c/html/sh6sepcrv_8c.js File Reference	3316
30.1719	Variable Documentation	3316
30.1719.1	sh6sepcrv_8c	3316
30.1720	10c/html/sh6setcnsd_8c.js File Reference	3316
30.1720	Variable Documentation	3316
30.1720.1	sh6setcnsd_8c	3316
30.1721	10c/html/sh6setdir_8c.js File Reference	3317

30.1721	Variable Documentation	3317
30.1721.1	sh6setdir_8c	3317
30.1722	c/html/sh6settop_8c.js File Reference	3317
30.1722	Variable Documentation	3317
30.1722.1	sh6settop_8c	3317
30.1723	c/html/sh6spltgeo_8c.js File Reference	3317
30.1723	Variable Documentation	3318
30.1723.1	sh6spltgeo_8c	3318
30.1724	c/html/sh6tohelp_8c.js File Reference	3318
30.1724	Variable Documentation	3318
30.1724.1	sh6tohelp_8c	3318
30.1725	c/html/sh6tomain_8c.js File Reference	3318
30.1725	Variable Documentation	3318
30.1725.1	sh6tomain_8c	3318
30.1726	c/html/sh6topohlp_8c.js File Reference	3319
30.1726	Variable Documentation	3319
30.1726.1	sh6topohlp_8c	3319
30.1727	c/html/sh6trmlist_8c.js File Reference	3319
30.1727	Variable Documentation	3319
30.1727.1	sh6trmlist_8c	3319
30.1728	c/html/sh__1d__div_8c.js File Reference	3319
30.1728	Variable Documentation	3320
30.1728.1	sh__1d__div_8c	3320
30.1729	c/html/sh__div__crv_8c.js File Reference	3320
30.1729	Variable Documentation	3320
30.1729.1	sh__div__crv_8c	3320
30.1730	c/html/sh__set__at_8c.js File Reference	3320
30.1730	Variable Documentation	3320
30.1730.1	sh__set__at_8c	3320
30.1731	c/html/shape_8c.js File Reference	3321

30.1731	Variable Documentation	3321
30.1731.1	shape_8c	3321
30.1732	30c/html/shcheckput_8c.js File Reference	3321
30.1732	Variable Documentation	3321
30.1732.1	shcheckput_8c	3321
30.1733	30c/html/shchecktyp_8c.js File Reference	3321
30.1733	Variable Documentation	3322
30.1733.1	shchecktyp_8c	3322
30.1734	30c/html/shcfsfing_8c.js File Reference	3322
30.1734	Variable Documentation	3322
30.1734.1	shcfsfing_8c	3322
30.1735	30c/html/shdivsurf_8c.js File Reference	3322
30.1735	Variable Documentation	3322
30.1735.1	shdivsurf_8c	3322
30.1736	30c/html/shevalc_8c.js File Reference	3323
30.1736	Variable Documentation	3323
30.1736.1	shevalc_8c	3323
30.1737	30c/html/shmkhlpts_8c.js File Reference	3323
30.1737	Variable Documentation	3323
30.1737.1	shmkhlpts_8c	3323
30.1738	30c/html/shsing_8c.js File Reference	3323
30.1738	Variable Documentation	3324
30.1738.1	shsing_8c	3324
30.1739	30c/html/sisl_8h.js File Reference	3324
30.1739	Variable Documentation	3324
30.1739.1	sisl_8h	3324
30.1740	30c/html/sisl__aux_8cpp.js File Reference	3324
30.1740	Variable Documentation	3324
30.1740.1	sisl__aux_8cpp	3324
30.1741	30c/html/sisl__aux_8h.js File Reference	3325

30.1741	Variable Documentation	3325
30.1741.1	sisl__aux_8h	3325
30.1742	c/html/sisl__file__io_8h.js File Reference	3325
30.1742	Variable Documentation	3325
30.1742.1	sisl__file__io_8h	3325
30.1743	c/html/sisl__view__demo_8cpp.js File Reference	3325
30.1743	Variable Documentation	3326
30.1743.1	sisl__view__demo_8cpp	3326
30.1744	c/html/sisl_p_8h.js File Reference	3326
30.1744	Variable Documentation	3326
30.1744.1	sisl_p_8h	3326
30.1745	c/html/sl__ex_8cpp.js File Reference	3326
30.1745	Variable Documentation	3326
30.1745.1	sl__ex_8cpp	3326
30.1746	c/html/sleep_8h.js File Reference	3326
30.1746	Variable Documentation	3327
30.1746.1	sleep_8h	3327
30.1747	c/html/solution_8cpp.js File Reference	3327
30.1747	Variable Documentation	3327
30.1747.1	solution_8cpp	3327
30.1748	c/html/sort_8cpp.js File Reference	3327
30.1748	Variable Documentation	3328
30.1748.1	sort_8cpp	3328
30.1749	c/html/spli__silh_8c.js File Reference	3328
30.1749	Variable Documentation	3328
30.1749.1	spli__silh_8c	3328
30.1750	c/html/spline2mesh_8h.js File Reference	3328
30.1750	Variable Documentation	3329
30.1750.1	spline2mesh_8h	3329
30.1751	c/html/struct_go_1_1_adjacency_info.js File Reference	3329

30.1751	Variable Documentation	3329
30.1751.1	struct_go_1_1_adjacency_info	3329
30.1752c/html/	struct_go_1_1_alg2_d_elem.js File Reference	3329
30.1752	Variable Documentation	3330
30.1752.1	struct_go_1_1_alg2_d_elem	3330
30.1753c/html/	struct_go_1_1_alg3_d_elem.js File Reference	3330
30.1753	Variable Documentation	3330
30.1753.1	struct_go_1_1_alg3_d_elem	3330
30.1754c/html/	struct_go_1_1_basis_derivs.js File Reference	3330
30.1754	Variable Documentation	3331
30.1754.1	struct_go_1_1_basis_derivs	3331
30.1755c/html/	struct_go_1_1_basis_derivs2.js File Reference	3331
30.1755	Variable Documentation	3331
30.1755.1	struct_go_1_1_basis_derivs2	3331
30.1756c/html/	struct_go_1_1_basis_derivs_sf.js File Reference	3331
30.1756	Variable Documentation	3332
30.1756.1	struct_go_1_1_basis_derivs_sf	3332
30.1757c/html/	struct_go_1_1_basis_derivs_sf2.js File Reference	3332
30.1757	Variable Documentation	3332
30.1757.1	struct_go_1_1_basis_derivs_sf2	3332
30.1758c/html/	struct_go_1_1_basis_pts.js File Reference	3332
30.1758	Variable Documentation	3333
30.1758.1	struct_go_1_1_basis_pts	3333
30.1759c/html/	struct_go_1_1_basis_pts_sf.js File Reference	3333
30.1759	Variable Documentation	3333
30.1759.1	struct_go_1_1_basis_pts_sf	3333
30.1760c/html/	struct_go_1_1_boundary_function_int.js File Reference	3333
30.1760	Variable Documentation	3334
30.1760.1	struct_go_1_1_boundary_function_int	3334
30.1761c/html/	struct_go_1_1_boundary_geom_int.js File Reference	3334

30.1761	Variable Documentation	3334
30.1761.1	struct_go_1_1_boundary_geom_int	3334
30.1762	c/html/struct_go_1_1_boundary_intersection_data.js File Reference	3334
30.1762	Variable Documentation	3335
30.1762.1	struct_go_1_1_boundary_intersection_data	3335
30.1763	c/html/struct_go_1_1_cached_interval.js File Reference	3335
30.1763	Variable Documentation	3335
30.1763.1	struct_go_1_1_cached_interval	3335
30.1764	c/html/struct_go_1_1_composite_model_file_handler_1_1sfvinfo.js File Reference	3335
30.1764	Variable Documentation	3336
30.1764.1	struct_go_1_1_composite_model_file_handler_1_1sfvinfo	3336
30.1765	c/html/struct_go_1_1_entity_list.js File Reference	3336
30.1765	Variable Documentation	3336
30.1765.1	struct_go_1_1_entity_list	3336
30.1766	c/html/struct_go_1_1_face_connectivity.js File Reference	3336
30.1766	Variable Documentation	3337
30.1766.1	struct_go_1_1_face_connectivity	3337
30.1767	c/html/struct_go_1_1_factorial.js File Reference	3337
30.1767	Variable Documentation	3337
30.1767.1	struct_go_1_1_factorial	3337
30.1768	c/html/struct_go_1_1_factorial_3_011_01_4.js File Reference	3337
30.1768	Variable Documentation	3338
30.1768.1	struct_go_1_1_factorial_3_011_01_4	3338
30.1769	c/html/struct_go_1_1_g_pos.js File Reference	3338
30.1769	Variable Documentation	3338
30.1769.1	struct_go_1_1_g_pos	3338
30.1770	c/html/struct_go_1_1_i_g_e_sdirentry.js File Reference	3338
30.1770	Variable Documentation	3339
30.1770.1	struct_go_1_1_i_g_e_sdirentry	3339
30.1771	c/html/struct_go_1_1_i_g_e_sheader.js File Reference	3339

30.1771	Variable Documentation	3339
30.1771.1	struct_go_1_1_i_g_e_sheader	3339
30.1772	c/html/struct_go_1_1_int_pt_info.js File Reference	3339
30.1772	Variable Documentation	3340
30.1772.1	struct_go_1_1_int_pt_info	3340
30.1773	c/html/struct_go_1_1_l_r_spline_surface_1_1_b_s_key.js File Reference	3340
30.1773	Variable Documentation	3340
30.1773.1	struct_go_1_1_l_r_spline_surface_1_1_b_s_key	3340
30.1774	c/html/struct_go_1_1_l_r_spline_surface_1_1_elem_key.js File Reference	3340
30.1774	Variable Documentation	3341
30.1774.1	struct_go_1_1_l_r_spline_surface_1_1_elem_key	3341
30.1775	c/html/struct_go_1_1_l_r_spline_surface_1_1_refinement2_d.js File Reference	3341
30.1775	Variable Documentation	3341
30.1775.1	struct_go_1_1_l_r_spline_surface_1_1_refinement2_d	3341
30.1776	c/html/struct_go_1_1_l_r_spline_surface_1_1_double__pair__hash.js File Reference	3341
30.1776	Variable Documentation	3342
30.1776.1	struct_go_1_1_l_r_spline_surface_1_1_double__pair__hash	3342
30.1777	c/html/struct_go_1_1_l_r_spline_utils_1_1_support__compare.js File Reference	3342
30.1777	Variable Documentation	3342
30.1777.1	struct_go_1_1_l_r_spline_utils_1_1_support__compare	3342
30.1778	c/html/struct_go_1_1_l_s_smooth_data.js File Reference	3342
30.1778	Variable Documentation	3342
30.1778.1	struct_go_1_1_l_s_smooth_data	3342
30.1779	c/html/struct_go_1_1_param_surface_1_1_degenerate__info.js File Reference	3343
30.1779	Variable Documentation	3343
30.1779.1	struct_go_1_1_param_surface_1_1_degenerate__info	3343
30.1780	c/html/struct_go_1_1_registration_input.js File Reference	3343
30.1780	Variable Documentation	3343
30.1780.1	struct_go_1_1_registration_input	3343
30.1781	c/html/struct_go_1_1_registration_result.js File Reference	3344

30.1781	Variable Documentation	3344
30.1781.1	struct_go_1_1_registration_result	3344
30.1782	c/html/struct_go_1_1_rotation_info.js File Reference	3344
30.1782	Variable Documentation	3344
30.1782.1	struct_go_1_1_rotation_info	3344
30.1783	c/html/struct_go_1_1_sample_point_data.js File Reference	3345
30.1783	Variable Documentation	3345
30.1783.1	struct_go_1_1_sample_point_data	3345
30.1784	c/html/struct_go_1_1_second_order_properties.js File Reference	3345
30.1784	Variable Documentation	3345
30.1784.1	struct_go_1_1_second_order_properties	3345
30.1785	c/html/struct_go_1_1_sing_box.js File Reference	3346
30.1785	Variable Documentation	3346
30.1785.1	struct_go_1_1_sing_box	3346
30.1786	c/html/struct_go_1_1_sing_union.js File Reference	3346
30.1786	Variable Documentation	3346
30.1786.1	struct_go_1_1_sing_union	3346
30.1787	c/html/struct_go_1_1_volume_adjacency_info.js File Reference	3346
30.1787	Variable Documentation	3347
30.1787.1	struct_go_1_1_volume_adjacency_info	3347
30.1788	c/html/struct_go_1_1cv_set_constraint.js File Reference	3347
30.1788	Variable Documentation	3347
30.1788.1	struct_go_1_1cv_set_constraint	3347
30.1789	c/html/struct_go_1_1go_iterator_traits.js File Reference	3348
30.1789	Variable Documentation	3348
30.1789.1	struct_go_1_1go_iterator_traits	3348
30.1790	c/html/struct_go_1_1go_iterator_traits_3_01_t_01_5_01_4.js File Reference	3348
30.1790	Variable Documentation	3348
30.1790.1	struct_go_1_1go_iterator_traits_3_01_t_01_5_01_4	3348
30.1791	c/html/struct_go_1_1go_iterator_traits_3_01const_01_t_01_5_01_4.js File Reference	3349

30.1791	Variable Documentation	3349
30.1791.1	struct_go_1_1go_iterator_traits_3_01const_01_t_01_5_01_4	3349
30.1792	c/html/struct_go_1_1pre_evaluation_sf.js File Reference	3349
30.1792	Variable Documentation	3349
30.1792.1	struct_go_1_1pre_evaluation_sf	3349
30.1793	c/html/struct_go_1_1pre_evaluation_vol.js File Reference	3350
30.1793	Variable Documentation	3350
30.1793.1	struct_go_1_1pre_evaluation_vol	3350
30.1794	c/html/struct_go_1_1raw_pointer_comp.js File Reference	3350
30.1794	Variable Documentation	3350
30.1794.1	struct_go_1_1raw_pointer_comp	3350
30.1795	c/html/struct_go_1_1side_constraint.js File Reference	3351
30.1795	Variable Documentation	3351
30.1795.1	struct_go_1_1side_constraint	3351
30.1796	c/html/struct_go_1_1side_constraint_set.js File Reference	3351
30.1796	Variable Documentation	3351
30.1796.1	struct_go_1_1side_constraint_set	3351
30.1797	c/html/struct_go_1_1tp_tolerances.js File Reference	3351
30.1797	Variable Documentation	3352
30.1797.1	struct_go_1_1tp_tolerances	3352
30.1798	c/html/struct_go_1_1tp_topological_info.js File Reference	3352
30.1798	Variable Documentation	3352
30.1798.1	struct_go_1_1tp_topological_info	3352
30.1799	c/html/struct_s_i_s_l_curve.js File Reference	3352
30.1799	Variable Documentation	3353
30.1799.1	struct_s_i_s_l_curve	3353
30.1800	c/html/struct_s_i_s_l_edge.js File Reference	3353
30.1800	Variable Documentation	3353
30.1800.1	struct_s_i_s_l_edge	3353
30.1801	c/html/struct_s_i_s_l_intcurve.js File Reference	3353

30.1801	Variable Documentation	3354
30.1801.1	struct_s_i_s_l_intcurve	3354
30.1802c/html/	struct_s_i_s_l_intdat.js File Reference	3354
30.1802	Variable Documentation	3354
30.1802.1	struct_s_i_s_l_intdat	3354
30.1803c/html/	struct_s_i_s_l_intlist.js File Reference	3354
30.1803	Variable Documentation	3355
30.1803.1	struct_s_i_s_l_intlist	3355
30.1804c/html/	struct_s_i_s_l_intpt.js File Reference	3355
30.1804	Variable Documentation	3355
30.1804.1	struct_s_i_s_l_intpt	3355
30.1805c/html/	struct_s_i_s_l_intsurf.js File Reference	3356
30.1805	Variable Documentation	3356
30.1805.1	struct_s_i_s_l_intsurf	3356
30.1806c/html/	struct_s_i_s_l_object.js File Reference	3356
30.1806	Variable Documentation	3356
30.1806.1	struct_s_i_s_l_object	3356
30.1807c/html/	struct_s_i_s_l_point.js File Reference	3356
30.1807	Variable Documentation	3357
30.1807.1	struct_s_i_s_l_point	3357
30.1808c/html/	struct_s_i_s_l_ptedge.js File Reference	3357
30.1808	Variable Documentation	3357
30.1808.1	struct_s_i_s_l_ptedge	3357
30.1809c/html/	struct_s_i_s_l_surf.js File Reference	3357
30.1809	Variable Documentation	3358
30.1809.1	struct_s_i_s_l_surf	3358
30.1810c/html/	struct_s_i_s_l_track.js File Reference	3358
30.1810	Variable Documentation	3358
30.1810.1	struct_s_i_s_l_track	3358
30.1811c/html/	struct_s_i_s_l_trimpar.js File Reference	3359

30.1811	Variable Documentation	3359
30.1811.1	struct_s_i_s_l_trimpar	3359
30.1812	c/html/struct_s_i_s_lbox.js File Reference	3359
30.1812	Variable Documentation	3359
30.1812.1	struct_s_i_s_lbox	3359
30.1813	c/html/struct_s_i_s_l_dir.js File Reference	3359
30.1813	Variable Documentation	3360
30.1813.1	struct_s_i_s_l_dir	3360
30.1814	c/html/structgv_color.js File Reference	3360
30.1814	Variable Documentation	3360
30.1814.1	structgv_color	3360
30.1815	c/html/structhed_1_1_t_t_traits.js File Reference	3360
30.1815	Variable Documentation	3361
30.1815.1	structhed_1_1_t_t_traits	3361
30.1816	c/html/structhetriang_1_1_t_t_traits.js File Reference	3361
30.1816	Variable Documentation	3361
30.1816.1	structhetriang_1_1_t_t_traits	3361
30.1817	c/html/structrank__info.js File Reference	3361
30.1817	Variable Documentation	3361
30.1817.1	structrank__info	3361
30.1818	c/html/submat_8cpp.js File Reference	3362
30.1818	Variable Documentation	3362
30.1818.1	submat_8cpp	3362
30.1819	c/html/svd_8cpp.js File Reference	3362
30.1819	Variable Documentation	3362
30.1819.1	svd_8cpp	3362
30.1820	c/html/test__exc_8cpp.js File Reference	3362
30.1820	Variable Documentation	3363
30.1820.1	test__exc_8cpp	3363
30.1821	c/html/test_suite_8h.js File Reference	3363

30.1821	Variable Documentation	3363
30.1821.1	test_suite_8h	3363
30.1821	doc/html/timeutils_8h.js File Reference	3363
30.1822	Variable Documentation	3363
30.1822.1	timeutils_8h	3363
30.1822	doc/html/tmt1_8cpp.js File Reference	3364
30.1823	Variable Documentation	3364
30.1823.1	tmt1_8cpp	3364
30.1823	doc/html/tmt2_8cpp.js File Reference	3364
30.1824	Variable Documentation	3364
30.1824.1	tmt2_8cpp	3364
30.1824	doc/html/tmt3_8cpp.js File Reference	3364
30.1825	Variable Documentation	3365
30.1825.1	tmt3_8cpp	3365
30.1825	doc/html/tmt4_8cpp.js File Reference	3365
30.1826	Variable Documentation	3365
30.1826.1	tmt4_8cpp	3365
30.1826	doc/html/tmt5_8cpp.js File Reference	3365
30.1827	Variable Documentation	3365
30.1827.1	tmt5_8cpp	3365
30.1827	doc/html/tmt6_8cpp.js File Reference	3366
30.1828	Variable Documentation	3366
30.1828.1	tmt6_8cpp	3366
30.1828	doc/html/tmt7_8cpp.js File Reference	3366
30.1829	Variable Documentation	3366
30.1829.1	tmt7_8cpp	3366
30.1829	doc/html/tmt8_8cpp.js File Reference	3366
30.1830	Variable Documentation	3367
30.1830.1	tmt8_8cpp	3367
30.1830	doc/html/tmt9_8cpp.js File Reference	3367

30.1831	Variable Documentation	3367
30.1831.1	tmt9_8cpp	3367
30.1832	c/html/tmt_8cpp.js File Reference	3367
30.1832	Variable Documentation	3368
30.1832.1	tmt_8cpp	3368
30.1833	c/html/tmt_8h.js File Reference	3368
30.1833	Variable Documentation	3368
30.1833.1	tmt_8h	3368
30.1834	c/html/tmta_8cpp.js File Reference	3368
30.1834	Variable Documentation	3369
30.1834.1	tmta_8cpp	3369
30.1835	c/html/tmtb_8cpp.js File Reference	3369
30.1835	Variable Documentation	3369
30.1835.1	tmtb_8cpp	3369
30.1836	c/html/tmtc_8cpp.js File Reference	3369
30.1836	Variable Documentation	3369
30.1836.1	tmtc_8cpp	3369
30.1837	c/html/tmtd_8cpp.js File Reference	3370
30.1837	Variable Documentation	3370
30.1837.1	tmtd_8cpp	3370
30.1838	c/html/tmte_8cpp.js File Reference	3370
30.1838	Variable Documentation	3370
30.1838.1	tmte_8cpp	3370
30.1839	c/html/tmtf_8cpp.js File Reference	3370
30.1839	Variable Documentation	3371
30.1839.1	tmtf_8cpp	3371
30.1840	c/html/tmtg_8cpp.js File Reference	3371
30.1840	Variable Documentation	3371
30.1840.1	tmtg_8cpp	3371
30.1841	c/html/tmth_8cpp.js File Reference	3371

30.1841	Variable Documentation	3371
30.1841.1	t1nth_8cpp	3371
30.1842	c/html/tmti_8cpp.js File Reference	3372
30.1842	Variable Documentation	3372
30.1842.1	t1nti_8cpp	3372
30.1843	c/html/tmtj_8cpp.js File Reference	3372
30.1843	Variable Documentation	3372
30.1843.1	t1ntj_8cpp	3372
30.1844	c/html/tmtk_8cpp.js File Reference	3372
30.1844	Variable Documentation	3373
30.1844.1	t1ntk_8cpp	3373
30.1845	c/html/tmtl_8cpp.js File Reference	3373
30.1845	Variable Documentation	3373
30.1845.1	t1ntl_8cpp	3373
30.1846	c/html/tmtm_8cpp.js File Reference	3373
30.1846	Variable Documentation	3374
30.1846.1	t1ntm_8cpp	3374
30.1847	c/html/tp_joint_type_8h.js File Reference	3374
30.1847	Variable Documentation	3374
30.1847.1	tp_joint_type_8h	3374
30.1848	c/html/tp_utils_8h.js File Reference	3374
30.1848	Variable Documentation	3375
30.1848.1	tp_utils_8h	3375
30.1849	c/html/transfutils_8cpp.js File Reference	3375
30.1849	Variable Documentation	3375
30.1849.1	t1transfutils_8cpp	3375
30.1850	c/html/transfutils_8h.js File Reference	3375
30.1850	Variable Documentation	3376
30.1850.1	t1transfutils_8h	3376
30.1851	c/html/tstcyclknt_8c.js File Reference	3376

30.1851	Variable Documentation	3376
30.1851.1	<code>ttl_cyclknt_8c</code>	3376
30.1852	<code>html/ttl_8h.js</code> File Reference	3376
30.1852	Variable Documentation	3377
30.1852.1	<code>ttl_8h</code>	3377
30.1853	<code>html/ttl__constr_8h.js</code> File Reference	3377
30.1853	Variable Documentation	3377
30.1853.1	<code>ttl__constr_8h</code>	3377
30.1854	<code>html/ttl__util_8h.js</code> File Reference	3377
30.1854	Variable Documentation	3377
30.1854.1	<code>ttl__util_8h</code>	3377
30.1855	<code>tools-core/include/GoTools/creators/AdaptCurve.h</code> File Reference	3378
30.1856	<code>tools-core/include/GoTools/creators/ApproxCrvToSeqs.h</code> File Reference	3378
30.1857	<code>tools-core/include/GoTools/creators/ApproxCurve.h</code> File Reference	3379
30.1858	<code>tools-core/include/GoTools/creators/ApproxSurf.h</code> File Reference	3380
30.1859	<code>tools-core/include/GoTools/creators/ConstraintDefinitions.h</code> File Reference	3380
30.1860	<code>tools-core/include/GoTools/creators/CoonsPatchGen.h</code> File Reference	3381
30.1861	<code>tools-core/include/GoTools/creators/CreatorsOffsetUtils.h</code> File Reference	3383
30.1861	Macro Definition Documentation	3383
30.1861.1	<code>_OFFSETUTILS_H</code>	3383
30.1862	<code>tools-core/include/GoTools/creators/CreatorsUtils.h</code> File Reference	3384
30.1863	<code>tools-core/include/GoTools/creators/CrossTangentOffset.h</code> File Reference	3385
30.1864	<code>tools-core/include/GoTools/creators/CrossTanOffDist.h</code> File Reference	3385
30.1865	<code>tools-core/include/GoTools/creators/CurveCreators.h</code> File Reference	3386
30.1866	<code>tools-core/include/GoTools/creators/EvalCurve.h</code> File Reference	3387
30.1867	<code>tools-core/include/GoTools/creators/EvalCurveSet.h</code> File Reference	3388
30.1868	<code>tools-core/include/GoTools/creators/EvalParamCurve.h</code> File Reference	3388
30.1869	<code>tools-core/include/GoTools/creators/EvalSurface.h</code> File Reference	3389
30.1870	<code>tools-core/include/GoTools/creators/HahnsSurfaceGen.h</code> File Reference	3390
30.1871	<code>tools-core/include/GoTools/creators/HermiteAppC.h</code> File Reference	3391

30.1872	tools-core/include/GoTools/creators/HermiteAppS.h File Reference	3391
30.1873	tools-core/include/GoTools/creators/HermiteGrid1D.h File Reference	3392
30.1874	tools-core/include/GoTools/creators/HermiteGrid1DMulti.h File Reference	3393
30.1875	tools-core/include/GoTools/creators/HermiteGrid2D.h File Reference	3394
30.1876	tools-core/include/GoTools/creators/IntCrvEvaluator.h File Reference	3395
30.1877	tools-core/include/GoTools/creators/Integrate.h File Reference	3396
30.1878	tools-core/include/GoTools/creators/LiftCurve.h File Reference	3397
30.1879	tools-core/include/GoTools/creators/LoftSurfaceCreator.h File Reference	3397
30.1880	tools-core/include/GoTools/creators/ModifySurf.h File Reference	3398
30.1881	tools-core/include/GoTools/creators/ProjectCurve.h File Reference	3399
30.1882	tools-core/include/GoTools/creators/ProjectCurveAndCrossTan.h File Reference	3399
30.1883	tools-core/include/GoTools/creators/ProjectIntersectionCurve.h File Reference	3400
30.1884	tools-core/include/GoTools/creators/SmoothCurve.h File Reference	3401
30.1885	tools-core/include/GoTools/creators/SmoothCurveSet.h File Reference	3402
30.1886	tools-core/include/GoTools/creators/SmoothSurf.h File Reference	3402
30.1887	tools-core/include/GoTools/creators/SmoothSurfSet.h File Reference	3403
30.1888	tools-core/include/GoTools/creators/SmoothTransition.h File Reference	3403
30.1889	tools-core/include/GoTools/creators/SolveBCG.h File Reference	3404
30.1890	tools-core/include/GoTools/creators/SolveCG.h File Reference	3405
30.1891	tools-core/include/GoTools/creators/SolveCGCO.h File Reference	3406
30.1892	tools-core/include/GoTools/creators/SpaceIntCrv.h File Reference	3406
30.1893	tools-core/include/GoTools/creators/SurfaceCreators.h File Reference	3407
30.1894	tools-core/include/GoTools/creators/TrimCurve.h File Reference	3408
30.1895	tools-core/include/GoTools/geometry/BoundedCurve.h File Reference	3408
30.1896	tools-core/include/GoTools/geometry/BoundedSurface.h File Reference	3409
30.1897	tools-core/include/GoTools/geometry/BoundedUtils.h File Reference	3410
30.1898	tools-core/include/GoTools/geometry/BsplineBasis.h File Reference	3412
30.1898	Macro Definition Documentation	3413
30.1898.1	CHECK	3413
30.1899	tools-core/include/GoTools/geometry/Circle.h File Reference	3414

30.1900	tools-core/include/GoTools/geometry/ClassType.h File Reference	3415
30.1901	tools-core/include/GoTools/geometry/ClosestPoint.h File Reference	3415
30.1902	tools-core/include/GoTools/geometry/CompositeSurface.h File Reference	3417
30.1903	tools-core/include/GoTools/geometry/Cone.h File Reference	3418
30.1904	tools-core/include/GoTools/geometry/copyright.h File Reference	3419
30.1904	Macro Definition Documentation	3419
30.1904.1	_COPYRIGHT_H	3419
30.1905	parametrization/include/GoTools/parametrization/copyright.h File Reference	3419
30.1906	tools-core/include/GoTools/geometry/Curvature.h File Reference	3419
30.1907	tools-core/include/GoTools/geometry/CurvatureAnalysis.h File Reference	3420
30.1907	Detailed Description	3420
30.1908	tools-core/include/GoTools/geometry/CurveBoundedDomain.h File Reference	3421
30.1909	tools-core/include/GoTools/geometry/CurveInterpolator.h File Reference	3421
30.1910	tools-core/include/GoTools/geometry/CurveLoop.h File Reference	3422
30.1911	tools-core/include/GoTools/geometry/CurveOnSurface.h File Reference	3423
30.1912	tools-core/include/GoTools/geometry/Cylinder.h File Reference	3424
30.1913	tools-core/include/GoTools/geometry/Disc.h File Reference	3425
30.1914	tools-core/include/GoTools/geometry/Domain.h File Reference	3426
30.1915	tools-core/include/GoTools/geometry/ElementaryCurve.h File Reference	3427
30.1916	tools-core/include/GoTools/geometry/ElementarySurface.h File Reference	3428
30.1917	tools-core/include/GoTools/geometry/Ellipse.h File Reference	3429
30.1918	tools-core/include/GoTools/geometry/examples_core_doxygen.h File Reference	3430
30.1919	tools-core/include/GoTools/geometry/examples_doxygen.h File Reference	3430
30.1920	tools-core/include/GoTools/geometry/extremalPtSurfSurf.h File Reference	3430
30.1920	Detailed Description	3430
30.1921	tools-core/include/GoTools/geometry/Factory.h File Reference	3431
30.1922	tools-core/include/GoTools/geometry/GapRemoval.h File Reference	3431
30.1923	tools-core/include/GoTools/geometry/geometry_doxygen.h File Reference	3433
30.1924	tools-core/include/GoTools/geometry/GeometryTools.h File Reference	3433
30.1925	tools-core/include/GoTools/geometry/GeomObject.h File Reference	3435

30.1926	tools-core/include/GoTools/geometry/GoIntersections.h File Reference	3436
30.1926	Detailed Description	3437
30.1927	tools-core/include/GoTools/geometry/GoTools.h File Reference	3438
30.1928	tools-core/include/GoTools/geometry/GoTools_version.h File Reference	3439
30.1928	Macro Definition Documentation	3439
30.1928.1	GO_VERSION_MAJOR	3439
30.1928.1	GO_VERSION_MINOR	3439
30.1928.1	GO_VERSION_PATCH	3439
30.1929	tools-core/include/GoTools/geometry/HermitInterpolator.h File Reference	3440
30.1930	tools-core/include/GoTools/geometry/Hyperbola.h File Reference	3440
30.1931	tools-core/include/GoTools/geometry/Interpolator.h File Reference	3441
30.1932	tools-core/include/GoTools/geometry/Line.h File Reference	3442
30.1933	tools-core/include/GoTools/geometry/LineCloud.h File Reference	3443
30.1934	tools-core/include/GoTools/geometry/LoopUtils.h File Reference	3444
30.1935	tools-core/include/GoTools/geometry/ObjectHeader.h File Reference	3445
30.1936	tools-core/include/GoTools/geometry/orientCurves.h File Reference	3447
30.1937	tools-core/include/GoTools/geometry/Parabola.h File Reference	3447
30.1938	tools-core/include/GoTools/geometry/ParamCurve.h File Reference	3448
30.1939	tools-core/include/GoTools/geometry/ParamSurface.h File Reference	3449
30.1940	tools-core/include/GoTools/geometry/Plane.h File Reference	3450
30.1941	tools-core/include/GoTools/geometry/PointCloud.h File Reference	3451
30.1942	tools-core/include/GoTools/geometry/PointOnCurve.h File Reference	3452
30.1943	tools-core/include/GoTools/geometry/PointSequence.h File Reference	3453
30.1944	tools-core/include/GoTools/geometry/RectDomain.h File Reference	3454
30.1945	tools-core/include/GoTools/geometry/RectGrid.h File Reference	3455
30.1946	tools-core/include/GoTools/geometry/SISL_code.h File Reference	3456
30.1947	tools-core/include/GoTools/geometry/sisl_file_io.h File Reference	3456
30.1947	Macro Definition Documentation	3456
30.1947.1	STD_FILE	3456
30.1947	Function Documentation	3457

30.1947.2	curve_to_file(STD_FILE *f, struct SISLCurve *c1)	3457
30.1947.2	file_to_obj(STD_FILE *fp, SISLObject **wo, int *jstat)	3457
30.1947.2	get_next_surface(STD_FILE *fp, SISLSurf **qc)	3457
30.1947.2	get_sisl_surfaces(STD_FILE *fp, std::vector< shared_ptr< SISLSurf > > &sisl_sfs)	3457
30.1947.2	read_non_comment(STD_FILE *fp, char *string)	3457
30.1947.2	surface_to_file(STD_FILE *f, struct SISLSurf *surf)	3457
30.1948	tools-core/include/GoTools/geometry/SISLconversion.h File Reference	3457
30.1948	Detailed Description	3458
30.1949	tools-core/include/GoTools/geometry/Sphere.h File Reference	3458
30.1950	tools-core/include/GoTools/geometry/SplineApproximator.h File Reference	3459
30.1951	tools-core/include/GoTools/geometry/SplineCurve.h File Reference	3460
30.1952	tools-core/include/GoTools/geometry/SplineDebugUtils.h File Reference	3460
30.1953	tools-core/include/GoTools/geometry/SplineInterpolator.h File Reference	3461
30.1954	tools-core/include/GoTools/geometry/SplineSurface.h File Reference	3462
30.1955	tools-core/include/GoTools/geometry/SplineUtils.h File Reference	3462
30.1956	tools-core/include/GoTools/geometry/Streamable.h File Reference	3464
30.1957	tools-core/include/GoTools/geometry/streamable_doxyman.h File Reference	3465
30.1958	tools-core/include/GoTools/geometry/SurfaceInterpolator.h File Reference	3465
30.1959	tools-core/include/GoTools/geometry/SurfaceOfLinearExtrusion.h File Reference	3465
30.1960	tools-core/include/GoTools/geometry/SurfaceOfRevolution.h File Reference	3466
30.1961	tools-core/include/GoTools/geometry/SurfaceTools.h File Reference	3468
30.1962	tools-core/include/GoTools/geometry/SweepSurfaceCreator.h File Reference	3469
30.1963	tools-core/include/GoTools/geometry/Torus.h File Reference	3470
30.1964	tools-core/include/GoTools/geometry/Utils.h File Reference	3471
30.1965	tools-core/include/GoTools/tesselator/2dpoly_for_s2m.h File Reference	3472
30.1965	Typedef Documentation	3473
30.1965.1	short_list	3473
30.1965.1	short_list_short_list	3473
30.1966	tools-core/include/GoTools/tesselator/CurveTesselator.h File Reference	3474
30.1967	tools-core/include/GoTools/tesselator/GeneralMesh.h File Reference	3475

30.1968	tools-core/include/GoTools/tesselator/GenericTriMesh.h File Reference	3476
30.1969	tools-core/include/GoTools/tesselator/LineCloudTesselator.h File Reference	3477
30.1970	tools-core/include/GoTools/tesselator/LineStrip.h File Reference	3477
30.1971	tools-core/include/GoTools/tesselator/NoopTesselator.h File Reference	3479
30.1972	tools-core/include/GoTools/tesselator/ParametricSurfaceTesselator.h File Reference	3479
30.1973	tools-core/include/GoTools/tesselator/QuadMesh.h File Reference	3480
30.1974	tools-core/include/GoTools/tesselator/RectangularSurfaceTesselator.h File Reference	3482
30.1975	tools-core/include/GoTools/tesselator/RectGridTesselator.h File Reference	3483
30.1976	tools-core/include/GoTools/tesselator/RegularMesh.h File Reference	3483
30.1977	tools-core/include/GoTools/tesselator/spline2mesh.h File Reference	3485
30.1978	tools-core/include/GoTools/tesselator/Tesselator.h File Reference	3485
30.1979	tools-core/include/GoTools/tesselator/TesselatorUtils.h File Reference	3486
30.1980	tools-core/include/GoTools/utills/Array.h File Reference	3487
30.1981	tools-core/include/GoTools/utills/BaryCoordSystem.h File Reference	3489
30.1982	tools-core/include/GoTools/utills/BaryCoordSystemTriangle3D.h File Reference	3490
30.1983	tools-core/include/GoTools/utills/binom.h File Reference	3491
30.1984	tools-core/include/GoTools/utills/BoundingBox.h File Reference	3491
30.1985	tools-core/include/GoTools/utills/brent_minimize.h File Reference	3493
30.1986	tools-core/include/GoTools/utills/checks.h File Reference	3493
30.1987	tools-core/include/GoTools/utills/ClosestPointUtils.h File Reference	3495
30.1988	tools-core/include/GoTools/utills/CompositeBox.h File Reference	3496
30.1989	tools-core/include/GoTools/utills/config.h File Reference	3497
30.1989	Macro Definition Documentation	3497
30.1989.1	GO_API	3497
30.1990	tools-core/include/GoTools/utills/CoordinateSystem.h File Reference	3498
30.1991	tools-core/include/GoTools/utills/CPUclock.h File Reference	3499
30.1992	tools-core/include/GoTools/utills/CurvatureUtils.h File Reference	3499
30.1993	tools-core/include/GoTools/utills/DirectionCone.h File Reference	3500
30.1994	tools-core/include/GoTools/utills/errormacros.h File Reference	3501
30.1994	Macro Definition Documentation	3502

30.1994.1	ALWAYS_ERROR_IF	3502
30.1994.1	ASSERT	3502
30.1994.1	ASSERT2	3502
30.1994.1	DEBUG_ERROR_IF	3502
30.1994.1	GO_NO_CHECKS	3502
30.1994.1	MESSAGE	3502
30.1994.1	MESSAGE_IF	3502
30.1994.1	REPORT	3502
30.1994.1	THROW	3502
30.1995	tools-core/include/GoTools/utis/Factorial.h File Reference	3503
30.1996	tools-core/include/GoTools/utis/GeneralFunctionMinimizer.h File Reference	3503
30.1997	tools-core/include/GoTools/utis/GeneralFunctionMinimizer_implementation.h File Reference	3504
30.1998	tools-core/include/GoTools/utis/gotools-doxymain.h File Reference	3506
30.1999	tools-core/include/GoTools/utis/Integration.h File Reference	3506
30.2000	tools-core/include/GoTools/utis/LUDecomp.h File Reference	3507
30.2001	tools-core/include/GoTools/utis/LUDecomp_implementation.h File Reference	3508
30.2002	tools-core/include/GoTools/utis/MatrixXD.h File Reference	3509
30.2003	tools-core/include/GoTools/utis/Point.h File Reference	3510
30.2004	tools-core/include/GoTools/utis/randomnoise.h File Reference	3511
30.2005	tools-core/include/GoTools/utis/Rational.h File Reference	3511
30.2006	tools-core/include/GoTools/utis/RegistrationUtils.h File Reference	3512
30.2007	tools-core/include/GoTools/utis/RotatedBox.h File Reference	3513
30.2008	tools-core/include/GoTools/utis/ScratchVect.h File Reference	3514
30.2009	tools-core/include/GoTools/utis/sleep.h File Reference	3515
30.2009	Macro Definition Documentation	3515
30.2009.1	msleep	3515
30.2010	tools-core/include/GoTools/utis/StreamUtils.h File Reference	3516
30.2010	Function Documentation	3517
30.2010.1	object_from_stream(std::istream &is, T &obj)	3517
30.2010.1	object_from_stream(std::wistream &is, T &obj)	3517

30.2010.1	object_from_stream(std::istream &is, std::vector< T > &v)	3517
30.2010.1	object_from_stream(std::wistream &is, std::vector< T > &v)	3517
30.2010.1	object_to_stream(std::ostream &os, const T &obj)	3517
30.2010.1	object_to_stream(std::wostream &os, const T &obj)	3517
30.2010.1	object_to_stream(std::ostream &os, const std::vector< T > &v)	3517
30.2010.1	object_to_stream(std::wostream &os, const std::vector< T > &v)	3517
30.2010	tools-core/include/GoTools/Utils/TimeUtils.h File Reference	3518
30.2010	tools-core/include/GoTools/Utils/Values.h File Reference	3518
30.2012	Detailed Description	3519
30.2012	Macro Definition Documentation	3519
30.2012.2	DEFAULT_PARAMETER_EPSILON	3519
30.2012.2	DEFAULT_SPACE_EPSILON	3519
30.2012	Variable Documentation	3519
30.2012.3	M_PI	3519
30.2012.3	MAXDOUBLE	3519
30.2012.3	MAXINT	3520
30.2010	tools-core/include/GoTools/Utils/Volumes.h File Reference	3520
30.2010	igeslib/include/GoTools/igeslib/ftGroupGeom.h File Reference	3521
30.2010	igeslib/include/GoTools/igeslib/ftTangPriority.h File Reference	3522
30.2010	igeslib/include/GoTools/igeslib/IGESconverter.h File Reference	3523
30.2010	igeslib/include/GoTools/igeslib/igeslib_doxyman.h File Reference	3524
30.2017	Macro Definition Documentation	3525
30.2017.1	IGESLIB	3525
30.2010	implicitization/include/GoTools/implicitization/BernsteinMulti.h File Reference	3525
30.2010	implicitization/include/GoTools/implicitization/BernsteinPoly.h File Reference	3526
30.2010	implicitization/include/GoTools/implicitization/BernsteinTetrahedralPoly.h File Reference	3528
30.2010	implicitization/include/GoTools/implicitization/BernsteinTriangularPoly.h File Reference	3529
30.2010	implicitization/include/GoTools/implicitization/BernsteinUtils.h File Reference	3531
30.2010	implicitization/include/GoTools/implicitization/BezierTriangle.h File Reference	3532
30.2010	implicitization/include/GoTools/implicitization/Binomial.h File Reference	3533

30.2025	implicitization/include/GoTools/implicitization/GpuMatrix.hpp File Reference	3534
30.2026	implicitization/include/GoTools/implicitization/implicitization-doxymain.h File Reference	3535
30.2027	implicitization/include/GoTools/implicitization/ImplicitizeCurveAlgo.h File Reference	3535
30.2028	implicitization/include/GoTools/implicitization/ImplicitizeCurveAndVectorAlgo.h File Reference	3535
30.2029	implicitization/include/GoTools/implicitization/ImplicitizePointCloudAlgo.h File Reference	3536
30.2030	implicitization/include/GoTools/implicitization/ImplicitizeSurfaceAlgo.h File Reference	3537
30.2031	implicitization/include/GoTools/implicitization/ImplicitUtils.h File Reference	3537
30.2032	intersections/include/GoTools/intersections/AlgObj2DInt.h File Reference	3539
30.2033	intersections/include/GoTools/intersections/AlgObj3DInt.h File Reference	3540
30.2034	intersections/include/GoTools/intersections/AlgObjectInt.h File Reference	3541
30.2035	intersections/include/GoTools/intersections/BoundaryFunctionInt.h File Reference	3541
30.2036	intersections/include/GoTools/intersections/BoundaryGeomInt.h File Reference	3543
30.2037	intersections/include/GoTools/intersections/Coincidence.h File Reference	3543
30.2038	intersections/include/GoTools/intersections/ComplexityInfo.h File Reference	3545
30.2039	intersections/include/GoTools/intersections/ConeInt.h File Reference	3545
30.2040	intersections/include/GoTools/intersections/CvCvIntersector.h File Reference	3545
30.2041	intersections/include/GoTools/intersections/CvPtIntersector.h File Reference	3546
30.2042	intersections/include/GoTools/intersections/CylinderInt.h File Reference	3546
30.2043	intersections/include/GoTools/intersections/generic_graph_algorithms.h File Reference	3547
30.2043	Function Documentation	3548
30.2043.1	get_fundamental_cycle_set(int num_nodes, FunctorConnectedTo connectedTo, std::vector< std::vector< int > > &result)	3548
30.2043.1	get_fundamental_cycle_set(int num_nodes, const std::vector< std::vector< int > > &table, std::vector< std::vector< int > > &result)	3548
30.2043.1	get_individual_paths(int num_nodes, FunctorConnectedTo connectedTo, std::vector< std::vector< int > > &paths, std::vector< std::vector< int > > &cycles, std::vector< int > &isolated_nodes)	3548
30.2043.1	is_path_connected(int node1_index, int node2_index, int num_nodes, FunctorConnectedTo connected_to)	3549
30.2044	intersections/include/GoTools/intersections/generic_graph_algorithms_implementation.h File Reference	3549
30.2044	Enumeration Type Documentation	3550
30.2044.1	NodeStatus	3550

30.2044	Function Documentation	3551
30.2044.2	get_fundamental_cycle_set(int num_nodes, FunctorConnectedTo connectedTo, std::vector< std::vector< int > > &result)	3551
30.2044.2	get_fundamental_cycle_set(int num_nodes, const std::vector< std::vector< int > > &table, std::vector< std::vector< int > > &result)	3551
30.2044.2	get_individual_paths(int num_nodes, FunctorConnectedTo connectedTo, std::vector< std::vector< int > > &paths, std::vector< std::vector< int > > &cycles, std::vector< int > &isolated_nodes)	3551
30.2044.2	is_path_connected(int node1_index, int node2_index, int num_nodes, FunctorConnectedTo connected_to)	3551
30.2045	Intersections/include/GoTools/intersections/GeomObjectInt.h File Reference	3551
30.2046	Intersections/include/GoTools/intersections/GeoTol.h File Reference	3552
30.2047	Intersections/include/GoTools/intersections/Identity.h File Reference	3553
30.2048	Intersections/include/GoTools/intersections/IntersectionCurve.h File Reference	3553
30.2049	Intersections/include/GoTools/intersections/IntersectionInterface.h File Reference	3555
30.2050	Intersections/include/GoTools/intersections/IntersectionLink.h File Reference	3556
30.2051	Intersections/include/GoTools/intersections/IntersectionPoint.h File Reference	3557
30.2052	Intersections/include/GoTools/intersections/IntersectionPointUtils.h File Reference	3558
30.2053	Intersections/include/GoTools/intersections/IntersectionPool.h File Reference	3560
30.2054	Intersections/include/GoTools/intersections/IntersectionPoolUtils.h File Reference	3561
30.2055	Intersections/include/GoTools/intersections/intersections-doxymain.h File Reference	3562
30.2056	Intersections/include/GoTools/intersections/IntersectionUtils.h File Reference	3562
30.2057	Intersections/include/GoTools/intersections/Intersector.h File Reference	3563
30.2058	Intersections/include/GoTools/intersections/Intersector2Obj.h File Reference	3564
30.2059	Intersections/include/GoTools/intersections/IntersectorAlgPar.h File Reference	3565
30.2060	Intersections/include/GoTools/intersections/IntersectorFuncConst.h File Reference	3565
30.2061	Intersections/include/GoTools/intersections/Line2DInt.h File Reference	3566
30.2062	Intersections/include/GoTools/intersections/LinkType.h File Reference	3567
30.2063	Intersections/include/GoTools/intersections/Par0FuncIntersector.h File Reference	3568
30.2064	Intersections/include/GoTools/intersections/Par1FuncIntersector.h File Reference	3568
30.2065	Intersections/include/GoTools/intersections/Par2FuncIntersector.h File Reference	3569
30.2066	Intersections/include/GoTools/intersections/Param0FunctionInt.h File Reference	3570
30.2067	Intersections/include/GoTools/intersections/Param1FunctionInt.h File Reference	3571

30.2068	ersections/include/GoTools/intersections/Param2FunctionInt.h File Reference	3572
30.2069	ersections/include/GoTools/intersections/ParamCurveInt.h File Reference	3573
30.2070	ersections/include/GoTools/intersections/ParamFunctionInt.h File Reference	3574
30.2071	ersections/include/GoTools/intersections/ParamGeomInt.h File Reference	3575
30.2072	ersections/include/GoTools/intersections/ParamObjectInt.h File Reference	3575
30.2073	ersections/include/GoTools/intersections/ParamPointInt.h File Reference	3576
30.2074	ersections/include/GoTools/intersections/ParamSurfaceInt.h File Reference	3578
30.2075	ersections/include/GoTools/intersections/PlaneInt.h File Reference	3578
30.2076	ersections/include/GoTools/intersections/PtPtIntersector.h File Reference	3579
30.2077	ersections/include/GoTools/intersections/SecondOrderProperties.h File Reference	3580
30.2078	ersections/include/GoTools/intersections/SfCvIntersector.h File Reference	3581
30.2079	ersections/include/GoTools/intersections/SfPtIntersector.h File Reference	3582
30.2080	ersections/include/GoTools/intersections/SfSelfIntersector.h File Reference	3583
30.2081	ersections/include/GoTools/intersections/SfSfIntersector.h File Reference	3583
30.2082	ersections/include/GoTools/intersections/Singular.h File Reference	3584
30.2083	ersections/include/GoTools/intersections/SingularityClassification.h File Reference	3585
30.2084	ersections/include/GoTools/intersections/SingularityInfo.h File Reference	3585
30.2085	ersections/include/GoTools/intersections/SingularityType.h File Reference	3587
30.2086	ersections/include/GoTools/intersections/SphereInt.h File Reference	3588
30.2087	ersections/include/GoTools/intersections/Spline1FunctionInt.h File Reference	3588
30.2088	ersections/include/GoTools/intersections/Spline2FunctionInt.h File Reference	3589
30.2089	ersections/include/GoTools/intersections/SplineCurveInt.h File Reference	3590
30.2090	ersections/include/GoTools/intersections/SplineSurfaceInt.h File Reference	3590
30.2091	ersections/include/GoTools/intersections/SubdivisionClassification.h File Reference	3591
30.2092	ersections/include/GoTools/intersections/SurfaceAssembly.h File Reference	3591
30.2093	ersections/include/GoTools/intersections/TorusInt.h File Reference	3592
30.2094	geometric_model/include/GoTools/isogeometric_model/BdCondFunctor.h File Reference	3592
30.2095	geometric_model/include/GoTools/isogeometric_model/BdConditionType.h File Reference . . .	3593
30.2096	geometric_model/include/GoTools/isogeometric_model/BlockBoundaryCondition.h File Reference	3594
30.2097	geometric_model/include/GoTools/isogeometric_model/BlockPointBdCond.h File Reference . .	3595

30.2098	Geometric_model/include/GoTools/isogeometric_model/BlockSolution.h File Reference	3596
30.2099	Geometric_model/include/GoTools/isogeometric_model/EvalFunctorCurve.h File Reference	3597
30.2100	Geometric_model/include/GoTools/isogeometric_model/EvalFunctorSurface.h File Reference	3598
30.2101	Geometric_model/include/GoTools/isogeometric_model/isogeometric_model-doxymain.h File Reference	3599
30.2102	Geometric_model/include/GoTools/isogeometric_model/IsogeometricBlock.h File Reference	3599
30.2103	Geometric_model/include/GoTools/isogeometric_model/IsogeometricModel.h File Reference	3600
30.2104	Geometric_model/include/GoTools/isogeometric_model/IsogeometricSfBlock.h File Reference	3601
30.2105	Geometric_model/include/GoTools/isogeometric_model/IsogeometricSfModel.h File Reference	3602
30.2106	Geometric_model/include/GoTools/isogeometric_model/IsogeometricVolBlock.h File Reference	3603
30.2107	Geometric_model/include/GoTools/isogeometric_model/IsogeometricVolModel.h File Reference	3604
30.2108	Geometric_model/include/GoTools/isogeometric_model/SfBoundaryCondition.h File Reference	3604
30.2109	Geometric_model/include/GoTools/isogeometric_model/SfPointBdCond.h File Reference	3606
30.2110	Geometric_model/include/GoTools/isogeometric_model/SfSolution.h File Reference	3607
30.2111	Geometric_model/include/GoTools/isogeometric_model/VolBoundaryCondition.h File Reference	3609
30.2112	Geometric_model/include/GoTools/isogeometric_model/VolPointBdCond.h File Reference	3610
30.2113	Geometric_model/include/GoTools/isogeometric_model/VolSolution.h File Reference	3611
30.2114	rsplines2D/include/GoTools/lrsplines2D/Direction2D.h File Reference	3613
30.2115	rsplines2D/include/GoTools/lrsplines2D/Element2D.h File Reference	3613
30.2116	rsplines2D/include/GoTools/lrsplines2D/IndexMesh2DIterator.h File Reference	3614
30.2117	rsplines2D/include/GoTools/lrsplines2D/LinDepUtils.h File Reference	3615
30.2118	rsplines2D/include/GoTools/lrsplines2D/LRApproxApp.h File Reference	3616
30.2119	rsplines2D/include/GoTools/lrsplines2D/LRBenchmarkUtils.h File Reference	3617
30.2120	rsplines2D/include/GoTools/lrsplines2D/LRBSpline2D.h File Reference	3617
30.2121	rsplines2D/include/GoTools/lrsplines2D/LRBSpline2DUtils.h File Reference	3618
30.2122	rsplines2D/include/GoTools/lrsplines2D/LRSplineEvalGrid.h File Reference	3619
30.2122M	Macro Definition Documentation	3620
30.2122.1	_1_LRSPLINEEVAVGRID_H	3620
30.2123	rsplines2D/include/GoTools/lrsplines2D/LRSplineMBA.h File Reference	3620
30.2124	rsplines2D/include/GoTools/lrsplines2D/LRSplinePlotUtils.h File Reference	3621
30.2125	rsplines2D/include/GoTools/lrsplines2D/lrsplines2d-doxymain.h File Reference	3621

30.2126	splines2D/include/GoTools/lrsplines2D/LRSplineSurface.h File Reference	3621
30.2127	splines2D/include/GoTools/lrsplines2D/LRSplineSurface_impl.h File Reference	3622
30.2128	splines2D/include/GoTools/lrsplines2D/LRSplineUtils.h File Reference	3623
30.2129	splines2D/include/GoTools/lrsplines2D/LRSurfApprox.h File Reference	3624
30.2130	splines2D/include/GoTools/lrsplines2D/LRSurfSmoothLS.h File Reference	3625
30.2131	splines2D/include/GoTools/lrsplines2D/LRSurfStitch.h File Reference	3626
30.2132	splines2D/include/GoTools/lrsplines2D/Mesh2D.h File Reference	3627
30.2133	splines2D/include/GoTools/lrsplines2D/Mesh2DIterator.h File Reference	3628
30.2134	splines2D/include/GoTools/lrsplines2D/Mesh2DUtils.h File Reference	3629
30.2135	wmat/app/example.cpp File Reference	3630
30.2135	Macro Definition Documentation	3631
30.2135.1	WANT_MATH	3631
30.2135.1	WANT_STREAM	3632
30.2135	Function Documentation	3632
30.2135.2	main()	3632
30.2135.2	test1(Real *y, Real *x1, Real *x2, int nobs, int npred)	3632
30.2135.2	test2(Real *y, Real *x1, Real *x2, int nobs, int npred)	3632
30.2135.2	test3(Real *y, Real *x1, Real *x2, int nobs, int npred)	3632
30.2135.2	test4(Real *y, Real *x1, Real *x2, int nobs, int npred)	3632
30.2135.2	test5(Real *y, Real *x1, Real *x2, int nobs, int npred)	3632
30.2136	wmat/app/garch.cpp File Reference	3633
30.2136	Macro Definition Documentation	3634
30.2136.1	WANT_FSTREAM	3634
30.2136.1	WANT_MATH	3634
30.2136.1	WANT_STREAM	3634
30.2136	Function Documentation	3634
30.2136.2	main()	3634
30.2136.2	Square(Real x)	3634
30.2137	wmat/app/nl_ex.cpp File Reference	3634
30.2137	Macro Definition Documentation	3635

30.2137.1.WANT_MATH	3635
30.2137.1.WANT_STREAM	3636
30.2137.Function Documentation	3636
30.2137.2.main()	3636
30.2138.00wmat/app/sl_ex.cpp File Reference	3636
30.2138.Macro Definition Documentation	3637
30.2138.1.WANT_MATH	3637
30.2138.1.WANT_STREAM	3637
30.2138.Function Documentation	3637
30.2138.2.main()	3637
30.2139.00wmat/app/test_exc.cpp File Reference	3637
30.2139.Macro Definition Documentation	3638
30.2139.1.WANT_STREAM	3638
30.2139.Function Documentation	3639
30.2139.2.main()	3639
30.2140.00wmat/app/tmt.cpp File Reference	3639
30.2140.Macro Definition Documentation	3640
30.2140.1.WANT_STREAM	3640
30.2140.Function Documentation	3640
30.2140.2.Clean(Matrix &A, Real c)	3640
30.2140.2.Clean(DiagonalMatrix &A, Real c)	3640
30.2140.2.main()	3640
30.2140.2.MCN("Number of matrices tested = ")	3640
30.2140.2.MCZ("Number of non-zero matrices (should be 1) = ")	3640
30.2140.2.BentiumCheck(Real N, Real D)	3640
30.2140.2.Fprint(const Matrix &X)	3641
30.2140.2.Bprint(const UpperTriangularMatrix &X)	3641
30.2140.2.Dprint(const DiagonalMatrix &X)	3641
30.2140.2.Pprint(const SymmetricMatrix &X)	3641
30.2140.2.Lprint(const LowerTriangularMatrix &X)	3641

30.2140.2.10	TestTypeAdd()	3641
30.2140.2.11	TestTypeConcat()	3641
30.2140.2.12	TestTypeKP()	3641
30.2140.2.13	TestTypeMulti()	3641
30.2140.2.14	TestTypeOrder()	3641
30.2140.2.15	TestTypeSP()	3642
30.2141	newmat/app/tmt1.cpp File Reference	3642
30.2141	Macro Definition Documentation	3642
30.2141.1	WANT_STREAM	3642
30.2141	Function Documentation	3643
30.2141.2	tfymat1()	3643
30.2142	newmat/app/tmt2.cpp File Reference	3643
30.2142	Function Documentation	3643
30.2142.1	tfymat2()	3643
30.2143	newmat/app/tmt3.cpp File Reference	3644
30.2143	Function Documentation	3644
30.2143.1	tfymat3()	3644
30.2144	newmat/app/tmt4.cpp File Reference	3645
30.2144	Function Documentation	3645
30.2144.1	tfymat4()	3645
30.2145	newmat/app/tmt5.cpp File Reference	3646
30.2145	Function Documentation	3646
30.2145.1	Returner0(const GenericMatrix &GM)	3646
30.2145.1	Returner1(const GenericMatrix &GM)	3647
30.2145.1	Returner2(const GenericMatrix &GM)	3647
30.2145.1	Returner3(const GenericMatrix &GM)	3647
30.2145.1	Returner4(const GenericMatrix &GM)	3647
30.2145.1	Returner5(const GenericMatrix &GM)	3647
30.2145.1	Returner6(const GenericMatrix &GM)	3647
30.2145.1	Returner7(const GenericMatrix &GM)	3647

30.2145.1	tymat5()	3647
30.2146	tymat/app/tmt6.cpp File Reference	3648
30.2146	Macro Definition Documentation	3648
30.2146.1	WANT_MATH	3648
30.2146	Function Documentation	3649
30.2146.2	tymat6()	3649
30.2147	tymat/app/tmt7.cpp File Reference	3649
30.2147	Function Documentation	3649
30.2147.1	tymat7()	3649
30.2148	tymat/app/tmt8.cpp File Reference	3650
30.2148	Function Documentation	3650
30.2148.1	TestReturn(const GeneralMatrix &gm)	3650
30.2148.1	Transposer(const GenericMatrix &GM1, GenericMatrix &GM2)	3651
30.2148.1	tymat8()	3651
30.2149	tymat/app/tmt9.cpp File Reference	3651
30.2149	Function Documentation	3652
30.2149.1	tymat9()	3652
30.2150	tymat/app/tmta.cpp File Reference	3652
30.2150	Function Documentation	3653
30.2150.1	tymata()	3653
30.2151	tymat/app/tmtb.cpp File Reference	3653
30.2151	Function Documentation	3654
30.2151.1	tymatb()	3654
30.2152	tymat/app/tmtc.cpp File Reference	3654
30.2152	Function Documentation	3654
30.2152.1	tymatc()	3654
30.2153	tymat/app/tmtd.cpp File Reference	3655
30.2153	Function Documentation	3655
30.2153.1	Inverter(const CroutMatrix &X)	3655
30.2153.1	tymatd()	3656

30.2154	wmat/app/tmte.cpp File Reference	3656
30.2154	Macro Definition Documentation	3657
30.2154.1	WANT_MATH	3657
30.2154	Function Documentation	3657
30.2154.2	ChecksSorted(const DiagonalMatrix &D, bool ascending=false)	3657
30.2154.2	ymate()	3657
30.2155	wmat/app/tmtf.cpp File Reference	3657
30.2155	Macro Definition Documentation	3658
30.2155.1	WANT_MATH	3658
30.2155	Function Documentation	3658
30.2155.2	tymatf()	3658
30.2156	wmat/app/tmtg.cpp File Reference	3658
30.2156	Macro Definition Documentation	3659
30.2156.1	WANT_MATH	3659
30.2156	Function Documentation	3660
30.2156.2	tymatg()	3660
30.2157	wmat/app/tmth.cpp File Reference	3660
30.2157	Function Documentation	3661
30.2157.1	BandFunctions(int l1, int u1, int l2, int u2)	3661
30.2157.1	LowerBandFunctions(int l1, int l2)	3661
30.2157.1	SymmetricBandFunctions(int l1, int l2)	3661
30.2157.1	ymath()	3661
30.2157.1	UpperBandFunctions(int u1, int u2)	3661
30.2158	wmat/app/tmti.cpp File Reference	3661
30.2158	Function Documentation	3662
30.2158.1	ReSizeMatrix(Matrix &A)	3662
30.2158.1	ymati()	3662
30.2158.1	WillNotConverge()	3663
30.2159	wmat/app/tmtj.cpp File Reference	3663
30.2159	Function Documentation	3663

30.2159.1	<code>tymatj()</code>	3663
30.2160	<code>wmat/app/tmtk.cpp</code> File Reference	3664
30.2160	Macro Definition Documentation	3664
30.2160.1	<code>WANT_STREAM</code>	3664
30.2160	Function Documentation	3665
30.2160.2	<code>tymatk()</code>	3665
30.2161	<code>wmat/app/tmtl.cpp</code> File Reference	3665
30.2161	Macro Definition Documentation	3666
30.2161.1	<code>WANT_MATH</code>	3666
30.2161.1	<code>WANT_STREAM</code>	3666
30.2161	Function Documentation	3666
30.2161.2	<code>TestMax(const GenericMatrix &GM)</code>	3666
30.2161.2	<code>tymatl()</code>	3666
30.2162	<code>wmat/app/tmtm.cpp</code> File Reference	3666
30.2162	Macro Definition Documentation	3667
30.2162.1	<code>WANT_MATH</code>	3667
30.2162.1	<code>WANT_STREAM</code>	3667
30.2162	Function Documentation	3667
30.2162.2	<code>tymatm()</code>	3667
30.2163	<code>wmat/include/boolean.h</code> File Reference	3667
30.2163	Macro Definition Documentation	3668
30.2163.1	<code>bool_LIB</code>	3668
30.2163	Variable Documentation	3668
30.2163.2	<code>false</code>	3668
30.2163.2	<code>true</code>	3668
30.2164	<code>wmat/include/controlw.h</code> File Reference	3668
30.2164	Macro Definition Documentation	3668
30.2164.1	<code>CONTROL_WORD_LIB</code>	3668
30.2165	<code>wmat/include/include.h</code> File Reference	3669
30.2165	Detailed Description	3669

30.2166	newmat/include/myexcept.h File Reference	3670
30.2166	Detailed Description	3671
30.2166	Macro Definition Documentation	3671
30.2166.2	Catch	3671
30.2166.2	CatchAll	3671
30.2166.2	CatchAndThrow	3671
30.2166.2	FREE_CHECK	3672
30.2166.2	MONITOR_INT_DELETE	3672
30.2166.2	MONITOR_INT_NEW	3672
30.2166.2	MONITOR_REAL_DELETE	3672
30.2166.2	MONITOR_REAL_NEW	3672
30.2166.2	NEW_DELETE	3672
30.2166.2	ReThrow	3672
30.2166.2	Throw	3672
30.2166.2	Try	3672
30.2167	newmat/include/newmat.h File Reference	3673
30.2167	Macro Definition Documentation	3675
30.2167.1	MatrixTypeUnSp	3675
30.2167.1	NEWMAT_LIB	3675
30.2167.1	ReturnMatrix	3675
30.2168	newmat/include/newmatap.h File Reference	3676
30.2168	Macro Definition Documentation	3677
30.2168.1	NEWMATAP_LIB	3677
30.2169	newmat/include/newmatio.h File Reference	3678
30.2169	Macro Definition Documentation	3679
30.2169.1	NEWMATIO_LIB	3679
30.2170	newmat/include/newmatnl.h File Reference	3679
30.2170	Macro Definition Documentation	3680
30.2170.1	NEWMATNL_LIB	3680
30.2171	newmat/include/newmatrc.h File Reference	3680

30.2171	Macro Definition Documentation	3681
30.2171.1	NEWMATRC_LIB	3681
30.2171	Enumeration Type Documentation	3681
30.2171.2	ISF	3681
30.2171	newmat/include/newmatrm.h File Reference	3681
30.2172	Macro Definition Documentation	3682
30.2172.1	NEWMATRM_LIB	3682
30.2172	Function Documentation	3682
30.2172.2	sign(Real x, Real y)	3682
30.2172.2	square(Real x)	3682
30.2171	newmat/include/precisio.h File Reference	3683
30.2173	Detailed Description	3684
30.2173	Macro Definition Documentation	3684
30.2173.2	PRECISION_LIB	3684
30.2173.2	WANT_MATH	3684
30.2171	newmat/include/solution.h File Reference	3684
30.2175	newmat/include/tmt.h File Reference	3685
30.2175	Function Documentation	3686
30.2175.1	Clean(Matrix &, Real)	3686
30.2175.1	Clean(DiagonalMatrix &, Real)	3686
30.2175.1	Print(const Matrix &X)	3686
30.2175.1	Print(const UpperTriangularMatrix &X)	3686
30.2175.1	Print(const DiagonalMatrix &X)	3686
30.2175.1	Print(const SymmetricMatrix &X)	3686
30.2175.1	Print(const LowerTriangularMatrix &X)	3687
30.2175.1	symat1()	3687
30.2175.1	symat2()	3687
30.2175.1	symat3()	3687
30.2175.1	symat4()	3687
30.2175.1	symat5()	3687

30.2175.119	format6()	3687
30.2175.119	format7()	3687
30.2175.119	format8()	3687
30.2175.119	format9()	3687
30.2175.119	formata()	3688
30.2175.119	formatb()	3688
30.2175.119	formatc()	3688
30.2175.119	formatd()	3688
30.2175.119	formate()	3688
30.2175.119	formatf()	3688
30.2175.119	formatg()	3688
30.2175.119	formati()	3688
30.2175.119	formatj()	3688
30.2175.119	formatk()	3689
30.2175.119	formatl()	3689
30.2175.119	formatm()	3689
30.2176	format/src/bandmat.cpp File Reference	3689
30.2176	Macro Definition Documentation	3690
30.2176.119	REPORT	3690
30.2176.119	WANT_MATH	3690
30.2176	Function Documentation	3690
30.2176.219	square(Real x)	3690
30.2177	format/src/cholesky.cpp File Reference	3690
30.2177	Macro Definition Documentation	3691
30.2177.119	REPORT	3691
30.2177.119	WANT_MATH	3691
30.2177	Function Documentation	3691
30.2177.219	Cholesky(const SymmetricMatrix &S)	3691
30.2177.219	Cholesky(const SymmetricBandMatrix &S)	3691

30.2177.2	Square(Real x)	3691
30.2177	src/evaluate.cpp File Reference	3692
30.2178	Macro Definition Documentation	3693
30.2178.1	REPORT	3693
30.2178.1	WANT_MATH	3693
30.2178	Function Documentation	3693
30.2178.2	EigenValues(const SymmetricMatrix &A, DiagonalMatrix &D, Matrix &Z)	3693
30.2178.2	EigenValues(const SymmetricMatrix &X, DiagonalMatrix &D)	3693
30.2178.2	EigenValues(const SymmetricMatrix &X, DiagonalMatrix &D, SymmetricMatrix &A)	3693
30.2179	src/fft.cpp File Reference	3693
30.2179	Macro Definition Documentation	3695
30.2179.1	REPORT	3695
30.2179.1	WANT_MATH	3695
30.2179	Function Documentation	3695
30.2179.2	DCT(const ColumnVector &U, ColumnVector &V)	3695
30.2179.2	DCT_II(const ColumnVector &U, ColumnVector &V)	3695
30.2179.2	DCT_II_inverse(const ColumnVector &V, ColumnVector &U)	3695
30.2179.2	DCT_inverse(const ColumnVector &V, ColumnVector &U)	3695
30.2179.2	DST(const ColumnVector &U, ColumnVector &V)	3695
30.2179.2	DST_II(const ColumnVector &U, ColumnVector &V)	3695
30.2179.2	DST_II_inverse(const ColumnVector &V, ColumnVector &U)	3695
30.2179.2	DST_inverse(const ColumnVector &V, ColumnVector &U)	3696
30.2179.2	FFT(const ColumnVector &U, const ColumnVector &V, ColumnVector &X, ColumnVector &Y)	3696
30.2179.2	IFT(const ColumnVector &U, const ColumnVector &V, ColumnVector &X, ColumnVector &Y)	3696
30.2179.2	RealFFT(const ColumnVector &U, ColumnVector &X, ColumnVector &Y)	3696
30.2179.2	RealIFT(const ColumnVector &A, const ColumnVector &B, ColumnVector &U)	3696
30.2180	src/hholder.cpp File Reference	3696
30.2180	Macro Definition Documentation	3697
30.2180.1	REPORT	3697

30.2180.1	<code>WANT_MATH</code>	3698
30.2180	Function Documentation	3698
30.2180.2	<code>QRZ(Matrix &X, UpperTriangularMatrix &U)</code>	3698
30.2180.2	<code>QRZ(const Matrix &X, Matrix &Y, Matrix &M)</code>	3698
30.2180.2	<code>QRZT(Matrix &X, LowerTriangularMatrix &L)</code>	3698
30.2180.2	<code>QRZT(const Matrix &X, Matrix &Y, Matrix &M)</code>	3698
30.2180.2	<code>square(Real x)</code>	3698
30.2180	<code>src/jacobi.cpp</code> File Reference	3698
30.2181	Macro Definition Documentation	3699
30.2181.1	<code>REPORT</code>	3699
30.2181.1	<code>WANT_MATH</code>	3700
30.2181	Function Documentation	3700
30.2181.2	<code>Jacobi(const SymmetricMatrix &X, DiagonalMatrix &D, SymmetricMatrix &A, Matrix &V, bool eivec)</code>	3700
30.2181.2	<code>Jacobi(const SymmetricMatrix &X, DiagonalMatrix &D)</code>	3700
30.2181.2	<code>Jacobi(const SymmetricMatrix &X, DiagonalMatrix &D, SymmetricMatrix &A)</code>	3700
30.2181.2	<code>Jacobi(const SymmetricMatrix &X, DiagonalMatrix &D, Matrix &V)</code>	3700
30.2181	<code>src/myexcept.cpp</code> File Reference	3700
30.2182	Detailed Description	3701
30.2182	Macro Definition Documentation	3701
30.2182.2	<code>WANT_STREAM</code>	3701
30.2182.2	<code>WANT_STRING</code>	3701
30.2182	Function Documentation	3701
30.2182.3	<code>Terminate()</code>	3701
30.2182	<code>src/newfft.cpp</code> File Reference	3702
30.2183	Macro Definition Documentation	3702
30.2183.1	<code>REPORT</code>	3702
30.2183.1	<code>WANT_MATH</code>	3703
30.2183.1	<code>WANT_STREAM</code>	3703
30.2183	Function Documentation	3703
30.2183.2	<code>square(Real x)</code>	3703

30.2183.2	square(int x)	3703
30.2184	src/newmat1.cpp File Reference	3703
30.2184	Macro Definition Documentation	3704
30.2184.1	REPORT	3704
30.2184	Function Documentation	3704
30.2184.2	Rectangular(MatrixType a, MatrixType b, MatrixType c)	3704
30.2185	src/newmat2.cpp File Reference	3704
30.2185	Macro Definition Documentation	3705
30.2185.1	MONITOR	3705
30.2185.1	REPORT	3705
30.2185.1	WANT_MATH	3705
30.2185	Function Documentation	3705
30.2185.2	DotProd(const MatrixRowCol &mrc1, const MatrixRowCol &mrc2)	3705
30.2186	src/newmat3.cpp File Reference	3706
30.2186	Macro Definition Documentation	3706
30.2186.1	MONITOR	3706
30.2186.1	REPORT	3706
30.2187	src/newmat4.cpp File Reference	3707
30.2187	Macro Definition Documentation	3707
30.2187.1	DO_SEARCH	3707
30.2187.1	REPORT	3708
30.2187	Function Documentation	3708
30.2187.2	Compare(const MatrixType &source, MatrixType &destination)	3708
30.2188	src/newmat5.cpp File Reference	3708
30.2188	Macro Definition Documentation	3709
30.2188.1	REPORT	3709
30.2189	src/newmat6.cpp File Reference	3709
30.2189	Macro Definition Documentation	3710
30.2189.1	REPORT	3710
30.2189	Function Documentation	3710

30.2189.2	<code>KP(const BaseMatrix &bm1, const BaseMatrix &bm2)</code>	3710
30.2189.2	<code>operator-(Real f, const BaseMatrix &bm)</code>	3710
30.2189.2	<code>SP(const BaseMatrix &bm1, const BaseMatrix &bm2)</code>	3710
30.2190	<code>newmat/src/newmat7.cpp</code> File Reference	3710
30.2190	Macro Definition Documentation	3711
30.2190.1	<code>REPORT</code>	3711
30.2190	Function Documentation	3711
30.2190.2	<code>Zero(const BaseMatrix &A)</code>	3711
30.2190.2	<code>operator==(const BaseMatrix &A, const BaseMatrix &B)</code>	3711
30.2190.2	<code>operator==(const GeneralMatrix &A, const GeneralMatrix &B)</code>	3711
30.2191	<code>newmat/src/newmat8.cpp</code> File Reference	3712
30.2191	Macro Definition Documentation	3712
30.2191.1	<code>REPORT</code>	3712
30.2191.1	<code>WANT_MATH</code>	3713
30.2191	Function Documentation	3713
30.2191.2	<code>DotProduct(const Matrix &A, const Matrix &B)</code>	3713
30.2191.2	<code>Square(Real x)</code>	3713
30.2192	<code>newmat/src/newmat9.cpp</code> File Reference	3713
30.2192	Macro Definition Documentation	3714
30.2192.1	<code>ios_format_flags</code>	3714
30.2192.1	<code>REPORT</code>	3714
30.2192.1	<code>WANT_FSTREAM</code>	3714
30.2192	Function Documentation	3714
30.2192.2	<code>operator<<(ostream &s, const BaseMatrix &X)</code>	3714
30.2192.2	<code>operator<<(ostream &s, const GeneralMatrix &X)</code>	3714
30.2193	<code>newmat/src/newmatex.cpp</code> File Reference	3715
30.2193	Macro Definition Documentation	3715
30.2193.1	<code>WANT_STREAM</code>	3715
30.2193	Function Documentation	3716
30.2193.2	<code>MatrixErrorNoSpace(void *v)</code>	3716

30.2194	wmat/src/newmatnl.cpp File Reference	3716
30.2194	Macro Definition Documentation	3717
30.2194.1	WANT_MATH	3717
30.2194.1	WANT_STREAM	3717
30.2195	wmat/src/newmatrm.cpp File Reference	3717
30.2195	Macro Definition Documentation	3718
30.2195.1	REPORT	3718
30.2195	Function Documentation	3718
30.2195.2	ComplexScale(RectMatrixCol &U, RectMatrixCol &V, Real x, Real y)	3718
30.2195.2	Rotate(RectMatrixCol &U, RectMatrixCol &V, Real tau, Real s)	3718
30.2196	wmat/src/solution.cpp File Reference	3718
30.2196	Macro Definition Documentation	3719
30.2196.1	WANT_MATH	3719
30.2196.1	WANT_STREAM	3719
30.2196	Function Documentation	3720
30.2196.2	square(Real x)	3720
30.2197	wmat/src/sort.cpp File Reference	3720
30.2197	Macro Definition Documentation	3721
30.2197.1	DoSimpleSort	3721
30.2197.1	MaxDepth	3721
30.2197.1	REPORT	3721
30.2197.1	WANT_MATH	3721
30.2197	Function Documentation	3721
30.2197.2	SortAscending(GeneralMatrix &GM)	3721
30.2197.2	SortDescending(GeneralMatrix &GM)	3721
30.2197.2	SortSV(DiagonalMatrix &D, Matrix &U, bool ascending)	3721
30.2197.2	SortSV(DiagonalMatrix &D, Matrix &U, Matrix &V, bool ascending)	3721
30.2198	wmat/src/submat.cpp File Reference	3722
30.2198	Macro Definition Documentation	3722
30.2198.1	REPORT	3722

30.2199	swmat/src/svd.cpp File Reference	3722
30.2199	Macro Definition Documentation	3723
30.2199.1	REPORT	3723
30.2199.1	WANT_MATH	3724
30.2199	Function Documentation	3724
30.2199.2	SVD(const Matrix &A, DiagonalMatrix &Q, Matrix &U, Matrix &V, bool withU, bool withV)	3724
30.2199.2	SVD(const Matrix &A, DiagonalMatrix &D)	3724
30.2200	parametrization/include/GoTools/parametrization/parametrization-doxymain.h File Reference	3724
30.2201	parametrization/include/GoTools/parametrization/PrBiCGStab.h File Reference	3724
30.2202	parametrization/include/GoTools/parametrization/PrCellStructure.h File Reference	3725
30.2203	parametrization/include/GoTools/parametrization/PrCG.h File Reference	3726
30.2204	parametrization/include/GoTools/parametrization/PrDijkstra.h File Reference	3726
30.2204	Macro Definition Documentation	3728
30.2204.1	GraphType	3728
30.2204.2	Typedef Documentation	3729
30.2204.2	HeapType	3729
30.2205	parametrization/include/GoTools/parametrization/PrExplicitConnectivity.h File Reference	3729
30.2206	parametrization/include/GoTools/parametrization/PrFaber_F.h File Reference	3730
30.2207	parametrization/include/GoTools/parametrization/PrFastUnorganized_OP.h File Reference	3730
30.2208	parametrization/include/GoTools/parametrization/PrFilterbank.h File Reference	3731
30.2209	parametrization/include/GoTools/parametrization/PrGeodesics.h File Reference	3732
30.2209	Macro Definition Documentation	3734
30.2209.1	EPS	3734
30.2209	Function Documentation	3734
30.2209.2	decreasing_length(const int nb_steps, const vector< double > &lengths)	3734
30.2209.2	DijkstraPT(PrTriangulation_OP &triangulation, int sce_vertex, int dest_vertex)	3734
30.2209.2	dist2D(Vector2D &a, Vector2D &b)	3734
30.2209.2	dist3D(Vector3D &a, Vector3D &b)	3734
30.2209.2	GeodesicPT(PrTriangulation_OP &triangulation, int sce_vertex, int dest_vertex, vector< Vector3D > ¤t_path, vector< int > &tr_seq3d)	3735

30.2209.210	length_polygonal_path(const vector< Vector3D > &path)	3735
30.2209.211	next_pivot(std::list< int > &list_pivots, std::list< int >::iterator &iter_pivot, int prec_pivot)	3735
30.2209.212	pivot_in_new_list(std::list< int > &list_pivots, std::vector< int >::const_iterator &tried_begin, std::vector< int >::const_iterator &tried_end, std::list< int >::iterator &iter_pivot)	3735
30.2209.213	pivot_in_new_list(std::list< int > &list_pivots, const std::list< int > &list_pivots, int prec, int nb_steps, const vector< double > &lengths, std::list< int >::iterator &iter_pivot, int pivot)	3735
30.2209.214	point3D_from_indice(PrTriangulation_OP &t, const vector< int > &ind_path)	3735
30.2209.215	pr(double x)	3735
30.2209.216	tr_crossed_by_path_vertices(PrTriangulation_OP &t, vector< int > &vert_path)	3735
30.2209.217	tr_sequence_around_vertex(PrTriangulation_OP &t, int vertex)	3736
30.2209.218	tr_sequence_around_vertex(PrTriangulation_OP &t, int vertex, int first_tr)	3736
30.2209.219	update_triangle_sequence_around_pivot(PrTriangulation_OP &t, vector< int > &tr_seq, int pivot)	3736
30.2210	parametrization/include/GoTools/parametrization/PrHeap.h File Reference	3736
30.2211	parametrization/include/GoTools/parametrization/PrInterface.h File Reference	3737
30.2211	Function Documentation	3739
30.2211.11	print(vector< int > &v)	3739
30.2211.12	print(vector< double > &v)	3739
30.2211.13	print(vector< Vector3D > &v)	3739
30.2211.14	print(vector< Vector2D > &v)	3739
30.2211.15	print(vector< PrTriangle > &t)	3739
30.2211.16	print_edges_lengths(vector< int > tr_seq, PrTriangulation_OP &t)	3739
30.2211.17	print_lengths(const vector< double > &lengths)	3739
30.2211.18	printGeomviewPath(vector< Vector3D > &path)	3739
30.2211.19	printGeomviewPath(std::ofstream &os, vector< Vector3D > &path)	3739
30.2211.110	printGeomviewTriangleSequence(PrTriangulation_OP &t, vector< int > tr_seq)	3739
30.2211.111	printGeomviewTriangleSequence(std::ofstream &os, PrTriangulation_OP &t, vector< int > tr_seq)	3739
30.2211.112	printGeomviewTriangulation(PrTriangulation_OP &t)	3739
30.2211.113	printGeomviewTriangulation(std::ofstream &os, PrTriangulation_OP &t)	3740

30.2212	parametrization/include/GoTools/parametrization/PrLevelTriangulation_OP.h File Reference	3740
30.2213	parametrization/include/GoTools/parametrization/PrMat.h File Reference	3741
30.2214	parametrization/include/GoTools/parametrization/PrMatrix.h File Reference	3742
30.2215	parametrization/include/GoTools/parametrization/PrMatSparse.h File Reference	3743
30.2216	parametrization/include/GoTools/parametrization/PrMultiDijkstra.h File Reference	3745
30.2216	Macro Definition Documentation	3745
30.2216.1	GraphType	3745
30.2216.1	Point	3746
30.2216.2	Typedef Documentation	3746
30.2216.2	HeapType2	3746
30.2217	parametrization/include/GoTools/parametrization/PrNestedNode.h File Reference	3746
30.2218	parametrization/include/GoTools/parametrization/PrNestedTriangulation.h File Reference	3747
30.2219	parametrization/include/GoTools/parametrization/PrNode.h File Reference	3748
30.2220	parametrization/include/GoTools/parametrization/PrOrganizedPoints.h File Reference	3749
30.2221	parametrization/include/GoTools/parametrization/PrParametrizeBdy.h File Reference	3750
30.2221	Enumeration Type Documentation	3751
30.2221.1	PrBdyParamKind	3751
30.2222	parametrization/include/GoTools/parametrization/PrParametrizeInt.h File Reference	3752
30.2222	Enumeration Type Documentation	3752
30.2222.1	PrParamStartVector	3752
30.2223	parametrization/include/GoTools/parametrization/PrParametrizeMesh.h File Reference	3753
30.2224	parametrization/include/GoTools/parametrization/PrParamTriangulation.h File Reference	3753
30.2225	parametrization/include/GoTools/parametrization/PrParamUtil.h File Reference	3755
30.2225	Function Documentation	3755
30.2225.1	area(const double &x0, const double &y0, const double &x1, const double &y1, const double &x2, const double &y2)	3755
30.2225.1	area(const Vector3D &a, const Vector3D &b, const Vector3D &c)	3755
30.2225.1	BaryCoords(double x, double y, double x0, double y0, double x1, double y1, double x2, double y2, double &tau0, double &tau1, double &tau2)	3756
30.2225.1	BaryCoords0(double &u0, double &v0, double &u1, double &v1, double &u2, double &v2, double &tau0, double &tau1, double &tau2)	3756
30.2225.1	Contangent(const Vector3D &a, const Vector3D &b, const Vector3D &c)	3756

30.2225.1	Get(const double &u1, const double &v1, const double &u2, const double &v2)	3756
30.2225.1	polarCoords(Vector2D v, double &r, double &theta)	3756
30.2225.1	polarCoords(double u, double v, double &r, double &theta)	3756
30.2225.1	TanThetaOverTwo(const Vector3D &a, const Vector3D &b, const Vector3D &c)	3756
30.2226	Parametrization/include/GoTools/parametrization/PrPathTriangleSeq.h File Reference	3756
30.2226	Function Documentation	3759
30.2226.1	abs_val(double a)	3759
30.2226.1	angle(const Go::Vector2D &a, const Go::Vector2D &b, const Go::Vector2D &c, const Go::Vector2D &d)	3759
30.2226.1	axial_symmetry(const Go::Vector2D c, const Go::Vector2D a, const Go::Vector2D b, Go::Vector2D &v)	3759
30.2226.1	common_edge(PrTriangle t1, PrTriangle t2)	3759
30.2226.1	deviation_from_pivots(const list< int > &list_pivots_unf, const vector< UnfNodeType > &nodes_unf)	3759
30.2226.1	edge_sequence(const std::vector< int > &tr_seq, const PrTriangulation_OP &t)	3760
30.2226.1	edge_sequence(std::vector< PrTriangle > tr_seq)	3760
30.2226.1	length_polygonal_path(const list< int > &list_pivots_unf, vector< UnfNodeType > &nodes_unf)	3760
30.2226.1	line_intersection(Go::Vector2D a, Go::Vector2D b, Go::Vector2D c, Go::Vector2D d, Go::Vector2D &i)	3760
30.2226.1	local_coordinates(const Go::Vector3D a, const Go::Vector3D &U, const Go::Vector3D &V, const Go::Vector3D &W)	3760
30.2226.1	local_frame(Go::Vector3D a, Go::Vector3D b, Go::Vector3D c, Go::Vector3D &U, Go::Vector3D &V, Go::Vector3D &W)	3760
30.2226.1	path_2Dpts_from_ratio_on_edge(const vector< UnfNodeType > &nodes_unf, vector< Vector2D > &path, const vector< double > &ratio_on_edge, const vector< EdgeType > &edge_seq2d)	3760
30.2226.1	path_3Dpts_from_ratio_on_edge(const PrTriangulation_OP &t, std::vector< Go::Vector3D > &sh_path, const std::vector< double > &ratio_on_edge, const std::vector< EdgeType > &edge_seq3d)	3760
30.2226.1	path_ratio_on_edge_from_pivots(std::list< int > &list_pivots_unf, std::vector< EdgeType > &edge_seq_unf, const std::vector< UnfNodeType > &nodes_unf)	3760
30.2226.1	path_to_edge(const int sce_vertex_unf, int i, std::vector< PathType > &p, const std::vector< UnfNodeType > &nodes_unf, const std::vector< PrTriangle > &tr_seq_unf, std::vector< EdgeType > &edge_seq_unf)	3761
30.2226.1	pivots3D_from_pivots_in_unf_seq(const list< int > &list_pivots_unf, const vector< UnfNodeType > &nodes_unf)	3761

30.2226.1	print_deviation(std::list< double > &list_deviation)	3761
30.2226.1	print_edges_lengths(std::vector< PrTriangle > tr, std::vector< UnfNodeType > nodes_unf)	3761
30.2226.1	print_list_int(std::list< int > &list_pivots_unf)	3761
30.2226.1	print_tr_vertices(std::vector< int > &tr, PrTriangulation_OP &t)	3761
30.2226.1	print_triangle_sequence(std::vector< PrTriangle > tr, std::vector< UnfNodeType > nodes_unf)	3761
30.2226.1	printNode(vector< UnfNodeType > &v)	3761
30.2226.1	printUnfoldedObjects(std::ofstream &os, std::vector< UnfNodeType > &nodes_unf, std::vector< PrTriangle > &tr_seq_unf, std::list< int > &list_pivots_unf)	3761
30.2226.1	printUnfoldedPath(std::ofstream &os, std::vector< UnfNodeType > &nodes_unf, std::list< int > &list_pivots_unf)	3761
30.2226.1	printUnfoldedPath(std::ofstream &os, std::vector< Go::Vector2D > path)	3761
30.2226.1	printUnfoldedSequence(std::ofstream &os, std::vector< UnfNodeType > &nodes_unf, std::vector< PrTriangle > &tr_seq_unf)	3761
30.2226.1	ratio2D_from_ratio_on_edge(double r, Vector2D a, Vector2D b)	3761
30.2226.1	ratio3D_from_ratio_on_edge(double r, Go::Vector3D A, Go::Vector3D B)	3762
30.2226.1	rotation2D(const Go::Vector2D a, const double theta, const Go::Vector2D c, Go::Vector2D &vect)	3762
30.2226.1	same_side(const Go::Vector2D pt1, const Go::Vector2D pt2, const Go::Vector2D ptA, const Go::Vector2D ptB)	3762
30.2226.1	segment_intersection(Go::Vector2D a, Go::Vector2D b, Go::Vector2D c, Go::Vector2D d)	3762
30.2226.1	shortest_path_triangle_sequence(const PrTriangulation_OP &t, int sce_vertex, int dest_vertex, const vector< int > &tr_seq3d, list< int > &list_pivots, double &length)	3762
30.2226.1	shortest_path_triangle_sequence(const PrTriangulation_OP &t, const int &sce_vertex, const int &dest_vertex, vector< int > &tr_seq3d, list< int > &list_pivots, std::vector< EdgeType > &edge_seq3d, std::vector< double > &ratio_on_edge, double &length)	3762
30.2226.1	shortest_path_triangle_sequence_2D(const PrTriangulation_OP &t, const int sce_vertex, const int dest_vertex, const int sce_vertexUnf, int dest_vertexUnf, const std::vector< int > &tr_seq3d, std::vector< PrTriangle > &tr_seq_unf, std::list< int > &list_pivots_unf, std::vector< UnfNodeType > nodes_unf, double &length)	3762
30.2226.1	isrd_vertex(PrTriangle Tr, EdgeType e)	3763
30.2226.1	unfolding_first_triangle(const PrTriangulation_OP &t, const vector< int > &tr_seq, vector< PrTriangle > &tr_seq_unf, vector< UnfNodeType > &nodes_unf)	3763

30.2226.1.37	<code>folding_triangle_sequence(const PrTriangulation_OP &t, const vector< int > &tr_seq3d, vector< PrTriangle > &tr_seq_unf, const vector< EdgeType > &edge_seq3d, vector< EdgeType > &edge_seq_unf, vector< UnfNodeType > &nodes_unf)</code>	3763
30.2226.1.38	<code>folding_vertex(const PrTriangulation_OP &t, const vector< int > &tr_seq3d, vector< PrTriangle > &tr_seq_unf, const int vertex, int &vertex_unf)</code>	3763
30.2226.1.39	<code>dot_prod2D(Go::Vector2D a, Go::Vector2D b, Go::Vector2D c, Go::Vector2D d)</code> .	3763
30.2226.1.40	<code>Vector_U(const Go::Vector3D W)</code>	3763
30.2226.1.41	<code>Vector_V(const Go::Vector3D W, const Go::Vector3D U)</code>	3763
30.2226.1.42	<code>Vector_W(Go::Vector3D a, Go::Vector3D b, Go::Vector3D c)</code>	3763
30.2227	Parametrization/include/GoTools/parametrization/PrPGNode.h File Reference	3764
30.2228	Parametrization/include/GoTools/parametrization/PrPlanarGraph_OP.h File Reference	3765
30.2229	Parametrization/include/GoTools/parametrization/PrPrewavelet_F.h File Reference	3765
30.2230	Parametrization/include/GoTools/parametrization/PrPrmEDDHLS.h File Reference	3766
30.2231	Parametrization/include/GoTools/parametrization/PrPrmExperimental.h File Reference	3767
30.2232	Parametrization/include/GoTools/parametrization/PrPrmLeastSquare.h File Reference	3767
30.2233	Parametrization/include/GoTools/parametrization/PrPrmMeanValue.h File Reference	3768
30.2234	Parametrization/include/GoTools/parametrization/PrPrmShpPres.h File Reference	3769
30.2235	Parametrization/include/GoTools/parametrization/PrPrmSurface.h File Reference	3770
30.2236	Parametrization/include/GoTools/parametrization/PrPrmSymMeanValue.h File Reference	3771
30.2237	Parametrization/include/GoTools/parametrization/PrPrmUniform.h File Reference	3772
30.2238	Parametrization/include/GoTools/parametrization/PrPrmWachspres.h File Reference	3773
30.2239	Parametrization/include/GoTools/parametrization/PrRectangularGrid_OP.h File Reference	3774
30.2240	Parametrization/include/GoTools/parametrization/PrSubTriangulation.h File Reference	3775
30.2241	Parametrization/include/GoTools/parametrization/PrTexture.h File Reference	3777
30.2241	Function Documentation	3777
30.2241.1	<code>printTexture(std::ostream &os, PrOrganizedPoints &triang)</code>	3777
30.2242	Parametrization/include/GoTools/parametrization/PrThin.h File Reference	3777
30.2243	Parametrization/include/GoTools/parametrization/PrThreshold.h File Reference	3778
30.2244	Parametrization/include/GoTools/parametrization/PrTriangle.h File Reference	3778
30.2245	Parametrization/include/GoTools/parametrization/PrTriangulation_NT.h File Reference	3779
30.2246	Parametrization/include/GoTools/parametrization/PrTriangulation_OP.h File Reference	3780

30.2247	parametrization/include/GoTools/parametrization/PrUnorganized_OP.h File Reference	3781
30.2248	parametrization/include/GoTools/parametrization/PrVec.h File Reference	3781
30.2249	parametrization/include/GoTools/parametrization/PrWaveletUtil.h File Reference	3782
30.2249	Function Documentation	3783
30.2249.1	theta(int j, int k, int i, int deg, bool isBoundary)	3783
30.2250	qualitymodule/include/GoTools/qualitymodule/FaceSetQuality.h File Reference	3783
30.2251	qualitymodule/include/GoTools/qualitymodule/FaceSetRepair.h File Reference	3783
30.2252	qualitymodule/include/GoTools/qualitymodule/ModelQuality.h File Reference	3784
30.2253	qualitymodule/include/GoTools/qualitymodule/ModelRepair.h File Reference	3785
30.2254	qualitymodule/include/GoTools/qualitymodule/qualitymodule-doxymain.h File Reference	3786
30.2255	qualitymodule/include/GoTools/qualitymodule/QualityResults.h File Reference	3786
30.2256	qualitymodule/include/GoTools/qualitymodule/QualityUtils.h File Reference	3787
30.2257	qualitymodule/include/GoTools/qualitymodule/testSuite.h File Reference	3788
30.2257	Macro Definition Documentation	3789
30.2257.1	TEST_SUITE_SIZE	3789
30.2258	isl/examples/example01.cpp File Reference	3789
30.2258	Function Documentation	3789
30.2258.1	main(int avnum, char **vararg)	3789
30.2259	isl/examples/example02.cpp File Reference	3790
30.2259	Function Documentation	3790
30.2259.1	main(int avnum, char **vararg)	3790
30.2260	isl/examples/example03.cpp File Reference	3790
30.2260	Function Documentation	3791
30.2260.1	main(int avnum, char **vararg)	3791
30.2261	isl/examples/example04.cpp File Reference	3791
30.2261	Function Documentation	3792
30.2261.1	main(int avnum, char **vararg)	3792
30.2262	isl/examples/example05.cpp File Reference	3792
30.2262	Function Documentation	3792
30.2262.1	main(int avnum, char **vararg)	3792

30.2263	30.2263/examples/example06.cpp File Reference	3793
	30.2263 Function Documentation	3793
	30.2263.1 main(int avnum, char **vararg)	3793
30.2264	30.2264/examples/example07.cpp File Reference	3793
	30.2264 Function Documentation	3794
	30.2264.1 main(int avnum, char **vararg)	3794
30.2265	30.2265/examples/example08.cpp File Reference	3794
	30.2265 Function Documentation	3795
	30.2265.1 main(int avnum, char **vararg)	3795
30.2266	30.2266/examples/example09.cpp File Reference	3795
	30.2266 Function Documentation	3795
	30.2266.1 main(int avnum, char **vararg)	3795
30.2267	30.2267/examples/example10.cpp File Reference	3796
	30.2267 Function Documentation	3796
	30.2267.1 main(int avnum, char **vararg)	3796
30.2268	30.2268/examples/example11.cpp File Reference	3796
	30.2268 Function Documentation	3797
	30.2268.1 main(int avnum, char **vararg)	3797
30.2269	30.2269/examples/example12.cpp File Reference	3797
	30.2269 Function Documentation	3798
	30.2269.1 main(int avnum, char **vararg)	3798
30.2270	30.2270/examples/example13.cpp File Reference	3798
	30.2270 Function Documentation	3798
	30.2270.1 main(int avnum, char **vararg)	3798
30.2271	30.2271/examples/example14.cpp File Reference	3799
	30.2271 Function Documentation	3799
	30.2271.1 main(int avnum, char **vararg)	3799
30.2272	30.2272/examples/example15.cpp File Reference	3799
	30.2272 Function Documentation	3800
	30.2272.1 main(int varnum, char **vararg)	3800

30.2273	/examples/streaming/include/GoReadWrite.h File Reference	3800
30.2273	Function Documentation	3801
30.2273.1	readGoCurve(std::istream &go_stream)	3801
30.2273.1	readGoPoints(std::vector< double > &coords, std::istream &go_stream)	3801
30.2273.1	readGoSurface(std::istream &go_stream)	3801
30.2273.1	writeGoCurve(SISLCurve *curve, std::ostream &go_stream)	3801
30.2273.1	writeGoPoints(int num_points, double *coords, std::ostream &go_stream)	3801
30.2273.1	writeGoSurface(SISLSurf *surf, std::ostream &go_stream)	3801
30.2274	/examples/streaming/src/GoReadWrite.cpp File Reference	3801
30.2274	Function Documentation	3802
30.2274.1	readGoCurve(istream &go_stream)	3802
30.2274.1	readGoPoints(std::vector< double > &coords, std::istream &go_stream)	3802
30.2274.1	readGoSurface(istream &go_stream)	3802
30.2274.1	writeGoCurve(SISLCurve *curve, std::ostream &go_stream)	3802
30.2274.1	writeGoPoints(int num_points, double *coords, std::ostream &go_stream)	3802
30.2274.1	writeGoSurface(SISLSurf *surf, ostream &go_stream)	3802
30.2275	/examples/viewer/include/aux2.h File Reference	3803
30.2275	Macro Definition Documentation	3804
30.2275.1	_ERRORMACROS_H	3804
30.2275.1	ASSERT	3804
30.2275.1	ASSERT2	3805
30.2275.1	ASSERT3	3805
30.2275.1	AUX2_H_INCLUDED	3805
30.2275.1	CRIT_ERR	3805
30.2275.1	DEQUAL	3805
30.2275.1	DEQUAL2	3805
30.2275.1	DEQUAL3	3805
30.2275.1	DEQUAL4	3805
30.2275.1	DEQUALX	3805
30.2275.1	DGREATER	3806

30.2275.1	D GREATEREQ	3806
30.2275.1	D GREATEREQ2	3806
30.2275.1	D LESS	3806
30.2275.1	D LESSEQ	3806
30.2275.1	D LESSEQ2	3806
30.2275.1	D NEQUAL	3806
30.2275.1	D NEQUAL2	3806
30.2275.1	D ERROR_IF	3806
30.2275.1	D MAX	3806
30.2275.1	D MESSAGE	3807
30.2275.1	D MIN	3807
30.2275.1	D NEQUAL	3807
30.2275.1	D NEQUAL2	3807
30.2275.1	D PI	3807
30.2275.1	D REPORT	3807
30.2275.1	D SQR	3807
30.2275.1	D TROW	3807
30.2275.2	Function Documentation	3808
30.2275.2	d eps_equal(const double &a, const double &b)	3808
30.2275.2	d eps_greater(const double &a, const double &b)	3808
30.2275.2	d eps_greater_eq(const double &a, const double &b)	3808
30.2275.2	d eps_less(const double &a, const double &b)	3808
30.2275.2	d eps_less_eq(const double &a, const double &b)	3808
30.2275.2	d fc(void)	3808
30.2275.2	d fc(void)	3808
30.2275.3	/examples/viewer/include/gl_aux.h File Reference	3808
30.2276	Function Documentation	3809
30.2276.1	d draw_cylinder(double x0, double y0, double z0, double x1, double y1, double z1, double radius, double radius2, int n)	3809
30.2276.1	d draw_gl_axes_old(int n=10, double r=0.7, double radius=0.01, double rim=0.04, double l=0.1)	3810

30.2276.1	<code>draw_grid(const int n1, const int n2, const int n3)</code>	3810
30.2276.1	<code>draw_grid_planes(const int n1, const int n2, const int n3)</code>	3810
30.2276.1	<code>gl_init(int argc, char *argv[], const int xs, const int ys, const int x0, const int y0, const int doubleBuffer, const int two_sided, const int lighting, const int normalize, const int smooth, const double xtrans, const double ytrans, const double ztrans, const double xscale, const double yscale, const double zscale, int &tx, int &ty, const int texture_mode, const char *const texfile=NULL)</code>	3810
30.2276.1	<code>print_gl_matrix(const int m)</code>	3810
30.2276.1	<code>read_gl_matrices(FILE *f)</code>	3810
30.2276.1	<code>reshape_window(int width, int height)</code>	3810
30.2276.1	<code>write_gl_matrices(FILE *f)</code>	3810
30.2276	Variable Documentation	3810
30.2276.2	<code>gl_setting_back</code>	3810
30.2276.2	<code>gl_setting_back_old</code>	3810
30.2276.2	<code>gl_preset_col_table</code>	3810
30.2276.2	<code>gl_predefined_colours</code>	3811
30.2276	<code>gl/examples/viewer/include/glutils.h</code> File Reference	3811
30.2277	Macro Definition Documentation	3812
30.2277.1	<code>ASSERT_GL</code>	3812
30.2277.1	<code>assert_gl</code>	3812
30.2277.1	<code>ASSERT_GL_DONT_EXIT</code>	3812
30.2277.1	<code>GLUTILS_H_INCLUDED</code>	3812
30.2277	Function Documentation	3812
30.2277.2	<code>assert_gl_dummy_and_empty(void)</code>	3812
30.2277.2	<code>assert_gl_m(int line, char *file, const bool do_exit=true)</code>	3812
30.2277.2	<code>draw_sphere(const vector3t< float > &pos, const float r, const int n, const int slot, const vector3t< float > &col, const bool wiremode)</code>	3813
30.2277.2	<code>list_FBConfigs(GLXFBConfig *config, int nelements)</code>	3813
30.2277	<code>gl/examples/viewer/include/jonvec.h</code> File Reference	3813
30.2277	<code>gl/examples/viewer/include/mouse.h</code> File Reference	3814
30.2279	Macro Definition Documentation	3814
30.2279.1	<code>MOUSE_H_INCLUDED</code>	3814
30.2279	Function Documentation	3815

30.2279.2	Mouse(int butt, int state, int x, int y)	3815
30.2279.2	MouseRotate(int x, int y)	3815
30.2279.2	MouseTranslate(int x, int y)	3815
30.2279.2	MouseZoom(int x, int y)	3815
30.2279.2	transversal_rotation(int x, int y, int last_xx, int last_yy)	3815
30.2279	Variable Documentation	3815
30.2279.3	allow_zrot	3815
30.2279.3	last_x	3815
30.2279.3	last_x0	3815
30.2279.3	last_y	3815
30.2279.3	last_y0	3816
30.2279.3	mouse_movement	3816
30.2280	sisl/examples/viewer/include/sisl_aux.h File Reference	3816
30.2280	Macro Definition Documentation	3817
30.2280.1	SISL_AUX_H_INCLUDED	3817
30.2280	Function Documentation	3817
30.2280.2	compute_surface_normals(SISLSurf *srf, double **ngrid)	3817
30.2280.2	lower_degree_and_subdivide(SISLSurf **srf, int new_knots_per_interval, int max_number_of_knots)	3817
30.2280.2	lower_degree_to_linear(SISLSurf **srf, double e)	3817
30.2281	sisl/examples/viewer/include/transfutils.h File Reference	3817
30.2281	Macro Definition Documentation	3818
30.2281.1	TRANSFUTILS_H_INCLUDED	3818
30.2281	Function Documentation	3818
30.2281.2	rotate(double y_ang, double x_ang, double z_ang)	3818
30.2281.2	scale(double x, double y, double z)	3818
30.2281.2	translate(double x, double y, double z)	3818
30.2281	Variable Documentation	3819
30.2281.3	xrot	3819
30.2281.3	xrot_eps	3819
30.2281.3	xscale	3819

30.2281.3	x trans	3819
30.2281.3	y rot	3819
30.2281.3	y rot_eps	3819
30.2281.3	y scale	3819
30.2281.3	y trans	3819
30.2281.3	z rot	3819
30.2281.3	z rot_eps	3819
30.2281.3	z scale	3820
30.2281.3	z trans	3820
30.2282	3 /examples/viewer/sisl_view_demo.cpp File Reference	3820
30.2282	Macro Definition Documentation	3821
30.2282.1	C TRL_STRING	3821
30.2282.1	C URVE_EVALUATIONS	3821
30.2282.1	E SC_STRING	3821
30.2282.1	M AX_CURVES	3821
30.2282.1	M AX_LINES	3821
30.2282.1	M AX_SURFACES	3822
30.2282.1	T AB_STRING	3822
30.2282	Function Documentation	3822
30.2282.2	d raw_all_curves(void)	3822
30.2282.2	m ain(int argc, char *argv[])	3822
30.2282	Variable Documentation	3822
30.2282.3	a xis_length	3822
30.2282.3	a xis_thickness	3822
30.2282.3	b ackground_col	3822
30.2282.3	c ol_setting	3822
30.2282.3	c ol_x_setting_back	3823
30.2282.3	c urve	3823
30.2282.3	c urve_enabled	3823
30.2282.3	c urve_highlights	3823

30.2282.3	curve_name	3823
30.2282.3	curves	3823
30.2282.3	discr_curve	3823
30.2282.3	draw_axes	3823
30.2282.3	draw_edges	3823
30.2282.3	frames_without_movement	3824
30.2282.3	hit_key_string	3824
30.2282.3	marker_size	3824
30.2282.3	normal	3824
30.2282.3	cloud	3824
30.2282.3	predef_colours	3824
30.2282.3	selected_curve	3824
30.2282.3	selected_surface	3824
30.2282.3	ztf_enabled	3824
30.2282.3	ztf	3824
30.2282.3	ztf_face_highlights	3825
30.2282.3	ztf_face_name	3825
30.2282.3	ztf_faces	3825
30.2282.3	ztf_col	3825
30.2283	/examples/viewer/src/aux2.cpp File Reference	3825
30.2283	Function Documentation	3826
30.2283.1	tlc(void)	3826
30.2283.1	tlc(void)	3826
30.2283	Variable Documentation	3826
30.2283.2	join_timer	3826
30.2284	/examples/viewer/src/gl_aux.cpp File Reference	3826
30.2284	Macro Definition Documentation	3827
30.2284.1	PI	3827
30.2284	Function Documentation	3827
30.2284.2	draw_cylinder(double x0, double y0, double z0, double x1, double y1, double z1, double radius, double radius2, int n)	3827

30.2284.2	draw_gl_axes_old(int n, double r, double radius, double rim, double l)	3827
30.2284.2	draw_grid(const int n1, const int n2, const int n3)	3827
30.2284.2	draw_grid_planes(const int n1, const int n2, const int n3)	3828
30.2284.2	gl_init(int argc, char *argv[], const int xs, const int ys, const int x0, const int y0, const int doubleBuffer, const int two_sided, const int lighting, const int normalize, const int smooth, const double xtrans, const double ytrans, const double ztrans, const double xscale, const double yscale, const double zscale, int &tx, int &ty, const int texture_mode, const char *const texfile)	3828
30.2284.2	print_gl_matrix(const int m)	3828
30.2284.2	read_gl_matrices(FILE *f)	3828
30.2284.2	reshape_window(int width, int height)	3828
30.2284.2	transpose_matrix(double *const d)	3828
30.2284.2	write_gl_matrices(FILE *f)	3828
30.2284.3	Variable Documentation	3828
30.2284.3	predef_col_table	3828
30.2284.3	predefined_colours	3829
30.2285	examples/viewer/src/glutils.cpp File Reference	3829
30.2285	Macro Definition Documentation	3829
30.2285.1	M_PI	3829
30.2285.2	Function Documentation	3830
30.2285.2	assert_gl_dummy_and_empty(void)	3830
30.2285.2	assert_gl_m(int line, char *file, const bool do_exit)	3830
30.2285.2	draw_sphere(const vector3t< float > &pos, const float r, const int n, const int slot, const vector3t< float > &col, const bool wiremode)	3830
30.2285.2	list_FBConfigs(GLXFBConfig *config, int nelements)	3830
30.2286	examples/viewer/src/mouse.cpp File Reference	3830
30.2286	Macro Definition Documentation	3831
30.2286.1	GLUT_DISABLE_ATEXIT_HACK	3831
30.2286.1	PI	3831
30.2286.2	Function Documentation	3831
30.2286.2	draw_cursor(int x, int y)	3831
30.2286.2	Mouse(int butt, int state, int x, int y)	3831
30.2286.2	MouseRotate(int x, int y)	3831

30.2286.2	<code>MouseTranslate(int x, int y)</code>	3832
30.2286.2	<code>MouseZoom(int x, int y)</code>	3832
30.2286.2	<code>transversal_rotation(int x, int y, int last_xx, int last_yy)</code>	3832
30.2286	Variable Documentation	3832
30.2286.3	<code>allow_zrot</code>	3832
30.2286.3	<code>aurx</code>	3832
30.2286.3	<code>aury</code>	3832
30.2286.3	<code>aurz</code>	3832
30.2286.3	<code>last_x</code>	3832
30.2286.3	<code>last_x0</code>	3832
30.2286.3	<code>last_y</code>	3832
30.2286.3	<code>last_y0</code>	3833
30.2286.3	<code>mouse_movement</code>	3833
30.2286	<code>isl/examples/viewer/src/sisl_aux.cpp</code> File Reference	3833
30.2287	Macro Definition Documentation	3833
30.2287.1	<code>DBGqqq</code>	3833
30.2287	Function Documentation	3834
30.2287.2	<code>compute_surface_normals(SISLSurf *srf, double **ngrid)</code>	3834
30.2287.2	<code>lower_degree_and_subdivide(SISLSurf **srf, int new_knots_per_interval, int max_number_of_knots)</code>	3834
30.2287.2	<code>lower_degree_to_linear(SISLSurf **srf, double e)</code>	3834
30.2287	<code>isl/examples/viewer/src/transfutils.cpp</code> File Reference	3834
30.2288	Function Documentation	3835
30.2288.1	<code>rotate(double y_ang, double x_ang, double z_ang)</code>	3835
30.2288.1	<code>scale(double x, double y, double z)</code>	3835
30.2288.1	<code>translate(double x, double y, double z)</code>	3835
30.2288	Variable Documentation	3835
30.2288.2	<code>xrot</code>	3835
30.2288.2	<code>xrot_eps</code>	3835
30.2288.2	<code>xscale</code>	3835
30.2288.2	<code>xtrans</code>	3836

30.2288.2	rot	3836
30.2288.2	rot_eps	3836
30.2288.2	scale	3836
30.2288.2	trans	3836
30.2288.2	rot	3836
30.2288.2	rot_eps	3836
30.2288.2	scale	3836
30.2288.2	trans	3836
30.2288	/include/sisl-copyright.h File Reference	3837
30.2288	/include/sisl.h File Reference	3837
30.2290	Macro Definition Documentation	3841
30.2290.1	GO_API	3841
30.2290.1	SISL_CRV_CLOSED	3841
30.2290.1	SISL_CRV_OPEN	3841
30.2290.1	SISL_CRV_PERIODIC	3841
30.2290.1	SISL_SURF_CLOSED	3841
30.2290.1	SISL_SURF_OPEN	3841
30.2290.1	SISL_SURF_PERIODIC	3841
30.2290	Typedef Documentation	3842
30.2290.2	SISLbox	3842
30.2290.2	SISLCurve	3842
30.2290.2	SISLdir	3842
30.2290.2	SISLIntcurve	3842
30.2290.2	SISLSurf	3842
30.2290	Enumeration Type Documentation	3842
30.2290.3	anonymous enum	3842
30.2290.3	anonymous enum	3842
30.2290	Function Documentation	3843
30.2290.4	copyCurve()	3843
30.2290.4	copySurface()	3843

30.2290.4.feeCurve()	3843
30.2290.4.feeIntcrvlist()	3843
30.2290.4.feeIntcurve()	3843
30.2290.4.feeSurf()	3843
30.2290.4.make_cv_cyclic()	3843
30.2290.4.newbox()	3843
30.2290.4.newCurve()	3843
30.2290.4.newdir()	3843
30.2290.4.newIntcurve()	3843
30.2290.4.newSurf()	3843
30.2290.4.s1001()	3843
30.2290.4.s1011()	3843
30.2290.4.s1012()	3843
30.2290.4.s1013()	3843
30.2290.4.s1014()	3843
30.2290.4.s1015()	3843
30.2290.4.s1016()	3843
30.2290.4.s1017()	3843
30.2290.4.s1018()	3843
30.2290.4.s1021()	3843
30.2290.4.s1022()	3844
30.2290.4.s1023()	3844
30.2290.4.s1024()	3844
30.2290.4.s1025()	3844
30.2290.4.s1721()	3844
30.2290.4.s1825()	3844
30.2290.4.s1826()	3844
30.2290.4.s1827()	3844
30.2290.4.s18233()	3844
30.2290.4.s18237()	3844

30.2290.4.~~SF~~238() 3844

30.2290.4.~~SF~~240() 3844

30.2290.4.~~SF~~241() 3844

30.2290.4.~~SF~~243() 3844

30.2290.4.~~SF~~302() 3844

30.2290.4.~~SF~~303() 3844

30.2290.4.~~SF~~310() 3844

30.2290.4.~~SF~~314() 3844

30.2290.4.~~SF~~315() 3844

30.2290.4.~~SF~~316() 3844

30.2290.4.~~SF~~317() 3844

30.2290.4.~~SF~~318() 3844

30.2290.4.~~SF~~319() 3844

30.2290.4.~~SF~~327() 3845

30.2290.4.~~SF~~328() 3845

30.2290.4.~~SF~~332() 3845

30.2290.4.~~SF~~356() 3845

30.2290.4.~~SF~~357() 3845

30.2290.4.~~SF~~360() 3845

30.2290.4.~~SF~~363() 3845

30.2290.4.~~SF~~364() 3845

30.2290.4.~~SF~~365() 3845

30.2290.4.~~SF~~369() 3845

30.2290.4.~~SF~~371() 3845

30.2290.4.~~SF~~372() 3845

30.2290.4.~~SF~~373() 3845

30.2290.4.~~SF~~374() 3845

30.2290.4.~~SF~~375() 3845

30.2290.4.~~SF~~379() 3845

30.2290.4.~~SF~~380() 3845

30.2290.4.8383()	3845
30.2290.4.8386()	3845
30.2290.4.8387()	3845
30.2290.4.8388()	3845
30.2290.4.8389()	3845
30.2290.4.8390()	3845
30.2290.4.8391()	3846
30.2290.4.70401()	3846
30.2290.4.71421()	3846
30.2290.4.72422()	3846
30.2290.4.73424()	3846
30.2290.4.74425()	3846
30.2290.4.75439()	3846
30.2290.4.76440()	3846
30.2290.4.77450()	3846
30.2290.4.78451()	3846
30.2290.4.79501()	3846
30.2290.4.80502()	3846
30.2290.4.81503()	3846
30.2290.4.82506()	3846
30.2290.4.83508()	3846
30.2290.4.84510()	3846
30.2290.4.85511()	3846
30.2290.4.86514()	3846
30.2290.4.87515()	3846
30.2290.4.88522()	3846
30.2290.4.89529()	3846
30.2290.4.90530()	3846
30.2290.4.91534()	3846
30.2290.4.92535()	3847

30.2290.4.s10536() 3847

30.2290.4.s10537() 3847

30.2290.4.s10538() 3847

30.2290.4.s10539() 3847

30.2290.4.s10542() 3847

30.2290.4.s10600() 3847

30.2290.4.s10601() 3847

30.2290.4.s10602() 3847

30.2290.4.s10603() 3847

30.2290.4.s10606() 3847

30.2290.4.s10607() 3847

30.2290.4.s10608() 3847

30.2290.4.s10609() 3847

30.2290.4.s10611() 3847

30.2290.4.s10673() 3847

30.2290.4.s10680() 3847

30.2290.4.s10680() 3847

30.2290.4.s11706() 3847

30.2290.4.s11710() 3847

30.2290.4.s117211() 3847

30.2290.4.s117312() 3847

30.2290.4.s117413() 3847

30.2290.4.s117514() 3848

30.2290.4.s117615() 3848

30.2290.4.s117716() 3848

30.2290.4.s117820() 3848

30.2290.4.s117930() 3848

30.2290.4.s12031() 3848

30.2290.4.s12032() 3848

30.2290.4.s12033() 3848

30.2290.4.s1250()	3848
30.2290.4.s1274()	3848
30.2290.4.s1275()	3848
30.2290.4.s1285()	3848
30.2290.4.s12851()	3848
30.2290.4.s12852()	3848
30.2290.4.s12853()	3848
30.2290.4.s12854()	3848
30.2290.4.s12855()	3848
30.2290.4.s12856()	3848
30.2290.4.s12857()	3848
30.2290.4.s12858()	3848
30.2290.4.s12859()	3848
30.2290.4.s12860()	3848
30.2290.4.s12870()	3848
30.2290.4.s12871()	3849
30.2290.4.s12920()	3849
30.2290.4.s12921()	3849
30.2290.4.s12940()	3849
30.2290.4.s12953()	3849
30.2290.4.s12954()	3849
30.2290.4.s12955()	3849
30.2290.4.s12957()	3849
30.2290.4.s12958()	3849
30.2290.4.s12961()	3849
30.2290.4.s12962()	3849
30.2290.4.s12963()	3849
30.2290.4.s12965()	3849
30.2290.4.s12966()	3849
30.2290.4.s12967()	3849

30.2290.4	s1986 ()	3849
30.2290.4	s1986 ()	3849
30.2290.4	s1987 ()	3849
30.2290.4	s1988 ()	3849
30.2290.4	s1989 ()	3849
30.2290.4	s2500 ()	3849
30.2290.4	s2502 ()	3849
30.2290.4	s2504 ()	3849
30.2290.4	s2506 ()	3850
30.2290.4	s2508 ()	3850
30.2290.4	s2510 ()	3850
30.2290.4	s2532 ()	3850
30.2290.4	s2536 ()	3850
30.2290.4	s2540 ()	3850
30.2290.4	s2542 ()	3850
30.2290.4	s2544 ()	3850
30.2290.4	s2545 ()	3850
30.2290.4	s2550 ()	3850
30.2290.4	s2553 ()	3850
30.2290.4	s2556 ()	3850
30.2290.4	s2559 ()	3850
30.2290.4	s2562 ()	3850
30.2290.4	s65 rawseq()	3850
30.2291	isl/include/sislP.h File Reference	3850
30.2291	Macro Definition Documentation	3860
30.2291.1	A EPSCO	3860
30.2291.1	A EPSGE	3860
30.2291.1	A NGULAR_TOLERANCE	3860
30.2291.1	C ONST	3860
30.2291.1	C ONSTVOIDP	3860

30.2291.1	B ZERO	3861
30.2291.1	M AXIMAL_RADIUS_OF_CURVATURE	3861
30.2291.1	B EL_COMP_RES	3861
30.2291.1	B EL_PAR_RES	3861
30.2291.1	S ISL_NULL	3861
30.2291.1	S ISLCURVE	3861
30.2291.1	S ISLPOINT	3861
30.2291.1	S ISLSURFACE	3861
30.2291.1	V OIDP	3861
30.2291.2	T ypedef Documentation	3862
30.2291.2	r ank_info	3862
30.2291.2	S ISLEdge	3862
30.2291.2	S ISLIntdat	3862
30.2291.2	S ISLIntlist	3862
30.2291.2	S ISLIntpt	3862
30.2291.2	S ISLIntsurf	3862
30.2291.2	S ISLObject	3862
30.2291.2	S ISLPoint	3862
30.2291.2	S ISLPtedge	3862
30.2291.2	S ISLTrack	3862
30.2291.2	S ISLTrimpar	3862
30.2291.3	E numeration Type Documentation	3862
30.2291.3	a nonymous enum	3862
30.2291.4	F unction Documentation	3863
30.2291.4	d copyIntpt()	3863
30.2291.4	a rc_tang()	3863
30.2291.4	c rv_crv_tang()	3863
30.2291.4	a rc_lin_tang()	3863
30.2291.4	e v_cv_off()	3863
30.2291.4	e val_2_crv()	3863

30.2291.4.17	eval_crv_arc()	3863
30.2291.4.18	FreeEdge()	3863
30.2291.4.19	FreeIntdat()	3863
30.2291.4.20	FreeIntlist()	3863
30.2291.4.21	FreeIntpt()	3863
30.2291.4.22	FreeIntsurf()	3863
30.2291.4.23	FreeObject()	3863
30.2291.4.24	FreePoint()	3863
30.2291.4.25	FreePtedge()	3863
30.2291.4.26	FreeTrack()	3863
30.2291.4.27	FreeTrimpar()	3863
30.2291.4.28	h18_copyIntpt()	3863
30.2291.4.29	h19_newIntpt()	3863
30.2291.4.30	h20_s1880()	3863
30.2291.4.31	h21_join_per()	3863
30.2291.4.32	h22_make3D()	3863
30.2291.4.33	h23_make_cv_kreg()	3863
30.2291.4.34	h24_make_sf_kreg()	3863
30.2291.4.35	h25_make_tracks()	3863
30.2291.4.36	h26wEdge()	3863
30.2291.4.37	h27wIntdat()	3863
30.2291.4.38	h28wIntlist()	3864
30.2291.4.39	h29wIntpt()	3864
30.2291.4.40	h30wIntsurf()	3864
30.2291.4.41	h31wknots()	3864
30.2291.4.42	h32wObject()	3864
30.2291.4.43	h33wPoint()	3864
30.2291.4.44	h34wPtedge()	3864
30.2291.4.45	h35wTrack()	3864
30.2291.4.46	h36wTrimpar()	3864

30.2291.4.517k_crv_sf()	3864
30.2291.4.518_crv_tang()	3864
30.2291.4.519_line_all()	3864
30.2291.4.520119()	3864
30.2291.4.521161()	3864
30.2291.4.522162()	3864
30.2291.4.523172()	3864
30.2291.4.524173()	3864
30.2291.4.525174()	3864
30.2291.4.526190()	3864
30.2291.4.527192()	3864
30.2291.4.52819()	3864
30.2291.4.52920()	3864
30.2291.4.530222()	3864
30.2291.4.531223()	3865
30.2291.4.53224()	3865
30.2291.4.53331()	3865
30.2291.4.53432()	3865
30.2291.4.53535()	3865
30.2291.4.53636()	3865
30.2291.4.53739()	3865
30.2291.4.53844()	3865
30.2291.4.53945()	3865
30.2291.4.54051()	3865
30.2291.4.54152()	3865
30.2291.4.542301()	3865
30.2291.4.543304()	3865
30.2291.4.544305()	3865
30.2291.4.545306()	3865
30.2291.4.546307()	3865

30.2291.4.87308() 3865

30.2291.4.87309() 3865

30.2291.4.87311() 3865

30.2291.4.87312() 3865

30.2291.4.87313() 3865

30.2291.4.87320() 3865

30.2291.4.87321() 3865

30.2291.4.87322() 3866

30.2291.4.87323() 3866

30.2291.4.87324() 3866

30.2291.4.87325() 3866

30.2291.4.87329() 3866

30.2291.4.87330() 3866

30.2291.4.87331() 3866

30.2291.4.87333() 3866

30.2291.4.87333_count() 3866

30.2291.4.87333_cyclic() 3866

30.2291.4.87334() 3866

30.2291.4.87340() 3866

30.2291.4.87341() 3866

30.2291.4.87342() 3866

30.2291.4.87343() 3866

30.2291.4.87345() 3866

30.2291.4.87346() 3866

30.2291.4.87347() 3866

30.2291.4.87348() 3866

30.2291.4.87349() 3866

30.2291.4.87350() 3866

30.2291.4.87351() 3866

30.2291.4.87352() 3866

30.2291.4.97353()	3867
30.2291.4.98354()	3867
30.2291.4.98355()	3867
30.2291.4.10358()	3867
30.2291.4.10359()	3867
30.2291.4.10361()	3867
30.2291.4.10362()	3867
30.2291.4.10366()	3867
30.2291.4.10367()	3867
30.2291.4.10370()	3867
30.2291.4.10376()	3867
30.2291.4.10377()	3867
30.2291.4.10378()	3867
30.2291.4.11381()	3867
30.2291.4.11382()	3867
30.2291.4.11384()	3867
30.2291.4.11385()	3867
30.2291.4.11393()	3867
30.2291.4.11399()	3867
30.2291.4.11435()	3867
30.2291.4.11436()	3867
30.2291.4.11437()	3867
30.2291.4.11438()	3867
30.2291.4.12052()	3868
30.2291.4.12300()	3868
30.2291.4.12304()	3868
30.2291.4.12305()	3868
30.2291.4.12307()	3868
30.2291.4.12512()	3868
30.2291.4.12513()	3868

30.2291.4.s1516() 3868

30.2291.4.s1517() 3868

30.2291.4.s1520() 3868

30.2291.4.s1521() 3868

30.2291.4.s1528() 3868

30.2291.4.s1531() 3868

30.2291.4.s1540() 3868

30.2291.4.s1541() 3868

30.2291.4.s1554() 3868

30.2291.4.s1555() 3868

30.2291.4.s1572() 3868

30.2291.4.s1581bez() 3868

30.2291.4.s15914() 3868

30.2291.4.s16015() 3868

30.2291.4.s16116() 3868

30.2291.4.s16217() 3868

30.2291.4.s16318() 3869

30.2291.4.s16419() 3869

30.2291.4.s16500() 3869

30.2291.4.s16601() 3869

30.2291.4.s16705() 3869

30.2291.4.s16807() 3869

30.2291.4.s16908() 3869

30.2291.4.s17011() 3869

30.2291.4.s17153() 3869

30.2291.4.s17254() 3869

30.2291.4.s17355() 3869

30.2291.4.s17470() 3869

30.2291.4.s17570_2D() 3869

30.2291.4.s17671() 3869

30.2291.4.s15772()	3869
30.2291.4.s15773()	3869
30.2291.4.s15780()	3869
30.2291.4.s15785()	3869
30.2291.4.s15786()	3869
30.2291.4.s15787()	3869
30.2291.4.s15788()	3869
30.2291.4.s15789()	3869
30.2291.4.s15790()	3869
30.2291.4.s15791()	3870
30.2291.4.s15792()	3870
30.2291.4.s15795()	3870
30.2291.4.s15796()	3870
30.2291.4.s1707()	3870
30.2291.4.s17830()	3870
30.2291.4.s17834()	3870
30.2291.4.s17839()	3870
30.2291.4.s17840()	3870
30.2291.4.s17880()	3870
30.2291.4.s17890()	3870
30.2291.4.s17891()	3870
30.2291.4.s17893()	3870
30.2291.4.s17894()	3870
30.2291.4.s18896()	3870
30.2291.4.s18897()	3870
30.2291.4.s18900()	3870
30.2291.4.s18901()	3870
30.2291.4.s18902()	3870
30.2291.4.s18903()	3870
30.2291.4.s18904()	3870

30.2291.4.1905() 3870

30.2291.4.1906() 3870

30.2291.4.1907() 3871

30.2291.4.1908() 3871

30.2291.4.1909() 3871

30.2291.4.1910() 3871

30.2291.4.1911() 3871

30.2291.4.1912() 3871

30.2291.4.1916() 3871

30.2291.4.1917() 3871

30.2291.4.1918() 3871

30.2291.4.1919() 3871

30.2291.4.1924() 3871

30.2291.4.2025() 3871

30.2291.4.2026() 3871

30.2291.4.2027() 3871

30.2291.4.2031() 3871

30.2291.4.2031unit() 3871

30.2291.4.2032() 3871

30.2291.4.2033() 3871

30.2291.4.2034() 3871

30.2291.4.2035() 3871

30.2291.4.2036() 3871

30.2291.4.2037() 3871

30.2291.4.2038() 3871

30.2291.4.21240() 3872

30.2291.4.21241() 3872

30.2291.4.21242() 3872

30.2291.4.21243() 3872

30.2291.4.21244() 3872

30.2291.4.21945()	3872
30.2291.4.21946()	3872
30.2291.4.21947()	3872
30.2291.4.21948()	3872
30.2291.4.21949()	3872
30.2291.4.21950()	3872
30.2291.4.21951()	3872
30.2291.4.21956()	3872
30.2291.4.21959()	3872
30.2291.4.21960()	3872
30.2291.4.21990()	3872
30.2291.4.21991()	3872
30.2291.4.21992()	3872
30.2291.4.21992cu()	3872
30.2291.4.21992su()	3872
30.2291.4.21993()	3872
30.2291.4.21994()	3872
30.2291.4.22501()	3872
30.2291.4.22503()	3873
30.2291.4.22505()	3873
30.2291.4.22507()	3873
30.2291.4.22509()	3873
30.2291.4.22511()	3873
30.2291.4.22512()	3873
30.2291.4.22513()	3873
30.2291.4.22514()	3873
30.2291.4.22515()	3873
30.2291.4.22516()	3873
30.2291.4.22533()	3873
30.2291.4.22534()	3873

30.2291.4.2535()	3873
30.2291.4.2541()	3873
30.2291.4.2543()	3873
30.2291.4.2551()	3873
30.2291.4.2554()	3873
30.2291.4.2555()	3873
30.2291.4.2557()	3873
30.2291.4.2558()	3873
30.2291.4.2560()	3873
30.2291.4.2561()	3873
30.2291.4.2571	3873
30.2291.4.2581	3874
30.2291.4.2591	3874
30.2291.4.2601	3874
30.2291.4.2611	3874
30.2291.4.2621	3874
30.2291.4.2631	3874
30.2291.4.2641	3874
30.2291.4.2651	3874
30.2291.4.2661	3874
30.2291.4.2671	3874
30.2291.4.2681	3874
30.2291.4.2691	3874
30.2291.4.2701	3874
30.2291.4.2711	3874
30.2291.4.2721	3874
30.2291.4.2731	3874
30.2291.4.2741	3874
30.2291.4.2751	3874
30.2291.4.2761	3874
30.2291.4.2771	3874
30.2291.4.2781	3874
30.2291.4.2791	3874
30.2291.4.2801	3874
30.2291.4.2811	3874
30.2291.4.2821	3874
30.2291.4.2831	3874
30.2291.4.2841	3874
30.2291.4.2851	3874
30.2291.4.2861	3874
30.2291.4.2871	3874
30.2291.4.2881	3874
30.2291.4.2891	3874
30.2291.4.2901	3874
30.2291.4.2911	3874
30.2291.4.2921	3874
30.2291.4.2931	3874
30.2291.4.2941	3874
30.2291.4.2951	3874
30.2291.4.2961	3874
30.2291.4.2971	3874
30.2291.4.2981	3874
30.2291.4.2991	3874
30.2291.4.3001	3874

30.2291.4.s877	indfac()	3874
30.2291.4.s878	indintvl()	3874
30.2291.4.s879	term()	3874
30.2291.4.s880	hermite_bezier()	3874
30.2291.4.s881	icon()	3875
30.2291.4.s882	acpt()	3875
30.2291.4.s883	edg()	3875
30.2291.4.s884	identify()	3875
30.2291.4.s885	iget()	3875
30.2291.4.s886	int()	3875
30.2291.4.s887	klist()	3875
30.2291.4.s888	kpt()	3875
30.2291.4.s889	llis()	3875
30.2291.4.s890	dnpt()	3875
30.2291.4.s891	dput()	3875
30.2291.4.s892	lv4()	3875
30.2291.4.s893	lvert()	3875
30.2291.4.s894	notmult()	3875
30.2291.4.s895	length()	3875
30.2291.4.s896	me()	3875
30.2291.4.s897	rj()	3875
30.2291.4.s898	ifacp()	3875
30.2291.4.s899	solp()	3875
30.2291.4.s900	metric()	3875
30.2291.4.s901	hove()	3875
30.2291.4.s902	ulvec()	3875
30.2291.4.s903	vec()	3875
30.2291.4.s904	ewbox()	3876
30.2291.4.s905	form()	3876
30.2291.4.s906	atder()	3876

30.2291.4.s307	tax()	3876
30.2291.4.s308	tmatrix()	3876
30.2291.4.s309	trhoen()	3876
30.2291.4.s310	trpr()	3876
30.2291.4.s311	trsortpar()	3876
30.2291.4.s312	trstatder()	3876
30.2291.4.s313	trstrider()	3876
30.2291.4.s314	trtakeunion()	3876
30.2291.4.s315	trvonorm()	3876
30.2291.4.s316	trvsimp()	3876
30.2291.4.s317	trvstep()	3876
30.2291.4.s318	trvundimp()	3876
30.2291.4.s319	trvundit()	3876
30.2291.4.s320	trvpimp()	3876
30.2291.4.s321	trvpit()	3876
30.2291.4.s322	trvnmarch()	3876
30.2291.4.s323	trvrate()	3876
30.2291.4.s324	trvrmp()	3876
30.2291.4.s325	trvmp_knot()	3876
30.2291.4.s326	trvrnmarch()	3876
30.2291.4.s327	trv260()	3877
30.2291.4.s328	trv261()	3877
30.2291.4.s329	trv262()	3877
30.2291.4.s330	trv263()	3877
30.2291.4.s331	trv365()	3877
30.2291.4.s332	trv369()	3877
30.2291.4.s333	trv371()	3877
30.2291.4.s334	trv372()	3877
30.2291.4.s335	trv373()	3877
30.2291.4.s336	trv374()	3877

30.2291.4.s387375()	3877
30.2291.4.s387460()	3877
30.2291.4.s387461()	3877
30.2291.4.s387462()	3877
30.2291.4.s387463()	3877
30.2291.4.s387464()	3877
30.2291.4.s387465()	3877
30.2291.4.s387466()	3877
30.2291.4.s387467()	3877
30.2291.4.s387502()	3877
30.2291.4.s387503()	3877
30.2291.4.s387510()	3877
30.2291.4.s387511()	3877
30.2291.4.s38761()	3878
30.2291.4.s38762()	3878
30.2291.4.s38779()	3878
30.2291.4.s38779_at()	3878
30.2291.4.s38780()	3878
30.2291.4.s38780_at()	3878
30.2291.4.s38781()	3878
30.2291.4.s38781_at()	3878
30.2291.4.s38782()	3878
30.2291.4.s38783()	3878
30.2291.4.s38784()	3878
30.2291.4.s38786()	3878
30.2291.4.s38787()	3878
30.2291.4.s38790()	3878
30.2291.4.s38830()	3878
30.2291.4.s38831()	3878
30.2291.4.s38834()	3878

30.2291.4.s167839() 3878

30.2291.4.s167850() 3878

30.2291.4.s167851() 3878

30.2291.4.s170852() 3878

30.2291.4.s171853() 3878

30.2291.4.s172854() 3878

30.2291.4.s173855() 3879

30.2291.4.s174856() 3879

30.2291.4.s175857() 3879

30.2291.4.s176858() 3879

30.2291.4.s177859() 3879

30.2291.4.s178860() 3879

30.2291.4.s179870() 3879

30.2291.4.s180871() 3879

30.2291.4.s181922() 3879

30.2291.4.s182923() 3879

30.2291.4.s183924() 3879

30.2291.4.s184925() 3879

30.2291.4.s185926() 3879

30.2291.4.s186927() 3879

30.2291.4.s187928() 3879

30.2291.4.s188929() 3879

30.2291.4.s189930() 3879

30.2291.4.s190992() 3879

30.2291.4.s191992cu() 3879

30.2291.4.s192992su() 3879

30.2291.4.s193993() 3879

30.2291.4.s194994() 3879

30.2291.4.s195closevert() 3879

30.2291.4.s196comedg() 3880

30.2291.4.415	connect()	3880
30.2291.4.416	count()	3880
30.2291.4.417	cvvert()	3880
30.2291.4.418	degen()	3880
30.2291.4.419	disconnect()	3880
30.2291.4.420	edgpoint()	3880
30.2291.4.421	edgred()	3880
30.2291.4.422	evalint()	3880
30.2291.4.423	findsplit()	3880
30.2291.4.424	floop()	3880
30.2291.4.425	getgeom()	3880
30.2291.4.426	getlist()	3880
30.2291.4.427	getmain()	3880
30.2291.4.428	getnext()	3880
30.2291.4.429	getnhbrs()	3880
30.2291.4.430	getother()	3880
30.2291.4.431	getprev()	3880
30.2291.4.432	gettop()	3880
30.2291.4.433	gettophp()	3880
30.2291.4.434	dalledg()	3880
30.2291.4.435	dcon()	3880
30.2291.4.436	dfcross()	3880
30.2291.4.437	dget()	3881
30.2291.4.438	dkpt()	3881
30.2291.4.439	dilis()	3881
30.2291.4.440	dnewunite()	3881
30.2291.4.441	dnpt()	3881
30.2291.4.442	dput()	3881
30.2291.4.443	drmcross()	3881
30.2291.4.444	dsplit()	3881

30.2291.4.417	dunite()	3881
30.2291.4.418	insert()	3881
30.2291.4.419	insertpt()	3881
30.2291.4.420	sconnect()	3881
30.2291.4.421	shelp()	3881
30.2291.4.422	sinside()	3881
30.2291.4.423	smain()	3881
30.2291.4.424	shmbhelp()	3881
30.2291.4.425	shmbmain()	3881
30.2291.4.426	stobj()	3881
30.2291.4.427	putsing()	3881
30.2291.4.428	puttouch()	3881
30.2291.4.429	red()	3881
30.2291.4.430	remcon()	3881
30.2291.4.431	removept()	3881
30.2291.4.432	sepcrv()	3882
30.2291.4.433	setcnsdir()	3882
30.2291.4.434	setdir()	3882
30.2291.4.435	settop()	3882
30.2291.4.436	splitgeom()	3882
30.2291.4.437	shohelp()	3882
30.2291.4.438	somain()	3882
30.2291.4.439	srimlist()	3882
30.2291.4.450	d_div()	3882
30.2291.4.451	div_crv()	3882
30.2291.4.452	div_surf()	3882
30.2291.4.453	set_at()	3882
30.2291.4.454	sape()	3882
30.2291.4.455	sheckput()	3882
30.2291.4.456	shecktype()	3882

30.2291.4.457	fsing()	3882
30.2291.4.458	valc()	3882
30.2291.4.459	khlppts()	3882
30.2291.4.460	ng()	3882
30.2291.4.461	silh()	3882
30.2291.4.462	cyclic_knots()	3882
30.2292	/src/construct.c File Reference	3883
30.2292	Macro Definition Documentation	3884
30.2292.1	CONSTRUCT	3884
30.2292	Function Documentation	3884
30.2292.2	copyCurve(SISLCurve *pcurve)	3884
30.2292.2	copyIntpt(SISLIntpt *ppt)	3884
30.2292.2	copySurface(SISLSurf *psurf)	3884
30.2292.2.1	copyIntpt(SISLIntpt *ppt)	3884
30.2292.2.5	newIntpt(int ipar, double *epar, double adist, int itype, int ileft1, int irlight1, int irlight2, int irlight2, int size_1, int size_2, double *egeom1, double *egeom2)	3884
30.2292.2.6	newbox(int idim)	3884
30.2292.2.7	newCurve(int in, int ik, double *et, double *ecoef, int ikind, int idim, int icopy)	3884
30.2292.2.8	newdir(int idim)	3884
30.2292.2.9	newEdge(int iedge)	3885
30.2292.2.10	newIntcurve(int ipoint, int ipar1, int ipar2, double *epar1, double *epar2, int itype)	3885
30.2292.2.11	newIntdat()	3885
30.2292.2.12	newIntlist(SISLIntpt *pfirst, SISLIntpt *plast, int itype)	3885
30.2292.2.13	newIntpt(int ipar, double *epar, double adist)	3885
30.2292.2.14	newIntsurf(SISLIntlist *pintlist)	3885
30.2292.2.15	newObject(int iobj)	3885
30.2292.2.16	newPoint(double *ecoef, int idim, int icopy)	3885
30.2292.2.17	newPtedge(SISLIntpt *ppt)	3885
30.2292.2.18	newSurf(int in1, int in2, int ik1, int ik2, double *et1, double *et2, double *ecoef, int ikind, int idim, int icopy)	3885

30.2292.2	NewTrack(SISLSurf *psurf_1, SISLSurf *psurf_2, SISLCurve *pcurve_3d, SISLCurve *pcurve_2d_1, SISLCurve *pcurve_2d_2, int ideg, eimpli, int sing_start, int sing_end, int turned)	3886
30.2292.2	NewTrimpar(int pt, int par)	3886
30.2293	src/crvarctang.c File Reference	3886
30.2293	Macro Definition Documentation	3886
30.2293.1	CRV_ARC_TANG	3886
30.2293	Function Documentation	3887
30.2293.2	drv_arc_tang(SISLCurve *pc1, center, double radius, double aepsge, guess_par, iter_par, int *jstat)	3887
30.2294	src/crvcrvtang.c File Reference	3887
30.2294	Macro Definition Documentation	3887
30.2294.1	CRV_CRV_TANG	3887
30.2294	Function Documentation	3888
30.2294.2	drv_crv_tang(SISLCurve *pc1, SISLCurve *pc2, double aepsge, guess_par, iter_par, int *jstat)	3888
30.2295	src/crvlintang.c File Reference	3888
30.2295	Macro Definition Documentation	3888
30.2295.1	CRV_LIN_TANG	3888
30.2295	Function Documentation	3889
30.2295.2	drv_lin_tang(SISLCurve *pc1, point, normal, double ang_tol, double guess_par, double *iter_par, int *jstat)	3889
30.2296	src/destruct.c File Reference	3889
30.2296	Macro Definition Documentation	3890
30.2296.1	DESTRUCT	3890
30.2296	Function Documentation	3890
30.2296.2	freeCurve(SISLCurve *pcurve)	3890
30.2296.2	freeEdge(SISLEdge *pedge)	3890
30.2296.2	freeIntcrvlist(SISLIntcurve **viclist, int icrv)	3890
30.2296.2	freeIntcurve(SISLIntcurve *pintc)	3890
30.2296.2	freeIntdat(SISLIntdat *pintdat)	3890
30.2296.2	freeIntlist(SISLIntlist *plist)	3890
30.2296.2	freeIntpt(SISLIntpt *ppt)	3890

30.2296.2fileIntsurf(SISLIntsurf *intsurf)	3890
30.2296.2fileObject(SISLObject *pobj)	3891
30.2296.2filePoint(SISLPoint *ppoint)	3891
30.2296.2filePtedge(SISLPtedge *p1)	3891
30.2296.2fileSurf(SISLSurf *psurf)	3891
30.2296.2fileTrack(SISLTrack *ppt)	3891
30.2296.2fileTrimpar(SISLTrimpar *trimpar)	3891
30.2297src/ev_cv_off.c File Reference	3891
30.2297Macro Definition Documentation	3892
30.2297.1EV CV OFF	3892
30.2297Function Documentation	3892
30.2297.2ev_cv_off(SISLCurve *pc1, int ider, double ax, int *ileft, double offset, eder, int *jstat)	3892
30.2298src/eval_2_crv.c File Reference	3892
30.2298Macro Definition Documentation	3893
30.2298.1EVAL_2_CRV	3893
30.2298Function Documentation	3893
30.2298.2eval_2_crv(SISLCurve *pc1, SISLCurve *pc2, int ider, epar, int *ilfs, int *ilft, eder, int *jstat)	3893
30.2299src/evalcrvarc.c File Reference	3893
30.2299Macro Definition Documentation	3894
30.2299.1EVAL_CRV_ARC	3894
30.2299Function Documentation	3894
30.2299.2eval_crv_arc(SISLCurve *pc1, center, double radius, int ider, epar, int *ilfs, eder, int *jstat)	3894
30.2300src/hp_s1880.c File Reference	3894
30.2300Macro Definition Documentation	3895
30.2300.1HIP_S1880	3895
30.2300Function Documentation	3895
30.2300.2hp_s1880(SISLObject *po1, SISLObject *po2, int ideg, int ipar1, int ipar2, SISLIntdat *pintdat, int *jpar, double **gpar1, double **gpar2, int **pretop, int *jcrv, SISLIntcurve ***wcrv, int *jsurf, SISLIntsurf ***wsurf, int *jstat)	3895
30.2301src/intjoinper.c File Reference	3895

30.2301	Macro Definition Documentation	3896
30.2301.1	JNT_JOIN_PER	3896
30.2301	Function Documentation	3896
30.2301.2	int_join_per(SISLIntdat **pintdat, SISLObject *po1, SISLObject *po2, eimpli, int ideg, double aepsge, int *jstat)	3896
30.2301	src/make3D.c File Reference	3896
30.2302	Macro Definition Documentation	3897
30.2302.1	MAKE3D	3897
30.2302	Function Documentation	3897
30.2302.2	make3D(SISLSurf *ps, SISLSurf **rsnew, int *jstat)	3897
30.2302	src/makecvkreg.c File Reference	3897
30.2303	Macro Definition Documentation	3898
30.2303.1	MAKE_CV_KREG	3898
30.2303	Function Documentation	3898
30.2303.2	make_cv_kreg(SISLCurve *pc, SISLCurve **rcnew, int *jstat)	3898
30.2303	src/makesfkreg.c File Reference	3898
30.2304	Macro Definition Documentation	3899
30.2304.1	MAKE_SF_KREG	3899
30.2304	Function Documentation	3899
30.2304.2	make_sf_kreg(SISLSurf *ps, SISLSurf **rsnew, int *jstat)	3899
30.2304	src/maketracks.c File Reference	3899
30.2305	Macro Definition Documentation	3900
30.2305.1	MAKE_TRACKS	3900
30.2305	Function Documentation	3900
30.2305.2	make_tracks(SISLObject *po1, SISLObject *po2, int ideg, eimpli, int icrv, SISLIntlist **vlist, int *jtrack, SISLTrack ***wcrv, double aepsge, int *jstat)	3900
30.2305	src/mk_cv_cycl.c File Reference	3900
30.2306	Macro Definition Documentation	3901
30.2306.1	MAKE_CV_CYCLIC	3901
30.2306	Function Documentation	3901
30.2306.2	make_cv_cyclic(SISLCurve *pcurve, int icont, int *jstat)	3901

30.2306 `isl/src/newknots.c` File Reference 3901

 30.2307 Macro Definition Documentation 3902

 30.2307.1 `NEWKNOTS` 3902

 30.2307 Function Documentation 3902

 30.2307.2 `newknots(et, int in, int ik, epar, int inpar, double aeps, double **ginsert, int *jinsert, int *jstat)` 3902

30.2308 `isl/src/pickcrvsf.c` File Reference 3902

 30.2308 Macro Definition Documentation 3903

 30.2308.1 `PICK_CRV_SF` 3903

 30.2308 Function Documentation 3903

 30.2308.2 `pick_crv_sf(SISLObject *po1, SISLObject *po2, int ipar, SISLIntpt *pt1, SISLIntpt *pt2, SISLCurve **rcrv, int *jstat)` 3903

30.2309 `isl/src/pocrvtang.c` File Reference 3903

 30.2309 Macro Definition Documentation 3904

 30.2309.1 `PO_CRV_TANG` 3904

 30.2309 Function Documentation 3904

 30.2309.2 `po_crv_tang(SISLCurve *pcurve, point, double ang_tol, double guess_par, double *iter_par, int *jstat)` 3904

30.2310 `isl/src/refine_all.c` File Reference 3904

 30.2310 Macro Definition Documentation 3905

 30.2310.1 `REFINE_ALL` 3905

 30.2310 Function Documentation 3905

 30.2310.2 `refine_all(SISLIntdat **pintdat, SISLObject *po1, SISLObject *po2, eimpli, int ideg, double aepsge, int *jstat)` 3905

30.2311 `isl/src/s1001.c` File Reference 3905

 30.2311 Macro Definition Documentation 3906

 30.2311.1 `S1001` 3906

 30.2311 Function Documentation 3906

 30.2311.2 `s1001(SISLSurf *ps, double min1, double min2, double max1, double max2, S↔ISLSurf **rsnew, int *jstat)` 3906

30.2312 `isl/src/s1011.c` File Reference 3906

 30.2312 Macro Definition Documentation 3907

 30.2312.1 `S1011` 3907

30.2312	Function Documentation	3907
30.2312.2	<code>s1011(start_pos, top_pos, end_pos, double shape, int dim, SISLCurve **arc_↵ seg, int *stat)</code>	3907
30.2313	<code>src/s1012.c</code> File Reference	3907
30.2313	Macro Definition Documentation	3908
30.2313.1	<code>s1012</code>	3908
30.2313	Function Documentation	3908
30.2313.2	<code>s1012(start_pos, axis_pos, axis_dir, double frequency, int numb_quad, int counter_clock, SISLCurve **helix, int *stat)</code>	3908
30.2314	<code>src/s1013.c</code> File Reference	3908
30.2314	Macro Definition Documentation	3909
30.2314.1	<code>s1013</code>	3909
30.2314	Function Documentation	3909
30.2314.2	<code>s1013(SISLCurve *pcurve, double ang, double ang_tol, double guess_par, dou- ble *iter_par, int *jstat)</code>	3909
30.2315	<code>src/s1014.c</code> File Reference	3909
30.2315	Macro Definition Documentation	3910
30.2315.1	<code>s1014</code>	3910
30.2315	Function Documentation	3910
30.2315.2	<code>s1014(SISLCurve *pc1, circ_cen, double circ_rad, double aepsge, eps1, eps2, double aradius, double *parpt1, double *parpt2, center, int *jstat)</code>	3910
30.2316	<code>src/s1015.c</code> File Reference	3910
30.2316	Macro Definition Documentation	3911
30.2316.1	<code>s1015</code>	3911
30.2316	Function Documentation	3911
30.2316.2	<code>s1015(SISLCurve *pc1, SISLCurve *pc2, double aepsge, eps1, eps2, double aradius, double *parpt1, double *parpt2, center, int *jstat)</code>	3911
30.2317	<code>src/s1016.c</code> File Reference	3911
30.2317	Macro Definition Documentation	3912
30.2317.1	<code>s1016</code>	3912
30.2317	Function Documentation	3912
30.2317.2	<code>s1016(SISLCurve *pc1, point, normal, double aepsge, eps1, eps2, double ara- dius, double *parpt1, double *parpt2, center, int *jstat)</code>	3912

30.2318 [src/s1017.c File Reference](#) 3912

 30.2318 [Macro Definition Documentation](#) 3913

 30.2318.1 [§1017](#) 3913

 30.2318 [Function Documentation](#) 3913

 30.2318.2 [§1017\(SISLCurve *pc, SISLCurve **rc, double apar, int *jstat\)](#) 3913

30.2319 [src/s1018.c File Reference](#) 3913

 30.2319 [Macro Definition Documentation](#) 3914

 30.2319.1 [§1018](#) 3914

 30.2319 [Function Documentation](#) 3914

 30.2319.2 [§1018\(SISLCurve *pc, epar, int inpar, SISLCurve **rcnew, int *jstat\)](#) 3914

30.2320 [src/s1021.c File Reference](#) 3914

 30.2320 [Macro Definition Documentation](#) 3915

 30.2320.1 [§1021](#) 3915

 30.2320 [Function Documentation](#) 3915

 30.2320.2 [§1021\(bottom_pos, bottom_axis, double ellipse_ratio, axis_dir, double height, SISLSurf **cyl, int *stat\)](#) 3915

30.2321 [src/s1022.c File Reference](#) 3915

 30.2321 [Macro Definition Documentation](#) 3916

 30.2321.1 [§1022](#) 3916

 30.2321 [Function Documentation](#) 3916

 30.2321.2 [§1022\(bottom_pos, bottom_axis, double ellipse_ratio, axis_dir, double cone_angle, double height, SISLSurf **cone, int *stat\)](#) 3916

30.2322 [src/s1023.c File Reference](#) 3916

 30.2322 [Macro Definition Documentation](#) 3917

 30.2322.1 [§1023](#) 3917

 30.2322 [Function Documentation](#) 3917

 30.2322.2 [§1023\(center, axis, equator, int latitude, int longitude, SISLSurf **sphere, int *stat\)](#) 3917

30.2323 [src/s1024.c File Reference](#) 3917

 30.2323 [Macro Definition Documentation](#) 3918

 30.2323.1 [§1024](#) 3918

 30.2323 [Function Documentation](#) 3918

30.2323.2	s1024(center, axis, equator, double minor_radius, int start_minor, int end_minor, int numb_major, SISLSurf **torus, int *stat)	3918
30.2324	src/s1025.c File Reference	3918
30.2324	Macro Definition Documentation	3919
30.2324.1	s1025	3919
30.2324	Function Documentation	3919
30.2324.2	s1025(SISLSurf *ps, epar1, int inpar1, epar2, int inpar2, SISLSurf **rsnew, int *jstat)	3919
30.2325	src/s1119.c File Reference	3919
30.2325	Macro Definition Documentation	3920
30.2325.1	s1119	3920
30.2325	Function Documentation	3920
30.2325.2	s1119(double *ecoef, double *et1, double *et2, int ik1, int in1, int ik2, int in2, int *jsimple, int *jind1, int *jind2, int *jstat)	3920
30.2326	src/s1161.c File Reference	3920
30.2326	Macro Definition Documentation	3921
30.2326.1	s1161	3921
30.2326	Function Documentation	3921
30.2326.2	s1161(SISLObject *po1, double *cmax, double aepsge, SISLIntdat **pintdat, int *jstat)	3921
30.2327	src/s1162.c File Reference	3921
30.2327	Macro Definition Documentation	3922
30.2327.1	s1162	3922
30.2327	Function Documentation	3922
30.2327.2	s1162(SISLObject *po1, double *cmax, double aepsge, SISLIntdat **pintdat, vedge, int ilevel, int inum, int *jstat)	3922
30.2328	src/s1172.c File Reference	3922
30.2328	Macro Definition Documentation	3923
30.2328.1	s1172	3923
30.2328	Function Documentation	3923
30.2328.2	s1172(SISLCurve *pcurve, double astart, double aend, double anext, double *cpos, int *jstat)	3923
30.2329	src/s1173.c File Reference	3923

30.2329 ~~30~~ Macro Definition Documentation 3924

 30.2329.1 ~~30~~ §1173 3924

30.2329 ~~30~~ Function Documentation 3924

 30.2329.2 ~~30~~ §1173(SISLPoint *ppoint, SISLSurf *psurf, double aepsge, estart, eend, enext, gpos, int *jstat) 3924

30.2329 ~~30~~ ~~30~~/src/s1174.c File Reference 3924

30.2330 ~~30~~ Macro Definition Documentation 3925

 30.2330.1 ~~30~~ §1174 3925

30.2330 ~~30~~ Function Documentation 3925

 30.2330.2 ~~30~~ §1174(SISLSurf *psurf, estart, eend, enext, gpos, int *jstat) 3925

30.2330 ~~30~~ ~~30~~/src/s1190.c File Reference 3925

30.2331 ~~30~~ Macro Definition Documentation 3926

 30.2331.1 ~~30~~ §1190 3926

30.2331 ~~30~~ Function Documentation 3926

 30.2331.2 ~~30~~ §1190(SISLObject *po1, double *cmax, double aepsge, int *jstat) 3926

30.2331 ~~30~~ ~~30~~/src/s1192.c File Reference 3926

30.2332 ~~30~~ Macro Definition Documentation 3927

 30.2332.1 ~~30~~ §1192 3927

30.2332 ~~30~~ Function Documentation 3927

 30.2332.2 ~~30~~ §1192(SISLObject *po, double aepsge, int *jstat) 3927

30.2332 ~~30~~ ~~30~~/src/s1219.c File Reference 3927

30.2333 ~~30~~ Macro Definition Documentation 3928

 30.2333.1 ~~30~~ §1219 3928

30.2333 ~~30~~ Function Documentation 3928

 30.2333.2 ~~30~~ §1219(double *et, int ik, int in, int *ileft, double ax, int *jstat) 3928

30.2333 ~~30~~ ~~30~~/src/s1220.c File Reference 3928

30.2334 ~~30~~ Macro Definition Documentation 3929

 30.2334.1 ~~30~~ §1220 3929

30.2334 ~~30~~ Function Documentation 3929

 30.2334.2 ~~30~~ §1220(double *et, int ik, int in, int *ileft, double ax, int ider, ebder, int *jstat) 3929

30.2334 ~~30~~ ~~30~~/src/s1221.c File Reference 3929

30.2335	Macro Definition Documentation	3930
30.2335.1	§1221	3930
30.2335	Function Documentation	3930
30.2335.2	§1221(SISLCurve *pc1, int ider, double ax, int *ileft, eder, int *jstat)	3930
30.2335	/src/s1222.c File Reference	3930
30.2336	Macro Definition Documentation	3931
30.2336.1	§1222	3931
30.2336	Function Documentation	3931
30.2336.2	§1222(et, int ik, int in, int ibase, double ax, int ider, ebder, int *jstat)	3931
30.2336	/src/s1223.c File Reference	3931
30.2337	Macro Definition Documentation	3932
30.2337.1	§1223	3932
30.2337	Function Documentation	3932
30.2337.2	§1223(et1, et2, int ik1, int ik2, int in1, int in2, int ibase1, int ibase2, par, int ider1, int ider2, ebder, int *jstat)	3932
30.2337	/src/s1224.c File Reference	3932
30.2338	Macro Definition Documentation	3933
30.2338.1	§1224	3933
30.2338	Function Documentation	3933
30.2338.2	§1224(et1, et2, int ik1, int ik2, int in1, int in2, int ibase1, int ibase2, par, int ider, ebder, int *jstat)	3933
30.2338	/src/s1225.c File Reference	3933
30.2339	Macro Definition Documentation	3934
30.2339.1	§1225	3934
30.2339	Function Documentation	3934
30.2339.2	§1225(SISLCurve *curve, int der, double parvalue, int *leftknot, derive, curvature, double *radius_of_curvature, int *jstat)	3934
30.2340	/src/s1226.c File Reference	3934
30.2340	Macro Definition Documentation	3935
30.2340.1	§1226	3935
30.2340	Function Documentation	3935

30.2340.2	s1226(SISLCurve *curve, int der, double parvalue, int *leftknot, derive, curvature, double *radius_of_curvature, int *jstat)	3935
30.2340	src/s1227.c File Reference	3935
30.2341	Macro Definition Documentation	3936
30.2341.1	s1227	3936
30.2341	Function Documentation	3936
30.2341.2	s1227(SISLCurve *pc1, int ider, double ax, int *ileft, eder, int *jstat)	3936
30.2340	src/s1231.c File Reference	3936
30.2342	Macro Definition Documentation	3937
30.2342.1	s1231	3937
30.2342	Function Documentation	3937
30.2342.2	s1231(SISLCurve *pc1, double apar, SISLCurve **rcnew1, SISLCurve **rcnew2, int *jstat)	3937
30.2340	src/s1232.c File Reference	3937
30.2343	Macro Definition Documentation	3938
30.2343.1	s1232	3938
30.2343	Function Documentation	3938
30.2343.2	s1232(et1, int in, int ik, double afak1, double afak2, et2, int *jstat)	3938
30.2340	src/s1233.c File Reference	3938
30.2344	Macro Definition Documentation	3939
30.2344.1	s1233	3939
30.2344	Function Documentation	3939
30.2344.2	s1233(SISLCurve *pc, double afak1, double afak2, SISLCurve **rc, int *jstat)	3939
30.2340	src/s1235.c File Reference	3939
30.2345	Macro Definition Documentation	3940
30.2345.1	s1235	3940
30.2345	Function Documentation	3940
30.2345.2	s1235(et, int in, int ik, int *jnbreak, double **gbreak, int *jstat)	3940
30.2340	src/s1236.c File Reference	3940
30.2346	Macro Definition Documentation	3941
30.2346.1	s1236	3941

30.2346	Function Documentation	3941
30.2346.2	s1236(et, int in, int ik, int inpar, epar, int *jstat)	3941
30.2346	/src/s1237.c File Reference	3941
30.2347	Macro Definition Documentation	3942
30.2347.1	s1237	3942
30.2347	Function Documentation	3942
30.2347.2	s1237(SISLSurf *psurf, int inmb1, int inmb2, double aepscu, int *jstat)	3942
30.2346	/src/s1238.c File Reference	3942
30.2348	Macro Definition Documentation	3943
30.2348.1	s1238	3943
30.2348	Function Documentation	3943
30.2348.2	s1238(SISLSurf *psurf, SISLCurve *pcurve, int inmb1, int inmb2, double aepsco, double aepscu, int *jstat)	3943
30.2346	/src/s1239.c File Reference	3943
30.2349	Macro Definition Documentation	3944
30.2349.1	s1239	3944
30.2349	Function Documentation	3944
30.2349.2	s1239(SISLCurve *pcpar, int ipar, double apar, SISLCurve *pcurve, double aepsco, double aepsge, SISLCurve **vpartc, int imax, int *jpartc, int *jstat)	3944
30.2350	/src/s1240.c File Reference	3944
30.2350	Macro Definition Documentation	3945
30.2350.1	s1240	3945
30.2350	Function Documentation	3945
30.2350.2	s1240(SISLCurve *pcurve, double aepsge, double *clength, int *jstat)	3945
30.2351	/src/s1241.c File Reference	3945
30.2351	Macro Definition Documentation	3946
30.2351.1	s1241	3946
30.2351	Function Documentation	3946
30.2351.2	s1241(SISLCurve *pcurve, point, int dim, double epsge, double *area, int *stat)	3946
30.2351	/src/s1243.c File Reference	3946
30.2352	Macro Definition Documentation	3947

30.2352.1	§1243	3947
30.2352	Function Documentation	3947
30.2352.2	§1243(SISLCurve *pcurve, point, int dim, double epsge, weight, double *area, double *moment, int *stat)	3947
30.2353	src/s1244.c File Reference	3947
30.2353	Macro Definition Documentation	3948
30.2353.1	§1244	3948
30.2353	Function Documentation	3948
30.2353.2	§1244(knots, int knot_reg, int first_order, int second_order, int in, int first_index, int second_index, double *integral, int *stat)	3948
30.2354	src/s1245.c File Reference	3948
30.2354	Macro Definition Documentation	3949
30.2354.1	§1245	3949
30.2354	Function Documentation	3949
30.2354.2	§1245(coef, int ik, int dim, point, double local_tol, int depth, weight, double *area, double *moment, int *stat)	3949
30.2355	src/s1251.c File Reference	3949
30.2355	Macro Definition Documentation	3950
30.2355.1	§1251	3950
30.2355	Function Documentation	3950
30.2355.2	§1251(SISLCurve *pcurve, double aepsco, double *clength, int *jstat)	3950
30.2356	src/s1252.c File Reference	3950
30.2356	Macro Definition Documentation	3951
30.2356.1	§1252	3951
30.2356	Function Documentation	3951
30.2356.2	§1252(SISLCurve *pcurve, double aepsge, double astart, double *cpos, int *jstat)	3951
30.2357	src/s1301.c File Reference	3951
30.2357	Macro Definition Documentation	3952
30.2357.1	§1301	3952
30.2357	Function Documentation	3952
30.2357.2	§1301(double areler, double angle, int idim, SISLCurve **pc, int *jstat)	3952
30.2358	src/s1302.c File Reference	3952

30.2358	Macro Definition Documentation	3953
30.2358.1	§1302	3953
30.2358	Function Documentation	3953
30.2358.2	§1302(SISLCurve *pc, double aepsge, double angle, ep, eaxis, SISLSurf **rs, int *jstat)	3953
30.2358	src/s1303.c File Reference	3953
30.2359	Macro Definition Documentation	3954
30.2359.1	§1303	3954
30.2359	Function Documentation	3954
30.2359.2	§1303(epstrt, double aepsge, double angle, epcnt, eaxis, int idim, SISLCurve **rc, int *jstat)	3954
30.2360	src/s1304.c File Reference	3954
30.2360	Macro Definition Documentation	3955
30.2360.1	§1304	3955
30.2360	Function Documentation	3955
30.2360.2	§1304(ep, eq, eparp, eparq, egeo3d, egeop, egeoq, int *jstat)	3955
30.2361	src/s1305.c File Reference	3955
30.2361	Macro Definition Documentation	3956
30.2361.1	§1305	3956
30.2361	Function Documentation	3956
30.2361.2	§1305(epar1, epar2, eval1, eval2, int *jbound, gpar, int *jstat)	3956
30.2362	src/s1306.c File Reference	3956
30.2362	Macro Definition Documentation	3957
30.2362.1	§1306	3957
30.2362	Function Documentation	3957
30.2362.2	§1306(ep, eparp, eimpli, int ideg, egeo3d, egeop, int *jstat)	3957
30.2363	src/s1307.c File Reference	3957
30.2363	Macro Definition Documentation	3958
30.2363.1	§1307	3958
30.2363	Function Documentation	3958
30.2363.2	§1307(ep, int idim, egeo, int *jstat)	3958

30.2364	/src/s1308.c File Reference	3958
30.2364	Macro Definition Documentation	3959
30.2364.1	S1308	3959
30.2364	Function Documentation	3959
30.2364.2	s1308(ep, int idim, eimpli, int ideg, enorm, int *jstat)	3959
30.2365	/src/s1309.c File Reference	3959
30.2365	Macro Definition Documentation	3960
30.2365.1	S1309	3960
30.2365	Function Documentation	3960
30.2365.2	s1309(epnt, edir, eimpli, int ideg, int *jstat)	3960
30.2366	/src/s1310.c File Reference	3960
30.2366	Macro Definition Documentation	3961
30.2366.1	S1310	3961
30.2366	Function Documentation	3961
30.2366.2	s1310(SISLSurf *psurf1, SISLSurf *psurf2, SISLIntcurve *pinter, double aepsge, double amax, int icur, int igrph, int *jstat)	3961
30.2367	/src/s1311.c File Reference	3961
30.2367	Macro Definition Documentation	3962
30.2367.1	S1311	3962
30.2367	Function Documentation	3962
30.2367.2	s1311(double arad, double aepsge, double amax, int *jstat)	3962
30.2368	/src/s1312.c File Reference	3962
30.2368	Macro Definition Documentation	3963
30.2368.1	S1312	3963
30.2368	Function Documentation	3963
30.2368.2	s1312(egeo, int idim, int inbinf, int ipar, epar, SISLCurve **rcurve, int *jstat)	3963
30.2369	/src/s1313.c File Reference	3963
30.2369	Macro Definition Documentation	3964
30.2369.1	S1313	3964
30.2369	Function Documentation	3964

30.2369.2	§1313(SISLSurf *ps1, eimpli, int ideg, double aepsco, double aepsge, double amax, SISLIntcurve *pintcr, int icur, int igrph, int *jstat)	3964
30.2370	§1/src/s1314.c File Reference	3964
30.2370	Macro Definition Documentation	3965
30.2370.1	§1314	3965
30.2370	Function Documentation	3965
30.2370.2	§1314(SISLSurf *ps1, double *epoint, double *enorm, int idim, double aepsco, double aepsge, double amax, SISLIntcurve *pintcr, int icur, int igrph, int *jstat)	3965
30.2371	§1/src/s1315.c File Reference	3965
30.2371	Macro Definition Documentation	3966
30.2371.1	§1315	3966
30.2371	Function Documentation	3966
30.2371.2	§1315(SISLSurf *ps1, double *ecentr, double aradiu, int idim, double aepsco, double aepsge, double amax, SISLIntcurve *pintcr, int icur, int igrph, int *jstat)	3966
30.2372	§1/src/s1316.c File Reference	3966
30.2372	Macro Definition Documentation	3967
30.2372.1	§1316	3967
30.2372	Function Documentation	3967
30.2372.2	§1316(SISLSurf *ps1, double *epoint, double *edirec, double aradiu, int idim, double aepsco, double aepsge, double amax, SISLIntcurve *pintcr, int icur, int igrph, int *jstat)	3967
30.2373	§1/src/s1317.c File Reference	3967
30.2373	Macro Definition Documentation	3968
30.2373.1	§1317	3968
30.2373	Function Documentation	3968
30.2373.2	§1317(SISLSurf *ps1, double *etop, double *eaxis, double *econe, int idim, double aepsco, double aepsge, double amax, SISLIntcurve *pintcr, int icur, int igrph, int *jstat)	3968
30.2374	§1/src/s1318.c File Reference	3968
30.2374	Macro Definition Documentation	3969
30.2374.1	§1318	3969
30.2374	Function Documentation	3969

30.2374.2	§1318(SISLSurf *ps1, double *ecentr, double *enorm, double abigr, double as- malr, int idim, double aepsco, double aepsge, double amax, SISLIntcurve *pinctr, int icur, int igrph, int *jstat)	3969
30.2375	src/s1319.c File Reference	3969
30.2375	Macro Definition Documentation	3970
30.2375.1	§1319	3970
30.2375	Function Documentation	3970
30.2375.2	§1319(SISLSurf *ps1, double *eview, int idim, double aepsco, double aepsge, double amax, SISLIntcurve *pinctr, int icur, int igrph, int *jstat)	3970
30.2376	src/s1320.c File Reference	3970
30.2376	Macro Definition Documentation	3971
30.2376.1	§1320	3971
30.2376	Function Documentation	3971
30.2376.2	§1320(SISLSurf *psurf, earray, int inarr, int ratflag, SISLSurf **rsurf, int *jstat)	3971
30.2377	src/s1321.c File Reference	3971
30.2377	Macro Definition Documentation	3972
30.2377.1	§1321	3972
30.2377	Function Documentation	3972
30.2377.2	§1321(ecentr, double aradiu, int idim, int inumb, carray, int *jstat)	3972
30.2378	src/s1322.c File Reference	3972
30.2378	Macro Definition Documentation	3973
30.2378.1	§1322	3973
30.2378	Function Documentation	3973
30.2378.2	§1322(epoint, edirec, double aradiu, int idim, int inumb, carray, int *jstat)	3973
30.2379	src/s1323.c File Reference	3973
30.2379	Macro Definition Documentation	3974
30.2379.1	§1323	3974
30.2379	Function Documentation	3974
30.2379.2	§1323(etop, eaxis, econe, int idim, int inumb, carray, int *jstat)	3974
30.2380	src/s1324.c File Reference	3974
30.2380	Macro Definition Documentation	3975
30.2380.1	§1324	3975

30.2380	Function Documentation	3975
30.2380.2	s1324(acentr, double aradiu, enorm, int idim, carray, int *jstat)	3975
30.2380	src/s1325.c File Reference	3975
30.2381	Macro Definition Documentation	3976
30.2381.1	s1325	3976
30.2381	Function Documentation	3976
30.2381.2	s1325(double aradiu, double angle)	3976
30.2381	src/s1327.c File Reference	3976
30.2382	Macro Definition Documentation	3977
30.2382.1	s1327	3977
30.2382	Function Documentation	3977
30.2382.2	s1327(SISLCurve *pcold, epoint, enorm1, enorm2, int idim, SISLCurve **rcnew, int *jstat)	3977
30.2382	src/s1328.c File Reference	3977
30.2383	Macro Definition Documentation	3978
30.2383.1	s1328	3978
30.2383	Function Documentation	3978
30.2383.2	s1328(SISLSurf *psold, epoint, enorm1, enorm2, int idim, SISLSurf **rsnew, int *jstat)	3978
30.2383	src/s1329.c File Reference	3978
30.2384	Macro Definition Documentation	3979
30.2384.1	s1329	3979
30.2384	Function Documentation	3979
30.2384.2	s1329(SISLSurf *psold, epoint, enorm, int idim, SISLSurf **rsnew, int *jstat)	3979
30.2385	src/s1330.c File Reference	3979
30.2385	Macro Definition Documentation	3980
30.2385.1	s1330	3980
30.2385	Function Documentation	3980
30.2385.2	s1330(epar11, epar12, epar21, epar22, eval11, eval12, eval21, eval22, int *jbound, gpar1, gpar2, int *jstat)	3980
30.2385	src/s1331.c File Reference	3980
30.2386	Macro Definition Documentation	3981

30.2386.1	§1331	3981
30.2386	Function Documentation	3981
30.2386.2	§1331(ep, eimpli, int ideg, int nder, gder, gnorm, int *jstat)	3981
30.2386	src/s1332.c File Reference	3981
30.2387	Macro Definition Documentation	3982
30.2387.1	§1332	3982
30.2387	Function Documentation	3982
30.2387.2	§1332(SISLCurve *pc1, SISLCurve *pc2, double aepsge, ep, SISLSurf **rs, int *jstat)	3982
30.2387	src/s1333.c File Reference	3982
30.2388	Macro Definition Documentation	3983
30.2388.1	§1333	3983
30.2388	Function Documentation	3983
30.2388.2	§1333(int inbcrv, vpcurv, nctyp, double astpar, int iopen, int iord2, int iflag, SISL← Surf **rsurf, double **gpar, int *jstat)	3983
30.2388	src/s1333count.c File Reference	3983
30.2389	Macro Definition Documentation	3984
30.2389.1	§1333_COUNT	3984
30.2389	Function Documentation	3984
30.2389.2	§1333_count(int inbcrv, vpcurv, int *jcont, int *jstat)	3984
30.2390	src/s1333cycli.c File Reference	3984
30.2390	Macro Definition Documentation	3985
30.2390.1	§1333_CYCLIC	3985
30.2390	Function Documentation	3985
30.2390.2	§1333_cyclic(SISLSurf *vsurf, int icontr, int *jstat)	3985
30.2391	src/s1334.c File Reference	3985
30.2391	Macro Definition Documentation	3986
30.2391.1	§1334	3986
30.2391	Function Documentation	3986
30.2391.2	§1334(epoint, int inbpnt, int idim, nptyp, int icnsta, int icnend, int iopen, int ik, double astpar, double *cendpar, SISLCurve **rc, double **gpar, int *jnbpar, int *jstat)	3986

30.2392	/src/s1340.c File Reference	3986
30.2392	Macro Definition Documentation	3987
30.2392.1	§1340	3987
30.2392	Function Documentation	3987
30.2392.2	§1340(SISLCurve *oldcurve, eps, int startfix, int endfix, double epsco, int itmax, SISLCurve **newcurve, maxerr, int *stat)	3987
30.2393	/src/s1341.c File Reference	3987
30.2393	Macro Definition Documentation	3988
30.2393.1	§1341	3988
30.2393	Function Documentation	3988
30.2393.2	§1341(ep, int im, int idim, int ipar, epar, eeps, int ilend, int irend, double afctol, double aepsco, int itmax, int ik, SISLCurve **rc, emxerr, int *jstat)	3988
30.2394	/src/s1342.c File Reference	3988
30.2394	Macro Definition Documentation	3989
30.2394.1	§1342	3989
30.2394	Function Documentation	3989
30.2394.2	§1342(ep, ev, int im, int idim, int ipar, epar, eeps, int ilend, int irend, double aepsco, int itmax, SISLCurve **rc, emxerr, int *jstat)	3989
30.2395	/src/s1343.c File Reference	3989
30.2395	Macro Definition Documentation	3990
30.2395.1	§1343	3990
30.2395	Function Documentation	3990
30.2395.2	§1343(SISLCurve *pc, eeps, int ilend, int irend, double aepsco, int itmax, SISLCurve **rc, int *jstat)	3990
30.2396	/src/s1345.c File Reference	3990
30.2396	Macro Definition Documentation	3991
30.2396.1	§1345	3991
30.2396	Function Documentation	3991
30.2396.2	§1345(SISLSurf *oldsurf, eps, edgefix, edgeps, double epsco, int opt, int itmax, SISLSurf **newsurf, maxerr, int *stat)	3991
30.2397	/src/s1346.c File Reference	3991
30.2397	Macro Definition Documentation	3992
30.2397.1	§1346	3992

30.2397	Function Documentation	3992
30.2397.2	s1346(ep, int im1, int im2, int idim, int ipar, epar1, epar2, eeps, nend, edgeps, double afctol, double aepsco, int iopt, int itmax, int ik1, int ik2, SISLSurf **rs, emxerr, int *jstat)	3992
30.2398	/src/s1347.c File Reference	3992
30.2398	Macro Definition Documentation	3993
30.2398.1	s1347	3993
30.2398	Function Documentation	3993
30.2398.2	s1347(ep, etang1, etang2, eder11, int im1, int im2, int idim, int ipar, epar1, epar2, eeps, nend, edgeps, double aepsco, int iopt, int itmax, SISLSurf **rs, emxerr, int *jstat)	3993
30.2399	/src/s1348.c File Reference	3993
30.2399	Macro Definition Documentation	3994
30.2399.1	s1348	3994
30.2399	Function Documentation	3994
30.2399.2	s1348(SISLSurf *ps, eeps, nend, edgeps, double aepsco, int iopt, int itmax, SISLSurf **rs, int *jstat)	3994
30.2400	/src/s1349.c File Reference	3994
30.2400	Macro Definition Documentation	3995
30.2400.1	s1349	3995
30.2400	Function Documentation	3995
30.2400.2	s1349(int inbcrv, vpcrv, int *jstat)	3995
30.2401	/src/s1350.c File Reference	3995
30.2401	Macro Definition Documentation	3996
30.2401.1	s1350	3996
30.2401	Function Documentation	3996
30.2401.2	s1350(ep, epar, int im, int idim, int ik, SISLCurve **rc, int *jstat)	3996
30.2402	/src/s1351.c File Reference	3996
30.2402	Macro Definition Documentation	3997
30.2402.1	s1351	3997
30.2402	Function Documentation	3997
30.2402.2	s1351(ep, int ipar, int im, int idim, int ik, SISLCurve **rc, int *jstat)	3997
30.2403	/src/s1352.c File Reference	3997

30.2403	Macro Definition Documentation	3998
30.2403.1	§1352	3998
30.2403	Function Documentation	3998
30.2403.2	§1352(t, int n, int k, inteps, lefteps, righteps, int dim, int leftfix, int rightfix, eps, int *stat)	3998
30.2403	src/s1353.c File Reference	3998
30.2404	Macro Definition Documentation	3999
30.2404.1	§1353	3999
30.2404	Function Documentation	3999
30.2404.2	§1353(SISLCurve *curve, eps, rank_info *ranking, int *stat)	3999
30.2404	src/s1354.c File Reference	3999
30.2405	Macro Definition Documentation	4000
30.2405.1	§1354	4000
30.2405	Function Documentation	4000
30.2405.2	§1354(SISLCurve *oldcurve, SISLCurve *rankcurve, rank_info *ranking, eps, epsco, int startfix, int endfix, int mini, int maxi, SISLCurve **newcurve, maxerr, int *stat)	4000
30.2405	src/s1355.c File Reference	4000
30.2406	Macro Definition Documentation	4001
30.2406.1	§1355	4001
30.2406	Function Documentation	4001
30.2406.2	§1355(SISLCurve *pc, eeps, double **epar, int *im, int *jstat)	4001
30.2406	src/s1356.c File Reference	4001
30.2407	Macro Definition Documentation	4002
30.2407.1	§1356	4002
30.2407	Function Documentation	4002
30.2407.2	§1356(epoint, int inbpt, int idim, nptyp, int icnsta, int icnend, int iopen, int ik, double astpar, double *cendpar, SISLCurve **rc, double **gpar, int *jnbpar, int *jstat)	4002
30.2407	src/s1357.c File Reference	4002
30.2408	Macro Definition Documentation	4003
30.2408.1	§1357	4003
30.2408	Function Documentation	4003

30.2408.2	s1357(epoint, int inbpnt, int idim, ntype, epar, int icnsta, int icnend, int iopen, int ik, double astpar, double *cendpar, SISLCurve **rc, double **gpar, int *jnbpar, int *jstat)	4003
30.2408	src/s1358.c File Reference	4003
30.2409	Macro Definition Documentation	4004
30.2409.1	S1358	4004
30.2409	Function Documentation	4004
30.2409.2	s1358(epoint, int inbpnt, int idim, ntype, epar, int icnsta, int icnend, int iopen, int ik, double astpar, double *cendpar, SISLCurve **rc, double **gpar, int *jnbpar, int *jstat)	4004
30.2409	src/s1359.c File Reference	4004
30.2410	Macro Definition Documentation	4005
30.2410.1	S1359	4005
30.2410	Function Documentation	4005
30.2410.2	s1359(egeo, double aepsge, int idim, int inbinf, int ipar, epar, SISLCurve **rcurve, int *jstat)	4005
30.2410	src/s1360.c File Reference	4005
30.2411	Macro Definition Documentation	4006
30.2411.1	S1360	4006
30.2411	Function Documentation	4006
30.2411.2	s1360(SISLCurve *pc, double aoffset, double aepsge, enorm, double amax, int idim, SISLCurve **rc, int *jstat)	4006
30.2411	src/s1361.c File Reference	4006
30.2412	Macro Definition Documentation	4007
30.2412.1	S1361	4007
30.2412	Function Documentation	4007
30.2412.2	s1361(epnt1, epnt2, int idim, gmidd, gmtang, int *jstat)	4007
30.2412	src/s1362.c File Reference	4007
30.2413	Macro Definition Documentation	4008
30.2413.1	S1362	4008
30.2413	Function Documentation	4008
30.2413.2	s1362(SISLCurve *pc1, double aoffset, enorm, int idim, int ider, double ax, int *ileft, eder, int *jstat)	4008

30.2414	src/s1363.c File Reference	4008
30.2414	Macro Definition Documentation	4009
30.2414.1	S1363	4009
30.2414	Function Documentation	4009
30.2414.2	s1363(SISLCurve *pc, double *cmin, double *cmax, int *jstat)	4009
30.2415	src/s1364.c File Reference	4009
30.2415	Macro Definition Documentation	4010
30.2415.1	S1364	4010
30.2415	Function Documentation	4010
30.2415.2	s1364(SISLCurve *pc, double aepsge, int *jstat)	4010
30.2416	src/s1365.c File Reference	4010
30.2416	Macro Definition Documentation	4011
30.2416.1	S1365	4011
30.2416	Function Documentation	4011
30.2416.2	s1365(SISLSurf *ps, double aoffset, double aepsge, double amax, int idim, SISLSurf **rs, int *jstat)	4011
30.2417	src/s1366.c File Reference	4011
30.2417	Macro Definition Documentation	4012
30.2417.1	S1366	4012
30.2417	Function Documentation	4012
30.2417.2	s1366(SISLSurf *ps, double aoffset, double aepsge, double amax, int idim, double *eknot13, int in13, int ik13, double *eknot24, int in24, int ik24, SISLSurf **rs, int *jstat)	4012
30.2418	src/s1367.c File Reference	4012
30.2418	Macro Definition Documentation	4013
30.2418.1	S1366	4013
30.2418	Function Documentation	4013
30.2418.2	s1367(SISLSurf *ps, double aoffset, double aepsge, int idim, epar, int ider, int *ilfs, int *ilft, eder, int *jstat)	4013
30.2419	src/s1369.c File Reference	4013
30.2419	Macro Definition Documentation	4014
30.2419.1	S1369	4014

30.2419 ~~2~~unction Documentation 4014

 30.2419.2 ~~s~~1369(SISLSurf *ps, ecentr, enorm, double abigr, double asmair, int idim, double
 aepsco, double aepsge, int *jpt, double **gpar, int *jcrv, SISLIntcurve ***wcurve,
 int *jstat) 4014

30.2420 ~~3~~/src/s1370.c File Reference 4014

30.2420 Macro Definition Documentation 4015

 30.2420.1 ~~S~~1370 4015

30.2420 ~~2~~unction Documentation 4015

 30.2420.2 ~~s~~1370(SISLCurve *pcurv, earray, int idim, int inarr, int ratflag, SISLCurve **rcurv,
 int *jstat) 4015

30.2421 ~~4~~/src/s1371.c File Reference 4015

30.2421 Macro Definition Documentation 4016

 30.2421.1 ~~S~~1371 4016

30.2421 ~~2~~unction Documentation 4016

 30.2421.2 ~~s~~1371(SISLCurve *pc1, ecentr, double aradiu, int idim, double aepsco, double
 aepsge, int *jpt, double **gpar, int *jcrv, SISLIntcurve ***wcurve, int *jstat) . . 4016

30.2422 ~~5~~/src/s1372.c File Reference 4016

30.2422 Macro Definition Documentation 4017

 30.2422.1 ~~S~~1372 4017

30.2422 ~~2~~unction Documentation 4017

 30.2422.2 ~~s~~1372(SISLCurve *pc1, epoint, edirec, double aradiu, int idim, double aepsco,
 double aepsge, int *jpt, double **gpar, int *jcrv, SISLIntcurve ***wcurve, int *jstat) . 4017

30.2423 ~~6~~/src/s1373.c File Reference 4017

30.2423 Macro Definition Documentation 4018

 30.2423.1 ~~S~~1373 4018

30.2423 ~~2~~unction Documentation 4018

 30.2423.2 ~~s~~1373(SISLCurve *pc1, etop, eaxis, econc, int idim, double aepsco, double
 aepsge, int *jpt, double **gpar, int *jcrv, SISLIntcurve ***wcurve, int *jstat) . . 4018

30.2424 ~~7~~/src/s1374.c File Reference 4018

30.2424 Macro Definition Documentation 4019

 30.2424.1 ~~S~~1374 4019

30.2424 ~~2~~unction Documentation 4019

30.2424.2	s1374(SISLCurve *pc1, earray, int idim, double aepsco, double aepsge, int *jpt, double **gpar, int *jcrv, SISLIntcurve ***wcurve, int *jstat)	4019
30.2425	/src/s1375.c File Reference	4019
30.2425	Macro Definition Documentation	4020
30.2425.1	s1375	4020
30.2425	Function Documentation	4020
30.2425.2	s1375(SISLCurve *pc1, ecentr, enorm, double abigr, double asmalr, int idim, double aepsco, double aepsge, int *jpt, double **gpar, int *jcrv, SISLIntcurve ***wcurve, int *jstat)	4020
30.2426	/src/s1376.c File Reference	4020
30.2426	Macro Definition Documentation	4021
30.2426.1	s1376	4021
30.2426	Function Documentation	4021
30.2426.2	s1376(et, int in, int ik, double **gt, int *jkn, int *jkk, int *jstat)	4021
30.2427	/src/s1377.c File Reference	4021
30.2427	Macro Definition Documentation	4022
30.2427.1	s1377	4022
30.2427	Function Documentation	4022
30.2427.2	s1377(SISLCurve *pcurv, econic, int ideg, int idim, SISLCurve **rcurv, int *jstat)	4022
30.2428	/src/s1378.c File Reference	4022
30.2428	Macro Definition Documentation	4023
30.2428.1	s1378	4023
30.2428	Function Documentation	4023
30.2428.2	s1378(SISLSurf *psurf, econic, int ideg, int idim, SISLSurf **rsurf, int *jstat)	4023
30.2429	/src/s1379.c File Reference	4023
30.2429	Macro Definition Documentation	4024
30.2429.1	s1379	4024
30.2429	Function Documentation	4024
30.2429.2	s1379(ep, ev, epar, int im, int idim, SISLCurve **rcurve, int *jstat)	4024
30.2430	/src/s1380.c File Reference	4024
30.2430	Macro Definition Documentation	4025
30.2430.1	s1380	4025

30.2430 ~~2~~unction Documentation 4025

 30.2430.2 ~~s~~1380(ep, ev, int im, int idim, int ipar, SISLCurve ~~**~~rcurve, int ~~*~~jstat) 4025

30.2431 ~~3~~l/src/s1381.c File Reference 4025

 30.2431 ~~M~~acro Definition Documentation 4026

 30.2431.1 ~~S~~1381 4026

 30.2431 ~~F~~unction Documentation 4026

 30.2431.2 ~~s~~1381(et, int in, int ik, double ~~**~~gt, int ~~*~~jkn, int jkk, int ~~*~~jstat) 4026

30.2432 ~~3~~l/src/s1382.c File Reference 4026

 30.2432 ~~M~~acro Definition Documentation 4027

 30.2432.1 ~~S~~1382 4027

 30.2432 ~~F~~unction Documentation 4027

 30.2432.2 ~~s~~1382(SISLSurf ~~*~~psurf, eview, int idim, SISLSurf ~~**~~rsurf, int ~~*~~jstat) 4027

30.2433 ~~3~~l/src/s1383.c File Reference 4027

 30.2433 ~~M~~acro Definition Documentation 4028

 30.2433.1 ~~S~~1383 4028

 30.2433 ~~F~~unction Documentation 4028

 30.2433.2 ~~s~~1383(SISLSurf ~~*~~psurf, SISLCurve ~~*~~pcurv, double aepsge, double amax, int ider, SISLCurve ~~**~~rcpos, SISLCurve ~~**~~rcder1, SISLCurve ~~**~~rcder2, int ~~*~~jstat) 4028

30.2434 ~~3~~l/src/s1384.c File Reference 4028

 30.2434 ~~M~~acro Definition Documentation 4029

 30.2434.1 ~~S~~1384 4029

 30.2434 ~~F~~unction Documentation 4029

 30.2434.2 ~~s~~1384(SISLCurve ~~*~~pcurve, SISLSurf ~~*~~psurf, int idim, int iside, double ax, int ~~*~~ileftc, int ~~*~~ilefts1, int ~~*~~ilefts2, eder, ederu, ederv, edern, int ~~*~~jstat) 4029

30.2435 ~~3~~l/src/s1385.c File Reference 4029

 30.2435 ~~M~~acro Definition Documentation 4030

 30.2435.1 ~~S~~1385 4030

 30.2435 ~~F~~unction Documentation 4030

 30.2435.2 ~~s~~1385(ep0, ept, ep1, double as, int idim, double aepsge, SISLCurve ~~**~~rc, int ~~*~~jstat) 4030

30.2436 ~~3~~l/src/s1386.c File Reference 4030

 30.2436 ~~M~~acro Definition Documentation 4031

30.2436.1	§1386	4031
30.2436	Function Documentation	4031
30.2436.2	§1386(SISLSurf *ps, int ider1, int ider2, SISLSurf **rsnew, int *jstat)	4031
30.2436	src/s1387.c File Reference	4031
30.2437	Macro Definition Documentation	4032
30.2437.1	§1387	4032
30.2437	Function Documentation	4032
30.2437.2	§1387(SISLSurf *ps, int ik1, int ik2, SISLSurf **rsnew, int *jstat)	4032
30.2437	src/s1388.c File Reference	4032
30.2438	Macro Definition Documentation	4033
30.2438.1	§1388	4033
30.2438	Function Documentation	4033
30.2438.2	§1388(SISLSurf *ps1, gcoons, int *jnumb1, int *jnumb2, int *jdim, int *jstat)	4033
30.2438	src/s1389.c File Reference	4033
30.2439	Macro Definition Documentation	4034
30.2439.1	§1389	4034
30.2439	Function Documentation	4034
30.2439.2	§1389(SISLCurve *pc1, gcubic, int *jnumb, int *jdim, int *jstat)	4034
30.2440	src/s1390.c File Reference	4034
30.2440	Macro Definition Documentation	4035
30.2440.1	§1390	4035
30.2440	Function Documentation	4035
30.2440.2	§1390(pc1, SISLSurf **ps1, nder, int *jstat)	4035
30.2441	src/s1391.c File Reference	4035
30.2441	Macro Definition Documentation	4036
30.2441.1	§1391	4036
30.2441	Predef Documentation	4036
30.2441.2	shapeProc	4036
30.2441	Function Documentation	4036
30.2441.3	§1391(SISLCurve **pc, SISLSurf ***ws, int icurv, nder, int *jstat)	4036

30.2441 [src/s1393.c File Reference](#) 4036

 30.2442 [Macro Definition Documentation](#) 4037

 30.2442.1 [S1393](#) 4037

 30.2442 [Function Documentation](#) 4037

 30.2442.2 [s1393\(int n1, pc1, sc1, ec1, int *jstat\)](#) 4037

30.2442 [src/s1399.c File Reference](#) 4037

 30.2443 [Macro Definition Documentation](#) 4038

 30.2443.1 [S1399](#) 4038

 30.2443 [Function Documentation](#) 4038

 30.2443.2 [s1399\(SISLCurve *pc, double astart, double astop\)](#) 4038

30.2443 [src/s1401.c File Reference](#) 4038

 30.2444 [Macro Definition Documentation](#) 4039

 30.2444.1 [S1401](#) 4039

 30.2444 [Function Documentation](#) 4039

 30.2444.2 [s1401\(vcurve, etwist, SISLSurf **rsurf, int *jstat\)](#) 4039

30.2444 [src/s1421.c File Reference](#) 4039

 30.2445 [Macro Definition Documentation](#) 4040

 30.2445.1 [S1421](#) 4040

 30.2445 [Function Documentation](#) 4040

 30.2445.2 [s1421\(SISLSurf *ps1, int ider, epar, int *ilfs, int *ilft, eder, enorm, int *jstat\)](#) 4040

30.2445 [src/s1422.c File Reference](#) 4040

 30.2446 [Macro Definition Documentation](#) 4041

 30.2446.1 [S1422](#) 4041

 30.2446 [Function Documentation](#) 4041

 30.2446.2 [s1422\(SISLSurf *ps1, int ider, int iside1, int iside2, epar, int *ilfs, int *ilft, eder, enorm, int *jstat\)](#) 4041

30.2446 [src/s1424.c File Reference](#) 4041

 30.2447 [Macro Definition Documentation](#) 4042

 30.2447.1 [S1424](#) 4042

 30.2447 [Function Documentation](#) 4042

 30.2447.2 [s1424\(SISLSurf *ps1, int ider1, int ider2, epar, int *ileft1, int *ileft2, eder, int *jstat\)](#) 4042

30.2448	/src/s1425.c File Reference	4042
30.2448	Macro Definition Documentation	4043
30.2448.1	S1425	4043
30.2448	Function Documentation	4043
30.2448.2	s1425(SISLSurf *ps1, int ilder1, int ilder2, int iside1, int iside2, epar, int *ileft1, int *ileft2, ilder, int *jstat)	4043
30.2449	/src/s1435.c File Reference	4043
30.2449	Macro Definition Documentation	4044
30.2449.1	S1435	4044
30.2449	Function Documentation	4044
30.2449.2	s1435(SISLSurf *ps1, int iedge, SISLCurve **rledge, double *cpar, int *jstat)	4044
30.2450	/src/s1436.c File Reference	4044
30.2450	Macro Definition Documentation	4045
30.2450.1	S1436	4045
30.2450	Function Documentation	4045
30.2450.2	s1436(SISLSurf *ps1, double apar, SISLCurve **rcurve, int *jstat)	4045
30.2451	/src/s1437.c File Reference	4045
30.2451	Macro Definition Documentation	4046
30.2451.1	S1437	4046
30.2451	Function Documentation	4046
30.2451.2	s1437(SISLSurf *ps1, double apar, SISLCurve **rcurve, int *jstat)	4046
30.2452	/src/s1438.c File Reference	4046
30.2452	Macro Definition Documentation	4047
30.2452.1	S1438	4047
30.2452	Function Documentation	4047
30.2452.2	s1438(SISLCurve *pc, int iedge, SISLPoint **rpedge, double *cpar, int *jstat)	4047
30.2453	/src/s1439.c File Reference	4047
30.2453	Macro Definition Documentation	4048
30.2453.1	S1439	4048
30.2453	Function Documentation	4048
30.2453.2	s1439(SISLSurf *ps1, double apar, int idirec, SISLCurve **rcurve, int *jstat)	4048

30.2454 [src/s1440.c File Reference](#) 4048

 30.2454 [Macro Definition Documentation](#) 4049

 30.2454.1 [S1440](#) 4049

 30.2454 [Function Documentation](#) 4049

 30.2454.2 [s1440\(SISLSurf *ps1, SISLSurf **rs2, int *jstat\)](#) 4049

30.2455 [src/s1450.c File Reference](#) 4049

 30.2455 [Macro Definition Documentation](#) 4050

 30.2455.1 [S1450](#) 4050

 30.2455 [Function Documentation](#) 4050

 30.2455.2 [s1450\(SISLSurf *ps1, double aepsge, int *jclos1, int *jclos2, int *jdgen1, int *jdgen2, int *jdgen3, int *jdgen4, int *jstat\)](#) 4050

30.2456 [src/s1451.c File Reference](#) 4050

 30.2456 [Macro Definition Documentation](#) 4051

 30.2456.1 [S1451](#) 4051

 30.2456 [Function Documentation](#) 4051

 30.2456.2 [s1451\(SISLCurve *pc1, double aepsge, int *jdgen, int *jstat\)](#) 4051

30.2457 [src/s1452.c File Reference](#) 4051

 30.2457 [Macro Definition Documentation](#) 4052

 30.2457.1 [S1452](#) 4052

 30.2457 [Function Documentation](#) 4052

 30.2457.2 [s1452\(SISLSurf *ps, double aepsge, double aoffset, SISLSurf **rs, int *jstat\)](#) . . 4052

30.2458 [src/s1500.c File Reference](#) 4052

 30.2458 [Macro Definition Documentation](#) 4053

 30.2458.1 [S1500](#) 4053

 30.2458 [Function Documentation](#) 4053

 30.2458.2 [s1500\(base, norm, axisA, double alpha, double ratio, int idim, int inumb, carray, int *jstat\)](#) 4053

30.2459 [src/s1501.c File Reference](#) 4053

 30.2459 [Macro Definition Documentation](#) 4054

 30.2459.1 [S1501](#) 4054

 30.2459 [Function Documentation](#) 4054

30.2459.2	s1501(SISLSurf *ps1, double *base, double *norm, double *axisA, double alpha, double ratio, int idim, double aepsco, double aepsge, double amax, SISLIntcurve *pintcr, int icur, int igrph, int *jstat)	4054
30.2460	src/s1502.c File Reference	4054
30.2460	Macro Definition Documentation	4055
30.2460.1	s1502	4055
30.2460	Function Documentation	4055
30.2460.2	s1502(SISLCurve *pc1, base, norm, axisA, alpha, ratio, int idim, double aepsco, double aepsge, int *jpt, double **gpar, int *jcrv, SISLIntcurve ***wcurve, int *jstat)	4055
30.2461	src/s1503.c File Reference	4055
30.2461	Macro Definition Documentation	4056
30.2461.1	s1503	4056
30.2461	Function Documentation	4056
30.2461.2	s1503(SISLSurf *ps1, base, norm, axisA, double alpha, double ratio, int idim, double aepsco, double aepsge, int *jpt, double **gpar, int *jcrv, SISLIntcurve ***wcurve, int *jstat)	4056
30.2462	src/s1504.c File Reference	4056
30.2462	Macro Definition Documentation	4057
30.2462.1	s1504	4057
30.2462	Function Documentation	4057
30.2462.2	s1504(double *et, int ik, int in, double *ax, int im, int ider, ebder, ileft, int *jstat)	4057
30.2463	src/s1505.c File Reference	4057
30.2463	Macro Definition Documentation	4058
30.2463.1	s1505	4058
30.2463	Function Documentation	4058
30.2463.2	s1505(SISLSurf *ps1, int ider, int m1, int m2, double *ebder1, double *ebder2, int *ileft1, int *ileft2, eder, norm, int *jstat)	4058
30.2464	src/s1506.c File Reference	4058
30.2464	Macro Definition Documentation	4059
30.2464.1	s1506	4059
30.2464	Function Documentation	4059
30.2464.2	s1506(SISLSurf *ps1, int ider, int m1, double *x, int m2, double *y, eder, norm, int *jstat)	4059

30.2465I/src/s1507.c File Reference 4059

 30.2465M Macro Definition Documentation 4060

 30.2465.1S1507 4060

 30.2465F Junction Documentation 4060

 30.2465.2s1507(SISLCurve **curves, int nc, int periodic, SISLCurve ***newcurves, int *jstat) 4060

30.2466I/src/s1508.c File Reference 4060

 30.2466M Macro Definition Documentation 4061

 30.2466.1S1508 4061

 30.2466F Junction Documentation 4061

 30.2466.2s1508(int inbcrv, SISLCurve **vpcurv, par_arr, SISLSurf **rsurf, int *jstat) . . . 4061

30.2467I/src/s1510.c File Reference 4061

 30.2467M Macro Definition Documentation 4062

 30.2467.1S1510 4062

 30.2467F Junction Documentation 4062

 30.2467.2s1510(SISLSurf *ps, eyepoint, int idim, double aepsco, double aepsge, int *jpt, double **gpar, int *jcrv, SISLIntcurve ***wcurve, int *jstat) 4062

30.2468I/src/s1511.c File Reference 4062

 30.2468M Macro Definition Documentation 4063

 30.2468.1S1511 4063

 30.2468F Junction Documentation 4063

 30.2468.2s1511(SISLSurf *ps, qpoint, bvec, int idim, double aepsco, double aepsge, int *jpt, double **gpar, int *jcrv, SISLIntcurve ***wcurve, int *jstat) 4063

30.2469I/src/s1512.c File Reference 4063

 30.2469M Macro Definition Documentation 4064

 30.2469.1S1512 4064

 30.2469F Junction Documentation 4064

 30.2469.2s1512(SISLSurf *psurf, eyepoint, int idim, SISLSurf **rsurf, int *jstat) 4064

30.2470I/src/s1513.c File Reference 4064

 30.2470M Macro Definition Documentation 4065

 30.2470.1S1513 4065

 30.2470F Junction Documentation 4065

30.2470.2	<code>s1513(SISLSurf *psurf, qpoint, bvec, int idim, SISLSurf **rsurf, int *jstat)</code>	4065
30.2470	<code>src/s1514.c</code> File Reference	4065
30.2471	Macro Definition Documentation	4066
30.2471.1	<code>s1514</code>	4066
30.2471	Function Documentation	4066
30.2471.2	<code>s1514(SISLSurf *ps1, eyepoint, int idim, double aepsco, double aepsge, double amax, SISLIntcurve *pintcr, int icur, int igrph, int *jstat)</code>	4066
30.2471	<code>src/s1515.c</code> File Reference	4066
30.2472	Macro Definition Documentation	4067
30.2472.1	<code>s1515</code>	4067
30.2472	Function Documentation	4067
30.2472.2	<code>s1515(SISLSurf *ps1, qpoint, bvec, int idim, double aepsco, double aepsge, double amax, SISLIntcurve *pintcr, int icur, int igrph, int *jstat)</code>	4067
30.2473	<code>src/s1516.c</code> File Reference	4067
30.2473	Macro Definition Documentation	4068
30.2473.1	<code>s1516</code>	4068
30.2473	Function Documentation	4068
30.2473.2	<code>s1516(ep, epar, int im, int idim, double **ev, int *jstat)</code>	4068
30.2474	<code>src/s1517.c</code> File Reference	4068
30.2474	Macro Definition Documentation	4069
30.2474.1	<code>s1517</code>	4069
30.2474	Function Documentation	4069
30.2474.2	<code>s1517(ep, ev, epar, int im, double mu, double **evnew, int *jstat)</code>	4069
30.2475	<code>src/s1518.c</code> File Reference	4069
30.2475	Macro Definition Documentation	4070
30.2475.1	<code>s1518</code>	4070
30.2475	Function Documentation	4070
30.2475.2	<code>s1518(SISLSurf *surf, point, dir, double epsge, start, end, parin, parout, int *stat)</code>	4070
30.2476	<code>src/s1520.c</code> File Reference	4070
30.2476	Macro Definition Documentation	4071
30.2476.1	<code>s1520</code>	4071

30.2476 ~~2~~unction Documentation 4071

 30.2476.2 ~~s~~1520(SISLCurve *pc, double angle, ep, eaxis, SISLSurf **rs, int *jstat) 4071

30.2476 ~~3~~/src/s1521.c File Reference 4071

30.2477 ~~M~~acro Definition Documentation 4072

 30.2477.1 ~~S~~1521 4072

30.2477 ~~2~~unction Documentation 4072

 30.2477.2 ~~s~~1521(SISLCurve *pc, int *jstat) 4072

30.2477 ~~3~~/src/s1522.c File Reference 4072

30.2478 ~~M~~acro Definition Documentation 4073

 30.2478.1 ~~S~~1522 4073

30.2478 ~~2~~unction Documentation 4073

 30.2478.2 ~~s~~1522(normal, centre, ellipaxis, double ratio, int dim, SISLCurve **ellipse, int *jstat) 4073

30.2478 ~~3~~/src/s1528.c File Reference 4073

30.2479 ~~M~~acro Definition Documentation 4074

 30.2479.1 ~~S~~1528 4074

30.2479 ~~2~~unction Documentation 4074

 30.2479.2 ~~s~~1528(int idim, int m1, int m2, points, int ipar, int iopen1, int iopen2, double **par1, double **par2, int *jstat) 4074

30.2479 ~~3~~/src/s1529.c File Reference 4074

30.2480 ~~M~~acro Definition Documentation 4075

 30.2480.1 ~~S~~1529 4075

30.2480 ~~2~~unction Documentation 4075

 30.2480.2 ~~s~~1529(ep, eder10, eder01, eder11, int im1, int im2, int idim, int ipar, SISLSurf **rsurf, int *jstat) 4075

30.2480 ~~3~~/src/s1530.c File Reference 4075

30.2481 ~~M~~acro Definition Documentation 4076

 30.2481.1 ~~S~~1530 4076

30.2481 ~~2~~unction Documentation 4076

 30.2481.2 ~~s~~1530(ep, eder10, eder01, eder11, epar1, epar2, int im1, int im2, int idim, SISL←
Surf **rsurf, int *jstat) 4076

30.2481 ~~3~~/src/s1531.c File Reference 4076

30.2482 ~~M~~acro Definition Documentation 4077

30.2482.1	§1531	4077
30.2482	Function Documentation	4077
30.2482.2	§1531(ea, int idim, int in1, int in2, double **eb, int *jstat)	4077
30.2483	src/s1534.c File Reference	4077
30.2483	Macro Definition Documentation	4078
30.2483.1	§1534	4078
30.2483	Function Documentation	4078
30.2483.2	§1534(points, der10, der01, der11, int im1, int im2, int idim, int ipar, int con1, int con2, int con3, int con4, int order1, int order2, int iopen1, int iopen2, SISLSurf **rsurf, int *jstat)	4078
30.2484	src/s1535.c File Reference	4078
30.2484	Macro Definition Documentation	4079
30.2484.1	§1535	4079
30.2484	Function Documentation	4079
30.2484.2	§1535(points, der10, der01, der11, int im1, int im2, int idim, par1, par2, int con1, int con2, int con3, int con4, int order1, int order2, int iopen1, int iopen2, SISLSurf **rsurf, int *jstat)	4079
30.2485	src/s1536.c File Reference	4079
30.2485	Macro Definition Documentation	4080
30.2485.1	§1536	4080
30.2485	Function Documentation	4080
30.2485.2	§1536(points, int im1, int im2, int idim, int ipar, int con1, int con2, int con3, int con4, int order1, int order2, int iopen1, int iopen2, SISLSurf **rsurf, int *jstat)	4080
30.2486	src/s1537.c File Reference	4080
30.2486	Macro Definition Documentation	4081
30.2486.1	§1537	4081
30.2486	Function Documentation	4081
30.2486.2	§1537(points, int im1, int im2, int idim, par1, par2, int con1, int con2, int con3, int con4, int order1, int order2, int iopen1, int iopen2, SISLSurf **rsurf, int *jstat)	4081
30.2487	src/s1538.c File Reference	4081
30.2487	Macro Definition Documentation	4082
30.2487.1	§1538	4082
30.2487	Function Documentation	4082

30.2487.2.s1538(int inbcrv, vpcurv, nctyp, double astpar, int iopen, int iord2, int iflag, SISL←
Surf **rsurf, double **gpar, int *jstat) 4082

30.2488B/src/s1539.c File Reference 4082

30.2488M Macro Definition Documentation 4083

30.2488.1.S1539 4083

30.2488F Function Documentation 4083

30.2488.2.s1539(int inbcrv, vpcurv, nctyp, epar, double astpar, int iopen, int iord2, int iflag,
SISLSurf **rsurf, double **gpar, int *jstat) 4083

30.2489B/src/s1540.c File Reference 4083

30.2489M Macro Definition Documentation 4084

30.2489.1.S1540 4084

30.2489F Function Documentation 4084

30.2489.2.s1540(et, int ik, int in, ax, int im, int ider, ebder, ileft, int *jstat) 4084

30.2490B/src/s1541.c File Reference 4084

30.2490M Macro Definition Documentation 4085

30.2490.1.S1541 4085

30.2490F Function Documentation 4085

30.2490.2.s1541(SISLCurve *pc1, int npol, ebder, ileft, eder, int *jstat) 4085

30.2491B/src/s1542.c File Reference 4085

30.2491M Macro Definition Documentation 4086

30.2491.1.S1542 4086

30.2491F Function Documentation 4086

30.2491.2.s1542(SISLCurve *pc1, int m, x, eder, int *jstat) 4086

30.2492B/src/s1600.c File Reference 4086

30.2492M Macro Definition Documentation 4087

30.2492.1.S1600 4087

30.2492F Function Documentation 4087

30.2492.2.s1600(SISLCurve *pc, epoint, enorm, int idim, SISLCurve **rc, int *jstat) 4087

30.2493B/src/s1601.c File Reference 4087

30.2493M Macro Definition Documentation 4088

30.2493.1.S1601 4088

30.2493	Function Documentation	4088
30.2493.2	s1601(SISLSurf *psurf, epoint, enorm, int idim, SISLSurf **rsurf, int *jstat)	4088
30.2494	src/s1602.c File Reference	4088
30.2494	Macro Definition Documentation	4089
30.2494.1	S1602	4089
30.2494	Function Documentation	4089
30.2494.2	s1602(estapt, endpt, int ik, int idim, double astpar, double *cendpar, SISLCurve **rc, int *jstat)	4089
30.2495	src/s1603.c File Reference	4089
30.2495	Macro Definition Documentation	4090
30.2495.1	S1603	4090
30.2495	Function Documentation	4090
30.2495.2	s1603(SISLSurf *psurf, double *cmin1, double *cmin2, double *cmax1, double *cmax2, int *jstat)	4090
30.2496	src/s1604.c File Reference	4090
30.2496	Macro Definition Documentation	4091
30.2496.1	S1604	4091
30.2496	Function Documentation	4091
30.2496.2	s1604(epoint, int inbpnt, double astpar, int iopen, int idim, int ik, SISLCurve **rc, int *jstat)	4091
30.2497	src/s1605.c File Reference	4091
30.2497	Macro Definition Documentation	4092
30.2497.1	S1605	4092
30.2497	Function Documentation	4092
30.2497.2	s1605(SISLCurve *pc, double aepsge, double **gpoint, int *jnbpnt, int *jstat)	4092
30.2498	src/s1606.c File Reference	4092
30.2498	Macro Definition Documentation	4093
30.2498.1	S1606	4093
30.2498	Function Documentation	4093
30.2498.2	s1606(SISLCurve *pc1, SISLCurve *pc2, double aepsge, epoint1, epoint2, int itype, int idim, int ik, SISLCurve **rc, int *jstat)	4093
30.2499	src/s1607.c File Reference	4093

30.2499 Macro Definition Documentation 4094

 30.2499.1 \S 1607 4094

30.2499 Function Documentation 4094

 30.2499.2 \S 1607(SISLCurve *pc1, SISLCurve *pc2, double aepsge, double aend1, double
 afil1, double aend2, double afil2, int itype, int idim, int ik, SISLCurve **rc, int *jstat) 4094

30.2500 ~~src/s1608.c~~ File Reference 4094

30.2500 Macro Definition Documentation 4095

 30.2500.1 \S 1608 4095

30.2500 Function Documentation 4095

 30.2500.2 \S 1608(SISLCurve *pc1, SISLCurve *pc2, double aepsge, ep11, epf1, ep21, epf2,
 int itype, int idim, int ik, SISLCurve **rc, double *ct11, double *ctf1, double *ct21,
 double *ctf2, int *jstat) 4095

30.2501 ~~src/s1609.c~~ File Reference 4095

30.2501 Macro Definition Documentation 4096

 30.2501.1 \S 1609 4096

30.2501 Function Documentation 4096

 30.2501.2 \S 1609(SISLCurve *pc1, SISLCurve *pc2, double aepsge, eps1, epf, eps2, double
 aradius, enorm, int itype, int idim, int ik, SISLCurve **rc, double *ct11, double
 *ctf1, double *ct21, double *ctf2, int *jstat) 4096

30.2502 ~~src/s1611.c~~ File Reference 4096

30.2502 Macro Definition Documentation 4097

 30.2502.1 \S 1611 4097

30.2502 Function Documentation 4097

 30.2502.2 \S 1611(epoint, int inbpnt, int idim, epty, int iopen, int ik, double astpar, double
 aepsge, double *cendpar, SISLCurve **rc, int *jstat) 4097

30.2503 ~~src/s1612.c~~ File Reference 4097

30.2503 Macro Definition Documentation 4098

 30.2503.1 \S 1612 4098

30.2503 Function Documentation 4098

 30.2503.2 \S 1612(SISLCurve *pc, double aepsge, double **gpoint, int *jnbpnt, int *jleng, int
 *jstat) 4098

30.2504 ~~src/s1613.c~~ File Reference 4098

30.2504 Macro Definition Documentation 4099

30.2504.1	§1613	4099
30.2504	Function Documentation	4099
30.2504.2	§1613(SISLCurve *pc, double aepsge, double **gpoint, int *jnbpnt, int *jstat)	4099
30.2505	src/s1613bez.c File Reference	4099
30.2505	Macro Definition Documentation	4100
30.2505.1	§1613BEZ	4100
30.2505	Function Documentation	4100
30.2505.2	§1613bez(SISLCurve *pc, int idiv, double aepsge, double **gpar, int *jnpar, int *jstat)	4100
30.2506	src/s1614.c File Reference	4100
30.2506	Macro Definition Documentation	4101
30.2506.1	§1614	4101
30.2506	Function Documentation	4101
30.2506.2	§1614(epoint, int inbpnt, int idim, eptyp, spoint, int *jnbpnt, sptyp, int *jstat)	4101
30.2507	src/s1615.c File Reference	4101
30.2507	Macro Definition Documentation	4102
30.2507.1	§1615	4102
30.2507	Function Documentation	4102
30.2507.2	§1615(epoint, int inbpnt, int idim, eptyp, int *jstat)	4102
30.2508	src/s1616.c File Reference	4102
30.2508	Macro Definition Documentation	4103
30.2508.1	§1616	4103
30.2508	Function Documentation	4103
30.2508.2	§1616(epoint, int inbpnt, int idim, eptyp, econic, int *jstat)	4103
30.2509	src/s1617.c File Reference	4103
30.2509	Macro Definition Documentation	4104
30.2509.1	§1617	4104
30.2509	Function Documentation	4104
30.2509.2	§1617(epoint, int inbpnt, int idim, eptyp, double aepsge, econic, estart, etang, estop, double *ashape, int *jstat)	4104
30.2510	src/s1618.c File Reference	4104

30.2510 Macro Definition Documentation 4105

 30.2510.1 \S 1618 4105

30.2510 Function Documentation 4105

 30.2510.2 \S 1618(ematrix, eright, esol, int in, double *adiff) 4105

30.2510 \S 1619/src/s1619.c File Reference 4105

30.2511 Macro Definition Documentation 4106

 30.2511.1 \S 1619 4106

30.2511 Function Documentation 4106

 30.2511.2 \S 1619(epoint, int inbpnt, int idim, epty, econic, int ityp, etang, double *ashape, int *jstat) 4106

30.2511 \S 1620/src/s1620.c File Reference 4106

30.2512 Macro Definition Documentation 4107

 30.2512.1 \S 1620 4107

30.2512 Function Documentation 4107

 30.2512.2 \S 1620(epoint, int inbpnt1, int inbpnt2, int ipar, int iopen1, int iopen2, int ik1, int ik2, int idim, SISLSurf **rs, int *jstat) 4107

30.2512 \S 1630/src/s1630.c File Reference 4107

30.2513 Macro Definition Documentation 4108

 30.2513.1 \S 1630 4108

30.2513 Function Documentation 4108

 30.2513.2 \S 1630(epoint, int inbpnt, double astpar, int iopen, int idim, int ik, SISLCurve **rc, int *jstat) 4108

30.2513 \S 1631/src/s1631.c File Reference 4108

30.2514 Macro Definition Documentation 4109

 30.2514.1 \S 1631 4109

30.2514 Function Documentation 4109

 30.2514.2 \S 1631(SISLCurve *pc, epoint, enorm, projdir, int idim, SISLCurve **rc, int *jstat) 4109

30.2514 \S 1700/src/s1700.c File Reference 4109

30.2515 Macro Definition Documentation 4110

 30.2515.1 \S 1700 4110

30.2515 Function Documentation 4110

30.2515.2	<code>s1700(int imy, int ik, int in, int iv, int *jpl, int *jfi, int *jla, double *et, double apar, double *galfa, int *jstat)</code>	4110
30.2515	<code>src/s1701.c</code> File Reference	4110
30.2516	Macro Definition Documentation	4111
30.2516.1	<code>s1701</code>	4111
30.2516	Function Documentation	4111
30.2516.2	<code>s1701(int ij, int imy, int ik, int in, int *jpl, int *jfi, int *jla, double *et, double *etau, double *ep, double *galfa, int *jstat)</code>	4111
30.2515	<code>src/s1705.c</code> File Reference	4111
30.2517	Macro Definition Documentation	4112
30.2517.1	<code>s1705</code>	4112
30.2517	Function Documentation	4112
30.2517.2	<code>s1705(SISLCurve *pc, int *jstat)</code>	4112
30.2515	<code>src/s1706.c</code> File Reference	4112
30.2518	Macro Definition Documentation	4113
30.2518.1	<code>s1706</code>	4113
30.2518	Function Documentation	4113
30.2518.2	<code>s1706(SISLCurve *pc)</code>	4113
30.2515	<code>src/s1707.c</code> File Reference	4113
30.2519	Macro Definition Documentation	4114
30.2519.1	<code>s1707</code>	4114
30.2519	Function Documentation	4114
30.2519.2	<code>s1707(SISLCurve *pc, int *jstat)</code>	4114
30.2515	<code>src/s1708.c</code> File Reference	4114
30.2520	Macro Definition Documentation	4115
30.2520.1	<code>s1708</code>	4115
30.2520	Function Documentation	4115
30.2520.2	<code>s1708(SISLSurf *ps, int *jstat)</code>	4115
30.2515	<code>src/s1710.c</code> File Reference	4115
30.2521	Macro Definition Documentation	4116
30.2521.1	<code>s1710</code>	4116

30.2521 [Function Documentation](#) 4116

 30.2521.2 [s1710\(SISLCurve *pc1, double apar, SISLCurve **rcnew1, SISLCurve **rcnew2, int *jstat\)](#) 4116

30.2522 [src/s1711.c File Reference](#) 4116

30.2522 [Macro Definition Documentation](#) 4117

 30.2522.1 [S1711](#) 4117

30.2522 [Function Documentation](#) 4117

 30.2522.2 [s1711\(SISLSurf *ps, int ipar, double apar, SISLSurf **rsnew1, SISLSurf **rsnew2, int *jstat\)](#) 4117

30.2523 [src/s1712.c File Reference](#) 4117

30.2523 [Macro Definition Documentation](#) 4118

 30.2523.1 [S1712](#) 4118

30.2523 [Function Documentation](#) 4118

 30.2523.2 [s1712\(SISLCurve *pc, double abeg, double aend, SISLCurve **rcnew, int *jstat\)](#) 4118

30.2524 [src/s1713.c File Reference](#) 4118

30.2524 [Macro Definition Documentation](#) 4119

 30.2524.1 [S1713](#) 4119

30.2524 [Function Documentation](#) 4119

 30.2524.2 [s1713\(SISLCurve *pc, double abeg, double aend, SISLCurve **rcnew, int *jstat\)](#) 4119

30.2525 [src/s1714.c File Reference](#) 4119

30.2525 [Macro Definition Documentation](#) 4120

 30.2525.1 [S1714](#) 4120

 30.2525.1 [SISL_CRV_CLOSED](#) 4120

 30.2525.1 [SISL_CRV_OPEN](#) 4120

 30.2525.1 [SISL_CRV_PERIODIC](#) 4120

30.2525 [Function Documentation](#) 4120

 30.2525.2 [s1714\(SISLCurve *pc, double apar1, double apar2, SISLCurve **rcnew1, SISLCurve **rcnew2, int *jstat\)](#) 4120

30.2526 [src/s1715.c File Reference](#) 4120

30.2526 [Macro Definition Documentation](#) 4121

 30.2526.1 [S1715](#) 4121

30.2526 [Function Documentation](#) 4121

30.2526.2	<code>s1715(SISLCurve *pc1, SISLCurve *pc2, int iend1, int iend2, SISLCurve **rcnew, int *jstat)</code>	4121
30.2527	<code>src/s1716.c</code> File Reference	4121
30.2527	Macro Definition Documentation	4122
30.2527.1	<code>s1716</code>	4122
30.2527	Function Documentation	4122
30.2527.2	<code>s1716(SISLCurve *pc1, SISLCurve *pc2, double aeps, SISLCurve **rcnew, int *jstat)</code>	4122
30.2528	<code>src/s1720.c</code> File Reference	4122
30.2528	Macro Definition Documentation	4123
30.2528.1	<code>s1720</code>	4123
30.2528	Function Documentation	4123
30.2528.2	<code>s1720(SISLCurve *pc, int nder, SISLCurve **rcnew, int *jstat)</code>	4123
30.2529	<code>src/s1730.c</code> File Reference	4123
30.2529	Macro Definition Documentation	4124
30.2529.1	<code>s1730</code>	4124
30.2529	Function Documentation	4124
30.2529.2	<code>s1730(SISLCurve *pc, SISLCurve **rcnew, int *jstat)</code>	4124
30.2530	<code>src/s1731.c</code> File Reference	4124
30.2530	Macro Definition Documentation	4125
30.2530.1	<code>s1731</code>	4125
30.2530	Function Documentation	4125
30.2530.2	<code>s1731(SISLSurf *ps, SISLSurf **rsnew, int *jstat)</code>	4125
30.2531	<code>src/s1732.c</code> File Reference	4125
30.2531	Macro Definition Documentation	4126
30.2531.1	<code>s1732</code>	4126
30.2531	Function Documentation	4126
30.2531.2	<code>s1732(SISLCurve *pc, int icon, double *cstart, double *cend, double *gcoef, int *jstat)</code>	4126
30.2532	<code>src/s1733.c</code> File Reference	4126
30.2532	Macro Definition Documentation	4127
30.2532.1	<code>s1733</code>	4127

30.2532 [Function Documentation](#) 4127

 30.2532.2 [s1733\(SISLSurf *ps, int icont1, int icont2, double *cstart1, double *cend1, double *cstart2, double *cend2, double *gcoef, int *jstat\)](#) 4127

30.2533 [src/s1741.c File Reference](#) 4127

 30.2533 [Macro Definition Documentation](#) 4128

 30.2533.1 [s1741](#) 4128

 30.2533 [Function Documentation](#) 4128

 30.2533.2 [s1741\(SISLObject *po1, SISLObject *po2, double aepsge, int *jstat\)](#) 4128

30.2534 [src/s1750.c File Reference](#) 4128

 30.2534 [Macro Definition Documentation](#) 4129

 30.2534.1 [s1750](#) 4129

 30.2534 [Function Documentation](#) 4129

 30.2534.2 [s1750\(SISLCurve *pc, int ikh, SISLCurve **rc, int *jstat\)](#) 4129

30.2535 [src/s1753.c File Reference](#) 4129

 30.2535 [Macro Definition Documentation](#) 4130

 30.2535.1 [s1753](#) 4130

 30.2535 [Function Documentation](#) 4130

 30.2535.2 [s1753\(et, ecf, int in, int ik, int idim, etr, ecf, int inr, ecc, ecw, int *jstat\)](#) 4130

30.2536 [src/s1754.c File Reference](#) 4130

 30.2536 [Macro Definition Documentation](#) 4131

 30.2536.1 [s1754](#) 4131

 30.2536 [Function Documentation](#) 4131

 30.2536.2 [s1754\(double *et, int in, int ik, int ikh, double **iknt, int *inh, int *jstat\)](#) 4131

30.2537 [src/s1755.c File Reference](#) 4131

 30.2537 [Macro Definition Documentation](#) 4132

 30.2537.1 [s1755](#) 4132

 30.2537 [Function Documentation](#) 4132

 30.2537.2 [s1755\(orknt, int in, int ik, extknt, int *inh, int *jstat\)](#) 4132

30.2538 [src/s1770.c File Reference](#) 4132

 30.2538 [Macro Definition Documentation](#) 4133

 30.2538.1 [s1770](#) 4133

30.2538	Function Documentation	4133
30.2538.2	s1770(SISLCurve *pcurve1, SISLCurve *pcurve2, double aepsge, double astart1, double astart2, double aend1, double aend2, double anext1, double anext2, double *cpos1, double *cpos2, int *jstat)	4133
30.2538	src/s17702d.c File Reference	4133
30.2539	Macro Definition Documentation	4134
30.2539.1	dopy2	4134
30.2539.1	dopy3	4134
30.2539.1	decr2	4134
30.2539.1	incr2	4134
30.2539.1	s1770_2D	4134
30.2539.1	set_order	4134
30.2539.1	SINGULAR	4134
30.2539	Function Documentation	4134
30.2539.2	s1770_2D(SISLCurve *pcurve1, SISLCurve *pcurve2, double aepsge, double astart1, double astart2, double aend1, double aend2, double anext1, double anext2, double *cpos1, double *cpos2, int *jstat)	4134
30.2540	src/s1771.c File Reference	4135
30.2540	Macro Definition Documentation	4135
30.2540.1	s1771	4135
30.2540	Function Documentation	4135
30.2540.2	s1771(SISLPoint *ppoint, SISLCurve *pcurve, double aepsge, double astart, double aend, double anext, double *cpos, int *jstat)	4135
30.2541	src/s1772.c File Reference	4136
30.2541	Macro Definition Documentation	4136
30.2541.1	dopy2	4136
30.2541.1	dopy3	4136
30.2541.1	decr2	4137
30.2541.1	incr2	4137
30.2541.1	s1772	4137
30.2541.1	set_order	4137
30.2541.1	SINGULAR	4137
30.2541	Function Documentation	4137

30.2541.2	src/s1772(SISLCurve *pcurve, SISLSurf *psurf, double aepsge, double astart1, estart2, double aend1, eend2, double anex1, enext2, double *cpos1, gpos2, int *jstat)	4137
30.2542	src/s1773.c File Reference	4137
30.2542	Macro Definition Documentation	4138
30.2542.1	S1773	4138
30.2542	Function Documentation	4138
30.2542.2	src/s1773(SISLPoint *ppoint, SISLSurf *psurf, double aepsge, estart, eend, enext, gpos, int *jstat)	4138
30.2543	src/s1774.c File Reference	4138
30.2543	Macro Definition Documentation	4139
30.2543.1	S1774	4139
30.2543	Function Documentation	4139
30.2543.2	src/s1774(SISLCurve *crv, point, int dim, double epsge, double start, double end, double guess, double *clpar, int *stat)	4139
30.2544	src/s1775.c File Reference	4139
30.2544	Macro Definition Documentation	4140
30.2544.1	S1775	4140
30.2544	Function Documentation	4140
30.2544.2	src/s1775(SISLSurf *surf, point, int dim, double epsge, start, end, guess, clpar, int *stat)	4140
30.2545	src/s1780.c File Reference	4140
30.2545	Macro Definition Documentation	4141
30.2545.1	S1780	4141
30.2545	Function Documentation	4141
30.2545.2	src/s1780(SISLCurve *pc1, SISLCurve *pc2, vipt, int *jstat)	4141
30.2546	src/s1785.c File Reference	4141
30.2546	Macro Definition Documentation	4142
30.2546.1	S1785	4142
30.2546	Function Documentation	4142
30.2546.2	src/s1785(SISLCurve *pcurve, SISLSurf *psurf, double aepsge, epar1, epar2, int icur, int *jstat)	4142
30.2547	src/s1786.c File Reference	4142

30.2547	Macro Definition Documentation	4143
30.2547.1	§1786	4143
30.2547	Predef Documentation	4143
30.2547.2	evalcProc	4143
30.2547	Function Documentation	4143
30.2547.3	§1786(SISLCurve *pc1, SISLCurve *pc2, double aepsge, epar1, epar2, int *jstat)	4143
30.2548	src/s1787.c File Reference	4143
30.2548	Macro Definition Documentation	4144
30.2548.1	§1787	4144
30.2548	Function Documentation	4144
30.2548.2	§1787(SISLSurf *ps, double alevel, double aepsge, epar, gpar1, gpar2, int *jstat)	4144
30.2549	src/s1788.c File Reference	4144
30.2549	Macro Definition Documentation	4145
30.2549.1	§1788	4145
30.2549	Function Documentation	4145
30.2549.2	§1788(SISLSurf *ps1, SISLSurf *ps2, double aepsge, epar, gpar1, gpar2, int *jstat)	4145
30.2550	src/s1789.c File Reference	4145
30.2550	Macro Definition Documentation	4146
30.2550.1	§1789	4146
30.2550	Function Documentation	4146
30.2550.2	§1789(SISLPoint *ppoint, SISLSurf *psurf, double aepsge, epar1, epar2, int *jstat)	4146
30.2551	src/s1790.c File Reference	4146
30.2551	Macro Definition Documentation	4147
30.2551.1	§1790	4147
30.2551	Function Documentation	4147
30.2551.2	§1790(SISLObject *po1, SISLObject *po2, double aepsge, int *jstat)	4147
30.2552	src/s1791.c File Reference	4147
30.2552	Macro Definition Documentation	4148
30.2552.1	§1791	4148
30.2552	Function Documentation	4148

30.2552.2	s1791(et, int ik, int in)	4148
30.2553	/src/s1792.c File Reference	4148
30.2553	Macro Definition Documentation	4149
30.2553.1	s1792	4149
30.2553	Function Documentation	4149
30.2553.2	s1792(et, int ik, int in)	4149
30.2554	/src/s1795.c File Reference	4149
30.2554	Macro Definition Documentation	4150
30.2554.1	s1795	4150
30.2554	Function Documentation	4150
30.2554.2	s1795(SISLSurf *ps1, SISLSurf *ps2, double aepsge, double aang, int *jstat)	4150
30.2555	/src/s1796.c File Reference	4150
30.2555	Macro Definition Documentation	4151
30.2555.1	s1796	4151
30.2555	Function Documentation	4151
30.2555.2	s1796(SISLCurve *pc1, SISLCurve *pc2, double aepsge, double aang, int *jstat)	4151
30.2556	/src/s1797.c File Reference	4151
30.2556	Macro Definition Documentation	4152
30.2556.1	s1797	4152
30.2556	Function Documentation	4152
30.2556.2	s1797(SISLSurf *ps1, SISLCurve *pc1, double aepsge, double aang, int *jstat)	4152
30.2557	/src/s1830.c File Reference	4152
30.2557	Macro Definition Documentation	4153
30.2557.1	s1830	4153
30.2557	Function Documentation	4153
30.2557.2	s1830(SISLSurf *psurf, SISLCurve *pcurve, int *jstat)	4153
30.2558	/src/s1834.c File Reference	4153
30.2558	Macro Definition Documentation	4154
30.2558.1	s1834	4154
30.2558	Function Documentation	4154

30.2558.2.s1834(ecoef1, int in1, ecoef2, int in2, int idim, edir1, edir2, int *jstat)	4154
30.2558 src/s1839.c File Reference	4154
30.2559Macro Definition Documentation	4155
30.2559.1.s1839	4155
30.2559F2unction Documentation	4155
30.2559.2.s1839(SISLSurf *ps1, epol, int in, int idim, int *jstat)	4155
30.2560 src/s1840.c File Reference	4155
30.2560Macro Definition Documentation	4156
30.2560.1.s1840	4156
30.2560F2unction Documentation	4156
30.2560.2.s1840(SISLCurve *pcurve, double *cdist, int *jstat)	4156
30.2561 src/s1850.c File Reference	4156
30.2561Macro Definition Documentation	4157
30.2561.1.s1850	4157
30.2561F2unction Documentation	4157
30.2561.2.s1850(SISLCurve *pc1, epoint, enorm, int idim, double aepsco, double aepsge, int *jpt, double **gpar, int *jcrv, SISLIntcurve ***wcurve, int *jstat)	4157
30.2562 src/s1851.c File Reference	4157
30.2562Macro Definition Documentation	4158
30.2562.1.s1851	4158
30.2562F2unction Documentation	4158
30.2562.2.s1851(SISLSurf *ps1, epoint, enorm, int idim, double aepsco, double aepsge, int *jpt, double **gpar, int *jcrv, SISLIntcurve ***wcurve, int *jstat)	4158
30.2563 src/s1852.c File Reference	4158
30.2563Macro Definition Documentation	4159
30.2563.1.s1852	4159
30.2563F2unction Documentation	4159
30.2563.2.s1852(SISLSurf *ps1, ecenter, double aradius, int idim, double aepsco, double aepsge, int *jpt, double **gpar, int *jcrv, SISLIntcurve ***wcurve, int *jstat)	4159
30.2564 src/s1853.c File Reference	4159
30.2564Macro Definition Documentation	4160
30.2564.1.s1853	4160

30.2564 [Function Documentation](#) 4160

 30.2564.2 [s1853\(SISLSurf *ps1, epoint, edirec, double aradius, int idim, double aepsco, double aepsge, int *jpt, double **gpar, int *jcrv, SISLIntcurve ***wcurve, int *jstat\)](#) 4160

30.2565 [src/s1854.c File Reference](#) 4160

30.2565 [Macro Definition Documentation](#) 4161

 30.2565.1 [S1854](#) 4161

30.2565 [Function Documentation](#) 4161

 30.2565.2 [s1854\(SISLSurf *ps1, etop, eaxis, econc, int idim, double aepsco, double aepsge, int *jpt, double **gpar, int *jcrv, SISLIntcurve ***wcurve, int *jstat\)](#) 4161

30.2566 [src/s1855.c File Reference](#) 4161

30.2566 [Macro Definition Documentation](#) 4162

 30.2566.1 [S1855](#) 4162

30.2566 [Function Documentation](#) 4162

 30.2566.2 [s1855\(SISLSurf *ps1, ecentr, double aradius, enorm, int idim, double aepsco, double aepsge, int *jpt, double **gpar, int *jcrv, SISLIntcurve ***wcurve, int *jstat\)](#) 4162

30.2567 [src/s1856.c File Reference](#) 4162

30.2567 [Macro Definition Documentation](#) 4163

 30.2567.1 [S1856](#) 4163

30.2567 [Function Documentation](#) 4163

 30.2567.2 [s1856\(SISLSurf *ps1, epoint, edir, int idim, double aepsco, double aepsge, int *jpt, double **gpar, int *jcrv, SISLIntcurve ***wcurve, int *jstat\)](#) 4163

30.2568 [src/s1857.c File Reference](#) 4163

30.2568 [Macro Definition Documentation](#) 4164

 30.2568.1 [S1857](#) 4164

30.2568 [Function Documentation](#) 4164

 30.2568.2 [s1857\(SISLCurve *pc1, SISLCurve *pc2, double aepsco, double aepsge, int *jpt, double **gpar1, double **gpar2, int *jcrv, SISLIntcurve ***wcurve, int *jstat\)](#) 4164

30.2569 [src/s1858.c File Reference](#) 4164

30.2569 [Macro Definition Documentation](#) 4165

 30.2569.1 [S1858](#) 4165

30.2569 [Function Documentation](#) 4165

 30.2569.2 [s1858\(SISLSurf *ps1, SISLCurve *pc1, double aepsco, double aepsge, int *jpt, double **gpar1, double **gpar2, int *jcrv, SISLIntcurve ***wcurve, int *jstat\)](#) 4165

30.2570	src/s1859.c File Reference	4165
30.2570	Macro Definition Documentation	4166
30.2570.1	\$1859	4166
30.2570	Function Documentation	4166
30.2570.2	\$1859(SISLSurf *ps1, SISLSurf *ps2, double aepsco, double aepsge, int *jpt, double **gpar1, double **gpar2, int *jcrv, SISLIntcurve ***wcurve, int *jstat)	4166
30.2571	src/s1860.c File Reference	4166
30.2571	Macro Definition Documentation	4167
30.2571.1	\$1860	4167
30.2571	Function Documentation	4167
30.2571.2	\$1860(SISLSurf *ps, eviiew, int idim, double aepsco, double aepsge, int *jpt, double **gpar, int *jcrv, SISLIntcurve ***wcurve, int *jstat)	4167
30.2572	src/s1870.c File Reference	4167
30.2572	Macro Definition Documentation	4168
30.2572.1	\$1870	4168
30.2572	Function Documentation	4168
30.2572.2	\$1870(SISLSurf *ps1, double *pt1, int idim, double aepsge, int *jpt, double **gpar1, int *jcrv, SISLIntcurve ***wcurve, int *jstat)	4168
30.2573	src/s1871.c File Reference	4168
30.2573	Macro Definition Documentation	4169
30.2573.1	\$1871	4169
30.2573	Function Documentation	4169
30.2573.2	\$1871(SISLCurve *pc1, double *pt1, int idim, double aepsge, int *jpt, double **gpar1, int *jcrv, SISLIntcurve ***wcurve, int *jstat)	4169
30.2574	src/s1880.c File Reference	4169
30.2574	Macro Definition Documentation	4170
30.2574.1	\$1880	4170
30.2574	Function Documentation	4170
30.2574.2	\$1880(int ipar1, int ipar2, int *jpt, SISLIntpt **vpoint, int *jlist, SISLIntlist **vlist, int *jpar, double **gpar1, double **gpar2, int *jcrv, SISLIntcurve ***wcrv, int *jstat)	4170
30.2575	src/s1890.c File Reference	4170
30.2575	Macro Definition Documentation	4171

30.2575.1 [§1890](#) 4171

30.2575 [Function Documentation](#) 4171

30.2575.2 [§1890\(oknots, int oik, int oin, par, der, int *jstat\)](#) 4171

30.2575 [src/s1891.c File Reference](#) 4171

30.2576 [Macro Definition Documentation](#) 4172

30.2576.1 [MAX_SIZE](#) 4172

30.2576.1 [§1891](#) 4172

30.2576 [Function Documentation](#) 4172

30.2576.2 [§1891\(etau, epoint, int idim, int inbpnt, int irlight, eder, int iopen, et, ebcoef, int *in, int ik, int inlr, int inrc, int *jstat\)](#) 4172

30.2576 [src/s1893.c File Reference](#) 4172

30.2577 [Macro Definition Documentation](#) 4173

30.2577.1 [§1893](#) 4173

30.2577 [Function Documentation](#) 4173

30.2577.2 [§1893\(SISLCurve *orig, earray, int dimp1, int narr, int der1, int der2, SISLCurve **ncurve, int *jstat\)](#) 4173

30.2577 [src/s1894.c File Reference](#) 4173

30.2578 [Macro Definition Documentation](#) 4174

30.2578.1 [§1894](#) 4174

30.2578 [Function Documentation](#) 4174

30.2578.2 [§1894\(oknots, int oik, int oin, int der1, int der2, earray, int dimp1, int narr, nknots, int *nik, int *nin, int *jstat\)](#) 4174

30.2578 [src/s1896.c File Reference](#) 4174

30.2579 [Macro Definition Documentation](#) 4175

30.2579.1 [§1896](#) 4175

30.2579 [Function Documentation](#) 4175

30.2579.2 [§1896\(SISLSurf *osurf, earray, int dimp1, int narr, ders1, dert1, ders2, dert2, S← ISLSurf **nsurf, int *jstat\)](#) 4175

30.2579 [src/s1897.c File Reference](#) 4175

30.2580 [Macro Definition Documentation](#) 4176

30.2580.1 [MAX_IK](#) 4176

30.2580.1 [§1897](#) 4176

30.2580	Function Documentation	4176
30.2580.2	s1897(et, int ik, double ax, int left, int deriv, ebiatx, int *jstat)	4176
30.2581	src/s1900.c File Reference	4176
30.2581	Macro Definition Documentation	4177
30.2581.1	S1900	4177
30.2581	Function Documentation	4177
30.2581.2	s1900(param, knots, econd, ntype, int inpt, int ik, int idim, int iopen, double *cendpar, SISLCurve **rcurve, double **gpar, int *jnbpar, int *jstat)	4177
30.2582	src/s1901.c File Reference	4177
30.2582	Macro Definition Documentation	4178
30.2582.1	S1901	4178
30.2582	Typedef Documentation	4178
30.2582.2	fknotsProc	4178
30.2582.2	fparamProc	4178
30.2582	Function Documentation	4178
30.2582.3	s1901(fparamProc fparam, fknotsProc fknots, econd, ntype, int inpt, double astpar, int ik, int idim, int iopen, double *cendpar, SISLCurve **rcurve, double **gpar, int *jnbpar, int *jstat)	4178
30.2583	src/s1902.c File Reference	4179
30.2583	Macro Definition Documentation	4179
30.2583.1	S1902	4179
30.2583	Function Documentation	4179
30.2583.2	s1902(epar, int in, int ik, int cuopen, double **eknots, int *jstat)	4179
30.2584	src/s1903.c File Reference	4180
30.2584	Macro Definition Documentation	4180
30.2584.1	S1903	4180
30.2584	Function Documentation	4180
30.2584.2	s1903(epar, int in, int ik, int cuopen, eknots, int *jstat)	4180
30.2585	src/s1904.c File Reference	4181
30.2585	Macro Definition Documentation	4181
30.2585.1	S1904	4181
30.2585	Function Documentation	4181

30.2585.2	<code>s1904(epar, int in, int ik, int cuopen, eknots, int *jstat)</code>	4181
30.2586	<code>/src/s1905.c</code> File Reference	4182
30.2586	Macro Definition Documentation	4182
30.2586.1	<code>s1905</code>	4182
30.2586	Function Documentation	4182
30.2586.2	<code>s1905(econd1, ntype1, int inpt1, int ik, int idim, int iopen, double **gcond2, int **mtype2, int *jnpt2, int *jstat)</code>	4182
30.2587	<code>/src/s1906.c</code> File Reference	4183
30.2587	Macro Definition Documentation	4183
30.2587.1	<code>s1906</code>	4183
30.2587	Function Documentation	4183
30.2587.2	<code>s1906(double *epoint, int *etype, int icnsta, int icnend, int inbpnt, int idim, double **opoint, int **otype, int *knbpnt, int *jstat)</code>	4183
30.2588	<code>/src/s1907.c</code> File Reference	4184
30.2588	Macro Definition Documentation	4184
30.2588.1	<code>s1907</code>	4184
30.2588	Function Documentation	4184
30.2588.2	<code>s1907(double *epoint, int *ntype, double *epar, int iopen, int icnsta, int icnend, int inbpnt, int idim, opoint, otype, opar, int *knbpnt, int *jstat)</code>	4184
30.2589	<code>/src/s1908.c</code> File Reference	4185
30.2589	Macro Definition Documentation	4185
30.2589.1	<code>MAX_SIZE</code>	4185
30.2589.1	<code>s1908</code>	4185
30.2589	Function Documentation	4185
30.2589.2	<code>s1908(econd1, ntype1, epar, int inpt1, int ik, int idim, int iopen, gcond2, mtype2, mpar, int *jnpt2, int *jstat)</code>	4185
30.2590	<code>/src/s1909.c</code> File Reference	4186
30.2590	Macro Definition Documentation	4186
30.2590.1	<code>s1909</code>	4186
30.2590	Function Documentation	4186
30.2590.2	<code>s1909(econd, ntype, int inpt, int idim, int iopen, double astpar, double *cendpar, epar1, epar2, int *jstat)</code>	4186
30.2591	<code>/src/s1910.c</code> File Reference	4187

30.2591	Macro Definition Documentation	4187
30.2591.1	§1910	4187
30.2591	Function Documentation	4187
30.2591.2	§1910(econd, ntype, int inpt, int idim, int iopen, double astpar, double *cendpar, epar1, epar2, int *jstat)	4187
30.2592	/src/s1911.c File Reference	4188
30.2592	Macro Definition Documentation	4188
30.2592.1	§1911	4188
30.2592	Function Documentation	4188
30.2592.2	§1911(econd, ntype, int inpt, int idim, int iopen, double astpar, double *cendpar, epar1, epar2, int *jstat)	4188
30.2593	/src/s1912.c File Reference	4189
30.2593	Macro Definition Documentation	4189
30.2593.1	§1912	4189
30.2593	Typedef Documentation	4189
30.2593.2	fknotsProc	4189
30.2593.2	fparamProc	4190
30.2593	Function Documentation	4190
30.2593.3	§1912(fparamProc fparam, fknotsProc fknots, econd, ntype, int inpt, double astpar, int ik, int idim, int iopen, double *cendpar, SISLCurve **rcurve, double **gpar, int *jnbpar, int *jstat)	4190
30.2594	/src/s1916.c File Reference	4190
30.2594	Macro Definition Documentation	4190
30.2594.1	§1916	4190
30.2594	Function Documentation	4191
30.2594.2	§1916(int inbcrv, et2, ecoef, int in2, int iord, int idim, int iopen, par, der, int *jstat)	4191
30.2595	/src/s1917.c File Reference	4191
30.2595	Macro Definition Documentation	4191
30.2595.1	§1917	4191
30.2595	Function Documentation	4192
30.2595.2	§1917(int inbcrv, ecoef, int in2, int idim, eptyp, double astpar, int iopen, par, der, int *inumb, int *jstat)	4192
30.2596	/src/s1918.c File Reference	4192

30.2596	Macro Definition Documentation	4192
30.2596.1	§1918	4192
30.2596	Function Documentation	4193
30.2596.2	§1918(int inbcrv, et2, ecoef, int in2, int iord, int idim, par, der, int *jstat)	4193
30.2597	/src/s1919.c File Reference	4193
30.2597	Macro Definition Documentation	4193
30.2597.1	§1919	4193
30.2597	Function Documentation	4194
30.2597.2	§1919(et, prev, curr, deriv, follow, int in, int ik, int idim, int iip, int iif, double ap, double ac, double af, int *jstat)	4194
30.2598	/src/s1920.c File Reference	4194
30.2598	Macro Definition Documentation	4194
30.2598.1	§1920	4194
30.2598	Function Documentation	4195
30.2598.2	§1920(SISLCurve *pc1, edir, int idim, double aepsco, double aepsge, int *jpt, double **gpar, int *jcrv, SISLIntcurve ***wcurve, int *jstat)	4195
30.2599	/src/s1921.c File Reference	4195
30.2599	Macro Definition Documentation	4195
30.2599.1	§1921	4195
30.2599	Function Documentation	4196
30.2599.2	§1921(SISLSurf *ps1, edir, int idim, double aepsco, double aepsge, int *jpt, double **gpar, int *jcrv, SISLIntcurve ***wcurve, int *jstat)	4196
30.2600	/src/s1924.c File Reference	4196
30.2600	Macro Definition Documentation	4196
30.2600.1	§1924	4196
30.2600	Function Documentation	4197
30.2600.2	§1924(int id1, int id2, int id3, int id4, int in1, int in2, double **ew, int *jstat)	4197
30.2601	/src/s1925.c File Reference	4197
30.2601	Macro Definition Documentation	4197
30.2601.1	§1925	4197
30.2601.1	§1925_MAX_ARRAY_SIZE	4198
30.2601	Function Documentation	4198

30.2601.2	s1925(etau, epoint, int inbpnt, eder, et, ebcoef, int in, int ik, int igrht, int dim, ew1, int nur, ed, ew2, int inrc, ew3, int inlr, int *jstat)	4198
30.2601	/src/s1926.c File Reference	4198
30.2602	Macro Definition Documentation	4198
30.2602.1	s1926	4198
30.2602	Function Documentation	4199
30.2602.2	s1926(double *w1, int nur, int ik, int *ed, double *w2, int nrc, double *w3, int nlr, int *jstat)	4199
30.2603	/src/s1927.c File Reference	4199
30.2603	Macro Definition Documentation	4199
30.2603.1	s1927	4199
30.2603	Function Documentation	4200
30.2603.2	s1927(double *w1, int nur, int ik, int *ed, double *w2, int nrc, double *w3, int nlr, ex, double *ey, int *jstat)	4200
30.2604	/src/s1930.c File Reference	4200
30.2604	Macro Definition Documentation	4200
30.2604.1	s1930	4200
30.2604	Function Documentation	4201
30.2604.2	s1930(int inbcrv, SISLCurve **vpcrv, double **gknot2, double **gcoef2, int *jn2, int *jord2, int *jstat)	4201
30.2605	/src/s1931.c File Reference	4201
30.2605	Macro Definition Documentation	4201
30.2605.1	s1931	4201
30.2605	Function Documentation	4202
30.2605.2	s1931(int inbcrv, SISLCurve **vpcrv, double **gknot2, double **gcoef2, int *jn2, int *jord2, int *jstat)	4202
30.2606	/src/s1931unit.c File Reference	4202
30.2606	Macro Definition Documentation	4202
30.2606.1	s1931UNIT	4202
30.2606	Function Documentation	4203
30.2606.2	s1931unit(int inbcrv, SISLCurve **vpcrv, double **gknot2, double **gcoef2, int *jn2, int *jord2, int *jstat)	4203
30.2607	/src/s1932.c File Reference	4203

30.2607	Macro Definition Documentation	4203
30.2607.1	§1932	4203
30.2607	Function Documentation	4204
30.2607.2	§1932(int inbcrv, SISLCurve **crvarr, double start, double stop, double *et, int in, int iordr, double **iright, int *jstat)	4204
30.2608	/src/s1933.c File Reference	4204
30.2608	Macro Definition Documentation	4204
30.2608.1	§1933	4204
30.2608	Function Documentation	4205
30.2608.2	§1933(int inbcrv, crvarr, double start, double stop, double **it, int *in, int *iordr, int *jstat)	4205
30.2609	/src/s1934.c File Reference	4205
30.2609	Macro Definition Documentation	4205
30.2609.1	§1934	4205
30.2609	Function Documentation	4206
30.2609.2	§1934(double *et, int in, int ik, double start, double end, int *jstat)	4206
30.2610	/src/s1935.c File Reference	4206
30.2610	Macro Definition Documentation	4206
30.2610.1	§1935	4206
30.2610	Function Documentation	4207
30.2610.2	§1935(double *et1, int in1, double *et2, int in2, knt, int *in, int ik, int *jstat)	4207
30.2611	/src/s1936.c File Reference	4207
30.2611	Macro Definition Documentation	4207
30.2611.1	MAX_SIZE	4207
30.2611.1	§1936	4208
30.2611	Function Documentation	4208
30.2611.2	§1936(SISLCurve *crv, etd, int ind, double *curvd, int *jstat)	4208
30.2612	/src/s1937.c File Reference	4208
30.2612	Macro Definition Documentation	4208
30.2612.1	§1937	4208
30.2612	Function Documentation	4209

30.2612.2	s1937(et, int iordr, int ref, int left, alfa, etref)	4209
30.2613	src/s1938.c File Reference	4209
30.2613	Macro Definition Documentation	4209
30.2613.1	MAX_SIZE	4209
30.2613.1	s1938	4210
30.2613	Function Documentation	4210
30.2613.2	s1938(SISLSurf *srf, etr1, int inr1, etr2, int inr2, double **surfr, int *jstat)	4210
30.2614	src/s1940.c File Reference	4210
30.2614	Macro Definition Documentation	4210
30.2614.1	s1940	4210
30.2614	Function Documentation	4211
30.2614.2	s1940(SISLCurve *oldcurve, eps, int startfix, int endfix, int iopen, int itmax, SISLCurve **newcurve, maxerr, int *stat)	4211
30.2615	src/s1941.c File Reference	4211
30.2615	Macro Definition Documentation	4211
30.2615.1	s1941	4211
30.2615	Function Documentation	4212
30.2615.2	s1941(SISLCurve *pcurve, int icont, int *jstat)	4212
30.2616	src/s1942.c File Reference	4212
30.2616	Macro Definition Documentation	4212
30.2616.1	s1942	4212
30.2616	Function Documentation	4213
30.2616.2	s1942(SISLCurve *pc1, SISLCurve *pc2, int idim, ea, nstart, nstop, emxerr, el2err, int *jstat)	4213
30.2617	src/s1943.c File Reference	4213
30.2617	Macro Definition Documentation	4213
30.2617.1	s1943	4213
30.2617	Function Documentation	4214
30.2617.2	s1943(SISLCurve *pcurve, etau, int ik, int in, int ileftfix, int irlightfix, int incon, SISLCurve **newcurve, gmaxerr, gl2err, int *jstat)	4214
30.2618	src/s1944.c File Reference	4214
30.2618	Macro Definition Documentation	4214

30.2618.1	§1944	4214
30.2618	Function Documentation	4215
30.2618.2	§1944(etau, int ik, int in, int idim, et, ed, int im, int inlc, int inlr, int inorm, ea, ew1, nfirst, nlast, eb, ew2, n2sta, ec, int *jstat)	4215
30.2618	src/s1945.c File Reference	4215
30.2619	Macro Definition Documentation	4215
30.2619.1	§1945	4215
30.2619	Function Documentation	4216
30.2619.2	§1945(etau, int ik, int in, int idim, et, ed, int im, int ilend, int irend, int inlc, int inlr, int inorm, ea, ew1, int inh, nfirst, nlast, eb, ew2, ec, n2sta, int *jstat)	4216
30.2619	src/s1946.c File Reference	4216
30.2620	Macro Definition Documentation	4216
30.2620.1	§1946	4216
30.2620	Function Documentation	4217
30.2620.2	§1946(ea, ew1, nfirst, nlast, ed, ec, int ik, int in, int im, int idim, int ilend, int irend, int inlr, int inlc, int *jstat)	4217
30.2620	src/s1947.c File Reference	4217
30.2621	Macro Definition Documentation	4217
30.2621.1	§1947	4217
30.2621	Function Documentation	4218
30.2621.2	§1947(ea, nfirst, nlast, int ik, int im, etau, int in, int incont, ew, int inlr, int *jnred, efac, int *jstat)	4218
30.2621	src/s1948.c File Reference	4218
30.2622	Macro Definition Documentation	4218
30.2622.1	§1948	4218
30.2622	Function Documentation	4219
30.2622.2	§1948(double *ea, double *ew, int in, int ik, int inlr, int *nstart, int *jstat)	4219
30.2622	src/s1949.c File Reference	4219
30.2623	Macro Definition Documentation	4219
30.2623.1	§1949	4219
30.2623	Function Documentation	4220
30.2623.2	§1949(double *ea, double *ew, double *eb, int in, int ik, int inlr, int idim, int *nstart, int *jstat)	4220

30.2624	src/s1950.c File Reference	4220
30.2624	Macro Definition Documentation	4220
30.2624.1	§1950	4220
30.2624	Function Documentation	4221
30.2624.2	§1950(SISLCurve *oldcurve, SISLCurve *rankcurve, rank_info *ranking, eps, epsco, int startfix, int endfix, int *jncont, int mini, int maxi, SISLCurve **newcurve, maxerr, int *stat)	4221
30.2625	src/s1951.c File Reference	4221
30.2625	Macro Definition Documentation	4221
30.2625.1	§1951	4221
30.2625	Function Documentation	4222
30.2625.2	§1951(etau, ecoef, int in, int ik, int idim, int ilend, int irend, int incont, efac)	4222
30.2626	src/s1953.c File Reference	4222
30.2626	Macro Definition Documentation	4222
30.2626.1	§1953	4222
30.2626	Function Documentation	4223
30.2626.2	§1953(SISLCurve *pcurve, epoint, int idim, double aepsco, double aepsge, int *jpt, double **gpar, int *jcrv, SISLIntcurve ***wcurve, int *jstat)	4223
30.2627	src/s1954.c File Reference	4223
30.2627	Macro Definition Documentation	4223
30.2627.1	§1954	4223
30.2627	Function Documentation	4224
30.2627.2	§1954(SISLSurf *psurf, epoint, int idim, double aepsco, double aepsge, int *jpt, double **gpar, int *jcrv, SISLIntcurve ***wcurve, int *jstat)	4224
30.2628	src/s1955.c File Reference	4224
30.2628	Macro Definition Documentation	4224
30.2628.1	§1955	4224
30.2628	Function Documentation	4225
30.2628.2	§1955(SISLCurve *pc1, SISLCurve *pc2, double aepsco, double aepsge, int *jpt, double **gpar1, double **gpar2, int *jcrv, SISLIntcurve ***wcurve, int *jstat)	4225
30.2629	src/s1956.c File Reference	4225
30.2629	Macro Definition Documentation	4225
30.2629.1	§1956	4225

30.2629	Function Documentation	4226
30.2629.2	§1956(SISLCurve *pc1, SISLCurve *pc2, SISLSurf **rsurf, int *jstat)	4226
30.2630	src/s1957.c File Reference	4226
30.2630	Macro Definition Documentation	4226
30.2630.1	§1957	4226
30.2630	Function Documentation	4227
30.2630.2	§1957(SISLCurve *pcurve, epoint, int idim, double aepsco, double aepsge, double *gpar, double *dist, int *jstat)	4227
30.2631	src/s1958.c File Reference	4227
30.2631	Macro Definition Documentation	4227
30.2631.1	§1958	4227
30.2631	Function Documentation	4228
30.2631.2	§1958(SISLSurf *psurf, epoint, int idim, double aepsco, double aepsge, gpar, double *dist, int *jstat)	4228
30.2632	src/s1959.c File Reference	4228
30.2632	Macro Definition Documentation	4228
30.2632.1	§1959	4228
30.2632	Function Documentation	4229
30.2632.2	§1959(SISLPoint *ppoint, SISLCurve *pcurve, double *gpos, int *jstat)	4229
30.2633	src/s1960.c File Reference	4229
30.2633	Macro Definition Documentation	4229
30.2633.1	§1960	4229
30.2633	Function Documentation	4230
30.2633.2	§1960(SISLPoint *ppoint, SISLSurf *psurf, gpos, int *jstat)	4230
30.2634	src/s1961.c File Reference	4230
30.2634	Macro Definition Documentation	4230
30.2634.1	§1961	4230
30.2634	Function Documentation	4231
30.2634.2	§1961(ep, int im, int idim, int ipar, epar, eeps, int ilend, int irend, int iopen, double afctol, int itmax, int ik, SISLCurve **rc, emxerr, int *jstat)	4231
30.2635	src/s1962.c File Reference	4231
30.2635	Macro Definition Documentation	4231

30.2635.1	S1962	4231
30.2635	Function Documentation	4232
30.2635.2	S1962(ep, ev, int im, int idim, int ipar, epar, eeps, int ilend, int irend, int iopen, int itmax, SISLCurve **rc, emxerr, int *jstat)	4232
30.2635	/src/s1963.c File Reference	4232
30.2636	Macro Definition Documentation	4232
30.2636.1	S1963	4232
30.2636	Function Documentation	4233
30.2636.2	S1963(SISLCurve *pc, eeps, int ilend, int irend, int iopen, int itmax, SISLCurve **rc, int *jstat)	4233
30.2636	/src/s1965.c File Reference	4233
30.2637	Macro Definition Documentation	4233
30.2637.1	S1965	4233
30.2637	Function Documentation	4234
30.2637.2	S1965(SISLSurf *oldsurf, eps, edgefix, int iopen1, int iopen2, edgeps, int opt, int itmax, SISLSurf **newsurf, maxerr, int *stat)	4234
30.2638	/src/s1966.c File Reference	4234
30.2638	Macro Definition Documentation	4234
30.2638.1	S1966	4234
30.2638	Function Documentation	4235
30.2638.2	S1966(ep, int im1, int im2, int idim, int ipar, epar1, epar2, eeps, nend, int iopen1, int iopen2, edgeps, double afctol, int iopt, int itmax, int ik1, int ik2, SISLSurf **rs, emxerr, int *jstat)	4235
30.2638	/src/s1967.c File Reference	4235
30.2639	Macro Definition Documentation	4235
30.2639.1	S1967	4235
30.2639	Function Documentation	4236
30.2639.2	S1967(ep, etang1, etang2, eder11, int im1, int im2, int idim, int ipar, epar1, epar2, eeps, nend, int iopen1, int iopen2, edgeps, int iopt, int itmax, SISLSurf **rs, emxerr, int *jstat)	4236
30.2640	/src/s1968.c File Reference	4236
30.2640	Macro Definition Documentation	4236
30.2640.1	S1968	4236

30.2640	Function Documentation	4237
30.2640.2	<code>s1968(SISLSurf *ps, eeps, nend, int iopen1, int iopen2, edgeps, int iopt, int itmax, SISLSurf **rs, int *jstat)</code>	4237
30.2640	<code>src/s1986.c</code> File Reference	4237
30.2641	Macro Definition Documentation	4237
30.2641.1	<code>S1986</code>	4237
30.2641	Function Documentation	4238
30.2641.2	<code>s1986(SISLCurve *pc, double aepsge, int *jgtpi, double **gaxis, double *cang, int *jstat)</code>	4238
30.2641	<code>src/s1987.c</code> File Reference	4238
30.2642	Macro Definition Documentation	4238
30.2642.1	<code>S1987</code>	4238
30.2642	Function Documentation	4239
30.2642.2	<code>s1987(SISLSurf *ps, double aepsge, int *jgtpi, double **gaxis, double *cang, int *jstat)</code>	4239
30.2642	<code>src/s1988.c</code> File Reference	4239
30.2643	Macro Definition Documentation	4239
30.2643.1	<code>S1988</code>	4239
30.2643	Function Documentation	4240
30.2643.2	<code>s1988(SISLCurve *pc, double **emax, double **emin, int *jstat)</code>	4240
30.2643	<code>src/s1989.c</code> File Reference	4240
30.2644	Macro Definition Documentation	4240
30.2644.1	<code>S1989</code>	4240
30.2644	Function Documentation	4241
30.2644.2	<code>s1989(SISLSurf *ps, double **emax, double **emin, int *jstat)</code>	4241
30.2644	<code>src/s1990.c</code> File Reference	4241
30.2645	Macro Definition Documentation	4241
30.2645.1	<code>S1990</code>	4241
30.2645	Function Documentation	4242
30.2645.2	<code>s1990(SISLSurf *ps, double aepsge, int *jstat)</code>	4242
30.2645	<code>src/s1991.c</code> File Reference	4242
30.2646	Macro Definition Documentation	4242

30.2646.1	§1991	4242
30.2646	Function Documentation	4243
30.2646.2	§1991(SISLCurve *pc, double aeptge, int *jstat)	4243
30.2647	/src/s1992.c File Reference	4243
30.2647	Macro Definition Documentation	4243
30.2647.1	§1992	4243
30.2647	Function Documentation	4244
30.2647.2	§1992(SISLObject *po, int *jstat)	4244
30.2647.2	§1992cu(SISLCurve *pc, int *jstat)	4244
30.2647.2	§1992su(SISLSurf *ps, int *jstat)	4244
30.2648	/src/s1993.c File Reference	4244
30.2648	Macro Definition Documentation	4245
30.2648.1	§1993	4245
30.2648	Function Documentation	4245
30.2648.2	§1993(SISLCurve *c1, int *jstat)	4245
30.2649	/src/s1994.c File Reference	4245
30.2649	Macro Definition Documentation	4246
30.2649.1	§1994	4246
30.2649	Function Documentation	4246
30.2649.2	§1994(SISLSurf *s1, int *jstat)	4246
30.2650	/src/s2500.c File Reference	4246
30.2650	Macro Definition Documentation	4247
30.2650.1	§2500	4247
30.2650	Function Documentation	4247
30.2650.2	§2500(SISLSurf *surf, int nder, int nder1, int nder2, parvalue, int *leftknot1, int *leftknot2, double *gaussian, int *jstat)	4247
30.2651	/src/s2501.c File Reference	4247
30.2651	Macro Definition Documentation	4248
30.2651.1	§2501	4248
30.2651	Function Documentation	4248
30.2651.2	§2501(SISLSurf *surf, int nder, derive, normal, double *gaussian, int *jstat)	4248

30.2652	/src/s2502.c File Reference	4248
30.2652	Macro Definition Documentation	4249
30.2652.1	\S 2502	4249
30.2652	Function Documentation	4249
30.2652.2	\S 2502(SISLSurf *surf, int nder, int iside1, int iside2, parvalue, int *leftknot1, int *leftknot2, double *meancurvature, int *jstat)	4249
30.2653	/src/s2503.c File Reference	4249
30.2653	Macro Definition Documentation	4250
30.2653.1	\S 2503	4250
30.2653	Function Documentation	4250
30.2653.2	\S 2503(SISLSurf *surf, int nder, derive, normal, double *meancurvature, int *jstat)	4250
30.2654	/src/s2504.c File Reference	4250
30.2654	Macro Definition Documentation	4251
30.2654.1	\S 2504	4251
30.2654	Function Documentation	4251
30.2654.2	\S 2504(SISLSurf *surf, int nder, int iside1, int iside2, parvalue, int *leftknot1, int *leftknot2, double *absCurvature, int *jstat)	4251
30.2655	/src/s2505.c File Reference	4251
30.2655	Macro Definition Documentation	4252
30.2655.1	\S 2505	4252
30.2655	Function Documentation	4252
30.2655.2	\S 2505(SISLSurf *surf, int der, derive, normal, double *absCurvature, int *jstat)	4252
30.2656	/src/s2506.c File Reference	4252
30.2656	Macro Definition Documentation	4253
30.2656.1	\S 2506	4253
30.2656	Function Documentation	4253
30.2656.2	\S 2506(SISLSurf *surf, int nder, int iside1, int iside2, parvalue, int *leftknot1, int *leftknot2, double *totalCurvature, int *jstat)	4253
30.2657	/src/s2507.c File Reference	4253
30.2657	Macro Definition Documentation	4254
30.2657.1	\S 2507	4254
30.2657	Function Documentation	4254

30.2657.2	s2507(SISLSurf *surf, int ider, derive, normal, double *totalCurvature, int *jstat)	4254
30.2658	/src/s2508.c File Reference	4254
30.2658	Macro Definition Documentation	4255
30.2658.1	s2508	4255
30.2658	Function Documentation	4255
30.2658.2	s2508(SISLSurf *surf, int ider, int iside1, int iside2, parvalue, int *leftknot1, int *leftknot2, double *mehlum, int *jstat)	4255
30.2659	/src/s2509.c File Reference	4255
30.2659	Macro Definition Documentation	4256
30.2659.1	s2509	4256
30.2659	Function Documentation	4256
30.2659.2	s2509(SISLSurf *surf, int ider, derive, normal, double *mehlum, int *jstat)	4256
30.2660	/src/s2510.c File Reference	4256
30.2660	Macro Definition Documentation	4257
30.2660.1	s2510	4257
30.2660	Function Documentation	4257
30.2660.2	s2510(SISLSurf *surf, int ider, int iside1, int iside2, parvalue, int *leftknot1, int *leftknot2, double *mehlum, int *jstat)	4257
30.2661	/src/s2511.c File Reference	4257
30.2661	Macro Definition Documentation	4258
30.2661.1	s2511	4258
30.2661	Function Documentation	4258
30.2661.2	s2511(SISLSurf *surf, int ider, derive, normal, double *mehlum, int *jstat)	4258
30.2662	/src/s2512.c File Reference	4258
30.2662	Macro Definition Documentation	4259
30.2662.1	s2512	4259
30.2662	Function Documentation	4259
30.2662.2	s2512(SISLSurf *surf, int ider, int iside1, int iside2, parvalue, int *leftknot1, int *leftknot2, gaussian, int *stat)	4259
30.2663	/src/s2513.c File Reference	4259
30.2663	Macro Definition Documentation	4260
30.2663.1	s2513	4260

30.2663	Function Documentation	4260
30.2663.2	s2513(SISLSurf *surf, int ider, int type, int normalized, derive, normal, fundform, int *stat)	4260
30.2664	/src/s2514.c File Reference	4260
30.2664	Macro Definition Documentation	4261
30.2664.1	s2514	4261
30.2664	Function Documentation	4261
30.2664.2	s2514(SISLSurf *surf, int ider, derive, normal, gaussian, int *stat)	4261
30.2665	/src/s2515.c File Reference	4261
30.2665	Macro Definition Documentation	4262
30.2665.1	s2515	4262
30.2665	Function Documentation	4262
30.2665.2	s2515(SISLSurf *surf, int ider, int iside1, int iside2, parvalue, int *leftknot1, int *leftknot2, mehlum, int *stat)	4262
30.2666	/src/s2516.c File Reference	4262
30.2666	Macro Definition Documentation	4263
30.2666.1	s2516	4263
30.2666	Function Documentation	4263
30.2666.2	s2516(SISLSurf *surf, int ider, derive, normal, mehlum, int *stat)	4263
30.2667	/src/s2532.c File Reference	4263
30.2667	Macro Definition Documentation	4264
30.2667.1	s2532	4264
30.2667	Function Documentation	4264
30.2667.2	s2532(SISLSurf *surf, int u_continuity, int v_continuity, int *u_surfnumb, int *v_surfnumb, SISLSurf ***gauss_surf, int *stat)	4264
30.2668	/src/s2533.c File Reference	4264
30.2668	Macro Definition Documentation	4265
30.2668.1	s2533	4265
30.2668	Function Documentation	4265
30.2668.2	s2533(double *et, int ik, int in, int multinc, int newik, int *newin, double **newet, int *stat)	4265
30.2669	/src/s2534.c File Reference	4265

30.2669	Macro Definition Documentation	4266
30.2669.1	<code>S2534</code>	4266
30.2669	Function Documentation	4266
30.2669.2	<code>S2534</code> (SISLSurf *surf, int u_multinc, int v_multinc, int newik1, int newik2, evalp, int eval_dim, SISLSurf **rsurf, int *stat)	4266
30.2670	<code>src/s2535.c</code> File Reference	4266
30.2670	Macro Definition Documentation	4267
30.2670.1	<code>S2535</code>	4267
30.2670	Function Documentation	4267
30.2670.2	<code>S2535</code> (SISLSurf *surf, int u_continuity, int v_continuity, int *u_surfnumb, int *v_surfnumb, SISLSurf ***patches, int *stat)	4267
30.2671	<code>src/s2536.c</code> File Reference	4267
30.2671	Macro Definition Documentation	4268
30.2671.1	<code>S2536</code>	4268
30.2671	Function Documentation	4268
30.2671.2	<code>S2536</code> (SISLSurf *surf, int u_continuity, int v_continuity, int *u_surfnumb, int *v_surfnumb, SISLSurf ***mehlum_surf, int *stat)	4268
30.2672	<code>src/s2540.c</code> File Reference	4268
30.2672	Macro Definition Documentation	4269
30.2672.1	<code>S2540</code>	4269
30.2672	Function Documentation	4269
30.2672.2	<code>S2540</code> (SISLSurf *surf, int curvature_type, int export_par_val, int pick_subpart, boundary, int n_u, int n_v, double **garr, int *stat)	4269
30.2673	<code>src/s2541.c</code> File Reference	4269
30.2673	Macro Definition Documentation	4270
30.2673.1	<code>S2541</code>	4270
30.2673	Function Documentation	4270
30.2673.2	<code>S2541</code> (SISLSurf *surf, evalp, int dim, int export_par_val, int pick_subpart, boundary, int n_u, int n_v, double **garr, int *stat)	4270
30.2674	<code>src/s2542.c</code> File Reference	4270
30.2674	Macro Definition Documentation	4271
30.2674.1	<code>S2542</code>	4271
30.2674	Function Documentation	4271

30.2674.2.s2542(SISLSurf *surf, int nder, int iside1, int iside2, parvalue, int *leftknot1, int *leftknot2, double *k1, double *k2, d1, d2, int *jstat)	4271
30.2675 /src/s2543.c File Reference	4271
30.2675M Macro Definition Documentation	4272
30.2675.1.s2543	4272
30.2675F Function Documentation	4272
30.2675.2.s2543(SISLSurf *surf, int nder, derive, normal, double *k1, double *k2, d1, d2, int *jstat)	4272
30.2676 /src/s2544.c File Reference	4272
30.2676M Macro Definition Documentation	4273
30.2676.1.s2544	4273
30.2676F Function Documentation	4273
30.2676.2.s2544(SISLSurf *surf, int nder, int iside1, int iside2, parvalue, int *leftknot1, int *leftknot2, norcurv, int *jstat)	4273
30.2677 /src/s2545.c File Reference	4273
30.2677M Macro Definition Documentation	4274
30.2677.1.s2545	4274
30.2677F Function Documentation	4274
30.2677.2.s2545(SISLSurf *surf, int curvature_type, int export_par_val, int pick_subpart, boundary, int n_u, int n_v, double scale, double **garr, int *stat)	4274
30.2678 /src/s2550.c File Reference	4274
30.2678M Macro Definition Documentation	4275
30.2678.1.s2550	4275
30.2678F Function Documentation	4275
30.2678.2.s2550(SISLCurve *curve, ax, int num_ax, curvature, int *jstat)	4275
30.2679 /src/s2551.c File Reference	4275
30.2679M Macro Definition Documentation	4276
30.2679.1.s2551	4276
30.2679F Function Documentation	4276
30.2679.2.s2551(SISLCurve *curve, double parvalue, int *leftknot, derive, double *curvature, int *jstat)	4276
30.2680 /src/s2553.c File Reference	4276
30.2680M Macro Definition Documentation	4277

30.2680.1	§2553	4277
30.2680	Function Documentation	4277
30.2680.2	§2553(SISLCurve *curve, ax, int num_ax, torsion, int *jstat)	4277
30.2680	src/s2554.c File Reference	4277
30.2681	Macro Definition Documentation	4278
30.2681.1	§2554	4278
30.2681	Function Documentation	4278
30.2681.2	§2554(SISLCurve *curve, double parvalue, int *leftknot, derive, double *torsion, int *jstat)	4278
30.2681	src/s2555.c File Reference	4278
30.2682	Macro Definition Documentation	4279
30.2682.1	§2555	4279
30.2682	Function Documentation	4279
30.2682.2	§2555(derive, double *torsion, int *jstat)	4279
30.2682	src/s2556.c File Reference	4279
30.2683	Macro Definition Documentation	4280
30.2683.1	§2556	4280
30.2683	Function Documentation	4280
30.2683.2	§2556(SISLCurve *curve, ax, int num_ax, VoC, int *jstat)	4280
30.2683	src/s2557.c File Reference	4280
30.2684	Macro Definition Documentation	4281
30.2684.1	§2557	4281
30.2684	Function Documentation	4281
30.2684.2	§2557(SISLCurve *curve, double parvalue, int *leftknot, derive, double *VoC, int *jstat)	4281
30.2684	src/s2558.c File Reference	4281
30.2685	Macro Definition Documentation	4282
30.2685.1	§2558	4282
30.2685	Function Documentation	4282
30.2685.2	§2558(derive, int idim, double *VoC, int *jstat)	4282
30.2685	src/s2559.c File Reference	4282

30.2686	Macro Definition Documentation	4283
30.2686.1	<code>S2559</code>	4283
30.2686	Function Documentation	4283
30.2686.2	<code>s2559(SISLCurve *curve, ax, int num_ax, p, t, n, b, int *jstat)</code>	4283
30.2687	<code>src/s2560.c</code> File Reference	4283
30.2687	Macro Definition Documentation	4284
30.2687.1	<code>S2560</code>	4284
30.2687	Function Documentation	4284
30.2687.2	<code>s2560(SISLCurve *curve, double parvalue, int *leftknot, derive, p, t, n, b, int *jstat)</code>	4284
30.2688	<code>src/s2561.c</code> File Reference	4284
30.2688	Macro Definition Documentation	4285
30.2688.1	<code>S2561</code>	4285
30.2688	Function Documentation	4285
30.2688.2	<code>s2561(derive, int idim, p, t, n, b, int *jstat)</code>	4285
30.2689	<code>src/s2562.c</code> File Reference	4285
30.2689	Macro Definition Documentation	4286
30.2689.1	<code>S2562</code>	4286
30.2689	Function Documentation	4286
30.2689.2	<code>s2562(SISLCurve *curve, ax, int num_ax, int val_flag, p, t, n, b, val, int *jstat)</code>	4286
30.2690	<code>src/s6addcurve.c</code> File Reference	4286
30.2690	Macro Definition Documentation	4287
30.2690.1	<code>S6ADDCURVE</code>	4287
30.2690	Function Documentation	4287
30.2690.2	<code>s6addcurve(SISLCurve *pc1, SISLCurve *pc2, int isign, SISLCurve **rcurve, int *jstat)</code>	4287
30.2691	<code>src/s6affdist.c</code> File Reference	4287
30.2691	Macro Definition Documentation	4288
30.2691.1	<code>S6AFFDIST</code>	4288
30.2691	Function Documentation	4288
30.2691.2	<code>s6affdist(e1, e2, emat, int idim)</code>	4288
30.2692	<code>src/s6ang.c</code> File Reference	4288

30.2692	Macro Definition Documentation	4289
30.2692.1	<code>S6ANG</code>	4289
30.2692	Function Documentation	4289
30.2692.2	<code>s6ang(evec1, evec2, int idim)</code>	4289
30.2693	<code>src/s6angle.c</code> File Reference	4289
30.2693	Macro Definition Documentation	4290
30.2693.1	<code>S6ANGLE</code>	4290
30.2693	Function Documentation	4290
30.2693.2	<code>s6angle(evec1, evec2, enorm, int idim, int *jstat)</code>	4290
30.2694	<code>src/s6castelja.c</code> File Reference	4290
30.2694	Macro Definition Documentation	4291
30.2694.1	<code>S6DECASTELJAU</code>	4291
30.2694	Function Documentation	4291
30.2694.2	<code>s6deCasteljau(C, double a, double b, double t, int k, D, int *jstat)</code>	4291
30.2695	<code>src/s6chpar.c</code> File Reference	4291
30.2695	Macro Definition Documentation	4292
30.2695.1	<code>S6CHPAR</code>	4292
30.2695	Function Documentation	4292
30.2695.2	<code>s6chpar(ecoef1, int in1, int in2, int idim, ecoef2)</code>	4292
30.2696	<code>src/s6crss.c</code> File Reference	4292
30.2696	Macro Definition Documentation	4293
30.2696.1	<code>S6CRSS</code>	4293
30.2696	Function Documentation	4293
30.2696.2	<code>s6crss(e1, e2, e3)</code>	4293
30.2697	<code>src/s6crvture.c</code> File Reference	4293
30.2697	Macro Definition Documentation	4294
30.2697.1	<code>S6CURVATURE</code>	4294
30.2697	Function Documentation	4294
30.2697.2	<code>s6curvature(eder, int idim, ecurv, int *jstat)</code>	4294
30.2698	<code>src/s6crvcheck.c</code> File Reference	4294

30.2698	Macro Definition Documentation	4295
30.2698.1	<code>S6CRVCHECK</code>	4295
30.2698	Function Documentation	4295
30.2698.2	<code>s6crvcheck(SISLCurve *pc, int *jstat)</code>	4295
30.2698	<code>src/s6currad.c</code> File Reference	4295
30.2699	Macro Definition Documentation	4296
30.2699.1	<code>S6CURVRAD</code>	4296
30.2699	Function Documentation	4296
30.2699.2	<code>s6currad(epnt1, epnt2, etang, int idim,*crad, int *jstat)</code>	4296
30.2700	<code>src/s6decomp.c</code> File Reference	4296
30.2700	Macro Definition Documentation	4297
30.2700.1	<code>S6DECOMP</code>	4297
30.2700	Function Documentation	4297
30.2700.2	<code>s6decomp(ea, gx, eb1, eb2, eb3, int *jstat)</code>	4297
30.2701	<code>src/s6degnorm.c</code> File Reference	4297
30.2701	Macro Definition Documentation	4298
30.2701.1	<code>S6DEGNORM</code>	4298
30.2701	Function Documentation	4298
30.2701.2	<code>s6degnorm(SISLSurf *ps1, int ider, epar, eder, utang, vtang, enorm, int *jstat)</code>	4298
30.2702	<code>src/s6dertopt.c</code> File Reference	4298
30.2702	Macro Definition Documentation	4299
30.2702.1	<code>S6DERTOPT</code>	4299
30.2702	Function Documentation	4299
30.2702.2	<code>s6dertopt(eder, ntype, int inpt, int idim, epoint, int *jstat)</code>	4299
30.2703	<code>src/s6diff.c</code> File Reference	4299
30.2703	Macro Definition Documentation	4300
30.2703.1	<code>S6DIFF</code>	4300
30.2703	Function Documentation	4300
30.2703.2	<code>s6diff(e1, e2, int idim, e3)</code>	4300
30.2704	<code>src/s6dist.c</code> File Reference	4300

30.2704	Macro Definition Documentation	4301
30.2704.1	<code>S6DIST</code>	4301
30.2704	Function Documentation	4301
30.2704.2	<code>s6dist(epoint1, epoint2, int idim)</code>	4301
30.2705	<code>src/s6dline.c</code> File Reference	4301
30.2705	Macro Definition Documentation	4302
30.2705.1	<code>S6DLINE</code>	4302
30.2705	Function Documentation	4302
30.2705.2	<code>s6dline(estart, eend, epoint, int idim, int *jstat)</code>	4302
30.2706	<code>src/s6dplane.c</code> File Reference	4302
30.2706	Macro Definition Documentation	4303
30.2706.1	<code>S6DPLANE</code>	4303
30.2706	Function Documentation	4303
30.2706.2	<code>s6dplane(eq1, eq2, eq3, epoint, int idim, int *jstat)</code>	4303
30.2707	<code>src/s6drawseq.c</code> File Reference	4303
30.2707	Macro Definition Documentation	4304
30.2707.1	<code>S6DRAWSEQ</code>	4304
30.2707	Function Documentation	4304
30.2707.2	<code>s6drawseq(epoint, int ipoint)</code>	4304
30.2707.2	<code>s6line()</code>	4304
30.2707.2	<code>s6move()</code>	4304
30.2708	<code>src/s6equal.c</code> File Reference	4304
30.2708	Macro Definition Documentation	4305
30.2708.1	<code>S6EQUAL</code>	4305
30.2708	Function Documentation	4305
30.2708.2	<code>s6equal(double a1, double a2, double aref)</code>	4305
30.2709	<code>src/s6err.c</code> File Reference	4305
30.2709	Macro Definition Documentation	4306
30.2709.1	<code>S6ERR</code>	4306
30.2709	Function Documentation	4306

30.2709.2	<code>s6err(char *rut, int jstat, int ipos)</code>	4306
30.2710	<code>src/s6existbox.c</code> File Reference	4306
30.2710	Macro Definition Documentation	4307
30.2710.1	<code>S6EXISTBOX</code>	4307
30.2710	Function Documentation	4307
30.2710.2	<code>s6existbox(SISLbox *pbox, int itype, double aepsge)</code>	4307
30.2711	<code>src/s6findfac.c</code> File Reference	4307
30.2711	Macro Definition Documentation	4308
30.2711.1	<code>S6FINDFAC</code>	4308
30.2711	Function Documentation	4308
30.2711.2	<code>s6findfac(evecu, evecv, evecw, etang, int idim, int isign, double *coef1, double *coef2, double *coef3, int *jstat)</code>	4308
30.2712	<code>src/s6fndintv.c</code> File Reference	4308
30.2712	Macro Definition Documentation	4309
30.2712.1	<code>S6FNDINTVL</code>	4309
30.2712	Function Documentation	4309
30.2712.2	<code>s6fndintvl(double *et, int ik, int in, int *ileft, double ax1, double ax2, int mu_max, int *jstat)</code>	4309
30.2713	<code>src/s6herm.c</code> File Reference	4309
30.2713	Macro Definition Documentation	4310
30.2713.1	<code>S6HERM</code>	4310
30.2713	Function Documentation	4310
30.2713.2	<code>s6herm(double *pt, double *uknots, double *vknots, int unum, int vnum, int dim, int uindex, int vindex, herminfo, int *jstat)</code>	4310
30.2714	<code>src/s6herm_bez.c</code> File Reference	4310
30.2714	Macro Definition Documentation	4311
30.2714.1	<code>S6HERMITE_BEZIER</code>	4311
30.2714	Function Documentation	4311
30.2714.2	<code>s6hermite_bezier(SISLSurf *s, a, b, int idim, c, int *jstat)</code>	4311
30.2715	<code>src/s6idcon.c</code> File Reference	4311
30.2715	Macro Definition Documentation	4312
30.2715.1	<code>S6IDCON</code>	4312

30.2715	Function Documentation	4312
30.2715.2	s6idcon(SISLIntdat **pintdat, SISLIntpt **pintpt1, SISLIntpt **pintpt2, int *jstat)	4312
30.2715	src/s6idcpt.c File Reference	4312
30.2716	Macro Definition Documentation	4313
30.2716.1	S6IDCPT	4313
30.2716	Function Documentation	4313
30.2716.2	s6idcpt(SISLIntdat *pintdat, SISLIntpt *pintpt, SISLIntpt **rintpt)	4313
30.2716	src/s6idedg.c File Reference	4313
30.2717	Macro Definition Documentation	4314
30.2717.1	S6IDEDG	4314
30.2717	Function Documentation	4314
30.2717.2	s6idedg(SISLObject *po1, SISLObject *po2, int iobj, int ipar, double apar, SISLIntdat *pintdat, SISLPtedge **rptedge, int *jnum, int *jstat)	4314
30.2716	src/s6identify.c File Reference	4314
30.2718	Macro Definition Documentation	4315
30.2718.1	S6IDENTIFY	4315
30.2718	Function Documentation	4315
30.2718.2	s6identify(SISLSurf *s, a, b, double level_val, double eps1, double eps2, int *jstat)	4315
30.2716	src/s6idget.c File Reference	4315
30.2719	Macro Definition Documentation	4316
30.2719.1	S6IDGET	4316
30.2719	Function Documentation	4316
30.2719.2	s6idget(SISLObject *po1, SISLObject *po2, int ipar, double apar, SISLIntdat *pintdat, SISLIntdat **rintdat, int *jstat)	4316
30.2716	src/s6idint.c File Reference	4316
30.2720	Macro Definition Documentation	4317
30.2720.1	S6IDINT	4317
30.2720	Function Documentation	4317
30.2720.2	s6idint(SISLObject *po1, SISLObject *po2, SISLIntdat *pintdat, SISLIntpt **rpt, int iob)	4317
30.2716	src/s6idklist.c File Reference	4317
30.2721	Macro Definition Documentation	4318

30.2721.1	<code>S6IDKLIST</code>	4318
30.2721	Function Documentation	4318
30.2721.2	<code>s6idklist(SISLIntdat **pintdat, SISLIntlist *pintlist, int *jstat)</code>	4318
30.2721	<code>src/s6idkpt.c</code> File Reference	4318
30.2722	Macro Definition Documentation	4319
30.2722.1	<code>S6IDKPT</code>	4319
30.2722	Function Documentation	4319
30.2722.2	<code>s6idkpt(SISLIntdat **pintdat, SISLIntpt **pintpt, SISLIntpt **rtpt, SISLIntpt **rfpt, int *jstat)</code>	4319
30.2722	<code>src/s6idlis.c</code> File Reference	4319
30.2723	Macro Definition Documentation	4320
30.2723.1	<code>S6IDLIS</code>	4320
30.2723	Function Documentation	4320
30.2723.2	<code>s6idlis(SISLObject *po1, SISLObject *po2, SISLIntdat **pintdat, int *jstat)</code>	4320
30.2723	<code>src/s6idnpt.c</code> File Reference	4320
30.2724	Macro Definition Documentation	4321
30.2724.1	<code>S6IDNPT</code>	4321
30.2724	Function Documentation	4321
30.2724.2	<code>s6idnpt(SISLIntdat **pintdat, SISLIntpt **pintpt, int itest, int *jstat)</code>	4321
30.2725	<code>src/s6idput.c</code> File Reference	4321
30.2725	Macro Definition Documentation	4322
30.2725.1	<code>S6IDPUT</code>	4322
30.2725	Function Documentation	4322
30.2725.2	<code>s6idput(SISLIntdat **rintdat, SISLIntdat *pintdat, int inr, double apar, int *jstat)</code>	4322
30.2725	<code>src/s6inv4.c</code> File Reference	4322
30.2726	Macro Definition Documentation	4323
30.2726.1	<code>S6INV4</code>	4323
30.2726	Function Documentation	4323
30.2726.2	<code>s6inv4(em, einv, int *jstat)</code>	4323
30.2726	<code>src/s6invert.c</code> File Reference	4323
30.2727	Macro Definition Documentation	4324

30.2727.1	<code>S6INVERT</code>	4324
30.2727	Function Documentation	4324
30.2727.2	<code>s6invert(emat, int im, einvertmat, int *jstat)</code>	4324
30.2728	<code>src/s6knotmult.c</code> File Reference	4324
30.2728	Macro Definition Documentation	4325
30.2728.1	<code>S6KNOTMULT</code>	4325
30.2728	Function Documentation	4325
30.2728.2	<code>s6knotmult(et, int ik, int in, int *ileft, double ax, int *jstat)</code>	4325
30.2729	<code>src/s6length.c</code> File Reference	4325
30.2729	Macro Definition Documentation	4326
30.2729.1	<code>S6LENGTH</code>	4326
30.2729	Function Documentation	4326
30.2729.2	<code>s6length(e1, int idim, int *jstat)</code>	4326
30.2730	<code>src/s6line.c</code> File Reference	4326
30.2730	Macro Definition Documentation	4327
30.2730.1	<code>S6LINE</code>	4327
30.2730	Function Documentation	4327
30.2730.2	<code>s6line(epoint)</code>	4327
30.2731	<code>src/s6lprj.c</code> File Reference	4327
30.2731	Macro Definition Documentation	4328
30.2731.1	<code>S6LPRJ</code>	4328
30.2731	Function Documentation	4328
30.2731.2	<code>s6lprj(e1, e2, int idim)</code>	4328
30.2732	<code>src/s6lufacp.c</code> File Reference	4328
30.2732	Macro Definition Documentation	4329
30.2732.1	<code>S6LUFACP</code>	4329
30.2732	Function Documentation	4329
30.2732.2	<code>s6lufacp(ea, nl, int im, int *jstat)</code>	4329
30.2733	<code>src/s6lusolp.c</code> File Reference	4329
30.2733	Macro Definition Documentation	4330

30.2733.1	S6LUSOLP	4330
30.2733	Function Documentation	4330
30.2733.2	s6lusolp(ea, eb, nl, int im, int *jstat)	4330
30.2733	src/s6metric.c File Reference	4330
30.2734	Macro Definition Documentation	4331
30.2734.1	S6METRIC	4331
30.2734	Function Documentation	4331
30.2734.2	s6metric(epoint, int in, int idim, emat, int *jstat)	4331
30.2734	src/s6move.c File Reference	4331
30.2735	Macro Definition Documentation	4332
30.2735.1	S6MOVE	4332
30.2735	Function Documentation	4332
30.2735.2	s6move(epoint)	4332
30.2735	src/s6mulvec.c File Reference	4332
30.2736	Macro Definition Documentation	4333
30.2736.1	S6MULVEC	4333
30.2736	Function Documentation	4333
30.2736.2	s6mulvec(ematix, evect, eright)	4333
30.2736	src/s6mvec.c File Reference	4333
30.2737	Macro Definition Documentation	4334
30.2737.1	S6MVEC	4334
30.2737	Function Documentation	4334
30.2737.2	s6mvec(emat, evect1, int invec, evect2)	4334
30.2737	src/s6newbox.c File Reference	4334
30.2738	Macro Definition Documentation	4335
30.2738.1	S6NEWBOX	4335
30.2738	Function Documentation	4335
30.2738.2	s6newbox(SISLbox *pbox, int inum, int itype, double aepsge, int *jstat)	4335
30.2738	src/s6norm.c File Reference	4335
30.2739	Macro Definition Documentation	4336

30.2739.1	S6NORM	4336
30.2739	Function Documentation	4336
30.2739.2	s6norm(e1, int idim, e2, int *jstat)	4336
30.2740	src/s6ratder.c File Reference	4336
30.2740	Macro Definition Documentation	4337
30.2740.1	S6RATDER	4337
30.2740	Function Documentation	4337
30.2740.2	s6ratder(eder, int idim, int ider, gder, int *jstat)	4337
30.2741	src/s6rotax.c File Reference	4337
30.2741	Macro Definition Documentation	4338
30.2741.1	S6ROTAX	4338
30.2741	Function Documentation	4338
30.2741.2	s6rotax(ep, eaxis, expnt, emat, int *jstat)	4338
30.2742	src/s6rotmat.c File Reference	4338
30.2742	Macro Definition Documentation	4339
30.2742.1	S6ROTMAT	4339
30.2742	Function Documentation	4339
30.2742.2	s6rotmat(eorigo, eaxis, enorm, ematrix, int *jstat)	4339
30.2743	src/s6schoen.c File Reference	4339
30.2743	Macro Definition Documentation	4340
30.2743.1	S6SCHOEN	4340
30.2743	Function Documentation	4340
30.2743.2	s6schoen(et, int ik, int index)	4340
30.2744	src/s6scpr.c File Reference	4340
30.2744	Macro Definition Documentation	4341
30.2744.1	S6SCPR	4341
30.2744	Function Documentation	4341
30.2744.2	s6scpr(e1, e2, int idim)	4341
30.2745	src/s6sortpar.c File Reference	4341
30.2745	Macro Definition Documentation	4342

30.2745.1	<code>S6SORTPAR</code>	4342
30.2745	Function Documentation	4342
30.2745.2	<code>s6sortpar(evec1, epar1, int ipar, int idim, evec2, epar2, int *jstat)</code>	4342
30.2745	<code>src/s6sratder.c</code> File Reference	4342
30.2746	Macro Definition Documentation	4343
30.2746.1	<code>S6SRATDER</code>	4343
30.2746	Function Documentation	4343
30.2746.2	<code>s6sratder(eder, int idim, int ider1, int ider2, gder, int *jstat)</code>	4343
30.2746	<code>src/s6strider.c</code> File Reference	4343
30.2747	Macro Definition Documentation	4344
30.2747.1	<code>S6STRIDER</code>	4344
30.2747	Function Documentation	4344
30.2747.2	<code>s6strider(eder, int idim, int ider, gder, int *jstat)</code>	4344
30.2748	<code>src/s6takunion.c</code> File Reference	4344
30.2748	Macro Definition Documentation	4345
30.2748.1	<code>S6TAKEUNION</code>	4345
30.2748	Function Documentation	4345
30.2748.2	<code>s6takeunion(evec1, int ielem1, evec2, int ielem2, **gunion, int *jnmelem, int *jstat)</code>	4345
30.2749	<code>src/s6twonorm.c</code> File Reference	4345
30.2749	Macro Definition Documentation	4346
30.2749.1	<code>S6TWONORM</code>	4346
30.2749	Function Documentation	4346
30.2749.2	<code>s6twonorm(evec, enorm1, enorm2, int *jstat)</code>	4346
30.2750	<code>src/s9adsimp.c</code> File Reference	4346
30.2750	Macro Definition Documentation	4347
30.2750.1	<code>S9ADSIMP</code>	4347
30.2750	Function Documentation	4347
30.2750.2	<code>s9adsimp(epnt1, epar1, eimpli, int ideg, egd1, epgd1, etang, eptan, double astep, int *jstat)</code>	4347
30.2751	<code>src/s9adstep.c</code> File Reference	4347
30.2751	Macro Definition Documentation	4348

30.2751.1	S9ADSTEP	4348
30.2751	Function Documentation	4348
30.2751.2	s9adstep(epnt1, epar1, epnt2, epar2, egd1, epgd1, egd2, epgd2, etang, eptan1, eptan2, double astep, int *jstat)	4348
30.2751	src/s9boundimp.c File Reference	4348
30.2752	Macro Definition Documentation	4349
30.2752.1	S9BOUNDIMP	4349
30.2752	Function Documentation	4349
30.2752.2	s9boundimp(epnt1, epar1, SISLSurf *psurf1, eimpli, int ideg, double apar, int idir, double aepsge, gpnt1, gpar1, int *jstat)	4349
30.2752	src/s9boundit.c File Reference	4349
30.2753	Macro Definition Documentation	4350
30.2753.1	S9BOUNDIT	4350
30.2753	Function Documentation	4350
30.2753.2	s9boundit(epnt1, epnt2, epar1, epar2, SISLSurf *psurf1, SISLSurf *psurf2, double apar, int idir, double aepsge, gpnt1, gpnt2, gpar1, gpar2, int *jstat)	4350
30.2754	src/s9clipimp.c File Reference	4350
30.2754	Macro Definition Documentation	4351
30.2754.1	S9CLIPIMP	4351
30.2754	Function Documentation	4351
30.2754.2	s9clipimp(epar1, epar2, SISLSurf *psurf1, eimpli, int ideg, euval, eval, double aepsge, gpnt1, gpar1, int *jstat)	4351
30.2755	src/s9clipit.c File Reference	4351
30.2755	Macro Definition Documentation	4352
30.2755.1	S9CLIPIT	4352
30.2755	Function Documentation	4352
30.2755.2	s9clipit(epar11, epar12, epar21, epar22, SISLSurf *psurf1, SISLSurf *psurf2, euval, eval, esval, etval, double aepsge, gpnt1, gpnt2, gpar1, gpar2, int *jstat)	4352
30.2756	src/s9conmarch.c File Reference	4352
30.2756	Macro Definition Documentation	4353
30.2756.1	S9CONMARCH	4353
30.2756	Function Documentation	4353

30.2756.2	s9conmarch(SISLSurf *ps, double alevel, epar, ndir, int ipoint, gpar, mpar, int *jpoint, int *jstat)	4353
30.2756	lib/src/s9iterate.c File Reference	4353
30.2757	Macro Definition Documentation	4354
30.2757.1	S9ITERATE	4354
30.2757	Function Documentation	4354
30.2757.2	s9iterate(epoint, epnt1, epnt2, epar1, epar2, SISLSurf *psurf1, SISLSurf *psurf2, double astepl, double aepsge, gpnt1, gpnt2, gpar1, gpar2, int *jstat)	4354
30.2758	lib/src/s9iterimp.c File Reference	4354
30.2758	Macro Definition Documentation	4355
30.2758.1	S9ITERIMP	4355
30.2758	Function Documentation	4355
30.2758.2	s9iterimp(epoint, epnt1, epar1, SISLSurf *psurf1, eimpli, int ideg, double astepl, double aepsge, gpnt1, gpar1, int *jstat)	4355
30.2759	lib/src/s9simplknot.c File Reference	4355
30.2759	Macro Definition Documentation	4356
30.2759.1	S9SIMPLE_KNOT	4356
30.2759	Function Documentation	4356
30.2759.2	s9simple_knot(SISLSurf *surf, int idiv, epar, int *fixflag, int *jstat)	4356
30.2760	lib/src/s9surmarch.c File Reference	4356
30.2760	Macro Definition Documentation	4357
30.2760.1	S9SURMARCH	4357
30.2760	Function Documentation	4357
30.2760.2	s9surmarch(SISLSurf *ps1, SISLSurf *ps2, epar, ndir, int ipoint, gpar, mpar, int *jpoint, int *jstat)	4357
30.2761	lib/src/sh1260.c File Reference	4357
30.2761	Macro Definition Documentation	4358
30.2761.1	SH1260	4358
30.2761	Function Documentation	4358
30.2761.2	sh1260(double aconst, vcurve, int icurve, int *jstat)	4358
30.2762	lib/src/sh1261.c File Reference	4358
30.2762	Macro Definition Documentation	4359

30.2762.1	SH1261	4359
30.2762	Function Documentation	4359
30.2762.2	sh1261(SISLCurve *pcurve1, SISLCurve *pcurve2, ecoef1, int ik1, ecoef2, int ik2, SISLCurve **rcrtanc, int *jstat)	4359
30.2763	/src/sh1262.c File Reference	4359
30.2763	Macro Definition Documentation	4360
30.2763.1	SH1262	4360
30.2763	Function Documentation	4360
30.2763.2	sh1262(vcurve, int iedge, int inmbx, ecoef, int *jstat)	4360
30.2764	/src/sh1263.c File Reference	4360
30.2764	Macro Definition Documentation	4361
30.2764.1	SH1263	4361
30.2764	Function Documentation	4361
30.2764.2	sh1263(vcurve, int iedge, vboundc, int *jstat)	4361
30.2765	/src/sh1365.c File Reference	4361
30.2765	Macro Definition Documentation	4362
30.2765.1	SH1365	4362
30.2765	Function Documentation	4362
30.2765.2	sh1365(SISLCurve *pcurve, etau, int ik, int in, int ileftfix, int irlightfix, SISLCurve **newcurve, double **gmaxerr, double **gl2err, int *jstat)	4362
30.2766	/src/sh1369.c File Reference	4362
30.2766	Macro Definition Documentation	4363
30.2766.1	SH1369	4363
30.2766	Function Documentation	4363
30.2766.2	sh1369(SISLSurf *ps, ecentr, enorm, double abigr, double asmalr, int idim, double aepsco, double aepsge, int trackflag, int *jtrack, SISLTrack ***wtrack, int *jpt, double **gpar, int **pretop, int *jcrv, SISLIntcurve ***wcurve, int *jsurf, SISLIntsurf ***wsurf, int *jstat)	4363
30.2767	/src/sh1371.c File Reference	4363
30.2767	Macro Definition Documentation	4364
30.2767.1	SH1371	4364
30.2767	Function Documentation	4364

30.2767.2sh1371(SISLCurve *pc1, ecentr, double aradiu, int idim, double aepsco, double aepsge, int trackflag, int *jtrack, SISLTrack ***wtrack, int *jpt, double **gpar, int **pretop, int *jcrv, SISLIntcurve ***wcurve, int *jstat) 4364

30.2768src/sh1372.c File Reference 4364

30.2768Macro Definition Documentation 4365

30.2768.1SH1372 4365

30.2768Function Documentation 4365

30.2768.2sh1372(SISLCurve *pc1, epoint, edirec, double aradiu, int idim, double aepsco, double aepsge, int trackflag, int *jtrack, SISLTrack ***wtrack, int *jpt, double **gpar, int **pretop, int *jcrv, SISLIntcurve ***wcurve, int *jstat) 4365

30.2769src/sh1373.c File Reference 4365

30.2769Macro Definition Documentation 4366

30.2769.1SH1373 4366

30.2769Function Documentation 4366

30.2769.2sh1373(SISLCurve *pc1, etop, eaxis, econc, int idim, double aepsco, double aepsge, int trackflag, int *jtrack, SISLTrack ***wtrack, int *jpt, double **gpar, int **pretop, int *jcrv, SISLIntcurve ***wcurve, int *jstat) 4366

30.2770src/sh1374.c File Reference 4366

30.2770Macro Definition Documentation 4367

30.2770.1SH1374 4367

30.2770Function Documentation 4367

30.2770.2sh1374(SISLCurve *pc1, earray, int idim, double aepsco, double aepsge, int trackflag, int *jtrack, SISLTrack ***wtrack, int *jpt, double **gpar, int **pretop, int *jcrv, SISLIntcurve ***wcurve, int *jstat) 4367

30.2771src/sh1375.c File Reference 4367

30.2771Macro Definition Documentation 4368

30.2771.1SH1375 4368

30.2771Function Documentation 4368

30.2771.2sh1375(SISLCurve *pc1, ecentr, enorm, double abigr, double asmalr, int idim, double aepsco, double aepsge, int trackflag, int *jtrack, SISLTrack ***wtrack, int *jpt, double **gpar, int **pretop, int *jcrv, SISLIntcurve ***wcurve, int *jstat) 4368

30.2772src/sh1460.c File Reference 4368

30.2772Macro Definition Documentation 4369

30.2772.1SH1460 4369

30.2772Predef Documentation 4369

30.2772.2	fevalmidProc	4369
30.2772.2	fshapeProc	4369
30.2772	Function Documentation	4369
30.2772.3	sh1460(fshapeProc fshape, vboundc, int icurv, SISLSurf ***wsurf, int *jstat)	4369
30.2772	src/sh1461.c File Reference	4369
30.2773	Macro Definition Documentation	4370
30.2773.1	SH1461	4370
30.2773	typedef Documentation	4370
30.2773.2	fshapeProc	4370
30.2773.2	initProc	4370
30.2773	Function Documentation	4370
30.2773.3	sh1461(fshapeProc fshape, initProc f_initmid, vboundc, int icurv, vsurf, int *jstat)	4370
30.2773	src/sh1462.c File Reference	4371
30.2774	Macro Definition Documentation	4371
30.2774.1	SH1462	4371
30.2774	typedef Documentation	4371
30.2774.2	fshapeProc	4371
30.2774	Function Documentation	4372
30.2774.3	sh1462(fshapeProc fshape, vboundc, int icurv, etwist, etang, eder, int *jstat)	4372
30.2775	src/sh1463.c File Reference	4372
30.2775	Macro Definition Documentation	4372
30.2775.1	SH1463	4372
30.2775	typedef Documentation	4373
30.2775.2	fshapeProc	4373
30.2775	Function Documentation	4373
30.2775.3	sh1463(fshapeProc fshape, vboundc, int icurv, etwist, etang, eder, int *jstat)	4373
30.2775	src/sh1464.c File Reference	4373
30.2776	Macro Definition Documentation	4374
30.2776.1	SH1464	4374
30.2776	typedef Documentation	4374

30.2776.2	fshapeProc	4374
30.2776	Function Documentation	4374
30.2776.3	sh1464(fshapeProc fshape, vboundc, int icurv, etwist, etang, eder, int *jstat)	4374
30.2776	src/sh1465.c File Reference	4374
30.2777	Macro Definition Documentation	4375
30.2777.1	SH1465	4375
30.2777	Predef Documentation	4375
30.2777.2	fshapeProc	4375
30.2777	Function Documentation	4375
30.2777.3	sh1465(fshapeProc fshape, vboundc, int icurv, etwist, etang, eder, int *jstat)	4375
30.2777	src/sh1466.c File Reference	4375
30.2778	Macro Definition Documentation	4376
30.2778.1	SH1466	4376
30.2778	Function Documentation	4376
30.2778.2	sh1466(ecurve, etwist, int ider, ebar, eval, int *jstat)	4376
30.2778	src/sh1467.c File Reference	4376
30.2779	Macro Definition Documentation	4377
30.2779.1	SH1467	4377
30.2779	Function Documentation	4377
30.2779.2	sh1467(ecurve, etwist, int ider, ebar, eval, int *jstat)	4377
30.2780	src/sh1502.c File Reference	4377
30.2780	Macro Definition Documentation	4378
30.2780.1	SH1502	4378
30.2780	Function Documentation	4378
30.2780.2	sh1502(SISLCurve *pc1, base, norm, axisA, alpha, ratio, int idim, double aepsco, double aepsge, int trackflag, int *jtrack, SISLTrack ***wtrack, int *jpt, double **gpar, int **pretop, int *jcrv, SISLIntcurve ***wcurve, int *jstat)	4378
30.2780	src/sh1503.c File Reference	4378
30.2781	Macro Definition Documentation	4379
30.2781.1	SH1503	4379
30.2781	Function Documentation	4379

30.2781.2	sh1503(SISLSurf *ps1, base, norm, axisA, double alpha, double ratio, int idim, double aepsco, double aepsge, int trackflag, int *jtrack, SISLTrack ***wtrack, int *jpt, double **gpar, int **pretop, int *jcrv, SISLIntcurve ***wcurve, int *jsurf, SISLIntsurf ***wsurf, int *jstat)	4379
30.2781	src/sh1510.c File Reference	4379
30.2782	Macro Definition Documentation	4380
30.2782.1	SH1510	4380
30.2782	Function Documentation	4380
30.2782.2	sh1510(SISLSurf *ps, eyepoint, int idim, double aepsco, double aepsge, int trackflag, int *jtrack, SISLTrack ***wtrack, int *jpt, double **gpar, int **pretop, int *jcrv, SISLIntcurve ***wcurve, int *jsurf, SISLIntsurf ***wsurf, int *jstat)	4380
30.2783	src/sh1511.c File Reference	4380
30.2783	Macro Definition Documentation	4381
30.2783.1	SH1511	4381
30.2783	Function Documentation	4381
30.2783.2	sh1511(SISLSurf *ps, qpoint, bvec, int idim, double aepsco, double aepsge, int trackflag, int *jtrack, SISLTrack ***wtrack, int *jpt, double **gpar, int **pretop, int *jcrv, SISLIntcurve ***wcurve, int *jsurf, SISLIntsurf ***wsurf, int *jstat)	4381
30.2784	src/sh1761.c File Reference	4381
30.2784	Macro Definition Documentation	4382
30.2784.1	SH1761	4382
30.2784	Function Documentation	4382
30.2784.2	sh1761(SISLObject *po1, SISLObject *po2, double aepsge, SISLIntdat **pintdat, int *jstat)	4382
30.2785	src/sh1762.c File Reference	4382
30.2785	Macro Definition Documentation	4383
30.2785.1	SH1762	4383
30.2785	Function Documentation	4383
30.2785.2	sh1762(SISLObject *po1, SISLObject *po2, double aepsge, SISLIntdat **pintdat, vedge, int *jstat)	4383
30.2786	src/sh1779.c File Reference	4383
30.2786	Macro Definition Documentation	4384
30.2786.1	SH1779	4384
30.2786	Function Documentation	4384

30.2786.2	sh1779(SISLObject *po1, SISLObject *po2, double aepsge, SISLIntdat **rintdat, SISLIntpt *pintpt, int *jnewpt, int *jstat)	4384
30.2786	src/sh1779_at.c File Reference	4384
30.2787	Macro Definition Documentation	4385
30.2787.1	SH1779_AT	4385
30.2787	Function Documentation	4385
30.2787.2	sh1779_at(SISLObject *po1, SISLObject *po2, SISLIntpt *pintpt, int *jstat)	4385
30.2788	src/sh1780.c File Reference	4385
30.2788	Macro Definition Documentation	4386
30.2788.1	SH1780	4386
30.2788	Function Documentation	4386
30.2788.2	sh1780(SISLObject *po1, SISLObject *po2, double aepsge, SISLIntdat **rintdat, SISLIntpt *pintpt, int *jnewpt, int *jstat)	4386
30.2789	src/sh1780_at.c File Reference	4386
30.2789	Macro Definition Documentation	4387
30.2789.1	SH1780_AT	4387
30.2789	Function Documentation	4387
30.2789.2	sh1780_at(SISLObject *po1, SISLObject *po2, SISLIntpt *pintpt, int *jstat)	4387
30.2790	src/sh1781.c File Reference	4387
30.2790	Macro Definition Documentation	4388
30.2790.1	SH1781	4388
30.2790	Function Documentation	4388
30.2790.2	sh1781(SISLObject *po1, SISLObject *po2, double aepsge, SISLIntdat **rintdat, SISLIntpt *pintpt, int *jnewpt, int *jstat)	4388
30.2791	src/sh1781_at.c File Reference	4388
30.2791	Macro Definition Documentation	4389
30.2791.1	SH1781_AT	4389
30.2791	Function Documentation	4389
30.2791.2	sh1781_at(SISLObject *po1, SISLObject *po2, SISLIntpt *pintpt, int *jstat)	4389
30.2792	src/sh1782.c File Reference	4389
30.2792	Macro Definition Documentation	4390
30.2792.1	SH1782	4390

30.2792	Function Documentation	4390
30.2792.2	sh1782(SISLObject *po1, SISLObject *po2, double aepsge, SISLIntdat *pintdat, int ipar, double apar, SISLIntdat **rintdat, int *jnewpt, int *jstat)	4390
30.2793	/src/sh1783.c File Reference	4390
30.2793	Macro Definition Documentation	4391
30.2793.1	SH1783	4391
30.2793	Typedef Documentation	4391
30.2793.2	fevalProc	4391
30.2793	Function Documentation	4391
30.2793.3	sh1783(SISLCurve *pc1, SISLCurve *pc2, double aepsge, epar, int idir1, int idir2, elast, enext, int *jstat)	4391
30.2794	/src/sh1784.c File Reference	4391
30.2794	Macro Definition Documentation	4392
30.2794.1	SH1784	4392
30.2794	Typedef Documentation	4392
30.2794.2	fevalcProc	4392
30.2794	Function Documentation	4392
30.2794.3	sh1784(SISLCurve *pcurve, SISLSurf *psurf, double aepsge, epar, int icur, int idirc, elast, enext, int *jstat)	4392
30.2795	/src/sh1786.c File Reference	4393
30.2795	Macro Definition Documentation	4393
30.2795.1	SH1786	4393
30.2795	Function Documentation	4393
30.2795.2	sh1786(SISLObject *po1, SISLObject *po2, double aepsge, SISLIntdat **rintdat, SISLIntpt *pintpt, int *jnewpt, int *jstat)	4393
30.2796	/src/sh1787.c File Reference	4394
30.2796	Macro Definition Documentation	4394
30.2796.1	SH1787	4394
30.2796	Function Documentation	4394
30.2796.2	sh1787(SISLObject *po1, SISLObject *po2, double aepsge, SISLIntdat **rintdat, SISLIntpt *pintpt, int *jnewpt, int *jstat)	4394
30.2797	/src/sh1790.c File Reference	4395
30.2797	Macro Definition Documentation	4395

30.2797.1	SH1790	4395
30.2797	Function Documentation	4395
30.2797.2	sh1790(SISLObject *po1, SISLObject *po2, int itype, double aepsge, int *jstat)	4395
30.2798	src/sh1830.c File Reference	4396
30.2798	Macro Definition Documentation	4396
30.2798.1	SH1830	4396
30.2798	Function Documentation	4396
30.2798.2	sh1830(SISLObject *po1, SISLObject *po2, double aepsge, int *jstat)	4396
30.2799	src/sh1831.c File Reference	4397
30.2799	Macro Definition Documentation	4397
30.2799.1	SH1831	4397
30.2799	Function Documentation	4397
30.2799.2	sh1831(SISLCurve *pc1, SISLCurve *pc2, int isign, epoint, enorm, double aepsge, int *jstat)	4397
30.2800	src/sh1834.c File Reference	4398
30.2800	Macro Definition Documentation	4398
30.2800.1	SH1834	4398
30.2800	Function Documentation	4398
30.2800.2	sh1834(SISLObject *po1, SISLObject *po2, double aepsge, int idim, edir1, edir2, int *jstat)	4398
30.2801	src/sh1839.c File Reference	4399
30.2801	Macro Definition Documentation	4399
30.2801.1	SH1839	4399
30.2801	Function Documentation	4399
30.2801.2	sh1839(SISLObject *po1, SISLObject *po2, double aepsge, int *jstat)	4399
30.2802	src/sh1850.c File Reference	4400
30.2802	Macro Definition Documentation	4400
30.2802.1	SH1850	4400
30.2802	Function Documentation	4400
30.2802.2	sh1850(SISLCurve *pc1, epoint, enorm, int idim, double aepsco, double aepsge, int trackflag, int *jtrack, SISLTrack ***wtrack, int *jpt, double **gpar, int **pretop, int *jcrv, SISLIntcurve ***wcurve, int *jstat)	4400

30.2803	/src/sh1851.c File Reference	4401
30.2803	Macro Definition Documentation	4401
30.2803.1	SH1851	4401
30.2803	Function Documentation	4401
30.2803.2	sh1851(SISLSurf *ps1, epoint, enorm, int idim, double aepsco, double aepsge, int trackflag, int *jtrack, SISLTrack ***wtrack, int *jpt, double **gpar, int **pretop, int *jcrv, SISLIntcurve ***wcurve, int *jsurf, SISLIntsurf ***wsurf, int *jstat)	4401
30.2804	/src/sh1852.c File Reference	4402
30.2804	Macro Definition Documentation	4402
30.2804.1	SH1852	4402
30.2804	Function Documentation	4402
30.2804.2	sh1852(SISLSurf *ps1, ecenter, double aradius, int idim, double aepsco, double aepsge, int trackflag, int *jtrack, SISLTrack ***wtrack, int *jpt, double **gpar, int **pretop, int *jcrv, SISLIntcurve ***wcurve, int *jsurf, SISLIntsurf ***wsurf, int *jstat)	4402
30.2805	/src/sh1853.c File Reference	4403
30.2805	Macro Definition Documentation	4403
30.2805.1	SH1853	4403
30.2805	Function Documentation	4403
30.2805.2	sh1853(SISLSurf *ps1, epoint, edirec, double aradius, int idim, double aepsco, double aepsge, int trackflag, int *jtrack, SISLTrack ***wtrack, int *jpt, double **gpar, int **pretop, int *jcrv, SISLIntcurve ***wcurve, int *jsurf, SISLIntsurf ***wsurf, int *jstat)	4403
30.2806	/src/sh1854.c File Reference	4404
30.2806	Macro Definition Documentation	4404
30.2806.1	SH1854	4404
30.2806	Function Documentation	4404
30.2806.2	sh1854(SISLSurf *ps1, etop, eaxis, econc, int idim, double aepsco, double aepsge, int trackflag, int *jtrack, SISLTrack ***wtrack, int *jpt, double **gpar, int **pretop, int *jcrv, SISLIntcurve ***wcurve, int *jsurf, SISLIntsurf ***wsurf, int *jstat)	4404
30.2807	/src/sh1855.c File Reference	4405
30.2807	Macro Definition Documentation	4405
30.2807.1	SH1855	4405
30.2807	Function Documentation	4405

30.2807.2sh1855(SISLSurf *ps1, ecentr, double aradius, enorm, int idim, double aepsco, double aepsge, int trackflag, int *jtrack, SISLTrack ***wtrack, int *jpt, double **gpar, int **pretop, int *jcrv, SISLIntcurve ***wcurve, int *jstat) 4405

30.2808src/sh1856.c File Reference 4406

30.2808Macro Definition Documentation 4406

30.2808.1SH1856 4406

30.2808Function Documentation 4406

30.2808.2sh1856(SISLSurf *ps1, epoint, edir, int idim, double aepsco, double aepsge, int trackflag, int *jtrack, SISLTrack ***wtrack, int *jpt, double **gpar, int **pretop, int *jcrv, SISLIntcurve ***wcurve, int *jstat) 4406

30.2809src/sh1857.c File Reference 4407

30.2809Macro Definition Documentation 4407

30.2809.1SH1857 4407

30.2809Function Documentation 4407

30.2809.2sh1857(SISLCurve *pc1, SISLCurve *pc2, double aepsco, double aepsge, int trackflag, int *jtrack, SISLTrack ***wtrack, int *jpt, double **gpar1, double **gpar2, int **pretop, int *jcrv, SISLIntcurve ***wcurve, int *jstat) 4407

30.2810src/sh1858.c File Reference 4408

30.2810Macro Definition Documentation 4408

30.2810.1SH1858 4408

30.2810Function Documentation 4408

30.2810.2sh1858(SISLSurf *ps1, SISLCurve *pc1, double aepsco, double aepsge, int trackflag, int *jtrack, SISLTrack ***wtrack, int *jpt, double **gpar1, double **gpar2, int **pretop, int *jcrv, SISLIntcurve ***wcurve, int *jstat) 4408

30.2811src/sh1859.c File Reference 4409

30.2811Macro Definition Documentation 4409

30.2811.1SH1859 4409

30.2811Function Documentation 4409

30.2811.2sh1859(SISLSurf *ps1, SISLSurf *ps2, double aepsco, double aepsge, int trackflag, int *jtrack, SISLTrack ***wtrack, int *jpt, double **gpar1, double **gpar2, int **pretop, int *jcrv, SISLIntcurve ***wcurve, int *jsurf, SISLIntsurf ***wsurf, int *jstat) 4409

30.2812src/sh1860.c File Reference 4410

30.2812Macro Definition Documentation 4410

30.2812.1SH1860 4410

30.2812	Function Documentation	4410
30.2812.2	sh1860(SISLSurf *ps, eview, int idim, double aepsco, double aepsge, int trackflag, int *jtrack, SISLTrack ***wtrack, int *jpt, double **gpar, int **pretop, int *jcrv, SISLIntcurve ***wcurve, int *jsurf, SISLIntsurf ***wsurf, int *jstat)	4410
30.2813	/src/sh1870.c File Reference	4411
30.2813	Macro Definition Documentation	4411
30.2813.1	SH1870	4411
30.2813	Function Documentation	4411
30.2813.2	sh1870(SISLSurf *ps1, double *pt1, int idim, double aepsco, double aepsge, int trackflag, int *jtrack, SISLTrack ***wtrack, int *jpt, double **gpar1, int **pretop, int *jcrv, SISLIntcurve ***wcurve, int *jstat)	4411
30.2814	/src/sh1871.c File Reference	4412
30.2814	Macro Definition Documentation	4412
30.2814.1	SH1871	4412
30.2814	Function Documentation	4412
30.2814.2	sh1871(SISLCurve *pc1, double *pt1, int idim, double aepsco, double aepsge, int trackflag, int *jtrack, SISLTrack ***wtrack, int *jpt, double **gpar1, int **pretop, int *jcrv, SISLIntcurve ***wcurve, int *jstat)	4412
30.2815	/src/sh1922.c File Reference	4413
30.2815	Macro Definition Documentation	4413
30.2815.1	SH1922	4413
30.2815	Function Documentation	4413
30.2815.2	sh1922(et, int im, int ik, etau, int in, ea, nfirst, nlast, int *jstat)	4413
30.2816	/src/sh1923.c File Reference	4414
30.2816	Macro Definition Documentation	4414
30.2816.1	SH1923	4414
30.2816	Function Documentation	4414
30.2816.2	sh1923(double *ea, int in, int ik, int *nstart, int *jstat)	4414
30.2817	/src/sh1924.c File Reference	4415
30.2817	Macro Definition Documentation	4415
30.2817.1	SH1924	4415
30.2817	Function Documentation	4415
30.2817.2	sh1924(double *ea, double *eb, int in, int ik, int idim, int *nstart, int *jstat)	4415

30.2818	/src/sh1925.c File Reference	4416
30.2818	Macro Definition Documentation	4416
30.2818.1	SH1925	4416
30.2818	Function Documentation	4416
30.2818.2	sh1925(SISLCurve *pc1, SISLCurve *pc2, int idim, ea, nstart, nstop, emxerr, el2err, int ilend, int irend, int *jstat)	4416
30.2819	/src/sh1926.c File Reference	4417
30.2819	Macro Definition Documentation	4417
30.2819.1	SH1926	4417
30.2819	Function Documentation	4417
30.2819.2	sh1926(etau, int ik, int in, int idim, et, ed, int im, ea, nfirst, nlast, eb, n2sta, ec, int *jstat)	4417
30.2820	/src/sh1927.c File Reference	4418
30.2820	Macro Definition Documentation	4418
30.2820.1	SH1927	4418
30.2820	Function Documentation	4418
30.2820.2	sh1927(etau, int ik, int in, int idim, SISLCurve *pcurve, int ilend, int irend, ec, int *jstat)	4418
30.2821	/src/sh1928.c File Reference	4419
30.2821	Macro Definition Documentation	4419
30.2821.1	SH1928	4419
30.2821	Function Documentation	4419
30.2821.2	sh1928(etau, int ik, int in, int idim, et, ed, int im, int ilend, int irend, ea, int inh, nfirst, nlast, eb, ec, n2sta, int *jstat)	4419
30.2822	/src/sh1929.c File Reference	4420
30.2822	Macro Definition Documentation	4420
30.2822.1	SH1929	4420
30.2822	Function Documentation	4420
30.2822.2	sh1929(etau, int in, int ik, int imu, et, int im, int ij, eah, int *jmuprm, int *jnu, int *jstat)	4420
30.2823	/src/sh1930.c File Reference	4421
30.2823	Macro Definition Documentation	4421
30.2823.1	SH1930	4421

30.2823	Function Documentation	4421
30.2823.2	sh1930(ea, nfirst, nlast, ed, ec, int ik, int in, int im, int idim, int ilend, int irend, int *jstat)	4421
30.2824	/src/sh1992.c File Reference	4422
30.2824	Macro Definition Documentation	4422
30.2824.1	SH1992	4422
30.2824	Function Documentation	4422
30.2824.2	sh1992(SISLObject *po, int itype, double aepsge, int *jstat)	4422
30.2824.2	sh1992cu(SISLCurve *pc, int itype, double aepsge, int *jstat)	4423
30.2824.2	sh1992su(SISLSurf *ps, int itype, double aepsge, int *jstat)	4423
30.2825	/src/sh1993.c File Reference	4423
30.2825	Macro Definition Documentation	4423
30.2825.1	SH1993	4423
30.2825	Function Documentation	4424
30.2825.2	sh1993(SISLCurve *c1, double aepsge, int *jstat)	4424
30.2826	/src/sh1994.c File Reference	4424
30.2826	Macro Definition Documentation	4424
30.2826.1	SH1994	4424
30.2826	Function Documentation	4425
30.2826.2	sh1994(SISLSurf *s1, double aepsge, int *jstat)	4425
30.2827	/src/sh6clvert.c File Reference	4425
30.2827	Macro Definition Documentation	4425
30.2827.1	SH6CLOSEVERT	4425
30.2827	Function Documentation	4426
30.2827.2	sh6closevert(SISLCurve *pcurve, SISLSurf *psurf, double *cpar1, epar2)	4426
30.2828	/src/sh6comedg.c File Reference	4426
30.2828	Macro Definition Documentation	4426
30.2828.1	SH6COMEDG	4426
30.2828	Function Documentation	4427
30.2828.2	sh6comedg(SISLObject *po1, SISLObject *po2, SISLIntpt *pt1, SISLIntpt *pt2, int *jstat)	4427

30.2828	src/sh6connect.c File Reference	4427
30.2829	Macro Definition Documentation	4427
30.2829.1	SH6CONNECT	4427
30.2829	Function Documentation	4428
30.2829.2	sh6connect(SISLIntpt *pt1, SISLIntpt *pt2, int *jstat)	4428
30.2830	src/sh6count.c File Reference	4428
30.2830	Macro Definition Documentation	4428
30.2830.1	SH6COUNT	4428
30.2830	Function Documentation	4429
30.2830.2	sh6count(SISLIntpt *pt, int *jstat)	4429
30.2831	src/sh6cvert.c File Reference	4429
30.2831	Macro Definition Documentation	4429
30.2831.1	SH6CVVERT	4429
30.2831	Function Documentation	4430
30.2831.2	sh6cvert(SISLCurve *pc1, SISLCurve *pc2, double *cpar1, double *cpar2)	4430
30.2832	src/sh6degen.c File Reference	4430
30.2832	Macro Definition Documentation	4430
30.2832.1	SH6DEGEN	4430
30.2832	Function Documentation	4431
30.2832.2	sh6degen(SISLObject *po1, SISLObject *po2, SISLIntdat **pintdat, double aepsge, int *jstat)	4431
30.2833	src/sh6disconn.c File Reference	4431
30.2833	Macro Definition Documentation	4431
30.2833.1	SH6DISCONNECT	4431
30.2833	Function Documentation	4432
30.2833.2	sh6disconnect(SISLIntpt *pt1, SISLIntpt *pt2, int *jstat)	4432
30.2834	src/sh6edgpnt.c File Reference	4432
30.2834	Macro Definition Documentation	4432
30.2834.1	SH6EDGPOINT	4432
30.2834	Function Documentation	4433
30.2834.2	sh6edgpoint(vedge, SISLIntpt ***wintpt, int *jnum, int *jstat)	4433

30.2835	/src/sh6edgred.c File Reference	4433
30.2835	Macro Definition Documentation	4433
30.2835.1	SH6EDGRED	4433
30.2835	Function Documentation	4434
30.2835.2	sh6edgred(SISLObject *po1, SISLObject *po2, SISLIntdat *pintdat, int *jstat)	4434
30.2836	/src/sh6evalint.c File Reference	4434
30.2836	Macro Definition Documentation	4434
30.2836.1	SH6EVALINT	4434
30.2836	Function Documentation	4435
30.2836.2	sh6evalint(SISLObject *ob1, SISLObject *ob2, eimpli, int ideg, SISLIntpt *pt, double aepsge, curve_3d, curve_2d_1, curve_2d_2, int *jstat)	4435
30.2837	/src/sh6floop.c File Reference	4435
30.2837	Macro Definition Documentation	4435
30.2837.1	SH6FLOOP	4435
30.2837	Function Documentation	4436
30.2837.2	sh6floop(vedgept, int inum, int *jpt, int *jstat)	4436
30.2838	/src/sh6findspl.c File Reference	4436
30.2838	Macro Definition Documentation	4436
30.2838.1	SH6FINDSPLIT	4436
30.2838	Function Documentation	4437
30.2838.2	sh6findsplit(SISLSurf *ps1, SISLSurf *ps2, double aepsge, int *jstat)	4437
30.2839	/src/sh6getgeom.c File Reference	4437
30.2839	Macro Definition Documentation	4437
30.2839.1	SH6GETGEOM	4437
30.2839	Function Documentation	4438
30.2839.2	sh6getgeom(SISLObject *ob, int obr, SISLIntpt *pt, double **geom, double **norm, double aepsge, int *jstat)	4438
30.2840	/src/sh6getlist.c File Reference	4438
30.2840	Macro Definition Documentation	4438
30.2840.1	SH6GETLIST	4438
30.2840	Function Documentation	4439

30.2840.2.sh6getlist(SISLIntpt *pt1, SISLIntpt *pt2, int *index1, int *index2, int *jstat) . . . 4439

30.2840.1./src/sh6getmain.c File Reference 4439

30.2841Macro Definition Documentation 4439

30.2841.1.SH6GETMAIN 4439

30.2841.2.Function Documentation 4440

30.2841.2.sh6getmain(SISLIntpt *pt) 4440

30.2842./src/sh6getnbrs.c File Reference 4440

30.2842Macro Definition Documentation 4440

30.2842.1.SH6GETNHBRIS 4440

30.2842.2.Function Documentation 4441

30.2842.2.sh6getnhbrs(SISLIntpt *pt, SISLIntpt **pt1, SISLIntpt **pt2, int *jstat) 4441

30.2843./src/sh6getnext.c File Reference 4441

30.2843Macro Definition Documentation 4441

30.2843.1.SH6GETNEXT 4441

30.2843.2.Function Documentation 4442

30.2843.2.sh6getnext(SISLIntpt *pt, int index) 4442

30.2844./src/sh6getothr.c File Reference 4442

30.2844Macro Definition Documentation 4442

30.2844.1.SH6GETOTHER 4442

30.2844.2.Function Documentation 4443

30.2844.2.sh6getother(SISLIntpt *pt, SISLIntpt *pt1, SISLIntpt **pt2, int *jstat) 4443

30.2845./src/sh6getprev.c File Reference 4443

30.2845Macro Definition Documentation 4443

30.2845.1.SH6GETPREV 4443

30.2845.2.Function Documentation 4444

30.2845.2.sh6getprev(SISLIntpt *pt1, SISLIntpt *pt2) 4444

30.2846./src/sh6gettop.c File Reference 4444

30.2846Macro Definition Documentation 4444

30.2846.1.SH6GETTOP 4444

30.2846.2.Function Documentation 4445

30.2846.2	<code>sh6gettop(SISLIntpt *pt, int ilist, int *left1, int *right1, int *left2, int *right2, int *jstat)</code>	4445
30.2846	<code>isl/src/sh6idaledg.c</code> File Reference	4445
30.2847	Macro Definition Documentation	4445
30.2847.1	<code>SH6ALLEDG</code>	4445
30.2847	Function Documentation	4446
30.2847.2	<code>sh6idalledg(SISLObject *pob1, SISLObject *pob2, SISLIntdat *pintdat, wedge, int *jstat)</code>	4446
30.2846	<code>isl/src/sh6idcon.c</code> File Reference	4446
30.2848	Macro Definition Documentation	4446
30.2848.1	<code>SH6IDCON</code>	4446
30.2848	Function Documentation	4447
30.2848.2	<code>sh6idcon(SISLIntdat **pintdat, SISLIntpt **pintpt1, SISLIntpt **pintpt2, int *jstat)</code>	4447
30.2846	<code>isl/src/sh6idfcros.c</code> File Reference	4447
30.2849	Macro Definition Documentation	4447
30.2849.1	<code>SH6IDFCROSS</code>	4447
30.2849	Function Documentation	4448
30.2849.2	<code>sh6idfcross(SISLIntdat *pintdat, vcross, int *jncross, int ipar1, int ipar2, int *jstat)</code>	4448
30.2850	<code>isl/src/sh6idget.c</code> File Reference	4448
30.2850	Macro Definition Documentation	4448
30.2850.1	<code>SH6IDGET</code>	4448
30.2850	Function Documentation	4449
30.2850.2	<code>sh6idget(SISLObject *po1, SISLObject *po2, int ipar, double apar, SISLIntdat *pintdat, SISLIntdat **rintdat, double aepsge, int *jstat)</code>	4449
30.2851	<code>isl/src/sh6idkpt.c</code> File Reference	4449
30.2851	Macro Definition Documentation	4449
30.2851.1	<code>SH6IDKPT</code>	4449
30.2851	Function Documentation	4450
30.2851.2	<code>sh6idkpt(SISLIntdat **pintdat, SISLIntpt **pintpt, int join, int *jstat)</code>	4450
30.2852	<code>isl/src/sh6idlis.c</code> File Reference	4450
30.2852	Macro Definition Documentation	4450
30.2852.1	<code>SH6IDLIS</code>	4450

30.2852	Function Documentation	4451
30.2852.2	sh6idlis(SISLObject *po1, SISLObject *po2, SISLIntdat **pintdat, double aepsge, int *jstat)	4451
30.2853	/src/sh6idnpt.c File Reference	4451
30.2853	Macro Definition Documentation	4451
30.2853.1	SH6IDNPT	4451
30.2853	Function Documentation	4452
30.2853.2	sh6idnpt(SISLIntdat **pintdat, SISLIntpt **pintpt, int itest, int *jstat)	4452
30.2854	/src/sh6idnwun.c File Reference	4452
30.2854	Macro Definition Documentation	4452
30.2854.1	SH6IDNEWUNITE	4452
30.2854	Function Documentation	4453
30.2854.2	sh6idnewunite(SISLObject *po1, SISLObject *po2, SISLIntdat **intdat, SISLIntpt **pt1, SISLIntpt **pt2, double weight, double aepsge, int *jstat)	4453
30.2855	/src/sh6idput.c File Reference	4453
30.2855	Macro Definition Documentation	4453
30.2855.1	SH6IDPUT	4453
30.2855	Function Documentation	4454
30.2855.2	sh6idput(SISLObject *po1, SISLObject *po2, SISLIntdat **rintdat, SISLIntdat *pintdat, int inr, double apar, SISLIntpt ***outintpt, int *npoint, int *jstat)	4454
30.2856	/src/sh6idrcros.c File Reference	4454
30.2856	Macro Definition Documentation	4454
30.2856.1	SH6IDRMCROSS	4454
30.2856	Function Documentation	4455
30.2856.2	sh6idrmcross(SISLObject *po1, SISLObject *po2, SISLIntdat **pintdat, vcross, int incross, vpt, int inpt, int *jstat)	4455
30.2857	/src/sh6idsplit.c File Reference	4455
30.2857	Macro Definition Documentation	4455
30.2857.1	SH6IDSPLIT	4455
30.2857	Function Documentation	4456
30.2857.2	sh6idsplit(SISLIntdat **pintdat, SISLIntpt *psource, int *jstat)	4456
30.2858	/src/sh6idunite.c File Reference	4456

30.2858	Macro Definition Documentation	4456
30.2858.1	<code>\$H6IDUNITE</code>	4456
30.2858	Function Documentation	4457
30.2858.2	<code>sh6idunite(SISLIntdat **intdat, SISLIntpt **pt1, SISLIntpt **pt2, double weight, int *jstat)</code>	4457
30.2858	<code>src/sh6insert.c</code> File Reference	4457
30.2859	Macro Definition Documentation	4457
30.2859.1	<code>\$H6INSERT</code>	4457
30.2859	Function Documentation	4458
30.2859.2	<code>sh6insert(SISLIntdat **pintdat, SISLIntpt *pt1, SISLIntpt *pt2, SISLIntpt **ptnew, int *jstat)</code>	4458
30.2860	<code>src/sh6inspnt.c</code> File Reference	4458
30.2860	Macro Definition Documentation	4458
30.2860.1	<code>\$H6INSERTPT</code>	4458
30.2860	Function Documentation	4459
30.2860.2	<code>sh6insertpt(SISLIntpt *pt1, SISLIntpt *pt2, SISLIntpt *ptnew, int *jstat)</code>	4459
30.2861	<code>src/sh6iscnect.c</code> File Reference	4459
30.2861	Macro Definition Documentation	4459
30.2861.1	<code>\$H6ISCONNECT</code>	4459
30.2861	Function Documentation	4460
30.2861.2	<code>sh6isconnect(SISLIntpt *pt0, SISLIntpt *pt1, SISLIntpt *pt2)</code>	4460
30.2862	<code>src/sh6ishelp.c</code> File Reference	4460
30.2862	Macro Definition Documentation	4460
30.2862.1	<code>\$H6ISHHELP</code>	4460
30.2862	Function Documentation	4461
30.2862.2	<code>sh6ishelp(SISLIntpt *pt)</code>	4461
30.2863	<code>src/sh6isinsid.c</code> File Reference	4461
30.2863	Macro Definition Documentation	4461
30.2863.1	<code>\$H6ISINSIDE</code>	4461
30.2863	Function Documentation	4462
30.2863.2	<code>sh6isinside(SISLObject *po1, SISLObject *po2, SISLIntpt *intpt, int *jstat)</code>	4462

30.2864	/src/sh6ismain.c File Reference	4462
30.2864	Macro Definition Documentation	4462
30.2864.1	SH6ISMAIN	4462
30.2864	Function Documentation	4463
30.2864.2	sh6ismain(SISLIntpt *pt)	4463
30.2865	/src/sh6nmbhelp.c File Reference	4463
30.2865	Macro Definition Documentation	4463
30.2865.1	SH6NMBHELP	4463
30.2865	Function Documentation	4464
30.2865.2	sh6nmbhelp(SISLIntpt *pt, int *jstat)	4464
30.2866	/src/sh6nmbmain.c File Reference	4464
30.2866	Macro Definition Documentation	4464
30.2866.1	SH6NMBMAIN	4464
30.2866	Function Documentation	4465
30.2866.2	sh6nmbmain(SISLIntpt *pt, int *jstat)	4465
30.2867	/src/sh6ptobj.c File Reference	4465
30.2867	Macro Definition Documentation	4465
30.2867.1	SH6PTOBJ	4465
30.2867	Function Documentation	4466
30.2867.2	sh6ptobj(double *point, SISLObject *obj, double aeapsge, start, result, int *jstat)	4466
30.2868	/src/sh6ptouchp.c File Reference	4466
30.2868	Macro Definition Documentation	4466
30.2868.1	SH6PUTTOUCH	4466
30.2868	Function Documentation	4467
30.2868.2	sh6puttouch(SISLIntpt *psource, SISLIntpt *pdest, int seq)	4467
30.2869	/src/sh6putsing.c File Reference	4467
30.2869	Macro Definition Documentation	4467
30.2869.1	SH6PUTSING	4467
30.2869	Function Documentation	4468
30.2869.2	sh6putsing(SISLIntpt *psource, SISLIntpt *pdest)	4468

30.2870	src/sh6red.c File Reference	4468
30.2870	Macro Definition Documentation	4468
30.2870.1	SH6RED	4468
30.2870	Function Documentation	4469
30.2870.2	sh6red(SISLObject *po1, SISLObject *po2, SISLIntdat *pintdat, int *jstat)	4469
30.2871	src/sh6remcon.c File Reference	4469
30.2871	Macro Definition Documentation	4469
30.2871.1	SH6REMCON	4469
30.2871	Function Documentation	4470
30.2871.2	sh6remcon(SISLIntdat **pintdat, SISLIntpt *pt1, SISLIntpt *pt2, int *jstat)	4470
30.2872	src/sh6rempt.c File Reference	4470
30.2872	Macro Definition Documentation	4470
30.2872.1	SH6REMOVEPT	4470
30.2872	Function Documentation	4471
30.2872.2	sh6removept(SISLIntpt *pt1, SISLIntpt *pt2, SISLIntpt *ptold, int *jstat)	4471
30.2873	src/sh6sepcrv.c File Reference	4471
30.2873	Macro Definition Documentation	4471
30.2873.1	SH6SEPCRV	4471
30.2873	Function Documentation	4472
30.2873.2	sh6sepcrv(SISLCurve *pc1, SISLCurve *pc2, double aeptsge, ecentre, double *crad, int *jstat)	4472
30.2874	src/sh6setcnsd.c File Reference	4472
30.2874	Macro Definition Documentation	4472
30.2874.1	SH6SETCNSDIR	4472
30.2874	Function Documentation	4473
30.2874.2	sh6setcnsdir(SISLIntpt *pt1, SISLIntpt *pt2, int ipar, int *jstat)	4473
30.2875	src/sh6setdir.c File Reference	4473
30.2875	Macro Definition Documentation	4473
30.2875.1	SH6SETDIR	4473
30.2875	Function Documentation	4474
30.2875.2	sh6setdir(SISLIntpt *pt1, SISLIntpt *pt2, int *jstat)	4474

30.2876	/src/sh6settop.c File Reference	4474
30.2876	Macro Definition Documentation	4474
30.2876.1	SH6SETTOP	4474
30.2876	Function Documentation	4475
30.2876.2	sh6settop(SISLIntpt *pt, int ilist, int left1, int right1, int left2, int right2, int *jstat)	4475
30.2877	/src/sh6spltgeo.c File Reference	4475
30.2877	Macro Definition Documentation	4475
30.2877.1	SH6SPLITGEOM	4475
30.2877	Function Documentation	4476
30.2877.2	sh6splitgeom(SISLSurf *ps1, SISLSurf *ps2, double aeapsge, ecentre, eaxis, double *cdist, double *crad, int *jstat)	4476
30.2878	/src/sh6tohelp.c File Reference	4476
30.2878	Macro Definition Documentation	4476
30.2878.1	SH6TOHELP	4476
30.2878	Function Documentation	4477
30.2878.2	sh6tohelp(SISLIntpt *pt, int *jstat)	4477
30.2879	/src/sh6tomain.c File Reference	4477
30.2879	Macro Definition Documentation	4477
30.2879.1	SH6TOMAIN	4477
30.2879	Function Documentation	4478
30.2879.2	sh6tomain(SISLIntpt *pt, int *jstat)	4478
30.2880	/src/sh6topohlp.c File Reference	4478
30.2880	Macro Definition Documentation	4478
30.2880.1	SH6GETTOPHLP	4478
30.2880	Function Documentation	4479
30.2880.2	sh6gettophp(SISLIntpt *pt, pretop, int case_2d, int *jstat)	4479
30.2881	/src/sh6trimlist.c File Reference	4479
30.2881	Macro Definition Documentation	4479
30.2881.1	SH6TRIMLIST	4479
30.2881	Function Documentation	4480
30.2881.2	sh6trimlist(SISLIntpt *pt, SISLIntpt ***ptlist, int *no_of_points, int *no_alloc)	4480

30.2882	/src/sh_1d_div.c File Reference	4480
30.2882	Macro Definition Documentation	4480
30.2882.1	\$SH_1D_DIV	4480
30.2882	Function Documentation	4481
30.2882.2	sh_1d_div(SISLObject *po1, SISLObject *po2, double aepsge, SISLIntdat **pintdat, vedge, int *jstat)	4481
30.2883	/src/sh_div_crv.c File Reference	4481
30.2883	Macro Definition Documentation	4481
30.2883.1	\$SH_DIV_CRV	4481
30.2883	Function Documentation	4482
30.2883.2	sh_div_crv(SISLCurve *pc, int which_end, double aepsge, SISLCurve **rcnew, int *jstat)	4482
30.2884	/src/sh_set_at.c File Reference	4482
30.2884	Macro Definition Documentation	4482
30.2884.1	\$SH_SET_AT	4482
30.2884	Function Documentation	4483
30.2884.2	sh_set_at(SISLObject *po1, SISLObject *po2, SISLIntdat *pintdat, int *jstat)	4483
30.2885	/src/shape.c File Reference	4483
30.2885	Macro Definition Documentation	4483
30.2885.1	\$SHAPE	4483
30.2885	Function Documentation	4484
30.2885.2	shape(emid, etang, int idim, int iedge, int *jstat)	4484
30.2886	/src/shcheckput.c File Reference	4484
30.2886	Macro Definition Documentation	4484
30.2886.1	\$SHCHECKPUT	4484
30.2886	Function Documentation	4485
30.2886.2	shcheckput(SISLObject *po1, SISLIntdat **rintdat, SISLIntdat *pintdat, int inr, double apar, int *jstat)	4485
30.2887	/src/shchecktyp.c File Reference	4485
30.2887	Macro Definition Documentation	4485
30.2887.1	\$SHCHECKTYPE	4485
30.2887	Function Documentation	4486

30.2887.2	shchecktype(SISLObject *pobj, double *parval)	4486
30.2887	isl/src/shcsfsing.c File Reference	4486
30.2888	Macro Definition Documentation	4486
30.2888.1	\$SHCSFSING	4486
30.2888	Function Documentation	4487
30.2888.2	shcsfsing(SISLCurve *pcurve, SISLSurf *psurf, limit, enext, gpos, int *jstat)	4487
30.2889	isl/src/shdivsurf.c File Reference	4487
30.2889	Macro Definition Documentation	4487
30.2889.1	\$SH_DIV_SURF	4487
30.2889	Function Documentation	4488
30.2889.2	sh_div_surf(SISLSurf *ps, int which_end_1, int which_end_2, double aepsge, SISLSurf **rsnew, int *jstat)	4488
30.2890	isl/src/shevalc.c File Reference	4488
30.2890	Macro Definition Documentation	4488
30.2890.1	\$SHEVALC	4488
30.2890	Function Documentation	4489
30.2890.2	shevalc(SISLCurve *pc1, int ider, double ax, double aepsge, int *ileft, eder, int *jstat)	4489
30.2891	isl/src/shmkhlppts.c File Reference	4489
30.2891	Macro Definition Documentation	4489
30.2891.1	\$SHMKHLPPTS	4489
30.2891	Function Documentation	4490
30.2891.2	shmkhlppts(SISLObject *po1, SISLObject *po2, double aepsge, SISLIntdat **rintdat, vedge, int *jnewpt, int *jstat)	4490
30.2892	isl/src/shsing.c File Reference	4490
30.2892	Macro Definition Documentation	4490
30.2892.1	\$HSING	4490
30.2892	Function Documentation	4491
30.2892.2	shsing(SISLSurf *psurf1, SISLSurf *psurf2, limit, enext, gpos, int *jstat)	4491
30.2893	isl/src/spli_silh.c File Reference	4491
30.2893	Macro Definition Documentation	4491
30.2893.1	\$PLI_SILH	4491

30.2893	Function Documentation	4492
30.2893.2	apli_silh(SISLIntdat **pintdat, SISLObject *po1, int *jstat)	4492
30.2894	/src/tstcyclknt.c File Reference	4492
30.2894	Macro Definition Documentation	4492
30.2894.1	TEST_CYCLIC_KNOTS	4492
30.2894	Function Documentation	4493
30.2894.2	test_cyclic_knots(et, int in, int ik, int *jstat)	4493
30.2895	topology/include/GoTools/topology/FaceAdjacency.h File Reference	4493
30.2896	topology/include/GoTools/topology/FaceConnectivity.h File Reference	4494
30.2897	topology/include/GoTools/topology/FaceConnectivityUtils.h File Reference	4495
30.2898	topology/include/GoTools/topology/topology_doxyman.h File Reference	4495
30.2899	topology/include/GoTools/topology/tpEdge.h File Reference	4495
30.2900	topology/include/GoTools/topology/tpFace.h File Reference	4496
30.2901	topology/include/GoTools/topology/tpJointType.h File Reference	4497
30.2902	topology/include/GoTools/topology/tpTolerances.h File Reference	4498
30.2903	topology/include/GoTools/topology/tpTopologyTable.h File Reference	4498
30.2904	topology/include/GoTools/topology/tpUtils.h File Reference	4499
30.2905	trivariate/include/GoTools/trivariate/ConeVolume.h File Reference	4500
30.2906	trivariate/include/GoTools/trivariate/CoonsPatchVolumeGen.h File Reference	4501
30.2907	trivariate/include/GoTools/trivariate/CurveOnVolume.h File Reference	4502
30.2908	trivariate/include/GoTools/trivariate/CylinderVolume.h File Reference	4502
30.2909	trivariate/include/GoTools/trivariate/ElementaryVolume.h File Reference	4503
30.2910	trivariate/include/GoTools/trivariate/examples_trivariate_doxyman.h File Reference	4504
30.2911	trivariate/include/GoTools/trivariate/GapRemovalVolume.h File Reference	4504
30.2912	trivariate/include/GoTools/trivariate/LoftVolumeCreator.h File Reference	4505
30.2913	trivariate/include/GoTools/trivariate/Parallelepiped.h File Reference	4506
30.2914	trivariate/include/GoTools/trivariate/ParamVolume.h File Reference	4507
30.2915	trivariate/include/GoTools/trivariate/RectangularVolumeTesselator.h File Reference	4507
30.2916	trivariate/include/GoTools/trivariate/RegularVolMesh.h File Reference	4509
30.2917	trivariate/include/GoTools/trivariate/SmoothVolume.h File Reference	4510

30.2918	trivariate/include/GoTools/trivariate/SphereVolume.h File Reference	4511
30.2919	trivariate/include/GoTools/trivariate/SplineVolume.h File Reference	4512
30.2919	Macro Definition Documentation	4513
30.2919.1	<code>SPLINE_VOLUME_BD_SFS_SIZE</code>	4513
30.2919.1	<code>SPLINE_VOLUME_DEGEN_SIZE</code>	4513
30.2919.1	<code>SPLINE_VOLUME_PERIOD_INFO_SIZE</code>	4513
30.2920	trivariate/include/GoTools/trivariate/SurfaceOnVolume.h File Reference	4513
30.2921	trivariate/include/GoTools/trivariate/SurfaceOnVolumeTools.h File Reference	4514
30.2922	trivariate/include/GoTools/trivariate/SweepVolumeCreator.h File Reference	4515
30.2923	trivariate/include/GoTools/trivariate/TorusVolume.h File Reference	4516
30.2924	trivariate/include/GoTools/trivariate/trivariate_doxyman.h File Reference	4516
30.2925	trivariate/include/GoTools/trivariate/VolumeInterpolator.h File Reference	4516
30.2926	trivariate/include/GoTools/trivariate/VolumeParameterCurve.h File Reference	4517
30.2927	trivariate/include/GoTools/trivariate/VolumeSpaceCurve.h File Reference	4518
30.2928	trivariate/include/GoTools/trivariate/VolumeTools.h File Reference	4519
30.2929	trivariatemodel/include/GoTools/trivariatemodel/examples_trivariatemodel_doxyman.h File Reference	4520
30.2930	trivariatemodel/include/GoTools/trivariatemodel/ftVolume.h File Reference	4520
30.2931	trivariatemodel/include/GoTools/trivariatemodel/ftVolumeTools.h File Reference	4521
30.2932	trivariatemodel/include/GoTools/trivariatemodel/trivariatemodel_doxyman.h File Reference	4522
30.2933	trivariatemodel/include/GoTools/trivariatemodel/VolumeAdjacency.h File Reference	4523
30.2934	trivariatemodel/include/GoTools/trivariatemodel/VolumeModel.h File Reference	4523
30.2935	trivariatemodel/include/GoTools/trivariatemodel/VolumeModelCreator.h File Reference	4524
30.2936	trivariatemodel/include/GoTools/trivariatemodel/VolumeModelFileHandler.h File Reference	4525
30.2936	Macro Definition Documentation	4526
30.2936.1	<code>_VOLUMEFILEHANDLER_H</code>	4526
30.2937	examples/hesimplest/main.cpp File Reference	4526
30.2937	Function Documentation	4526
30.2937.1	<code>EqPoints(hed::Node *&p1, hed::Node *&p2)</code>	4526
30.2937.1	<code>LexPoint(const hed::Node *p1, const hed::Node *p2)</code>	4526
30.2937.1	<code>main()</code>	4527

30.2938	include/ttl/api.h File Reference	4527
30.2939	include/ttl/halfedge/HeDart.h File Reference	4527
30.2940	include/ttl/halfedge/HeTraits.h File Reference	4528
30.2941	include/ttl/halfedge/HeTriang.h File Reference	4529
30.2941	Macro Definition Documentation	4530
30.2941.1	TTL_USE_NODE_FLAG	4530
30.2941.1	TTL_USE_NODE_ID	4531
30.2942	include/ttl/halfedge/main_he_ref.h File Reference	4531
30.2943	include/ttl/halfedge/mainpage_hed.h File Reference	4531
30.2944	include/ttl/mainpage_ttl.h File Reference	4531
30.2945	include/ttl/ttl.h File Reference	4531
30.2946	include/ttl/ttl_constr.h File Reference	4534
30.2947	include/ttl/ttl_util.h File Reference	4535
30.2948	include/ttl/utlis/Handle.h File Reference	4536
30.2949	include/ttl/utlis/HandleId.h File Reference	4538
30.2950	src/halfedge/HeTriang.cpp File Reference	4539
30.2951	src/utlis/HandleId.cpp File Reference	4539
30.2952	viewlib/include/GoTools/viewlib/CurveResolutionSheet.h File Reference	4540
30.2953	viewlib/include/GoTools/viewlib/DataHandler.h File Reference	4540
30.2954	viewlib/include/GoTools/viewlib/DefaultDataHandler.h File Reference	4541
30.2955	viewlib/include/GoTools/viewlib/gvActionSheet.h File Reference	4543
30.2956	viewlib/include/GoTools/viewlib/gvApplication.h File Reference	4543
30.2956	Typedef Documentation	4544
30.2956.1	ColContainer	4544
30.2956.1	ObjContainer	4544
30.2957	viewlib/include/GoTools/viewlib/gvCamera.h File Reference	4545
30.2958	viewlib/include/GoTools/viewlib/gvColor.h File Reference	4546
30.2959	viewlib/include/GoTools/viewlib/gvCurvePaintable.h File Reference	4547
30.2960	viewlib/include/GoTools/viewlib/gvData.h File Reference	4548
30.2961	viewlib/include/GoTools/viewlib/gvGenericTriPaintable.h File Reference	4549

30.2962	viewlib/include/GoTools/viewlib/gvGenericTriQuadMesh.h File Reference	4550
30.2963	viewlib/include/GoTools/viewlib/gvGenericTriQuadPaintable.h File Reference	4551
30.2964	viewlib/include/GoTools/viewlib/gvGroup.h File Reference	4552
30.2965	viewlib/include/GoTools/viewlib/gvGroupPropertySheet.h File Reference	4553
30.2966	viewlib/include/GoTools/viewlib/gvLineCloudPaintable.h File Reference	4553
30.2967	viewlib/include/GoTools/viewlib/gvNoopPaintable.h File Reference	4554
30.2968	viewlib/include/GoTools/viewlib/gvObjectList.h File Reference	4554
30.2969	viewlib/include/GoTools/viewlib/gvObserver.h File Reference	4555
30.2970	viewlib/include/GoTools/viewlib/gvPaintable.h File Reference	4555
30.2971	viewlib/include/GoTools/viewlib/gvPainter.h File Reference	4556
30.2972	viewlib/include/GoTools/viewlib/gvParametricSurfacePaintable.h File Reference	4558
30.2972	Typedef Documentation	4558
30.2972.1	genMesh	4558
30.2973	viewlib/include/GoTools/viewlib/gvPointCloudPaintable.h File Reference	4559
30.2974	viewlib/include/GoTools/viewlib/gvPropertySheet.h File Reference	4559
30.2975	viewlib/include/GoTools/viewlib/gvQuadsPaintable.h File Reference	4560
30.2976	viewlib/include/GoTools/viewlib/gvRectangularSurfacePaintable.h File Reference	4560
30.2977	viewlib/include/GoTools/viewlib/gvResolutionDialog.h File Reference	4561
30.2978	viewlib/include/GoTools/viewlib/gvStandardMouseHandler.h File Reference	4562
30.2978	Macro Definition Documentation	4562
30.2978.1	GV_STANDARD_MOUSEHANDLER_H_INCLUDED	4562
30.2979	viewlib/include/GoTools/viewlib/gvTexture.h File Reference	4562
30.2979	Enumeration Type Documentation	4563
30.2979.1	EnvModeSet	4563
30.2979.1	MagFilterSet	4564
30.2979.1	MinFilterSet	4564
30.2979.1	WrapModeSet	4564
30.2980	viewlib/include/GoTools/viewlib/gvUtilities.h File Reference	4564
30.2980	Typedef Documentation	4565
30.2980.1	FLOAT_TYPE	4565

30.2980.1	MATRIX3	4565
30.2980.1	MATRIX4	4565
30.2980	Function Documentation	4565
30.2980.2	draw_cylinder(double x0, double y0, double z0, double x1, double y1, double z1, double radius, double radius2, int n)	4565
30.2980.2	draw_gl_axes(int n, double r, double radius, double rim, double l)	4565
30.2980.2	draw_gl_axes(double relscale)	4566
30.2980.2	m3_det(MATRIX3 mat)	4566
30.2980.2	m3_identity(MATRIX3 mat)	4566
30.2980.2	m3_inverse(MATRIX3 mr, MATRIX3 ma)	4566
30.2980.2	m4_det(MATRIX4 mr)	4566
30.2980.2	m4_inverse(MATRIX4 mr, MATRIX4 ma)	4566
30.2980.2	m4_submat(MATRIX4 mr, MATRIX3 mb, int i, int j)	4566
30.2980	viewlib/include/GoTools/viewlib/gvView.h File Reference	4566
30.2980	viewlib/include/GoTools/viewlib/ParametricSurfacePropertySheet.h File Reference	4567
30.2980	viewlib/include/GoTools/viewlib/PointCloudPropertySheet.h File Reference	4567
30.2980	viewlib/include/GoTools/viewlib/raster.h File Reference	4568
30.2984	Macro Definition Documentation	4568
30.2984.1	RASTERUTILS_H_INCLUDED	4568
30.2984	Function Documentation	4568
30.2984.2	read_ppm_file(const char *const name, int *const xs, int *const ys)	4568
30.2984.2	write_ppm_file(const char *const name, const unsigned char *const d, const int xs, const int ys, bool rgb=true)	4568
30.2985	viewlib/include/GoTools/viewlib/RectangularSurfacePropertySheet.h File Reference	4569
30.2986	viewlib/include/GoTools/viewlib/SplineCurvePropertySheet.h File Reference	4569
30.2987	viewlib/include/GoTools/viewlib/SurfaceResolutionSheet.h File Reference	4570
30.2988	viewlib/include/GoTools/viewlib/viewlib_doxyman.h File Reference	4570
30.2989	viewlib/include/GoTools/viewlib/vol_and_lr/DataHandlerVolAndLR.h File Reference	4570
30.2990	viewlib/include/GoTools/viewlib/vol_and_lr/gvApplicationVolAndLR.h File Reference	4571
30.2991	viewlib/include/GoTools/viewlib/vol_and_lr/gvRectangularVolumePaintable.h File Reference	4572
30.2992	viewlib/include/GoTools/viewlib/vol_and_lr/RectangularVolumePropertySheet.h File Reference	4574

31 Example Documentation	4575
31.1 adapt_curve	4575
31.2 append_curve	4575
31.3 approx_curve	4575
31.4 approx_surface	4576
31.5 BrepToTrivariate	4576
31.6 circle	4576
31.7 closestpoint_curve	4576
31.8 closestpoint_degenerate_sf	4577
31.9 closestpoint_surface	4577
31.10cone	4577
31.11const_param_curves	4577
31.12coons_patch_gen	4578
31.13coons_patch_volume_gen	4578
31.14createBlockStructuredDisc	4578
31.15createCoonsVolume	4578
31.16createLinSweptVol	4579
31.17createMidShip	4579
31.18createRotationalVol	4579
31.19createSplitDisc	4579
31.20createVolumeBlocks	4580
31.21createVolumeBoundaries	4580
31.22cylinder	4580
31.23ellipse	4580
31.24face2splineset	4581
31.25interpol_curve_free	4581
31.26interpol_curve_hermite	4581
31.27linear_swept_surface	4581
31.28linear_swept_volume	4582
31.29linearBrepToTrivariate	4582
31.30loft_volume_creator	4582
31.31mirrorAndLoft	4582
31.32multiPatchSweep	4583
31.33project_curve	4583
31.34rotational_swept_surface	4583
31.35rotational_swept_volume	4583
31.36sphere	4584
31.37surface_of_revolution	4584
31.38torus	4584

Chapter 1

GoTools Library

1.1 Introduction

The newest version of GoTools is available from <http://www.sintef.no/Geometry-Toolkits>.

GoTools is licensed under the [GNU General Public License](#)

GoTools is the group name of many interdependent C++ software modules developed by the geometry group at SINTEF ICT, Dept. of Applied Mathematics. GoTools software has been developed for a range of different applications in many different projects. However, a few key modules are used by almost all the others; these have been grouped together in [GoTools Core](#).

At the moment, the following GoTools modules are offered with GPL license:

- [GoTools Core](#). Parametric curves and surfaces including construction methods and operations on these entities
- [GoTools Parametrization](#). Parametrization of scattered data points
- [GoTools Implicitization](#). Approximating spline curves and surfaces by implicitly defined algebraic entities
- [GoTools Intersections](#). Intersection functionality involving spline curves and surfaces and computations of self intersections.
- [GoTools Igeslib](#). Read from and write to iges format.
- [GoTools Trivariate](#). Spline volumes and elementary volumes including some creation methods
- [GoTools Topology](#). Adjacency analysis for surface sets
- [GoTools CompositeModel](#). Representation of a surface set including topological entities and operations on a set of surfaces as one unit.
- [GoTools Trivariatemodel](#). A volume model including topology and some operations on the model
- [GoTools Viewlib](#). A utility viewer to visualize curves and surfaces
- [GoTools QualityModule](#). A set of tools to check the quality of CAD models
- [GoTools IsogeometricModel](#). A set of tools related to isogeometric analysis
- [GoTools LRsplines2D](#). LR spline surfaces

GoTools depends on:

- [SISL](#) (available with GPL license), SINTEF's spline library, for various spline related functionality,
- [TTL](#) (available with GPL license), a generic triangulation library,
- [newmat](#) (available with a permissive license), for various matrix operations.

For convenience, these libraries are included in the GPL version of GoTools.

1.2 Building GoTools

This GoTools package uses CMake to generate a Makefile (on Linux) or MS Visual Studio project file (on Windows).

For information on using CMake, see www.cmake.org.

As a Quick Start Guide, on Linux, make a build directory somewhere:

```
$ cd some_dir
$ mkdir build
$ cd build
$ ccmake <path_to_source_code>
```

Follow the instructions of 'ccmake' - the CMake "GUI". Then:

```
$ make
$ sudo make install
```

On Windows, add a new build folder somewhere. Start the CMake executable and fill in the paths to the source and build folders. When you run CMake, a Visual Studio project solution file will be generated in the build folder.

1.2.1 Compilers

The code uses certain features of the new C++ standard C++11, most notably the smart pointer `std::shared_ptr`. It has been tested on GCC 4.6.1 on Linux and Visual Studio 2010 on Windows.

A set of options to control the build can be accessed in CMake (names starting with `GoTools`). For example, you can turn on/off building the various modules by checking/unchecking `GoTools_COMPILE_MODULE_<modulename>`.

Also provided is the option `GoTools_USE_BOOST`. If this option is turned on, the building process uses `boost::shared_ptr` instead of `std::shared_ptr`. If a C++11 compliant compiler is not available, you may try this option and see if it works. Requires Boost: www.boost.org.

Chapter 2

GoTools CompositeModel

Contains tools for representing composite models.

This module depends on the following GoTools modules:

- GoTools Core
- GoTools Implicitization
- GoTools Intersection
- GoTools Parametrization
- GoTools Igeslib
- GoTools Topology

and the following modules external to GoTools:

- SISL (SINTEF)
- TTL (SINTEF)
- The newmat matrix library, which is freely available from www.robertnz.net.

SISL, TTL and newmat is included for convenience.

Chapter 3

GoTools example files

A number of example programs are created to illustrate the use of various GoTools functionality.

Example programs in gotools-core:

- [adapt_curve](#)
- [append_curve](#)
- [approx_curve](#)
- [approx_surface](#)
- [circle](#)
- [closestpoint_curve](#)
- [closestpoint_degenerate_sf](#)
- [closestpoint_surface](#)
- [cone](#)
- [const_param_curves](#)
- [coons_patch_gen](#)
- [cylinder](#)
- [ellipse](#)
- [interpol_curve_free](#)
- [interpol_curve_hermite](#)
- [linear_swept_surface](#)
- [project_curve](#)
- [rotational_swept_surface](#)
- [sphere](#)
- [surface_of_revolution](#)
- [torus](#)

The module trivariate has the following example programs:

- [coons_patch_volume_gen](#)
- [createCoonsVolume](#)
- [linear_swept_volume](#)
- [loft_volume_creator](#)
- [rotational_swept_volume](#)

The example programs related to `compositemodel`:

- [createSplitDisc](#)
- [createBlockStructuredDisc](#)
- [createVolumeBoundaries](#)
- [face2splineset](#)

The example programs in the `trivariatemodel` module are:

- [createMidShip](#)
- [mirrorAndLoft](#)
- [multiPatchSweep](#)

Chapter 4

GoTools Core

All GoTools modules depend on the `gtools-core` module. This module contains parametric curves and surfaces and functionality related to these entities. For clarity it is divided into four parts:

- *geometry* — contains classes and functionality for representing, storing and manipulating parameterized geometrical objects
- *utils* — contains general, low-level functionality
- *creators* — contains functionality for generating curves and surfaces by approximation, blending, etc.
- *tesselator* — contains functionality for making tessellations of geometry entities

4.1 Geometric Data Structures

The figure above shows the main geometry classes in GoTools.

All geometric objects are of type `GeomObject`. This class has a function `instanceType`, and by calling this function, it is possible to check the concrete type of a given object. A `GeomObject` is `Streamable`, which means that they can be written to and read from a stream (typically a file) in a uniform way. See [the g2-format documentation](#) for more information on this topic. Similarly, they can be created in a uniform way by the `Factory`, which is useful when you want to generate objects whose exact kind is unknown at compile time.

4.1.1 Parametric Curves

`ParamCurve` is the base class for all parametric curves in GoTools and defines a common interface. Many operations do not need to distinguish between different types of parametric curves. A parametric curve has (with some exceptions) the following functionality:

- operations on the curve
 - `clone`: make a copy of this curve
 - `subCurve`: create a subcurve of this curve
 - `appendCurve`: append another curve to this curve
 - `split`: split the curve into two curves at a given parameter
- operations in the parameter interval

- [startparam](#): fetch start parameter
- [endparam](#): fetch end parameter
- [setParameterInterval](#): linearly reparametrize the parameter interval
- [reverseParameterDirection](#): reverse the parameter interval
- evaluations
 - [point](#): evaluate the curve's position (and perhaps a certain number of derivatives) at a given parameter
 - [uniformEvaluator](#): evaluate the curve's position at a regular set of parameter values
- geometric information retrieval
 - [length](#): compute the length of (a segment of) the curve
 - [estimatedCurveLength](#): estimate the length of (a segment of) the curve
 - [compositeBox](#): compute a composite box (bounding box) enclosing the curve
 - [directionCone](#): compute a directional cone containing all tangent directions of this curve
- closest point computations
 - [closestPointGeneric](#): compute the closest point on (a segment of) this curve to a specified point — generic implementation
 - [closestPoint](#): compute the closest point on (a segment of) this curve to a specified point — specific implementation for each parametric curve type
- checks for degeneracy, symmetry, and type
 - [isLinear](#): check if the curve is linear
 - [isInPlane](#): check if the curve is contained in a plane belonging to a given pencil of planes
 - [isAxisRotational](#): check if the curve is axis rotational
 - [isDegenerate](#): check whether the curve degenerates to a single point
 - [isClosed](#): check whether the curve is closed

4.1.2 Parametric Surfaces

[ParamSurface](#) is the base class for a parametric surface and defines the common interface for all parametric surfaces. A parametric surface has (with some exceptions) the following functionality:

- operations on the surface
 - [clone](#): make a copy of this surface
 - [subSurfaces](#): create subsurfaces of this surface
 - [asSplineSurface](#): create a spline surface represented by this surface, if any
 - [outerBoundaryLoop](#): fetch the anticlockwise, outer boundary loop of the surface
 - [allBoundaryLoops](#): fetch the anticlockwise outer boundary loop of the surface, together with clockwise loops of any interior boundaries
 - [getBoundaryInfo](#): compute the boundary curve segment between two points on the boundary, as well as the cross-tangent curve
 - [mirrorSurface](#): reflect the surface through a given plane
 - [getInternalPoint](#): fetch an (unspecified) internal point in the surface
 - [constParamCurves](#): fetch the curve(s) obtained by intersecting the surface with one of its isoparametric curves
 - [nextSegmentVal](#): determine the parameter value of the start of the 'next segment' from a parameter value, along a given parameter direction

- [turnOrientation](#): flip the direction of the normal of the surface
- [getCornerPoints](#): fetch surface corners, geometric and parametric points
- operations in the parameter domain
 - [setParameterDomain](#): set the parameter domain to a given rectangle
 - [parameterDomain](#): fetch the parameter domain
 - [containingDomain](#): fetch a rectangular domain that contains the parameter domain
 - [inDomain](#): check if a parameter pair lies in the parameter domain
 - [inDomain2](#): check if a parameter pair lies inside, on the boundary, or outside of the parameter domain
 - [onBoundary](#): check if a parameter pair lies on the boundary of the parameter domain
 - [closestInDomain](#): fetch the parameter value in the parameter domain closest to a given parameter pair
 - [reverseParameterDirection](#): reverse one of the parameter directions
 - [swapParameterDirection](#): swap the two parameter directions
- geometric information retrieval
 - [normalCone](#): create a direction cone containing all surface normals
 - [tangentCone](#): create a direction cone containing all surface tangents in a given parameter direction
 - [compositeBox](#): compute a composite box (bounding box) enclosing the surface
 - [area](#): compute the total area of this surface
 - [estimateSfSize](#): estimate the size of the surface in the two parameter directions
- evaluations
 - [point](#): evaluate the surface's position (and perhaps a certain number of derivatives) at a given parameter pair
 - [normal](#): evaluate the surface normal for a given parameter pair
 - [evalGrid](#): evaluate the surface's position at a grid in the parameter domain
- closest point computations
 - [closestPoint](#): iteratively find the closest point on the surface to a given point
 - [closestBoundaryPoint](#): iteratively find the closest point on the boundary of the surface to a given point
 - [setIterator](#): set type of closest point iterator
- checks for degeneracy, symmetry, and type
 - [isDegenerate](#): check whether one of the four boundary curves is degenerate (i.e., has zero length)
 - [getDegenerateCorners](#): find degenerate surface corners by checking for parallel and anti-parallel partial derivatives
 - [isIsoTrimmed](#): check if the surface is trimmed along constant parameter curves
 - [isSpline](#): check if the surface is of type spline
 - [isAxisRotational](#): check if the surface is axis rotational
 - [isPlanar](#): check if the surface is planar
 - [isLinear](#): check if the surface is linear in one or both directions
 - [ElementOnBoundary](#): check if a polynomial element (for spline surfaces) intersects the (trimming) boundaries
 - [ElementBoundaryStatus](#): check if a polynomial element (for spline surfaces) intersects the (trimming) boundaries, is inside or outside

4.2 B-spline Curves

A B-spline curve is represented by the [SplineCurve](#) class.

Mathematically, the curve is defined by the parametrization

$$\mathbf{c}(t) = \sum_{i=1}^n \mathbf{p}_i B_{i,k,t}(t).$$

The dimension dim of the curve c is equal to that of its *control points* \mathbf{p}_i . For example, if the dimension of the control points is one, the curve is a function, if the dimension is two, the curve is planar, and if the dimension is three, the curve is spatial. Usually the dimension of the curve will be at most three, but higher dimensions are allowed.

A B-spline curve is a linear combination of a sequence of [B-splines](#) $B_{i,k,t}$ (called a B-basis) uniquely determined by a knot vector \mathbf{t} and the order k . The order is equal to the polynomial degree plus one. For example, if the order is two, the degree is one and the B-splines and the curve c they generate are (piecewise) linear. If the order is three, the degree is two and the B-splines and the curve are quadratic. Cubic B-splines and curves have order 4 and degree 3, etc.

The parameter range of a B-spline curve c is the interval $[t_k, t_{n+1}]$, and so mathematically, the curve is a mapping $c : [t_k, t_{n+1}] \rightarrow \mathbf{R}^{dim}$, where dim is the Euclidean space dimension of its control points.

The complete representation of a B-spline curve consists of

- dim : The dimension of the underlying Euclidean space, 1, 2, 3, ...
- n : The number of control points (also the number of B-splines)
- k : The order (polynomial degree + 1) of the B-splines.
- \mathbf{t} : The knot vector of the B-splines. $\mathbf{t} = (t_1, t_2, \dots, t_{n+k})$.
- \mathbf{p} : The control points of the B-spline curve. $p_{d,i}$, $d = 1, \dots, dim$, $i = 1, \dots, n$. E.g. when $dim = 3$, we have $\mathbf{p} = (x_1, y_1, z_1, x_2, y_2, z_2, \dots, x_n, y_n, z_n)$.

We note that arrays in c start at index 0 which means, for example, that if the array t holds the knot vector, then $t[0] = t_1, \dots, t[n+k-1] = t_{n+k}$ and the parameter interval goes from $t[k-1]$ to $t[n]$. Similar considerations apply to the other arrays.

The data in the curve representation must satisfy certain conditions:

- The knot vector must be non-decreasing: $t_i \leq t_{i+1}$. Moreover, the knots t_i and t_{i+k} must be distinct: $t_i < t_{i+k}$.
- The number of control points should be greater than or equal to the order of the curve: $n \geq k$.

To understand the representation better, we will look at three parts of the representation: the B-splines (the basis functions), the knot vector and the control polygon.

4.2.1 B-spline Basis Functions

The spline space is represented by the class [BsplineBasis](#). A set of B-spline basis functions is determined by the order k and the knots.

4.2.1.1 Linear B-splines

For example, to define a single basis function of degree one, we need three knots.

In the figure the three knots are marked as dots.

In the figure above the two knots at each end are equal.

By taking a linear combination of the three basis functions shown above, we can generate a linear spline function as shown in the next figure.

4.2.1.2 Quadratic B-splines

A quadratic B-spline basis function is a linear combination of two linear basis functions, in which the coefficients are linear affine in the parameter t . In the next figure, we will see a quadratic B-spline defined by four knots. A quadratic B-spline is the sum of two products, the first product between the linear B-spline on the left and a corresponding line from 0 to 1, the second product between the linear B-spline on the right and a corresponding line from 1 to 0.

4.2.1.3 Higher-order B-splines

For higher-order B-splines there is a similar definition. A B-spline of order $k \geq 2$ is a weighted linear combination of two B-splines of order $k - 1$. Explicitly, we define B-splines of order $k = 1$ as box functions,

$$B_{i,1}(t) = 1 \quad \text{if } t_i \leq t < t_{i+1} \quad \text{and } 0 \quad \text{otherwise}$$

and then the B-splines of order $k \geq 2$ as

$$B_{i,k}(t) = \frac{t - t_i}{t_{i+k-1} - t_i} B_{i,k-1}(t) + \frac{t_{i+k} - t}{t_{i+k} - t_{i+1}} B_{i-1,k-1}(t).$$

4.2.1.4 Basic properties

B-spline basis functions satisfy some important properties for curve and surface design. Each basis function is non-negative and it can be shown that they sum to one,

$$\sum_{i=1}^n B_{i,k,t}(t) = 1.$$

These properties combined mean that B-spline curves satisfy the *convex hull property*: the curve lies in the convex hull of its control points. Furthermore, the support of the B-spline basis function $B_{i,k,t}$ is the interval $[t_i, t_{i+k}]$ which means that B-spline curves has *local control*: moving one control point only alters the curve locally.

With k multiple knots at the ends of the knot vector, B-spline curves also have the *endpoint property*: the start point of the B-spline curve equals the first control point and the end point equals the last control point, in other words

$$\mathbf{c}(t_k) = \mathbf{p}_1 \quad \text{and} \quad \mathbf{c}(t_{n+1}) = \mathbf{p}_n.$$

4.2.2 The Control Polygon

The *control polygon* of a B-spline curve is the polygonal arc formed by its control points, $\mathbf{p}_0, \mathbf{p}_1, \dots, \mathbf{p}_n$. This means that the control polygon, regarded as a parametric curve, is itself a piecewise linear B-spline curve (order two), identical to its own control polygon. If we increase the order, the distance between the control polygon and the curve increases. A higher order B-spline curve tends to smooth the control polygon and at the same time mimic its shape. For example, if the control polygon is convex, so is the B-spline curve. Another property of the control polygon is that it will get closer to the curve if it is redefined by inserting knots into the curve and thereby increasing the number of control points. This can be seen in the next figure. In the limit, the refinement of the control polygon converges to its B-spline curve.

4.2.3 The Knot Vector

The knots of a B-spline curve describe the following properties of the curve:

- The parameterization of the B-spline curve
- The continuity at the joins between the adjacent polynomial segments of the B-spline curve.

Although the curves in the figure below have the same control polygon, they have different knot vectors and therefore different parametrizations.

This example is not meant as an encouragement to use parameterization for modelling, rather to make users aware of the effect of parameterization. Something close to curve length parameterization is in most cases preferable. For interpolation, chord-length parameterization is used in most cases.

The number of equal knots determines the degree of continuity. If k consecutive internal knots are equal, the curve is discontinuous. Similarly if $k - 1$ consecutive internal knots are equal, the curve is continuous but not in general differentiable. A continuously differentiable curve with a discontinuity in the second derivative can be modelled using $k - 2$ equal knots etc. Normally, B-spline curves in GoTools are expected to be continuous. For many algorithms, curves should be continuously differentiable (C^1).

4.2.4 NURBS Curves

A NURBS (Non-Uniform Rational B-Spline) curve is a generalization of a B-spline curve,

$$\mathbf{c}(t) = \frac{\sum_{i=1}^n w_i \mathbf{p}_i B_{i,k,\mathbf{t}}(t)}{\sum_{i=1}^n w_i B_{i,k,\mathbf{t}}(t)}.$$

In addition to the data of a B-spline curve, the NURBS curve \mathbf{c} has a sequence of weights w_1, \dots, w_n . One of the advantages of NURBS curves over B-spline curves is that they can be used to represent conic sections exactly (taking the order k to be three). A disadvantage is that NURBS curves depend nonlinearly on their weights, making some calculations, like the evaluation of derivatives, more complicated and less efficient than with B-spline curves.

The representation of a NURBS curve is the same as for a B-spline curve except that it also includes

- \mathbf{w} : A sequence of weights $\mathbf{w} = (w_1, w_2, \dots, w_n)$.

We make the assumption that the weights are strictly positive: $w_i > 0$.

Under this condition, a NURBS curve, like its B-spline cousin, enjoys the convex hull property. With k -fold knots at the ends of the knot vector, also NURBS curves have the endpoint interpolation property.

A NURBS curve is in GoTools stored using the same entities as non-rational spline curves. Note that the constructors of these entities assume that the NURBS coefficients are given in the format: $w_i \mathbf{p}_{i,1}, \dots, w_i \mathbf{p}_{i,dim}, w_i$ for $i = 1, \dots, n$, i.e., the coefficients are multiplied with the weights.

4.2.5 Spline Curve Functionality

In GoTools, [SplineCurve](#) inherits [ParamCurve](#) and has thereby the functionality specified for ParamCurve objects. Functionality specific for a B-spline curve can be found in the doxygen information. This functionality includes:

- Compute the derivative curve corresponding to a given curve
- Fetch information related to the spline space
- Fetch the control polygon of a spline curve
- Insert new knots into the knot vector of the curve and update the curve accordingly
- Increase the polynomial degree of the curve
- Make sure that the curve has got an open knot vector, i.e., knot multiplicity equal to the order in the ends of the curve

4.3 B-spline Surfaces

A tensor product B-spline surface is represented by the class [SplineSurface](#).

Mathematically, the B-spline surface is defined by the parametrization

$$\mathbf{s}(u, v) = \sum_{i=1}^{n_1} \sum_{j=1}^{n_2} \mathbf{p}_{i,j} B_{i,k_1,\mathbf{u}}(u) B_{j,k_2,\mathbf{v}}(v)$$

with control points $\mathbf{p}_{i,j}$ and two variables (or parameters) u and v . The formula shows that a basis function of a B-spline surface is a product of two basis functions of B-spline curves (B-splines). This is why a B-spline surface is called a tensor-product surface. The following is a list of the components of the representation:

- dim : The dimension of the underlying Euclidean space.
- n_1 : The number of control points with respect to the first parameter.
- n_2 : The number of control points with respect to the second parameter.
- k_1 : The order (polynomial degree + 1) of the B-splines in the first parameter.
- k_2 : The order of the B-splines in the second parameter.
- \mathbf{u} : The knot vector of the B-splines with respect to the first parameter, $\mathbf{u} = (u_1, u_2, \dots, u_{n_1+k_1})$.
- \mathbf{v} : The knot vector of the B-splines with respect to the second parameter, $\mathbf{v} = (v_1, v_2, \dots, v_{n_2+k_2})$.
- \mathbf{p} : The control points of the B-spline surface, $c_{d,i,j}$, $d = 1, \dots, dim$, $i = 1, \dots, n_1$, $j = 1, \dots, n_2$. When $dim = 3$, we have $\mathbf{p} = (x_{1,1}, y_{1,1}, z_{1,1}, x_{2,1}, y_{2,1}, z_{2,1}, \dots, x_{n_1,1}, y_{n_1,1}, z_{n_1,1}, \dots, x_{n_1,n_2}, y_{n_1,n_2}, z_{n_1,n_2})$.

The data of the B-spline surface must fulfill the following requirements:

- Both knot vectors must be non-decreasing.
- The number of control points must be greater than or equal to the order with respect to both parameters:
 $n_1 \geq k_1$ and $n_2 \geq k_2$.

The properties of the representation of a B-spline surface are similar to the properties of the representation of a B-spline curve. The control points $\mathbf{p}_{i,j}$ form a *control net* as shown in the figure below. The control net has similar properties to the control polygon of a B-spline curve, described above.

4.3.1 The Basis Functions

A basis function of a B-spline surface is the product of two basis functions corresponding to B-spline curves,

$$B_{i,k_1,\mathbf{u}}(u)B_{j,k_2,\mathbf{v}}(v).$$

Its support is the rectangle $[u_i, u_{i+k_1}] \times [v_j, v_{j+k_2}]$. If the basis functions in both directions are of degree one and all knots have multiplicity one, then the surface basis functions are pyramid-shaped. For higher degrees, the surface basis functions are bell shaped.

4.3.2 NURBS Surfaces

A NURBS (Non-Uniform Rational B-Spline) surface is a generalization of a B-spline surface,

$$s(u, v) = \frac{\sum_{i=1}^{n_1} \sum_{j=1}^{n_2} w_{i,j} \mathbf{P}_{i,j} B_{i,k_1,\mathbf{u}}(u) B_{j,k_2,\mathbf{v}}(v)}{\sum_{i=1}^{n_1} \sum_{j=1}^{n_2} w_{i,j} B_{i,k_1,\mathbf{u}}(u) B_{j,k_2,\mathbf{v}}(v)}.$$

In addition to the data of a B-spline surface, the NURBS surface has a weights $w_{i,j}$. NURBS surfaces can be used to exactly represent several common 'analytic' surfaces such as spheres, cylinders, tori, and cones. A disadvantage is that NURBS surfaces depend nonlinearly on their weights, making some calculations less efficient.

The representation of a NURBS surface is the same as for a B-spline surface except that it also includes

- \mathbf{w} : The weights of the NURBS surface, $w_{i,j}$, $i = 1, \dots, n_1$, $j = 1, \dots, n_2$, so $\mathbf{w} = (w_{1,1}, w_{2,1}, \dots, w_{n_1,1}, \dots, w_{1,2}, \dots, w_{n_1,n_2})$.

The weights are required to be strictly positive: $w_{i,j} > 0$.

The NURBS surface is represented by [SISLSurf](#) and [SplineSurface](#) in SISL and GoTools, respectively. As for the curve case, the constructors of [SISLSurf](#) and [SplineSurface](#) expect the coefficients to be multiplied with the weights.

4.3.3 Spline Surface Functionality

[SplineSurface](#) inherits [ParamSurface](#) in the GoTools data structure and implements the functionality described for [ParamSurface](#). Important additional functionality is listed below:

- Compute the derivative surface and normal surface corresponding to a given surface
- Fetch information related to the spline spaces
- Fetch the control polygon of a spline surface
- Fetch all weights of a NURBS surface
- Grid evaluation of the basis functions related to a surface
- Grid evaluation of the surface
- Insert new knots into the knot vectors of the surface and adapt the surface description accordingly
- Increase the polynomial degrees of the surface
- Fetch information related to the boundary curves of a surface

4.4 Other Curve Types

As shown in the class hierarchy, a spline curve is not the only parametric curve in GoTools although it is the most important one. There are also other curves that share parts of the public interface:

- **BoundedCurve** A bounded curve is a restriction of a parametric curve to a given interval. The interval can be specified either in parameter space, in geometry space or both. In the last case, it must be specified whether the parameter space bound or the geometry space bound is the master.
- **ElementaryCurve** An elementary curve is a container for a conic section or a line. This entity will be described in some detail later.
- **CurveOnSurface** This class represents a curve lying on a surface.

4.4.1 Curve On Surface

The **CurveOnSurface** entity is often related to a bounded, or trimmed, surface. A bounded surface is limited by a curve loop where each curve in the loop most often is represented by a **CurveOnSurface**. However, a **CurveOnSurface** is an entity in its own right. It simply represents a curve lying on a surface. It can for instance originate from intersecting the surface with a plane.

A **CurveOnSurface** consists of a parametric surface and two corresponding curves, one curve in the parameter domain of the surface and one spatial curve. The master representation is specified. Some operations may require the existence of one particular of these curves, in particular the parameter curve is requested. It exists functions that compute the missing curve from the existing one.

4.4.2 Elementary Curve

An **elementary curve** is a fairly new concept in GoTools, and until now such curves have been entered as entities in an IGES or STEP file. The elementary curves may also be represented as spline curves although non-bounded curves must be given a finite extension. The elementary curves is of the types:

- **Circle** The circle is defined by its center and radius and the plane in which it lies if the dimension of the geometry space is larger than 2. A circle is parameterized in terms of the angle. This is a bounded parameterization.
- **Ellipse** An ellipse is represented by a centre, the direction of one semi-axis and the length of the two semi-axis. The plane in which the ellipse lies is required if the dimension of the geometry space is larger than 2. An ellipse is parameterized in terms of the angle which gives a bounded parameterization.
- **Line** A line is given by a point and a direction. It has an unbounded parameterization, $t \in [-\infty, \infty]$.
- **Parabola** A parabola is given by a position, C , a focal distance, F , and a coordinate system, $(\mathbf{x}, \mathbf{y}, \mathbf{z})$. It is described as $f(t) = C + F(t^2\mathbf{x} + 2t\mathbf{y})$ and has an unbounded parameterization, $t \in [-\infty, \infty]$.
- **Hyperbola** A hyperbola is given by a position, a coordinate system and two distances. The parameterization is unbounded.

4.5 Other Surface Types

Similar to the curve case, GoTools supports also other types of parametric surfaces than spline surfaces. The additional surface types are bounded surface, elementary surface and composite surface. The elementary surfaces are conic surfaces, plane and torus. A composite surface consists of a number of spline surfaces where the associated parameter domains are mapped into one composite domain. The class is not complete in the sense that it does not support all functions specified in the interface for a parametric surface.

4.5.1 Bounded Surface

A [bounded surface](#) is a trimmed surface defined by an underlying surface and a trimming loop. The underlying surface is a parametric surface and the loop consists of a closed sequence of ordered parametric curves, often of type `CurveOnSurface`. The trimming loop must lie on the surface and be continuous with respect to a given tolerance.

4.5.2 Elementary Surface

Also [elementary surfaces](#) are quite new in GoTools, and have been entered as entities from IGES or STEP. The elementary surfaces may also be represented as spline surfaces although non-bounded surfaces must be given a finite extension. Operations involving this curve may, however, be made more efficiently by the knowledge of the surface type. The elementary surfaces is of the types:

- [Cone](#) A cone is described by a location, the cone axis, the radius of the cross section corresponding to the location and the cone angle. The parameterization is given by the angle and the distance along the axis, thus it is unbounded in one parameter direction.
- [Cylinder](#) A cylinder is given by its centre and radius and the cylinder axis. It is parameterized by the angle around the circle and the distance in the axis direction.
- [Plane](#) A plane is given by a point and a normal. It has an unbounded parameterization in both parameter directions.
- [Sphere](#) A sphere is given by its centre and radius. It has an angular parameterization in both parameter directions and thereby bounded.
- [Torus](#) A torus is given by the minor and major radius, a location and an axis. The parameterization is angular in both parameter directions.
- [Disc](#) A disc is a part of a plane limited by a circle and given by its centre, radius, plane normal and an additional vector to generate a coordinate system.

The elementary surfaces described above is placed in a coordinate system to facilitate the parameterization. In addition to the geometric information given for each entity, remaining coordinate system information must be specified.

4.6 Geometry Construction

The combination of SISL and GoTools provide a lot of methods for geometry construction. They partly overlap. The GoTools functionality is outlined here. More details can be found in the appropriate classes and namespaces. The SISL geometry construction methods are described in the SISL manual.

The classes for parametric curves, surfaces and volumes in GoTools do not contain construction facilities. They are solely for operating on these entities. The construction is performed in separate classes.

4.6.1 Curve Construction

Construction of elementary curves are being done by their definition or they are read from an IGES file. The curve construction methods in GoTools are concerned with spline curves. The methods are split between two sub modules in gotools-core, namely geometry and creators. The relevant classes are

- [HermitInterpolator](#) The class is placed in geometry. Interpolates a set of points where each point is associated a tangent vector. The points must be parameterized. This is a local curve construction method.
- [SplineInterpolator](#) A set of methods to interpolate a set of points. Some methods accept tangent vectors associated to some points. The points must be parameterized. The class gives the possibility to set particular conditions in the endpoints of the curve. This is a global method for curve construction.
- [SplineApproximator](#) Approximation in the least squares sense. Parameterized points and a spline space must be given. This class and the two proceeding ones inherit the same class, link [Go::Interpolator](#) Interpolator, and share parts of the interface.
- [ApproxCurve](#) Make a curve approximating a set of points with a given accuracy. The points must be parameterized and it is possible to give an initial spline space. The method iteratively applies the method in the class [SmoothCurve](#). This class and the following ones lie in the sub module creators.
- [SmoothCurve](#) Approximate a set of parameterized data points with a spline curve using a least squares approximation with a smoothing term. The spline space must be given. The method can also be used to perform smoothing on a given curve.
- [AdaptCurve](#) Approximates an evaluator based curve with a given accuracy. The method is based on least squares approximation with a smoothing term and spline space refinement.
- [ApproxCrvToSeqs](#) Approximate a set of curve sequences by a set of B-spline curves within a given accuracy. The curves are represented in the same spline space.
- [HermiteAppC](#) Approximate an evaluator based curve represented by a sub class of the class [EvalCurve](#), by a spline curve. An hermite method is used, thus a C^1 continuous curve will be generated. The existing evaluator based curves are
 1. [A curve in the parameter domain](#) of a surface is represented as the projection of a 3D curve into the parameter domain of a given surface. The output curve lies in the parameter domain of the surface.
 2. Given data describing a [surface-surface intersection curve](#), the curve lying given offset distances from the two surface involved in the intersection and parameterized similar to the intersection curve is represented. This construction can be used to create a rolling ball blend.
 3. Given two approximations of the same intersection curves, reapproximate the [intersection curve](#).
 4. Given a [curve in the parameter domain](#) of a surface, the geometry space curve is evaluated.
 5. Represent [an offset curve](#) from a given space curve. The direction of the offset is given as an expression of corresponding curves.
 6. An [offset curve](#) to a given curve in the direction of a cross tangent curve constructed as a blend between tangent curves.
 7. A [parameter curve](#) of any type that is to be approximated by a spline curve.
- [HermiteAppS](#) Approximate an evaluator based curve set represented by a sub class of the class [EvalCurveSet](#). A number of curves defined in the same spline space are generated. One concrete evaluator based curve set exists currently:
 1. Given a surface, a curve in geometry space and a corresponding cross tangent curve, [the parameter curve](#) representing the projection of the geometry curve into the surface and the corresponding projection of the cross tangent in the parameter domain is represented.
 2. Given an [intersection curve](#) represented as two [CurveOnSurface](#) instances, reapproximate this intersection curve along with the two parameter domain curves representing this curve.
 3. Improve the accuracy of an existing [trimming curve](#).
 4. Given an intersection curve between two surfaces, offset these surfaces given distances and create the [intersections curve](#) between the offset surfaces and corresponding cross tangent curves that will give a smooth transition between the offset surfaces.

4.6.2 Surface Construction

Similar to the curve construction methods, these methods are placed in the sub modules geometry and creators.

- [SweepSurfaceCreator](#) The class lies in geometry. The class contains two methods; sweep a curve along another curve to create a surface and rotational sweep.
- [ApproxSurf](#) This class and the following lie in creators. A set of parameterized scattered data are approximated by a spline surface. The boundary curves to the surface can be defined a priori. The spline space must be given. The method iteratively applies the method in [SmoothSurface](#) and performs parameter iteration. It may also refine the spline space.
- [SmoothSurf](#) Approximate a set of parameterized scattered data points using a least squares approach with a smoothing term. The spline space must be given. The method can also be used to smooth a given surface.
- [LoftSurfaceCreator](#) constructs a surface by interpolating a number of non-rational spline curves.
- [CoonsPatchGen](#) Construct a surface interpolating 4 boundary curves. The curves may be attached cross tangent curves. To achieve interpolation, the boundary conditions must be consistent.
- [HahnsSurfaceGen](#) Fill an n -sided hole with n surfaces. $n = 3, 5$ or 6 .

4.7 The utils module

The 'utility' module was developed in parallel with the 'geometry' module, and contains various useful classes and functions related to math. / computational geometry, while not being directly related to splines.

4.7.1 Data structures

You can find templated arrays and optimised vectors.

4.7.2 Geometrical objects

The module contain some objects that are much used in computational geometry, like

- [points](#)
- [bounding boxes](#) , axis-aligned or [rotated](#) , eventually [composite](#)
- [rational](#) numbers
- [barycentric](#) coordinate systems
- [direction cones](#)

4.7.3 Misc.

- Compile-time computation of [factorials](#)
- A general, nonlinear [function minimizer](#) in an arbitrary number of variables. This can be useful in a great number of different settings!

4.8 Tessellation

The geometry entities are tessellated in `gotools-core` in the sub module `tessellator`. Depending on the type of the geometry object, the result of the tessellation is stored in [GenericTriMesh](#), [LineStrip](#), [QuadMesh](#) or [RegularMesh](#) which all inherit the abstract class [GeneralMesh](#). The tessellators in this module relate to a given resolution in each parameter direction of the geometry object.

A curve is tessellated by the class [CurveTessellator](#) and the tessellated curve is returned by the function `CurveTessellator::getMesh()` as a shared pointer to a `LineStrip` object. A rectangular surface is tessellated in [RectGridTessellator](#) and the tessellator returns a shared pointer to a `RegularMesh`. The triangles produced during tessellation is organized in a set of triangle strips. This structure is taken advantage of to improve the drawing efficiency. A trimmed surface is tessellated by [ParametricSurfaceTessellator](#) and the tessellation is represented as an unorganized bunch of triangles. The output from the functionality in the tessellation module is taken as input to the `GoTools` viewers and the actual drawing is performed using `openGL`.

Chapter 5

The g2-format, GoTools file format for geometry entities

The following geometry entities have read and write functionality using the g2-format:

- *SplineCurve*, entity code = 100,
- *CurveOnSurface*, entity code = 110,
- *Line*, entity code = 120,
- *Circle*, entity code = 130,
- *Ellipse*, entity code = 140,
- *BoundedCurve*, entity code = 150,
- *Hyperbola*, entity code = 160,
- *Parabola*, entity code = 170,
- *SplineSurface*, entity code = 200,
- *BoundedSurface*, entity code = 210,
- *SurfaceOnVolume*, entity code = 211,
- *GoBaryPolSurface*, entity code = 220,
- *GoHBSplineParamSurface*, entity code = 230,
- *CompositeSurface*, entity code = 240,
- *Plane*, entity code = 250,
- *Cylinder*, entity code = 260,
- *Sphere*, entity code = 270,
- *Cone*, entity code = 280,
- *Torus*, entity code = 290,
- *SurfaceOfRevolution*, entity code = 291,
- *Disc*, entity code = 292,
- *LRsplineSurface*, entity code = 293,

- *TSplineSurface*, entity code = 294,
- *Go3dsObject*, entity code = 300,
- *GoHeTriang*, entity code = 310,
- *GoSdTriang*, entity code = 320,
- *GoQuadMesh*, entity code = 330,
- *GoHybridMesh*, entity code = 340,
- *ParamTriang*, entity code = 350,
- *GoVrmlGeometry*, entity code = 360,
- *PointCloud*, entity code = 400,
- *LineCloud*, entity code = 410,
- *GoTriangleSets*, entity code = 500,
- *RectGrid*, entity code = 510,
- *SplineVolume*, entity code = 700,
- *Parallelepiped*, entity code = 720,
- *SphereVolume*, entity code = 721,
- *CylinderVolume*, entity code = 722,
- *ConeVolume*, entity code = 723,
- *TorusVolume*, entity code = 724

In the following the file format corresponding to most of the entities listed above will be described. Note that the g2-file format is a simple format that does not support pointers. Thus, duplication of information may occur.

5.1 The object header

All geometry entities are preceded by the object [ObjectHeader](#) in a stream. `ObjectHeader` contains information about the class type of the geometry object to follow and the version number of the file format. The format of the header is as follows:

- `class type` . The entity code given in the list above.
- `major version` . Always 1.
- `minor version` . Always 0.
- `auxillary data` . 0 if the default colour is chosen, 4 if the rgb-colour code is chosen. In the latter case, the entity is followed by 4 integer digits between 0 and 255 giving the colours red, green and blue and the xxx.

5.2 SplineCurve

The entity code 100 is given in the header. [SplineCurve](#) continues with the following information

- dimension of geometry space, whether or not the curve is rational: 1=rational, 0=non-rational
- the number of coefficients, the polynomial order (i.e. degree+1)
- the knot vector, multiple knots are represented by giving the knot value several times
- the curve coefficients

The curve coefficients are given continuously, normally divided by a carriage return for each coefficient. The sequence is, in the non-rational case, (x_i, y_i, z_i) , $i = 1, \dots, n$ where n is the number of coefficients. In the rational case, the coefficients are given as $(x_i h_i, y_i h_i, z_i h_i, h_i)$, $i = 1, \dots, n$. h_i is the weight associated the current coefficient. Note that the coefficients are represented with the weight multiplied with the x , y and z -value.

A linear non-rational spline curve with no inner knots in the parameter interval $[0,1]$ is including the header given as:

```
100 1 0 0
3 0
2 2
0 0 1 1
0 0 0
1 0 0
```

5.3 CurveOnSurface

The entity enumeration of a [CurveOnSurface](#) is 110. The remaining file format is as follows:

- whether or not the parameter space curve is the master information (1=parameter curve is master, 0=space curve is master), the entity type of the parameter curve (0 if no such curve exists), the entity type of the space curve (0 if no such curve exists)
- the description of the parameter curve according to the rules for that curve type
- the description of the space curve according to the rules for that curve type

Either the parameter curve, the space curve or both are given. Information regarding constant parameter information for the curve on surface is not covered by the file format.

Note that the CurveOnSurface entity does also contain information about a [ParamSurface](#). Thus, the g2-representation of a CurveOnSurface is not stand alone. It needs to be combined with a [BoundedSurface](#). That section gives an example of how the CurveOnSurface entity is used in this context.

5.4 Line

The [line](#) is given by the following information:

- the dimension of the geometry space
- a point on the line
- the direction of the line
- a flag, 0 or 1, for unbounded or bounded
- if the line is bounded, the start and end parameters

The entity enumeration for a line is 120.

5.5 Circle

The [circle](#) is given by:

- the dimension of the geometry space
- the radius of the circle
- the centre of the circle
- the normal of the plane in which the circle lies
- the vector from the centre of the circle towards the "x-axis" - the start point of the default parametrization of the circle
- the start and end parameters

The circle has a default implicit parametrization from 0 to 2π .

The entity enumeration for a circle is 130.

Example:

```
130 1 0 0
dim
rad
c_x c_y c_z
n_x n_y n_z
v_x v_y v_z
```

5.6 Ellipse

The [ellipse](#) is given by:

- the dimension of the geometry space
- the major radius of the ellipse
- the minor radius of the ellipse
- the centre of the ellipse
- the normal of the plane in which the ellipse lies
- the vector from the centre of the ellipse towards the "x-axis" - the start point of the default parametrization

The entity enumeration for an ellipse is 130.

5.7 Bounded Curve

A [bounded curve](#) are expected to be bounded both parametrically and geometrically. Thus, a parameter on whether or not the limitation in the parameter space or in geometry space is preferred is included in the format. The format is:

- Whether the limitation in the parameter space is the master (1 if true)
- The start parameter of the bounded curve, the end parameter of the bounded curve
- The start point of the bounded curve in geometry space
- The end point of the bounded curve in geometry space
- The underlying curve described according to its entity type

All the information listed above is expected to be given in a g2-file. The corresponding entity enumeration is 150.

A bounded line from the point (0,0,0) to (1,0,0) where the position in space is the master regarding the bounding points is expressed as

```
150 1 0 0
120 0
0 1
3
0 0 0
1 0 0

3
0 0 0
1 0 0
```

120 is the entity number for a line and the dimension of the geometry space is 3.

5.8 SplineSurface

The entity enumeration for a [spline surface](#) is 200. The body of a spline surface is as follows:

- dimension of geometry space, whether or not the surface is rational: 1=rational, 0=non-rational
- the number of coefficients in the first parameter direction, the polynomial order in this direction (i.e. degree+1)
- the knot vector in the first parameter direction, multiple knots are represented by giving the knot value several times
- the number of coefficients in the second parameter direction, the polynomial order in this direction (i.e. degree+1)
- the knot vector in the second parameter direction
- the surface coefficients

The surface coefficients are given continuously and the 1. parameter direction runs fastest. The sequence is $(x_{(1,1)}, y_{(1,1)}, z_{(1,1)}), (x_{(2,1)}, y_{(2,1)}, z_{(2,1)}), \dots, (x_{(n,1)}, y_{(n,1)}, z_{(n,1)}), (x_{(1,2)}, y_{(1,2)}, z_{(1,2)}), \dots, (x_{(n,2)}, y_{(n,2)}, z_{(n,2)}), \dots, (x_{(1,m)})$. Here n is the number of coefficients in the 1. parameter direction, and m is the number of coefficients in the 2. parameter direction. The weights are multiplied with the coefficients similarly to the case for the spline curve and is given as an extra entity for each coefficient, again similar to the curve case.

A simple non-rational, bilinear surface with no inner knots is represented as:

```

200 1 0 0
3 0
2 2
0 0 1 1
2 2
0 0 1 1
0 0 0
1 0 0
0 1 0
1 1 0

```

5.9 BoundedSurface

[BoundedSurface](#) has entity enumeration 210. A bounded surface consists of an underlying rectangular parametric surface and one or more trimming loops. The file format body consists of the following information:

- The underlying surface represented according to its type
- The number of trimming loops
- For each trimming loop
 1. The number of curves in the current trimming loop
 2. The tolerance within which the loop is found to be continuous
 3. The curves in the trimming loop, one by one

The trimming curves are of type [ParamCurve](#), but in most cases they will be represented as [CurveOnSurface](#).

The following example illustrates the format a bounded surface. The underlying surface is a plane as can be seen from the entity number in line two. The 3 curves in the trimming loop are all of type [CurveOnSurface](#). Note that the entity number of [CurveOnSurface](#) does not appear. The space curve corresponding to the curve on surface is the master, and the only curve describing the trimming curve. This is given by the line `0 0 100` which occur for every curve and is to be interpreted: *the space curve is the master, the parameter curve does not exist, the space curve is a spline curve*. Following this line is the body of a spline curve in the g2-format.

```

210 1 0 0
250
3
0 0 0
0 0 1
1 0 0
1
-3 3
-3 3
0

1
3 0.0001

0 0 100
3 0
2 2
0 0 1 1
-1 0 0
0 -1 0

0 0 100
3 0
2 2
0 0 1 1
0 -1 0
-0.5 2 0

0 0 100

```



```

3 0
4 3
0 0 0 0.5 1 1 1
-0.5 2 0
-1 2 0
-2 1 0
-1 0 0

```

In this case the two first trimming curves are linear spline curves while the last one is of order 3 (degree 2) and has one inner knot. All the curves are non-rational.

5.10 Plane

A [Plane](#) has entity enumeration 250 and the body related to the g2-formation contains:

- The dimension of the geometry space
- A point in the plane
- The normal of the plane
- One vector in the set of vectors spanning the plane
- A flag, 0 or 1, for unbounded or bounded
- If the plane is bounded, the four bounds: *umin*, *umax*, *vmin*, *vmax*
- A flag, 0 or 1, indicating whether or not the *u* and *v* parameter directions are swapped.

The plane used as an underlying surface in the previous example has the format:

```

250 1 0 0
3
0 0 0
0 0 1
1 0 0
1
-3 3
-3 3
0

```

5.11 Cylinder

A [Cylinder](#) has entity enumeration 260 and the body related to the g2-formation contains:

- The dimension of the geometry space
- The cylinder radius
- A point on the cylinder axis
- The cylinder axis
- The vector from the cylinder axis towards the cylinder surface giving the start point of the default parametrization
- A flag, 0 or 1, for unbounded or bounded in the linear direction
- If unbounded, the bounding parameters: *umin*, *umax*
- If bounded, the bounding parameters: *umin*, *umax*, *vmin*, *vmax*
- A flag, 0 or 1, indicating whether or not the *u* and *v* parameter directions are swapped

5.12 Sphere

A [Sphere](#) has entity enumeration 270 and the body related to the g2-formation contains:

- The dimension of the geometry space
- The radius of the sphere
- The centre of the sphere
- The axis on which the degeneracies of a NURBS representation of the sphere will lie, i.e. the sphere axis
- The vector from the sphere axis towards the sphere surface giving the start point of the default parametrization
- The bounding parameters: *umin*, *umax*, *vmin*, *vmax*
- A flag, 0 or 1, indicating whether or not the *u* and *v* parameter directions are swapped

A sphere with centre in (1.5, 0, 0) and radius 1.5 represented in the g2-format is

```
270 1 0 0
3
1.5
1.5 0 0
0 0 1
1 0 0
0 6.283185307179586
-1.5707963267948966 1.5707963267948966
0
```

5.13 Cone

A [Cone](#) has entity enumeration 280 and the body related to the g2-formation contains:

- The dimension of the geometry space
- The cone radius at the position of the point on the cone axis
- A point on the cone axis
- The cone axis
- The vector from the cone axis towards the cone surface giving the start point of the default parametrization
- The opening angle of the cone
- A flag, 0 or 1, for unbounded or bounded in the linear direction
- If unbounded, the bounding parameters: *umin*, *umax*
- If bounded, the bounding parameters: *umin*, *umax*, *vmin*, *vmax*
- A flag, 0 or 1, indicating whether or not the *u* and *v* parameter directions are swapped

5.14 Torus

A [Torus](#) has entity enumeration 290 and the body related to the g2-formation contains:

- The dimension of the geometry space
- The major radius of the torus
- The minor radius of the torus
- The torus centre
- The mid axis of the torus
- The vector from the torus centre to the torus surface giving the startpoint of the default parametrization
- Select outer (1=yes, 0=no): Defines the parameter domain of the torus in degenerate cases.
- A flag, 0 or 1, for unbounded or bounded in the linear direction
- If unbounded, the bounding parameters: $umin, umax$
- If bounded, the bounding parameters: $umin, umax, vmin, vmax$
- A flag, 0 or 1, indicating whether or not the u and v parameter directions are swapped

5.15 SurfaceOfRevolution

A [SurfaceOfRevolution](#) has entity enumeration 291 and the body contains:

- The dimension of the geometry space
- A point of the axis of revolution
- The direction of the axis of revolution
- The *body* of the *spline curve* to rotate around this axis

A surface of revolution is

```

291 1 0 0
3
0 0 0
0 0 1

3 0
2 2
0 0 1 1
1 0 0
3 0 0

```

5.16 Disc

The entity number for [Disc](#) is 292. The g2 body contains:

- The dimension of the geometry space
- The centre of the disc
- The disc radius
- The normal to the disc surface
- A vector from the disc centre to the disc boundary to give a start point for the parametrization
- Whether a NURBS representation should have a degeneracy in the centre (=1) or have degenerate corners (=0)
- The angles giving the four degeneracy points at the boundary. Not used if the flag for centre degeneracy is false.

5.17 PointCloud

A [PointCloud](#) has enumeration 400. It has the following body:

- Number of points
- The coordinates of each point. The points are given one by one.

The dimension of a point in a point cloud is 3 by default. An example of a point cloud with 3 points is

```
400 1 0 0
3
0 0 0
1 0 0
1 1 0
```

5.18 LineCloud

A [LineCloud](#) has enumeration 410. It has the following body:

- Number of line segments
- The coordinates of the endpoints of each line segments. The lines are given one by one.

The dimension of a line in a line cloud is 3 by default. A line cloud with 3 lines follow:

```
410 1 0 0
3
0 0 0 0 0 1
1 0 0 1 0 1
1 1 0 1 1 1
```

5.19 SplineVolume

The entity enumeration for a [spline volume](#) is 700. The spline volume entity is placed in the module trivariate. The body of a spline volume is as follows:

- dimension of geometry space, whether or not the volume is rational: 1=rational, 0=non-rational
- the number of coefficients in the first parameter direction, the polynomial order in this direction (i.e. degree+1)
- the knot vector in the first parameter direction, multiple knots are represented by giving the knot value several times
- the number of coefficients in the second parameter direction, the polynomial order in this direction (i.e. degree+1)
- the knot vector in the second parameter direction
- the number of coefficients in the third parameter direction, the polynomial order in this direction (i.e. degree+1)
- the knot vector in the third parameter direction
- the volume coefficients

The volume coefficients are given continuously and the 1. parameter direction runs fastest, the 2. parameter direction is next and the 3. parameter direction is the most slow. For a non-rational spline volume in the 3-dimensional space, the sequence is $(x_{(1,1,1)}, y_{(1,1,1)}, z_{(1,1,1)}), (x_{(2,1,1)}, y_{(2,1,1)}, z_{(2,1,1)}), \dots, (x_{(n,1,1)}, y_{(n,1,1)}, z_{(n,1,1)}), (x_{(1,2,1)}, y_{(1,2,1)}, z_{(1,2,1)})$. Here n is the number of coefficients in the 1. parameter direction, m is the number of coefficients in the 2. parameter direction and p is the number of coefficients in the 3. parameter direction. The weights are multiplied with the coefficients similarly to the case for the spline curve and the spline surface and is given as an extra entity for each coefficient, again similar to the curve and surface case.

A simple non-rational, trilinear volume with no inner knots is represented as:

```
700 1 0 0
3 0
2 2
0 0 1 1
2 2
0 0 1 1
2 2
0 0 1 1
0 0 0
1 0 0
0 1 0
1 1 1
0 0 1
1 0 1
0 1 1
1 1 1
```

5.20 Parallelepiped

[Parallelepiped](#) has enumeration 720 and the following g2-format:

- Dimension of geometry space
- Lower right corner
- Unit vector in geometry space representing the first parameter direction
- Length of parallelepiped in the first direction
- Unit vector in geometry space representing the second parameter direction
- Length of parallelepiped in the second direction
- Unit vector in geometry space representing the third parameter direction
- Length of parallelepiped in the third direction

5.21 SphereVolume

[SphereVolume](#) has enumeration 721 and the following g2-format:

- Dimension of geometry space
- Sphere radius
- Centre of sphere
- Axis of sphere, i.e. the vector along which the degeneracies of a NURBS representation of the sphere will lie
- Vector from the centre to the outer surface giving input to the sphere parametrization

5.22 CylinderVolume

[CylinderVolume](#) has enumeration 722 and the following g2-format:

- Dimension of geometry space
- Cylinder centre
- Cylinder axis
- Vector from the centre to the outer surface giving input to the parametrization of the cylinder
- Minimum cylinder radius. If this number is larger than 0, the cylinder will have a hole along the axis
- Maximum radius, the radius of the outer cylinder surface
- Minimum height, =1: the cylinder is infinite, =0: the cylinder is finite and the distance from the centre to the bottom of the cylinder follows (positive or negative number)
- Maximum height, =1: the cylinder is infinite, =0: the cylinder is finite and the total height of the cylinder follows
- Whether a NURBS representation should have a degeneracy in the centre (=1) or have degenerate corners (=0)
- The angles giving the four degeneracy points at the boundary. Not used if the flag for centre degeneracy is false.

5.23 ConeVolume

[ConeVolume](#) has enumeration 723 and the following g2-format:

- Dimension of geometry space
- Radius of the cone at the centre
- A point at the cone axis (the centre)
- Cone axis
- The vector from the cone axis towards the boundary surface of the cone giving the start point of the cone parametrization
- The opening angle of the cone
- Minimum height the distance from the centre to the smallest end disc of the cone (positive or negative number)
- Maximum height, =1: the cone is infinite, =0: the cone is finite and the cone height follows
- Whether a NURBS representation should have a degeneracy in the centre (=1) or have degenerate corners (=0)
- The angles giving the four degeneracy points at the boundary. Not used if the flag for centre degeneracy is false.

5.24 TorusVolume

[TorusVolume](#) has enumeration 724 and the following g2-format:

- Dimension of geometry space
- The torus centre
- The normal to the plane through the big circle of the torus
- A vector from the torus centre to the torus surface giving the parametrization
- The major radius
- The minor radius
- Angle of revolution for the small circle
- Angle of revolution for the large circle
- Whether a NURBS representation should have a degeneracy in the centre (=1) or have degenerate corners (=0)
- The angles giving the four degeneracy points at the boundary. Not used if the flag for centre degeneracy is false.

Chapter 6

GoTools Igeslib

The [IGES converter](#) read an IGES file and represents its entites in the internal data structure of GoTools. It can also write a model represented in GoTools to an IGES file or convert between an IGES file and the [internal file format](#) of GoTools.

GoTools represent only geometric entities. Thus, IGES entities like annotation, structure, property, associativity, view, drawing and figure will be neglected. Neither are constructive solid geometry or finite element modelling entites handled. If such entities exist in a file read by the IGES converter, warning messages will be issued.

The topological entities specified in IGES 5.3 is not handled by the current version of the IGES converter. Thus, the entities vertex, edge, edge list, loop, face and shell is not handled. However, the geometric entities corresponding to these topological entities will be read. Colour information is read.

The content of an IGES file is transferred to the application as a vector of [GeomObjects](#). By checking the type of each object and acting thereafter, the model can be stored and handled in the GoTools environment.

To write an IGES file, the file entities are added one by one to the IGES converter using the function `addGeom` which takes a `GeomObject` as parameter. The actual file is written by the command `writeIGES`.

Chapter 7

GoTools Implicitization

Contains tools for implicitizing spline curves and surfaces. Also contains various classes for Bernstein polynomials.

Historically, the underlying ideas for implicitization and approximate implicitization were developed by Tor Dokken at SINTEF, ICT, in his Ph.D. thesis: [Aspects of Intersection Algorithms and Approximation](#), University of Oslo, Norway, 1997.

The implementations in this library was done in connection with the EU projects [GAIA I and II](#).

This module depends on the newmat matrix library, which is freely available from www.robertnz.net.

Chapter 8

GoTools Intersections

Contains tools for finding intersections of spline curves and surfaces.

This module depends on the newmat matrix library, which is freely available from www.robertnz.net.

Chapter 9

GoTools IsogeometricModel

Contains tools related to isogeometric analysis.

Chapter 10

GoTools LRsplines2D

This module contains classes and tools for working with LR spline surfaces.

Chapter 11

GoTools Parametrization

11.1 Purpose

The purpose of the GoTools Parametrization Library is to compute a reasonable parametrization for a discrete geometrical object. This geometrical object can be a planar graph embedded in R^2 or R^3 (triangulation, rectangular grid, etc.) or a point cloud without any explicit connectivity, but with an identified border.

There are many contexts for which such a parametrization is useful, for example when mapping textures onto polygonal meshes or when triangulating a surface from a set of scattered 3D points. The algorithms contained in this software are based on the work of Dr. Michael Floater.

11.2 How it can be used

This library has three degrees of freedom for specifying the parametrization:

- choice of *data structure*
- choice of *boundary parametrization*
- choice of *parametrization of the interior*

11.2.1 Choice of data structure

As mentioned in the last section, the data to be parametrized can be of various kinds, from general polygonal meshes (3D points and connectivity) to point clouds (no connectivity information, but boundary nodes must be identified and ordered). The various possible data types are given as sub-classes of [PrOrganizedPoints](#). You should choose the one that corresponds to the data that you want to parametrize. Many of these classes have read/write functionality for a generic format; you might want to reimplement these to correspond with the format of the data you are using. The list of currently available data structures are:

- [PrUnorganized_OP](#)
- [PrFastUnorganized_OP](#)
- [PrTriangulation_OP](#)
- [PrRectangularGrid_OP](#)
- [PrPlanarGraph_OP](#)

You can of course write a totally *new* specialization of the [PrOrganizedPoints](#) class, as long as you implement all the required member functions.

11.2.2 Choice of boundary parametrization

The boundary of the surface must be parametrized before the surface interior. This is done by the class [PrParametrizeBdy](#). There are several ways of parametrizing the boundary. You can specify the method you want to use through the [PrParametrizeBdy::setParamKind\(\)](#) member function. There are currently three options:

- `PrCHORDLENGTHBDY` - parametrize by chordlength (generally preferred)
- `PrCENTRIPETAL` - parametrize using square root of boundary point distances
- `PrUNIFBDY` - parametrize by imposing a fixed distance between each boundary point

Another thing that must be specified when determining the parametrization of the boundary is the *shape* of the parametrical domain. It can be a circle (default), or a rectangle (corners must be specified).

11.2.3 Choice of interior parametrization

The parametrization of the interior can also be done in several ways, which are each implemented as a sub-class of [PrParametrizeInt](#). Currently available methods are: (click on the links for individual explanations)

- [PrPrmMeanValue](#) (generally preferred)
- [PrPrmEDDHLS](#)
- [PrPrmLeastSquare](#)
- [PrPrmShpPres](#)
- [PrPrmUniform](#)

11.2.4 Putting it all together

The general procedure for parametrizing a data set can be summarized as follows:

- Prepare the data structure
 - establish an object of the appropriate datastructure (sub-class of [PrOrganizedPoints](#)). This object should be owned by a shared pointer (as found in the `boost` library).
 - read your data into the structure, either by using one of the provided member functions like [PrTriangulation_OP::scanRawData\(\)](#), [PrTriangulation_OP::scan\(\)](#), or by adding your own I/O-routines.
- Parametrize the boundary
 - Establish a [PrParametrizeBdy](#) object.
 - Associate it with your data by using the [PrParametrizeBdy::attach\(\)](#) function.
 - Choose parametrization type by calling the [PrParametrizeBdy::setParamKind\(\)](#) function.
 - Execute the parametrization by calling [PrParametrizeBdy\(\)](#). When calling this function without arguments, the parametrical domain will be a circle. A rectangular parameter domain can also be used. This is specified by additional arguments to the function.

- Parametrize the interior
 - When the boundary has been successfully parametrized, the interior can be treated. First, you need to instantiate an object of the desired sub-class of `PrParametrizeInt`.
 - Associate your data to this object by using the `PrParametrizeInt::attach()` function.
 - Carry out the parametrization by calling the `PrParametrizeInt::parametrize()` function.
- Accessing the result
 - The parametrization can be accessed in several ways, through the `PrOrganizedPoints` interface of your data structure:
 - * Several print routines can be used to dump all nodes with/without coordinates and associated parameters (`PrOrganizedPoints::printUVNodes`, `PrOrganizedPoints::printUVXYZNodes`, etc.).
 - * The parameters of an individual node can be accessed by the `PrOrganizedPoints::getU()` and `PrOrganizedPoints::getV()` functions.
 - Most of the sub-classes of `PrOrganizedPoints` also have `print()` and `scan()` functions that can be used for reading and writing their data to a stream.
 - You can write your own customised data access and streaming functions.

11.3 Example program

The procedure mentioned above can be observed in a working program located in the `examples` folder. The source file name is `demo.C`. The program is run from the command line, where you can also specify the data. It can handle both point clouds and triangulations. A sample triangulation is provided in the `data/` folder.

For instance, try

```
$ ./demo -i data/gjoevik_triang
```

(There are also more options you can specify; to see the list of options, run the program without any arguments).

If the input data is a triangulation (the case above), the resulting, parametrized triangulation will be saved in the file `triangulation`. In any case, the parametrization for each node/point will be dumped to the file `uv_nodes`.

Good luck!

Chapter 12

GoTools QualityModule

This GoTools module consists of a set of tools to check the quality of CAD models.

12.1 Tolerances

The tests make use of the tolerances associated with a surface model, i.e.

gap CAD models are seen as continuous if they are continuous within this tolerance. If the distance between two points are less than this tolerance, they are viewed as identical. A reasonable tolerance is in the specter $[10^{-6}, 10^{-2}]$, but it depends on the expected quality of a model.

neighbour This tolerance is used in adjacency analysis and it represents the maximum distance between neighbouring surfaces or curves. If two curves or surfaces lie more distant than this tolerance, the entities are found not to be adjacent. The neighbour tolerance should be larger than the gap tolerance. If nothing specific is known, a factor of 10 makes sense, but if the gap tolerance is really small, a larger factor should be used. Surfaces that lie closer to each other than the neighbouring tolerance is found to be adjacent, but if the distance between the surfaces somewhere is larger than the gap tolerance, the surface set contains gaps. This is an error in the model.

bend If two surfaces meet along a common boundary and corresponding surface normals form an angle which is larger than this tolerance, it is assumed that there is an intended sharp edge between the surfaces. Similarly, if two curves in a composite curve meet with an angle larger than this tolerance, there is an intentional corner.

kink If two adjacent curves or surfaces meet with an angle less than this tolerance, they are seen as G^1 continuous. If the angle is larger than this tolerance, but less than the bend tolerance, the intended G^1 continuity is broken and it is an error in the model. The tolerance depends on the continuity requirements of the application. One suggestion is 10^{-2} . The bend tolerance must be larger than the kink tolerance, for instance by a factor of 10. Both angular tolerances are given in radians.

12.2 Tests

The available quality tests can be classified as follows:

Face continuity Checks for continuity between faces in a surface model. Positional and tangential discontinuities with respect to the gap and the kink tolerance, respective, are returned. Discontinuities larger than the neighbour tolerance or bend tolerance are assumed to be intentional, and thus no error is reported.

Edge continuity Checks for positional and tangential discontinuity between edges in the model. The same tolerances are used as for face continuity.

Accuracy of bounding entities Whether or not the distance between an edge and its associated face, a vertex and its associated edges and a vertex and its associated faces is less than the gap tolerance.

Acute angles Check for acute angles between adjacent faces or edges. The kink tolerance is applied.

Degeneracies Identify surfaces with boundary curves degenerating to a point and surfaces with degenerate corners. Identify vanishing surface normals and curve tangents, intersections between boundary loops belonging to a trimmed surface and self intersections of boundary loops. The gap tolerance is used in the intersection and self intersection tests while the neighbour and kink tolerance is used in degeneracy tests. The gap tolerance is used in tests for vanishing tangents and normals.

Small entities Identify small edges, small faces, sliver surfaces, and narrow regions in faces. The narrow region test relates to the neighbour tolerance while the other tests use specific tolerances.

Consistency of orientation Check for consistency in loops, i.e. the curves defining the loop are head to tail oriented, or in surface models. For closed face sets, all surface normals should point out of the model or into it, preferable out. Also open face sets should have consistent surface normal directions at face boundaries. However, the face orientation test is currently not up to date and the result can unfortunately not be trusted.

Identical entities Check for identity of vertices, and identical or embedded faces or edges. The neighbourhood tolerance is used.

Spline entity testing Check spline curves and surfaces for G^1 and C^1 discontinuities. The test is localized to knots of high multiplicity. The gap and the kink tolerances are used for these tests. Check also for spline entities with close, but not identical knots. Such entities can create problems for certain operations. A specific tolerance is used in this test.

Compute curvature information with respect to single entities The minimum curvature radius and curvature radii less than a given threshold can be obtained. The test is applied to all curve or surface entities in a model.

The tests described above are localized in the class [FaceSetQuality](#). See the documentation of this class for more information.

Chapter 13

The filter routines main page

13.1 Introduction

This is a collection of routines for handling various (3x3x3) 3d filters.

020114: This REALLY REALLY must be cleaned up... 021231: Not yet, but soon, it will only be a question of removing commented out stuff... Hopefully... 050120: Changed return types for boolean functions from int to bool. Hope this doesn't break anything.

Chapter 14

GoTools Topology

This module contains functionality to perform topology analysis on a collection of faces. The adjacency analysis itself is performed in [FaceAdjacency](#), and the operations uses template faces and edges. The interfaces are given in [tpFace](#) and [tpEdge](#), respectively.

The module depends on the GoTools core library

Chapter 15

GoTools Trivariate

This module trivariate represents NURBS volumes and contains construction methods and operations related to such volumes.

This module depends on:

- GoTools Core library
- SISL library

15.1 Data Structures

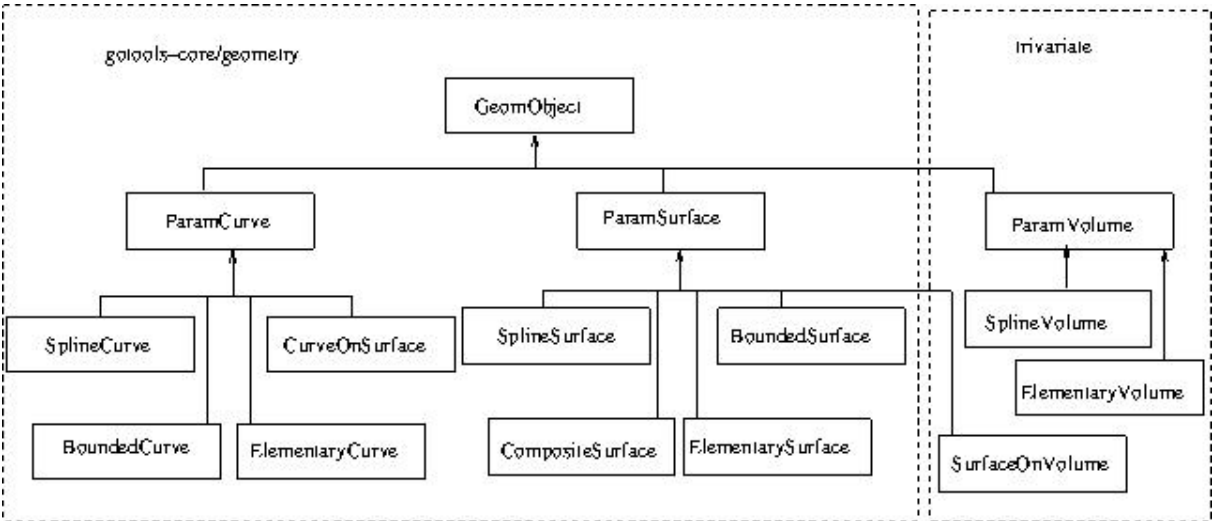


Figure 15.1 Simplified overview of the geometry class hierarchy

The figure above shows the main geometric classes in GoTools and how they are divided between modules.

15.2 B-spline Volumes

A B-spline volume is represented in [SplineVolume](#) in the GoTools module trivariate.

The volume is defined by the formula

$$\mathbf{V}(u, v, w) = \sum_{i=1}^{n_1} \sum_{j=1}^{n_2} \sum_{h=1}^{n_3} \mathbf{p}_{i,j,h} B_{i,k_1,\mathbf{u}}(u) B_{j,k_2,\mathbf{v}}(v) B_{h,k_3,\mathbf{w}}(w)$$

with control points $\mathbf{p}_{i,j,h}$ and three variables (or parameters) u , v and w . A basis function of a B-spline volume is a product of three basis functions of B-spline curves (B-splines).

The following is a list of the components of the representation:

- dim : The dimension of the underlying Euclidean space.
- n_1 : The number of control points with respect to the first parameter.
- n_2 : The number of control points with respect to the second parameter.
- n_3 : The number of control points with respect to the third parameter.
- k_1 : The order (polynomial degree + 1) of the B-splines in the first parameter.
- k_2 : The order of the B-splines in the second parameter.
- k_3 : The order of the B-splines in the third parameter.
- \mathbf{u} : The knot vector of the B-splines with respect to the first parameter, $\mathbf{u} = (u_1, u_2, \dots, u_{n_1+k_1})$.
- \mathbf{v} : The knot vector of the B-splines with respect to the second parameter, $\mathbf{v} = (v_1, v_2, \dots, v_{n_2+k_2})$.
- \mathbf{w} : The knot vector of the B-splines with respect to the third parameter, $\mathbf{w} = (w_1, w_2, \dots, w_{n_3+k_3})$.
- \mathbf{p} : The control points of the B-spline volume, $c_{d,i,j,h}$, $d = 1, \dots, dim$, $i = 1, \dots, n_1$, $j = 1, \dots, n_2$, $h = 1, \dots, n_3$. When $dim = 3$, we have $\mathbf{p} = (x_{1,1,1}, y_{1,1,1}, z_{1,1,1}, x_{2,1,1}, y_{2,1,1}, z_{2,1,1}, \dots, x_{n_1,1,1}, y_{n_1,1,1}, z_{n_1,1,1}, \dots, x_{n_1,n_2,1}, y_{n_1,n_2,1}, z_{n_1,n_2,1}, \dots, x_{n_1,n_2,n_3}, y_{n_1,n_2,n_3}, z_{n_1,n_2,n_3})$.

The data of the B-spline volume must fulfill the following requirements:

- All knot vectors must be non-decreasing.
- The number of control points must be greater than or equal to the order with respect to all three parameters: $n_1 \geq k_1$, $n_2 \geq k_2$ and $n_3 \geq k_3$.

The properties of the representation of a B-spline volume are similar to the properties of the representation of a B-spline curve or surface. The control points $\mathbf{p}_{i,j,h}$ form a *control net*. The control net has similar properties to the control polygon of a B-spline curve, described in the module `gotools-core`. A B-spline volume has three knot vectors, one for each parameter.

15.2.1 The Basis Functions

A [basis function](#) of a B-spline volume is the product of three basis functions corresponding to B-spline curves,

$$B_{i,k_1,\mathbf{u}}(u) B_{j,k_2,\mathbf{v}}(v) B_{h,k_3,\mathbf{w}}(w)$$

Its support is the box $[u_i, u_{i+k_1}] \times [v_j, v_{j+k_2}] \times [w_h, w_{h+k_3}]$.

15.2.2 NURBS Volumes

A NURBS (Non-Uniform Rational B-Spline) volume is a generalization of a B-spline volume,

$$\mathbf{V}(u, v, w) = \frac{\sum_{i=1}^{n_1} \sum_{j=1}^{n_2} \sum_{r=1}^{n_3} h_{i,j,r} \mathbf{P}^{i,j,r} B_{i,k_1,u}(u) B_{j,k_2,v}(v) B_{r,k_3,w}(w)}{\sum_{i=1}^{n_1} \sum_{j=1}^{n_2} \sum_{r=1}^{n_3} h_{i,j,r} B_{i,k_1,u}(u) B_{j,k_2,v}(v) B_{r,k_3,w}(w)}.$$

In addition to the data of a B-spline surface, the NURBS surface has a weights $h_{i,j,r}$. NURBS volumes can be used to exactly represent volumes that have common 'analytic' surfaces such as spheres, cylinders, tori, and cones as boundary surfaces. A disadvantage is that NURBS volume depend nonlinearly on their weights, making some calculations less efficient.

The representation of a NURBS volume is the same as for a B-spline volume except that it also includes

h: The weights of the NURBS volume, $h_{i,j,r}$, $i = 1, \dots, n_1$, $j = 1, \dots, n_2$, $r = 1, \dots, n_3$, so $\mathbf{h} = (h_{1,1,1}, h_{2,1,1}, \dots, h_{n_1,1,1}, h_{1,2,1}, \dots, h_{n_1,n_2,1}, \dots, h_{n_1,n_2,n_3})$.

The weights are required to be strictly positive: $h_{i,j,r} > 0$.

The NURBS volume is represented by `SplineVolume`. As for the curve and surface cases, the constructor expects the coefficients to be multiplied with the weights.

15.2.3 Spline Volume Functionality

The functionality of a [spline volume](#) to a large extend corresponds to the functionality of a spline surface. Important functionality is:

- A NURBS volume is able to make a copy of itself
- Compute the bounding box of the volume
- Evaluation and grid evaluation
- Grid evaluation of basis functions
- Compute the derivative volume corresponding to a volume
- Closest point computation
- Fetch a sub volume of a given volume
- Fetch information related to the spline spaces
- Swap and reverse parameter directions in a volume
- Fetch the control polygon of the volume
- Fetch all weights of a NURBS volume
- Insert knots into the spline spaces of the volume and adapt the volume description accordingly
- Increase the polynomial degree of the volume in one parameter direction
- Fetch a constant parameter surface from the volume
- Fetch all boundary surfaces surrounding a volume
- Check for periodicity and degeneracy

15.2.4 Construction Methods for SplineVolume

The following methods exist for construction of a spline volume. The corresponding GoTools class names are given in brackets.

- Sweep a NURBS surface along a NURBS curve ([SweepVolumeCreator](#)).
- Rotational sweep of a NURBS surface ([SweepVolumeCreator](#)).
- Lofting to interpolate a number of NURBS surfaces ([LoftVolumeCreator](#)).
- Interpolate 6 boundary surface to create a volume using a Coons patch approach ([CoonsPatchVolumeGen](#)). This functionality applies only to non-rational spline surfaces.
- Represent an [elementary volume](#) as a spline volume. An elementary volume is a [parametric volume](#) similarly to spline volumes. The elementary volumes are:
 1. [Sphere](#)
 2. [Cylinder](#)
 3. [Cone](#)
 4. [Parallelepiped](#)
 5. [Torus](#)

A spline volume may have a well behaved outer boundary, but a bad distribution of coefficients in the inner. This is in particular the case if the volume is constructed by a Coons patch approach. The positioning of the internal coefficients may be improved by smoothing. The coefficients at the boundaries are kept fixed and the coefficients in the inner are redistributed by solving a minimization problem. The smoothing is performed in the class [SmoothVolume](#).

15.3 Surface on Volume

The module trivariate provides a surface which extends the class of parametric surfaces defined in gotools-core, namely [SurfaceOnVolume](#). This surface inherits most of the functionality defined for parametric surfaces and takes the same role as [CurveOnSurface](#) for parametric curves. The surface possesses information about

- The associated volume
- The geometric description of this surface and/or
- The description of this surface in the parameter domain of the given volume
- Constant parameter and volume boundary information. If this surface for instance happens to be a constant parameter surface in the given volume, it knows the parameter direction and the associated constant parameter value.

15.4 Curves Related to a Parametric Volume

Two evaluator based curves can be used in together with [HermiteAppC](#) in gotools-core/creators exist, namely [VolumeParameterCurve](#) and [VolumeSpaceCurve](#).

Chapter 16

GoTools Trivariatemodel

The module trivariatemodel contains the representation format for a set of volumes and tools to act upon the volumemodel.

The module depends on:

- The GoTools Core library
- GoTools Composite Model
- GoTools Trivariate
- Modules that Composite Model depend upon

16.1 Volume Topology

The volume topology in GoTools is an extension to the boundary represented topology implemented in the module compositemodel. However, in contrast to the boundary represented topology, the volume topology is not manifold. Thus, we must expect more than two faces to meet in an edge.

The top entity in the topology structure is the volume model ([VolumeModel](#)) which consists of a set of volumes. Each volume has a topological entity implemented in [ftVolume](#) and a geometrical representation implemented as [ParamVolume](#). Information about the geometric representation of a volume can be found in the module trivariate.

[ftVolume](#) inherits [Body](#) in compositemodel and is, as [Body](#), surrounded by one or more shells represented as [SurfaceModels](#). A shell is a collection of faces ([ftSurface](#)) which have a geometric representation as a [Param↔Surface](#). [ParamSurface](#) is described in the module gotools-core.

A face is limited by a number of [loops](#) that are sequences of edges ([ftEdge](#)). In this context, the face is used to represent adjacency between volumes. An edge is limited by two [vertices](#).

16.2 Volume Model

`VolumeModel` is a [composite model](#) as illustrated in the figure above. See also the description in the documentation of composite model. As such it inherits a function interface, but some methods are not implemented. Thus, the class is incomplete, but not in the sense of being a topological entity.

`VolumeModel` has the following functionality:

- Fetch one entity in the set, either as a topological or geometric volume.
- Evaluate the volume model given information about which entity to evaluate
- Compute [bounding box](#)
- Add a new volume to the volume model
- Remove one volume from the volume model
- Check if all geometrical volumes are NURBS
- Fetch all [vertices](#) in the model
- Fetch all [radial edges](#) in the model
- Fetch non-radial edges, i.e. edges that does not belong to at least two volumes.
- Fetch all inner faces in the model
- Fetch all boundary faces in the model
- Check if the model has a corner-to-corner configuration
- Ensure that the model has a corner-to-corner configuration
- Ensure that adjacent spline volumes share common spline spaces at common boundaries
- Fetch the boundary of the volume model described as a number of `SurfaceModels`
- Divide the volume model into connected volume models

The constructor of the volume model requires a number of `comp_sec2`, namely `gap`, `neighbour`, `kink` and `bend`. These tolerances are the same as the ones required for a composite model, and the tolerances are explained in the documentation of the `compositemodel` module.

16.3 Topological Volume Entity

The topological volume entity is implemented in the class `ftVolume` and it inherits `Body` from the `compositemodel` module. The name `ftVolume` is chosen to be in line with `ftSurface` and `ftEdge`. Those entities have got their names for historical reasons. The prefix `ft` has no deeper meaning.

An `ftVolume` is a `Body` and has thus access to its shells, i.e. boundaries, and can check whether two `ftVolumes` are adjacent. An `ftVolume` can also

- Fetch the corresponding geometry volume
- Fetch the bounding box of this volume
- Fetch all adjacent `ftVolumes`
- Fetch all radial edges belonging to this volume or being common between this volume and another volume

- Fetch edges belonging only to the current volume
- Compute adjacency information between the current volume and another volume
- Return information about outer boundaries
- Get local information about correspondence of coefficients of two adjacent spline volumes
- Return the relation between the current volume and a given vertex

A `ftVolume` may be trimmed. That is, the boundary faces of the volume may limit the extent of the underlying parametric volume. In this context the following functionality apply:

- Check if the volume is hexahedral
- Approximate a hexahedral trimmed volume by one spline volume
- Split a trimmed volumes into a set of hexahedral trimmed volumes

Note that the functionality related to trimmed volumes are under development and currently very unstable and limited.

16.4 Extensions to Face

The initial implementation of a face as one entity in a boundary representation solid was no longer sufficient when the entity should serve as part of the boundary of a volume belonging to a volume model. Some extensions turned out to be required. The face `ftSurface` has knowledge about the `Body` it belongs to, if any. It has also a pointer to a boundary face belonging to an adjacent body. This pointer is used in the context of a volume model. During the assembly of a volume model, information about coincident boundary faces of the `ftVolumes` are computed, and the pointer representing adjacency is set accordingly.

An `ftSurface` provides access to its twin and to the, at most, two bodies sharing the common face represented by this `ftSurface`.

When the `ftSurface` represents the boundary surface of an `ftVolume`, the corresponding parametric surface will be of type `SurfaceOnVolume`. This class is implemented in the module `trivariate`. A `SurfaceOnVolume` has knowledge about the spatial representation of the surface, the `ParamVolume` on which it belongs and the position of the surface in the parameter domain of the volume. Currently, a `SurfaceOnVolume` is constructed only as a boundary surface of a `SplineVolume`. It contains enough information to identify which volume boundary it represents and the orientation of the surface compared to this boundary.

16.5 Radial Edge

In a non-manifold model, more than two faces can meet in an edge, and thus the half edge representation implemented in `ftEdge` is not sufficient to hold the model. The radial edge, `EdgeVertex`, is an extension to the topology structures of `compositemodel` and the class itself is placed in `compositemodel`. An `EdgeVertex` contains information of all pairs of half edges, `ftEdge`, meeting in an edge, and each `ftEdge` belonging to an `EdgeVertex` has access to this `EdgeVertex`.

An `EdgeVertex` instance is defined when adjacency between two `ftSurfaces` is found. Thus, at edges in an `ftVolume` where no adjacent `ftVolume` exists, no `EdgeVertex` instance will be defined. Then the edge is represented with a pair of `ftEdge` instances.

The `EdgeVertex` class is placed in the module `compositemodel`, but is used in connection with volume models. In addition to functionality related to topology build, the class has some access functionality:

- Fetch edges meeting in the radial edge, either uniquely defined, i.e. one instance for a pair or twins, or absolutely all edges.
- Fetch a specified edge
- Check if a ftEdge belongs to a radial edge
- Check if a ftEdge belongs to a radial edge and has no information about a corresponding ftEdge. In that case the corresponding surface does not belong to a volume.
- Fetch adjacent faces
- Fetch adjacent bodies

Chapter 17

Application Programming Interface to TTL (API)

There are two interface channels between TTL and the application data structure: `DartType` and `TraitsType`, which are template arguments to the TTL functions. Function templates in TTL use either `DartType` or both `DartType` and `TraitsType` depending on the complexity of the algorithms. `DartType` is used for topological queries, and `TraitsType` is used for topological modifiers and geometric calculations. The documentation in TTL tells which functionality is needed in `DartType` and `TraitsType` for each function template.

TTL consists of two namespaces with generic functions: `tll` contains the main interface and `tll_util` contains utility functions that can also be used by the application programmer. The figure below shows the (simple) architecture of an application using TTL.

In the following we explain how `DartType` and `TraitsType` must be implemented as an interface to the actual data structure. For more details see [literature](#).

17.1 DartType

The generic functions in TTL navigates in the application data structure through a "topological element" call a *dart*, which must be implemented as a struct or a class by the application programmer. A dart can be considered as a unique triple $d = (V, E, T)$, where V is a node of the edge E , and V and E is a node and an edge of the triangle T ; see figure a) below where a dart is indicated as an arrow.

TTL expects that three functions, which we call *alpha-iterators*, are present in the dart class: `alpha0()`, `alpha1()` and `alpha2()`. These functions reposition the dart in the triangulation as shown in figure b) above and return a reference to the modified dart itself. Thus, `alpha0()`, `alpha1()` and `alpha2()` change the node, the edge and the triangle of the triple $d = (V, E, T)$ respectively.

Note

- The *dart* is not part of the data structure. It is a "dynamic" element that "moves" around in the actual application data structure when the alpha-iterators are applied to it. Thus, this mechanism does not require any extra memory other than that occupied by the (few) darts that are involved in a function template of TTL that is called.

In addition to the alpha-iterators, TTL expects that standard class member functions such as constructors, assignment operators and the like are also implemented in the dart class.

The following syntax is required:

```

class MyDart {
    ...
public:
    // Constructors and destructors
    ...
    MyDart(const MyDart& dart) {...} // copy constructor
    MyDart& operator= (const MyDart& dart) {...} // assignment operator

    // comparing dart objects
    bool operator==(const MyDart& dart) const {...}
    bool operator!=(const MyDart& dart) const {...}

    // alpha-iterators
    MyDart& alpha0() {...; return *this;}
    MyDart& alpha1() {...; return *this;}
    MyDart& alpha2() {...; return *this;}
};

```

Thus, the alpha-iterators change the content of the dart and return a reference to the dart itself.

Note

- Note that `alpha2()` must return the dart itself without changing its content (when checked with the `==` operator) if the edge associated with the dart is at the boundary of the triangulation. The source code of [ttl::isBoundaryEdge](#) illustrates this:

```

template <class DartType>
bool isBoundaryEdge(const DartType& dart) {

    DartType dart_iter = dart;
    if (dart_iter.alpha2() == dart)
        return true;
    else
        return false;
}

```

Consult the definition and source code of class [hed::Dart](#), which implements the dart class for the half-edge data structure, for a thorough example.

17.2 TraitsType

`TraitsType` can be a static class or a struct that contains functionality required by the functions in TTL. The purpose with `TraitsType` is twofold:

- Let the application programmer provide topological modifiers on the actual data structure that are not resolved by TTL, e.g., remove a triangle from the triangulation,
- Let the application programmer provide basic geometric calculations and thus control the level of accuracy for such calculations.

For example, assume that a function in TTL requires a scalar product between vectors in the plane, represented as darts, with the following syntax:

```

real_type scalarProduct2d(const DartType& d1, const DartType& d2)

```

Then this function must be implemented in the traits class together with the definition of `real_type`:

```

struct MyTraitsType {
    typedef double real_type;
    ...
    static real_type scalarProduct2d(const MyDart& v1, const MyDart& v2) {
        return ::my_scalarProduct2d(v1,v2);
    }
    ...
    static bool swapTestDelaunay(const MyDart& dart) {
        if (dart.getEdge().isConstrained())
            return false;
        else
            return ttl::swapTestDelaunay<MyTraitsType>(dart);
    }
};

```

The second function, `swapTestDelaunay`, implements the Delaunay swapping test. The function decides if the edge associated with the given dart should be swapped according to the *circumcircle* criterion (or equivalently according to the *MaxMin* angle criterion). In the example above, it is first examined if the edge is constrained, in which case `false` is returned. Then the test is directed back to TTL again since the circumcircle test is present there; see [ttl::swapTestDelaunay](#). Of course, the user could also make his/her own implementation of the circumcircle test, e.g., by using robust schemes with exact arithmetic as Jonathan Richard Shewchuk does (<http://www-2.cs.cmu.edu/~quake/robust.html>).

This example also illustrates the syntax `functionName<MyTraitsType>(...arguments...)` when calling function templates using a traits class.

Actually, scalar product between vectors, as well as cross product and other point and vector algebra, are also present in the namespace `ttl_util`. Assume that `MyDart::x()` and `MyDart::y()` deliver the coordinates in the plane of the node associated with a dart. Then the scalar product in `MyTraitsType` can be implemented as:

```

static real_type scalarProduct2d(const MyDart& v1, const MyDart& v2) {
    MyDart v10 = v1; v10.alpha0();
    Dart v20 = v2; v20.alpha0();
    return ttl_util::scalarProduct2d(v10.x()-v1.x(), v10.y()-v1.y(),
                                    v20.x()-v2.x(), v20.y()-v2.y());
}

```

Consult the definition and source code of struct [hed::TTLtraits](#), which implements the traits class for the half-edge data structure. This example shows which members that are required in the traits class for running incremental Delaunay triangulation and removing nodes in a triangulation.

Chapter 18

Example using TTL and the half-edge data structure

Chapter 19

The Half-Edge Data Structure and Adaption to TTL

An implementation of the the well-known *half-edge* data structure for triangulations is documented briefly on these pages together with adaption of this data structure to [TTL, The Triangulation Template Library](#). Only the class [hed::Dart](#) and the (static) struct [hed::TTLtraits](#) are documented thoroughly. These compounds represent the interface channels between TTL and the application data structure. See [Application Programming Interface to TTL \(API\)](#) for a general documentation of this mechanism. Follow the links to the source code for classes that are not documented in detail - most member functions are self-explanatory.

19.1 The Class Diagram

The classes [hed::Node](#), [hed::Edge](#) and [hed::Triangulation](#) represent the half-edge data structure shown as a class diagram in the figure below. ("Edge" denotes a half-edge). These classes, together with [hed::Dart](#) and [hed::TTLtraits](#) for adaption to TTL, are encapsulated in the namespace **hed**.

Each half-edge has a pointer to the [hed::Node](#) it starts from, a pointer to the next half-edge belonging to the same triangle (counterclockwise), and a pointer to its "twin-edge" belonging to another triangle.

The class [hed::Triangulation](#) contains a list of (half-) edges, each of which represents one triangle in the triangulation. In addition, the class [hed::Triangulation](#) also serves as an interface to the TTL as it implements [hed::TTLtraits::swapEdge](#), [hed::TTLtraits::splitTriangle](#) etc., which are functions required on the application side by function templates in the [TTL](#) (via the traits class).

19.2 Example

Consult the file [main.cpp](#) for a program using TTL and the half-edge data structure.

Chapter 20

GoTools Viewlib

The Viewlib module contains the application 'goview', which is a utility to support visualization of curves, surface, point clouds and line clouds.

goview can read curves and surfaces from an IGES file or from the GoTools internal file format g2 and visualize those. Currently, goview is not able to visualize volumes. In the volume case, it is recommended to pick the boundary surfaces corresponding to the volume and draw them.

For more information on the g2 file format, see [The g2-format, GoTools file format for geometry entities](#).

The program uses Qt for representing the GUI and OpenGL for graphics. Curves and surfaces are tessellated in the submodule tessellate in the module gotools-core. According to their type, curves and surfaces are approximated by triangles or line segments that are convenient for visualization using OpenGL.

The model in the viewer is manipulated using the mouse keys. The left one rotates the model, the middle one is for zooming and the right one for translation of the model.

goview has got a graphical user interface. It has a graphical window, a window containing an object list and a number of pull down menus:

- `file` commands to read and write geometry and to close the current session and make the viewer ready to read a new geometry file.
- `view` options to choose shaded or wire frame mode for visualization. A highlight modus may be toggled and some focusing facilities exist.
- `select` Selection of entities can be done using this menu, in the object list or by using the control key in combination with the left mouse key.
- `group` grouping of objects. This menu is not really used
- `object` this menu offers the possibility to alter the resolution of curves and surfaces and to enable/disable selected entities from the view. Some commands have a key pad short cut.

goview is a utility and not a product. This implies unfortunately that the help functionality is not implemented. Visualization of trimmed surfaces can have some flaws, but they are normally repaired or minimized by increasing the resolution.

20.1 Dependencies

Viewlib requires the following libraries to be installed on the system:

- Qt4, qt.nokia.com
- OpenGL and GLUT, www.opengl.org
- Boost, www.boost.org

Chapter 21

Bug List

Class [Go::ftSSfEdge](#)

Not tested

Chapter 22

Module Index

22.1 Modules

Here is a list of all modules:

RBD common library 157

Chapter 23

Namespace Index

23.1 Namespace List

Here is a list of all namespaces with brief descriptions:

Go	159
Go::AdaptSurface	222
Go::BoundedUtils	224
Go::boxStructuring	233
Go::ClosestPoint	
Namespace for computing closest points	233
Go::cmUtils	
Various utility functions for the compositemodel module	238
Go::CoonsPatchGen	240
Go::CoonsPatchVolumeGen	
This namespace contains functions used to create a Coons Patch volume	247
Go::CreatorsUtils	
Related to the generation of cross tangent curves	249
Go::Curvature	
Curvature analysis related to curves	252
Go::CurvatureAnalysis	253
Go::CurveCreators	255
Go::CurveInterpolator	
Curve interpolation functionality not adapted to the Interpolator class	259
Go::FaceUtilities	259
Go::ftVolumeTools	
This namespace contains service functions related to ftVolume	260
Go::GapRemoval	
Functionality for removal of gaps between two adjacent surfaces of various types	261
Go::GeometryTools	264
Go::HahnsSurfaceGen	276
Go::IntersectionUtils	
Various functions related to the intersection algorithms	278
Go::LinDepUtils	279
Go::LoftSurfaceCreator	
This namespace contains functions used to create lofted surfaces	279
Go::LoftVolumeCreator	
This namespace contains functions used to create lofted volumes	282
Go::LoopUtils	285
Go::LRApproxApp	287

Go::LRBSpline2DUtills	288
Go::LRSplineMBA	289
Go::LRSplineUtills	290
Go::LRSurfStitch	292
Go::Mesh2DUtills	293
Go::ModifySurf	293
Go::OffsetSurfaceUtills	294
Go::OffsetUtills	
Related to the generation of cross tangent curves	295
Go::orientCurves	
Sorts and orients a set of curves	295
Go::Path	
Functions related to sequence of edges	296
Go::PointSetApp	
Point set and triangulation applications	297
Go::qualityUtills	298
Go::RegularizeUtills	
Utility functionality for RegularizeFace and RegularizeFaceSet	298
Go::Singular	302
Go::SplineDebugUtills	302
Go::SplineUtills	304
Go::SplitModelUtills	
Utility functionality for splitting of surface models	309
Go::SurfaceCreators	310
Go::SurfaceInterpolator	
Functionality for creating interpolating surfaces	312
Go::SurfaceModelUtills	
Utility functionality for splitting of surface models	313
Go::SurfaceOnVolumeTools	313
Go::SurfaceTools	
Free functions operating on parametric surfaces	313
Go::TesselatorUtills	
Related to the relative resolution of tessellation	318
Go::tpUtills	318
Go::Utills	
Namespace for some utility functions	319
Go::VolumeInterpolator	
This namespace contains functions used to interpolate a set of points	320
Go::VolumeModelCreator	321
Go::VolumeTools	
This namespace contains free functions operating on parametric volumes	321
hed	327
hetriang	327
NEWMAT	328
RBD_COMMON	335
RBD_LIBRARIES	336
std	336
ttl	
Main interface to TTL	338
ttl_constr	
Constrained Delaunay triangulation	352
ttl_util	
Utilities	353

Chapter 24

Hierarchical Index

24.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Go::AdaptCurve	357
Go::AdjacencyInfo	363
Go::Alg2DElem	365
Go::Alg3DElem	366
Go::AlgObjectInt	374
Go::AlgObj2DInt	368
Go::Line2DInt	1332
Go::AlgObj3DInt	370
Go::CylinderInt	729
Go::PlaneInt	1690
Go::SphereInt	2090
Go::ApproxCrvToSeqs	375
Go::ApproxCurve	378
Go::ApproxSurf	382
Go::Array< T, Dim >	386
Go::Array< double, 2 >	386
Go::Array< double, 3 >	386
Go::Array< double, Dim >	386
Go::Array< Go::BernsteinTriangularPoly, N >	386
Go::Array< int, 2 >	386
Go::Array< int, 3 >	386
NEWMAT::ArrayLengthSpecifier	394
Go::BaryCoordSystem< Dim >	407
Go::BaryCoordSystem< 2 >	407
Go::BaryCoordSystem< 3 >	407
Go::BaryCoordSystemTriangle3D	409
RBD_COMMON::BaseException	411
RBD_COMMON::Bad_alloc	395
RBD_COMMON::Logic_error	1353
NEWMAT::CannotBuildException	538
NEWMAT::IncompatibleDimensionsException	1182
NEWMAT::IndexException	1184
NEWMAT::InternalException	1190
NEWMAT::NotDefinedException	1497
NEWMAT::NotSquareException	1499

NEWMAT::ProgramException	1768
NEWMAT::SubMatrixDimensionException	2226
NEWMAT::VectorException	2411
RBD_COMMON::Domain_error	785
RBD_COMMON::Invalid_argument	1286
RBD_COMMON::Length_error	1317
RBD_COMMON::Out_of_range	1509
RBD_COMMON::Runtime_error	1950
NEWMAT::ConvergenceException	637
NEWMAT::NPDException	1500
NEWMAT::OverflowException	1512
NEWMAT::SingularException	2002
RBD_COMMON::Overflow_error	1511
RBD_COMMON::Range_error	1880
RBD_COMMON::SolutionException	2061
Go::BaseSurface	425
Go::BasisDerivs	425
Go::BasisDerivs2	427
Go::BasisDerivsSf	430
Go::BasisDerivsSf2	431
Go::BasisPts	433
Go::BasisPtsSf	434
Go::BdCondFunctor	436
Go::BernsteinMulti	436
Go::BernsteinPoly	446
Go::BernsteinTetrahedralPoly	452
Go::BernsteinTriangularPoly	457
Go::BezierTriangle< N >	462
Go::Binomial	464
Go::BlockBoundaryCondition	466
Go::SfBoundaryCondition	1963
Go::VolBoundaryCondition	2420
Go::BlockPointBdCond	469
Go::SfPointBdCond	1971
Go::VolPointBdCond	2422
Go::BlockSolution	470
Go::SfSolution	1987
Go::VolSolution	2424
Go::Body	474
Go::ftVolume	988
bool	479
Go::BoundaryFunctionInt	480
Go::BoundaryGeomInt	482
Go::BoundaryIntersectionData	484
Go::BoundingBox	515
Go::boxStructuring::BoundingBoxStructure	519
Go::LRSplineSurface::BSKey	522
Go::CachedInterval	536
Go::CellDivision	539
Go::ClosestPointCalculator	552
Go::CompleteEdgeNet	560
Go::ComplexityInfo	561
Go::CompositeBox	564
Go::CompositeModel	580
Go::CompositeCurve	568
Go::CurveModel	667
Go::SurfaceModel	2238

Go::VolumeModel	2434
Go::CompositeModelFactory	588
Go::CompositeModelFileHandler	592
Go::VolumeModelFileHandler	2444
Go::ConnectionFunctor	633
ControlWord	634
LoadAndStoreFlag	1349
Go::CoordinateSystem< Dim >	638
Go::CPUclock	641
Go::Creator	642
Go::ConcreteCreator< T >	612
Go::CrossesValue	643
Go::CurveLoop	662
Go::cvSetConstraint	714
hed::Dart	739
hetriang::Dart	742
DataHandler	746
DefaultDataHandler	750
DataHandlerVolAndLR	749
Go::ParamSurface::degenerate_info	752
Dijkstra	764
Go::DirectionCone	768
Go::Domain	782
Go::CurveBoundedDomain	655
Go::RectDomain	1894
Go::LRSplineSurface::double_pair_hash	786
hed::Edge	787
hetriang::Edge	790
EdgeType	792
Go::EdgeVertex	792
Go::Element2D	796
Go::LRSplineSurface::ElemKey	816
Go::EntityList	827
Go::Streamable::EofException	829
Go::EvalCurve	829
Go::CrossTangentOffset	644
Go::CrossTanOffDist	647
Go::EvalFunctorCurve	836
Go::EvalParamCurve	846
Go::LiftCurve	1319
Go::ProjectCurve	1770
Go::ProjectIntersectionCurve	1777
Go::SpaceIntCrv	2074
Go::VolumeParameterCurve	2445
Go::VolumeSpaceCurve	2449
Go::EvalCurveSet	833
Go::IntCrvEvaluator	1187
Go::ProjectCurveAndCrossTan	1773
Go::SmoothTransition	2055
Go::TrimCurve	2381
Go::EvalSurface	849
Go::EvalFunctorSurface	839
Go::EvalOffsetSurface	842
Go::FaceAdjacency< edgeType, faceType >	852
Go::FaceConnectivity< edgeType >	857
Go::FaceConnectivityUtils< edgeType, faceType >	860

Go::Factorial< N >	871
Go::Factorial< 1 >	872
Go::Factory	873
NEWMAT::FFT_Controller	874
NEWMAT::FindMaximum2	876
NEWMAT::MLE_D_FI	1459
NEWMAT::NonLinearLeastSquares	1494
NEWMAT::FloatingPointPrecision	877
Go::ftCell	879
Go::ftCellInfo	881
Go::ftCurve	888
Go::ftCurveSegment	894
Go::ftEdgeBase	910
Go::ftEdge	899
Go::ftSSfEdge	956
Go::ftFaceBase	918
Go::ftSurface	961
Go::ftSurfaceSet	978
Go::ftChartSurface	883
Go::ftGraphEdge	922
Go::ftGroupGeom	924
Go::ftLine	926
Go::ftMessage	928
Go::ftPlanarGraph	930
Go::ftPlane	932
Go::ftPoint	934
Go::ftSamplePoint	945
Go::ftSurfaceSetPoint	985
Go::ftSearchNode	951
Go::ftSmoothSurf	952
Go::Fun2Fun< Functor >	996
Go::FunctionMinimizer< Functor >	997
Go::GeneralMesh	1021
Go::GenericTriMesh	1027
Go::LineStrip	1344
Go::QuadMesh	1871
Go::RegularMesh	1927
Go::RegularVolMesh	1931
Go::GeomObjectInt	1034
Go::ParamObjectInt	1612
Go::ParamFunctionInt	1602
Go::Param0FunctionInt	1539
Go::Param1FunctionInt	1548
Go::Spline1FunctionInt	2099
Go::Param2FunctionInt	1560
Go::Spline2FunctionInt	2104
Go::ParamGeomInt	1606
Go::ParamCurveInt	1583
Go::SplineCurveInt	2131
Go::ParamPointInt	1621
Go::ParamSurfaceInt	1647
Go::SplineSurfaceInt	2182
Go::GeoTol	1034
Go::go_iterator_traits< Iterator >	1041
Go::go_iterator_traits< const T * >	1042
Go::go_iterator_traits< T * >	1042

Go::GoTools	1043
Go::GPos	1045
Go::GpuMatrix< T >	1046
gvCamera	1057
gvColor	1063
gvData	1067
gvGenericTriQuadMesh< FloatType >	1074
gvGroup	1078
gvObserver	1087
gvObjectList	1085
gvView	1112
gvPaintable	1088
gvCurvePaintable	1064
gvGenericTriPaintable< FloatType >	1072
gvGenericTriQuadPaintable< FloatType >	1076
gvLineCloudPaintable	1081
gvNoopPaintable	1083
gvParametricSurfacePaintable	1094
gvPointCloudPaintable	1096
gvQuadsPaintable	1101
gvRectangularSurfacePaintable	1103
gvRectangularVolumePaintable	1105
gvPainter	1092
gvPropertySheet	1100
gvGroupPropertySheet	1079
ParametricSurfacePropertySheet	1598
PointCloudPropertySheet	1707
RectangularSurfacePropertySheet	1886
RectangularVolumePropertySheet	1890
SplineCurvePropertySheet	2138
SurfaceResolutionSheet	2308
gvStandardMouseHandler	1109
gvTexture	1110
Handle< T >	1122
Handle< hed::Node >	1122
HandleId	1126
hed::Node	1486
HeapNode	1128
HeapNode2	1130
Go::HermiteAppC	1131
Go::HermiteApprEvalSurf	1133
Go::HermiteAppS	1135
Go::HermiteGrid1D	1136
Go::HermiteGrid1DMulti	1139
Go::HermiteGrid2D	1141
Go::Identity	1157
Go::IGESconverter	1162
Go::IGESdirentry	1166
Go::IGESheader	1168
Go::ImplicitizeCurveAlgo	1172
Go::ImplicitizeCurveAndVectorAlgo	1174
Go::ImplicitizePointCloudAlgo	1177
Go::ImplicitizeSurfaceAlgo	1179
Go::IndexMesh2DIterator	1185
Go::Interpolator	1196
Go::HermiteInterpolator	1144
Go::SplineApproximator	2110

Go::SplineInterpolator	2140
Go::IntersectionCurve	1197
Go::DegeneratedIntersectionCurve	754
Go::InterpolatedIntersectionCurve	1192
Go::IsoparametricIntersectionCurve	1307
Go::NonEvaluableIntersectionCurve	1490
Go::IntersectionLink	1203
Go::IntersectionPoint	1205
Go::IntersectionPool	1224
Go::Intersector	1246
Go::Intersector2Obj	1256
Go::CvCvIntersector	707
Go::CvPtIntersector	711
Go::PtPtIntersector	1867
Go::SfCvIntersector	1967
Go::SfPtIntersector	1973
Go::SfSfIntersector	1981
Go::IntersectorAlgPar	1263
Go::IntersectorFuncConst	1268
Go::Par0FuncIntersector	1514
Go::Par1FuncIntersector	1516
Go::Par2FuncIntersector	1519
Go::SfSelfIntersector	1976
Go::IntPtInfo	1272
Go::IntResultsModel	1278
Go::IntResultsCompCv	1274
Go::IntResultsSfModel	1282
Go::InverseFactorial < T, N >	1287
Go::IsogeometricBlock	1291
Go::IsogeometricSfBlock	1295
Go::IsogeometricVolBlock	1301
Go::IsogeometricModel	1294
Go::IsogeometricSfModel	1299
Go::IsogeometricVolModel	1305
RBD_COMMON::Janitor	1312
NEWMAT::BaseMatrix	413
NEWMAT::GeneralMatrix	1003
NEWMAT::BandLUMatrix	397
NEWMAT::BandMatrix	400
NEWMAT::LowerBandMatrix	1359
NEWMAT::UpperBandMatrix	2395
NEWMAT::CroutMatrix	651
NEWMAT::DiagonalMatrix	759
NEWMAT::IdentityMatrix	1158
NEWMAT::LowerTriangularMatrix	1364
NEWMAT::Matrix	1420
NEWMAT::ColumnVector	555
NEWMAT::nricMatrix	1502
NEWMAT::RowVector	1946
NEWMAT::SymmetricBandMatrix	2312
NEWMAT::SymmetricMatrix	2320
NEWMAT::UpperTriangularMatrix	2399
NEWMAT::GenericMatrix	1024
NEWMAT::LinearEquationSolver	1335
NEWMAT::MultipliedMatrix	1476
NEWMAT::AddedMatrix	360

NEWMAT::SPMatrix	2218
NEWMAT::SubtractedMatrix	2231
NEWMAT::ConcatenatedMatrix	608
NEWMAT::StackedMatrix	2221
NEWMAT::KPMatrix	1314
NEWMAT::SolvedMatrix	2071
NEWMAT::NegatedMatrix	1480
NEWMAT::ColedMatrix	553
NEWMAT::DiagedMatrix	757
NEWMAT::GetSubMatrix	1037
NEWMAT::InvertedMatrix	1288
NEWMAT::MatedMatrix	1416
NEWMAT::ReversedMatrix	1937
NEWMAT::RowedMatrix	1943
NEWMAT::TransposedMatrix	2370
NEWMAT::ReturnMatrixX	1935
NEWMAT::ShiftedMatrix	1993
NEWMAT::NegShiftedMatrix	1483
NEWMAT::ScaledMatrix	1954
NEWMAT::SimpleIntArray	1998
LL_D_FI	
GARCH11_LL	1002
NEWMAT::LL_D_FI	1347
Go::LockedDirDistFunc	1350
NEWMAT::LogAndSign	1351
Go::Loop	1354
Go::LRSplineEvalGrid	1376
Go::LRSurfApprox	1398
Go::LRSurfSmoothLS	1405
Go::LSSmoothData	1408
Go::MarchPoint	1414
material_appearance	1418
NEWMAT::MatrixBandWidth	1425
NEWMAT::MatrixInput	1431
MatrixRowCol	1434
MatrixCol	1427
MatrixColX	1429
MatrixRow	1432
NEWMAT::MatrixType	1442
Go::MatrixXD< T, Dim >	1448
Go::MatrixXD< double, 2 >	1448
Go::MatrixXD< double, 3 >	1448
Go::MatrixXD< double, Dim >	1448
Go::Mesh2DIterator	1457
Go::ModelQuality	1462
Go::FaceSetQuality	861
Go::ModelRepair	1470
Go::FaceSetRepair	869
MultiDijkstra	1472
NEWMAT::MultiRadixCounter	1479
hetriang::Node	1489
RBD_COMMON::OneDimSolve	1508
PathType	1676
Go::Point	1693
Go::PointOnCurve	1709
Go::PointOnEdge	1711
Go::PointSequence	1712

PrBiCGStab	1715
PrCellStructure	1718
PrCG	1722
Go::preEvaluationSf	1724
Go::preEvaluationVol	1726
PrFilterbank	1738
PrFaber_F	1731
PrPrewavelet_F	1809
PrHeap	1740
PrintCounter	1743
PrMatrix	1752
PrMat	1749
PrMatSparse	1754
PrNestedNode	1758
PrNestedTriangulation	1762
PrTriangulation_NT	1849
PrNode	1765
PrOrganizedPoints	1780
Go::ftPointSet	937
PrExplicitConnectivity	1728
PrLevelTriangulation_OP	1744
PrPlanarGraph_OP	1805
PrRectangularGrid_OP	1826
PrSubTriangulation	1834
PrTriangulation_OP	1853
PrFastUnorganized_OP	1733
PrUnorganized_OP	1860
PrParametrizeBdy	1786
PrParametrizeInt	1790
PrPrmEDDHLS	1811
PrPrmExperimental	1813
PrPrmLeastSquare	1815
PrPrmMeanValue	1816
PrPrmShpPres	1818
PrPrmSurface	1820
PrPrmSymMeanValue	1821
PrPrmUniform	1823
PrPrmWachspress	1824
PrParametrizeMesh	1795
PrParamTriangulation	1798
PrPGNode	1802
PrThin	1839
PrThreshold	1842
PrTriangle	1845
PrVec	1864
QDialog	
gvResolutionDialog	1107
QGLWidget	
gvView	1112
QObject	
CurveResolutionSheet	704
gvActionSheet	1048
gvGroupPropertySheet	1079
ParametricSurfacePropertySheet	1598
PointCloudPropertySheet	1707
RectangularSurfacePropertySheet	1886
RectangularVolumePropertySheet	1890

SplineCurvePropertySheet	2138
SurfaceResolutionSheet	2308
Go::QualityResults	1874
QWidget	
gvApplication	1049
gvApplicationVolAndLR	1055
gvObjectList	1085
R1_Col_I_D	
Model_3pe	1460
NEWMAT::R1_Col_I_D	1876
R1_R1	
Cube	655
RBD_COMMON::R1_R1	1878
rank_info	1882
Go::Rational	1882
Go::raw_pointer_comp< T >	1885
RectMatrixRowCol	1912
RectMatrixCol	1906
RectMatrixDiag	1908
RectMatrixRow	1910
Go::LRSplineSurface::Refinement2D	1915
Go::RegistrationInput	1916
Go::RegistrationResult	1919
Go::Registrar< T >	1921
Go::RegularizeFace	1922
Go::RegularizeFaceSet	1925
Go::RotatedBox	1939
Go::RotationInfo	1942
Go::SamplePointData	1952
Go::ScratchVect< T, N >	1957
Go::SecondOrderProperties	1961
Go::CompositeModelFileHandler::sfcvinfo	1965
Go::sideConstraint	1996
Go::sideConstraintSet	1997
Go::SingBox	2001
Go::SingularityInfo	2004
Go::SingUnion	2009
SISLbox	2010
SISLCurve	2011
SISLdir	2013
SISLEdge	2014
SISLIntcurve	2015
SISLIntdat	2017
SISLIntlist	2019
SISLIntpt	2020
SISLIntsurf	2024
SISLObject	2025
SISLPoint	2027
SISLPtedge	2028
SISLSurf	2029
SISLTrack	2032
SISLTrimpar	2034
Go::SmoothCurve	2034
Go::SmoothCurveSet	2037
Go::SmoothSurf	2039
Go::SmoothSurfSet	2050
Go::SmoothVolume	2059
Go::SolveCG	2064

Go::SolveBCG	2063
Go::SolveCGCO	2070
Go::Streamable	2224
Go::BsplineBasis	524
Go::GeomObject	1031
Go::LineCloud	1337
Go::ParamCurve	1573
Go::BoundedCurve	485
Go::CurveOnSurface	675
Go::CurveOnVolume	691
Go::ElementaryCurve	805
Go::Circle	541
Go::Ellipse	817
Go::Hyperbola	1147
Go::Line	1322
Go::Parabola	1522
Go::SplineCurve	2113
Go::ParamSurface	1631
Go::BoundedSurface	494
Go::CompositeSurface	595
Go::ElementarySurface	808
Go::Cone	613
Go::Cylinder	716
Go::Disc	771
Go::Plane	1677
Go::Sphere	2077
Go::Torus	2328
Go::LRSplineSurface	1379
Go::SplineSurface	2145
Go::SurfaceOfLinearExtrusion	2267
Go::SurfaceOfRevolution	2279
Go::SurfaceOnVolume	2291
Go::ParamVolume	1670
Go::ElementaryVolume	814
Go::ConeVolume	626
Go::CylinderVolume	732
Go::Parallelepiped	1532
Go::SphereVolume	2092
Go::TorusVolume	2341
Go::SplineVolume	2193
Go::PointCloud< Dim >	1702
Go::RectGrid	1899
Go::PointCloud< 3 >	1702
Go::LRBSpline2D	1368
Go::Mesh2D	1453
Go::ObjectHeader	1505
Go::boxStructuring::SubSurfaceBoundingBox	2228
Go::LRSplineUtils::support_compare	2233
Go::SurfaceAssembly	2234
Go::boxStructuring::SurfaceData	2236
Go::SweepSurfaceCreator	2309
Go::SweepVolumeCreator	2311
NEWMAT::SymmetricEigenAnalysis	2319
Go::Tessellator	2324
Go::CurveTessellator	705
Go::LineCloudTessellator	1342
Go::NoopTessellator	1496

Go::ParametricSurfaceTesselator	1600
Go::RectangularSurfaceTesselator	1887
Go::RectangularVolumeTesselator	1892
Go::RectGridTesselator	1904
TestClass	2325
Go::TestInDomain	2326
time_lapse	2327
Go::tpEdge	2349
Go::tpFace	2355
Go::tpMarchPoint	2357
Go::tpTableEntry< edgeType >	2359
Go::tpTolerances	2360
Go::tpTopologicalInfo	2362
Go::tpTopologyTable< edgeType, faceType >	2363
RBD_COMMON::Tracer	2368
Go::Triangle	2372
hetriang::Triangulation	2373
hed::Triangulation	2376
Go::ttlPoint	2384
hed::TTLtraits	2386
hetriang::TTLtraits	2389
UnfNodeType	2393
Go::CoonsPatchGen::UnknownError	2394
vector3t< T >	2403
Go::Vertex	2413
Go::VolumeAdjacency	2429
Go::VolumeAdjacencyInfo	2431
Go::Zero_Parameter_Span_Error	2452

Chapter 25

Class Index

25.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Go::AdaptCurve	357
NEWMAT::AddedMatrix	360
Go::AdjacencyInfo	
Struct to store information about adjacency relations between two faces	363
Go::Alg2DElem	365
Go::Alg3DElem	366
Go::AlgObj2DInt	
Class for 2-dimensional algebraic intersection objects	368
Go::AlgObj3DInt	370
Go::AlgObjectInt	374
Go::ApproxCrvToSeqs	375
Go::ApproxCurve	378
Go::ApproxSurf	382
Go::Array< T, Dim >	386
NEWMAT::ArrayLengthSpecifier	394
RBD_COMMON::Bad_alloc	395
NEWMAT::BandLUMatrix	397
NEWMAT::BandMatrix	400
Go::BaryCoordSystem< Dim >	407
Go::BaryCoordSystemTriangle3D	409
RBD_COMMON::BaseException	411
NEWMAT::BaseMatrix	413
Go::BaseSurface	425
Go::BasisDerivs	425
Go::BasisDerivs2	427
Go::BasisDerivsSf	430
Go::BasisDerivsSf2	431
Go::BasisPts	433
Go::BasisPtsSf	434
Go::BdCondFunctor	436
Go::BernsteinMulti	436
Go::BernsteinPoly	446
Go::BernsteinTetrahedralPoly	452
Go::BernsteinTriangularPoly	457
Go::BezierTriangle< N >	
Not documented	462

Go::Binomial	464
Go::BlockBoundaryCondition	466
Go::BlockPointBdCond	469
Go::BlockSolution	470
Go::Body	
A boundary represented solid	474
bool	479
Go::BoundaryFunctionInt	480
Go::BoundaryGeomInt	482
Go::BoundaryIntersectionData	484
Go::BoundedCurve	
A bounded curve. Both parameter values and end points may be given to define the boundaries. Assuming that both points prefer parameter, or both points prefer points. Typically used to bound infinite curves, for instance lines	485
Go::BoundedSurface	494
Go::BoundingBox	515
Go::boxStructuring::BoundingBoxStructure	519
Go::LRSplineSurface::BSKey	522
Go::BsplineBasis	524
Go::CachedInterval	
Helper struct for saving bracketed bounds of influence areas	536
NEWMAT::CannotBuildException	538
Go::CellDivision	539
Go::Circle	
Class that represents a circle. It is a subclass of ElementaryCurve and thus has a parametrization	541
Go::ClosestPointCalculator	552
NEWMAT::ColedMatrix	553
NEWMAT::ColumnVector	555
Go::CompleteEdgeNet	
Complete the edge net of a SurfaceModel . Fetches the wire frame model corresponding to a surface model, and extends it such that the extended wire frame will become the wire frame model corresponding to a volume model have the given surface model as its outer boundary. The extension to the wireframe is represented as pairs of vertices where these vertices lie at the endpoints of the missing edges. NB! This solution is currently not expected to handle all configurations	560
Go::ComplexityInfo	561
Go::CompositeBox	564
Go::CompositeCurve	568
Go::CompositeModel	580
Go::CompositeModelFactory	588
Go::CompositeModelFileHandler	592
Go::CompositeSurface	595
NEWMAT::ConcatenatedMatrix	608
Go::ConcreteCreator< T >	612
Go::Cone	
Class that represents a cone. It is a subclass of ElementarySurface , and thus has a parametrization and is non-selfintersecting	613
Go::ConeVolume	
Class that represents a solid cone. It is a subclass of ElementaryVolume , and has a natural parametrization in terms of a radius u , an angle v , and distance w : $\mathbf{p}(u, v, w) = \mathbf{C} + u (R + w \tan \alpha)((\cos v) \mathbf{x} + (\sin v) \mathbf{y}) + w \mathbf{z}$, where \mathbf{C} is the cone apex, R is the radius when $w = 0$, α is the cone angle, and \mathbf{x} , \mathbf{y} and \mathbf{z} are the (local) axes. The parametrization is bounded by: $0 \leq u \leq 1$, $0 \leq v \leq 2\pi$, $-\infty < w < \infty$. The dimension is 3	626
Go::ConnectionFuncior	633
ControlWord	634
NEWMAT::ConvergenceException	637

Go::CoordinateSystem< Dim >	638
Defines a Cartesian coordinate system	
Go::CPUclock	641
A class for measuring CPU time in programs	
Go::Creator	642
Go::CrossesValue	643
IntersectionPoolUtils. predicate for STL function	
Go::CrossTangentOffset	644
Go::CrossTanOffDist	647
NEWMAT::CroutMatrix	651
Cube	655
Go::CurveBoundedDomain	655
Go::CurveLoop	662
Go::CurveModel	667
Go::CurveOnSurface	675
A curve living on a parametric surface. It either has got information about the curve in geometry space and in the parameter domain of the surface or the ability to compute the other representation given one. The curve may have information on whether it is a constant parameter or boundary curve on the surface	
Go::CurveOnVolume	691
A curve living on a parametric volume. It either has got information about the curve in geometry space and in the parameter domain of the volume or the ability to compute the other representation given one	
CurveResolutionSheet	704
Go::CurveTesselator	705
Go::CvCvIntersector	707
Class that performs intersection between two parametric curves	
Go::CvPtIntersector	711
Go::cvSetConstraint	714
Go::Cylinder	716
Class that represents a cylinder. It is a subclass of ElementarySurface , and thus has a parametrization and is non-selfintersecting	
Go::CylinderInt	729
Class for cylindrical algebraic intersection objects	
Go::CylinderVolume	732
Class that represents a solid cylinder, maybe with empty interior like a straight tube. It is a subclass of ElementaryVolume , and has a natural parametrization in terms of a radius u , an angle v , and distance w : $\mathbf{p}(u, v, w) = \mathbf{C} + u(\cos(v) \mathbf{x} + \sin(v) \mathbf{y}) + w \mathbf{z}$, where \mathbf{C} is a position vector, and \mathbf{x} , \mathbf{y} and \mathbf{z} are the (local) axes. The parametrization is bounded by: $R \leq u \leq S$, $0 \leq v \leq 2\pi$, $-\infty < w < \infty$, where R and S are the inner and outer radius. The dimension is 3	
hed::Dart	739
Dart class for the half-edge data structure	
hetriang::Dart	742
Dart class for the half-edge data structure	
DataHandler	746
DataHandlerVolAndLR	749
DefaultDataHandler	750
Go::ParamSurface::degenerate_info	752
Degeneracy information regarding one boundary surface of the current surface	
Go::DegeneratedIntersectionCurve	754
IntersectionCurve that is degenerated into a single point	
NEWMAT::DiagedMatrix	757
NEWMAT::DiagonalMatrix	759
Dijkstra	764
Go::DirectionCone	768

Go::Disc	
	Class that represents a circular disc. It is a subclass of ElementarySurface , and has a natural parametrization by polar coordinates in terms of a radius r and angle v : $\mathbf{p}(r, v) = \mathbf{C} + r((\cos v) \mathbf{x} + (\sin v) \mathbf{y})$, where \mathbf{C} is the centre position vector and \mathbf{x} and \mathbf{y} are the (local) axes. The parametrization is bounded by: $0 \leq r \leq R$ and $0 \leq v \leq 2\pi$, where R is the disc radius. The dimension is 2 or 3
	771
Go::Domain	782
RBD_COMMON::Domain_error	785
Go::LRSplineSurface::double_pair_hash	786
hed::Edge	
	Edge class in the in the half-edge data structure
	787
hetriang::Edge	
	Edge class in the half-edge data structure
	790
EdgeType	792
Go::EdgeVertex	792
Go::Element2D	796
Go::ElementaryCurve	
	ElementaryCurve is a base class for elementary curves like lines and circles. Such curves have natural parametrizations and ElementaryCurve is therefore a subclass of ParamCurve
	805
Go::ElementarySurface	
	ElementarySurface is a base class for elementary surfaces like planes and cylinders. Such surfaces have natural parametrizations and ElementarySurface is therefore a subclass of ParamSurface . These surfaces are non-self-intersecting
	808
Go::ElementaryVolume	
	ElementaryVolume is a base class for elementary volumes like boxes and solid cylinders. Such volumes have natural parametrizations and ElementaryVolume is therefore a subclass of ParamVolume . These volumes are non-self-intersecting
	814
Go::LRSplineSurface::ElemKey	816
Go::Ellipse	
	Class that represents an ellipse. It is a subclass of ElementaryCurve and thus has a parametrization
	817
Go::EntityList	
	The entity number of all supported IGES entites
	827
Go::Streamable::EofException	829
Go::EvalCurve	829
Go::EvalCurveSet	833
Go::EvalFunctorCurve	836
Go::EvalFunctorSurface	839
Go::EvalOffsetSurface	842
Go::EvalParamCurve	846
Go::EvalSurface	849
Go::FaceAdjacency< edgeType, faceType >	
	Compute adjacency information for a set of surfaces
	852
Go::FaceConnectivity< edgeType >	857
Go::FaceConnectivityUtils< edgeType, faceType >	
	Utilities used in adjacency computations of face sets
	860
Go::FaceSetQuality	861
Go::FaceSetRepair	869
Go::Factorial< N >	
	Compile-time factorial calculations
	871
Go::Factorial< 1 >	872
Go::Factory	873
NEWMAT::FFT_Controller	874
NEWMAT::FindMaximum2	876
NEWMAT::FloatingPointPrecision	
	Floating point precision (type double)
	877
Go::ftCell	
	Helper class handling one of the cells in class CellDivision
	879

Go::ftCellInfo		
Function object for sorting ftCells	881
Go::ftChartSurface	883
Go::ftCurve	888
Go::ftCurveSegment	894
Go::ftEdge		
The ftEdge is a half-edge implementation of a topological data structure	899
Go::ftEdgeBase		
Base class for edges. Defines the interface used in topology analysis	910
Go::ftFaceBase	918
Go::ftGraphEdge	922
Go::ftGroupGeom		
A group of geometrical objects	924
Go::ftLine	926
Go::ftMessage	928
Go::ftPlanarGraph	930
Go::ftPlane	932
Go::ftPoint	934
Go::ftPointSet	937
Go::ftSamplePoint	945
Go::ftSearchNode	951
Go::ftSmoothSurf	952
Go::ftSSfEdge	956
Go::ftSurface	961
Go::ftSurfaceSet	978
Go::ftSurfaceSetPoint	985
Go::ftVolume		
A topological solid with a trivariate geometry description	988
Go::Fun2Fun< Functor >	996
Go::FunctionMinimizer< Functor >	997
GARCH11_LL	1002
NEWMAT::GeneralMatrix	1003
Go::GeneralMesh	1021
NEWMAT::GenericMatrix	1024
Go::GenericTriMesh	1027
Go::GeomObject	1031
Go::GeomObjectInt	1034
Go::GeoTol		
Class handling various tolerances	1034
NEWMAT::GetSubMatrix	1037
Go::go_iterator_traits< literator >	1041
Go::go_iterator_traits< const T * >	1042
Go::go_iterator_traits< T * >	1042
Go::GoTools		
Class containing some service functions for the GoTools "system"	1043
Go::GPos	1045
Go::GpuMatrix< T >	1046
gvActionSheet	1048
gvApplication	1049
gvApplicationVolAndLR	1055
gvCamera	1057
gvColor		
Represents a color by four floats, like in OpenGL	1063
gvCurvePaintable	1064
gvData	1067
gvGenericTriPaintable< FloatType >	1072
gvGenericTriQuadMesh< FloatType >	1074
gvGenericTriQuadPaintable< FloatType >	1076

gvGroup	1078
Structure used to store name and index of members of a group of objects	
gvGroupPropertySheet	1079
gvLineCloudPaintable	1081
gvNoopPaintable	1083
gvObjectList	1085
gvObserver	1087
gvPaintable	1088
gvPainter	
The gvPainter is responsible for painting the 3D scene	1092
gvParametricSurfacePaintable	1094
gvPointCloudPaintable	1096
gvPropertySheet	1100
gvQuadsPaintable	1101
gvRectangularSurfacePaintable	1103
gvRectangularVolumePaintable	1105
gvResolutionDialog	1107
gvStandardMouseHandler	1109
gvTexture	1110
gvView	1112
Handle< T >	
Template class for smart pointers. The actual class must inherit from HandleId	1122
HandleId	
Base class with reference counting for smart pointers	1126
HeapNode	1128
HeapNode2	1130
Go::HermiteAppC	1131
Go::HermiteApprEvalSurf	1133
Go::HermiteAppS	1135
Go::HermiteGrid1D	1136
Go::HermiteGrid1DMulti	1139
Go::HermiteGrid2D	1141
Go::HermiteInterpolator	1144
Go::Hyperbola	
Class that represents a hyperbola. It is a subclass of ElementaryCurve and thus has a parametrization	1147
Go::Identity	
Check coincidence	1157
NEWMAT::IdentityMatrix	1158
Go::IGESconverter	1162
Go::IGESdirentry	
Storage of all data contained in an IGES directory entity	1166
Go::IGESheader	
Storage of all data contained in the IGES header	1168
Go::ImplicitizeCurveAlgo	1172
Go::ImplicitizeCurveAndVectorAlgo	1174
Go::ImplicitizePointCloudAlgo	1177
Go::ImplicitizeSurfaceAlgo	1179
NEWMAT::IncompatibleDimensionsException	1182
NEWMAT::IndexException	1184
Go::IndexMesh2DIterator	1185
Go::IntCrvEvaluator	1187
NEWMAT::InternalException	1190
Go::InterpolatedIntersectionCurve	1192
Go::Interpolator	
Base class for spline interpolators or approximators	1196
Go::IntersectionCurve	1197
Go::IntersectionLink	1203

Go::IntersectionPoint	1205
Go::IntersectionPool	1224
Go::Intersector	1246
Go::Intersector2Obj	1256
Go::IntersectorAlgPar	1263
Go::IntersectorFuncConst	1268
Go::IntPtInfo	1272
Go::IntResultsCompCv	1274
Go::IntResultsModel	1278
Go::IntResultsSfModel	1282
RBD_COMMON::Invalid_argument	1286
Go::InverseFactorial< T, N >	1287
NEWMAT::InvertedMatrix	1288
Go::IsogeometricBlock	1291
Go::IsogeometricModel	1294
Go::IsogeometricSfBlock	1295
Go::IsogeometricSfModel	1299
Go::IsogeometricVolBlock	1301
Go::IsogeometricVolModel	1305
Go::IsoparametricIntersectionCurve	1307
RBD_COMMON::Janitor	1312
NEWMAT::KPMatrix	1314
RBD_COMMON::Length_error	1317
Go::LiftCurve	1319
Go::Line	
Class that represents a line. It is a subclass of ElementaryCurve and thus has a parametrization	1322
Go::Line2DInt	
Class representing an algebraic line in 2-dimensional space	1332
NEWMAT::LinearEquationSolver	1335
Go::LineCloud	1337
Go::LineCloudTesselator	1342
Go::LineStrip	1344
NEWMAT::LL_D_FI	1347
LoadAndStoreFlag	1349
Go::LockedDirDistFunc	1350
NEWMAT::LogAndSign	1351
RBD_COMMON::Logic_error	1353
Go::Loop	
Primarily a loop connected to a face in a boundary represented solid or face set. May also be used to represent a general closed sequence of edges	1354
NEWMAT::LowerBandMatrix	1359
NEWMAT::LowerTriangularMatrix	1364
Go::LRBSpline2D	1368
Go::LRSplineEvalGrid	1376
Go::LRSplineSurface	1379
Go::LRSurfApprox	1398
Go::LRSurfSmoothLS	1405
Go::LSSmoothData	1408
Go::MarchPoint	
Helper class for marching	1414
NEWMAT::MatedMatrix	1416
material_appearance	1418
NEWMAT::Matrix	1420
NEWMAT::MatrixBandWidth	1425
MatrixCol	1427
MatrixColX	1429
NEWMAT::MatrixInput	1431
MatrixRow	1432

MatrixRowCol	1434
NEWMAT::MatrixType	1442
Go::MatrixXD< T, Dim >	1448
Go::Mesh2D	1453
Go::Mesh2DIterator	1457
NEWMAT::MLE_D_FI	1459
Model_3pe	1460
Go::ModelQuality	1462
Go::ModelRepair	1470
MultiDijkstra	1472
NEWMAT::MultipliedMatrix	1476
NEWMAT::MultiRadixCounter	1479
NEWMAT::NegatedMatrix	1480
NEWMAT::NegShiftedMatrix	1483
hed::Node	
Node class for data structures (Inherits from HandleId)	1486
hetriang::Node	
Node class in the half-edge data structure	1489
Go::NonEvaluableIntersectionCurve	
IntersectionCurve that cannot be evaluated	1490
NEWMAT::NonLinearLeastSquares	1494
Go::NoopTesselator	1496
NEWMAT::NotDefinedException	1497
NEWMAT::NotSquareException	1499
NEWMAT::NPDException	1500
NEWMAT::nricMatrix	1502
Go::ObjectHeader	1505
RBD_COMMON::OneDimSolve	1508
RBD_COMMON::Out_of_range	1509
RBD_COMMON::Overflow_error	1511
NEWMAT::OverflowException	1512
Go::Par0FuncIntersector	
This class is performing intersections between two constants	1514
Go::Par1FuncIntersector	1516
Go::Par2FuncIntersector	1519
Go::Parabola	
Class that represents a parabola. It is a subclass of ElementaryCurve and thus has a parametrization	1522
Go::Parallelepiped	
Class that represents a solid parallelepiped. It is a subclass of ElementaryVolume , and thus has a parametrization	1532
Go::Param0FunctionInt	1539
Go::Param1FunctionInt	1548
Go::Param2FunctionInt	1560
Go::ParamCurve	1573
Go::ParamCurveInt	1583
ParametricSurfacePropertySheet	1598
Go::ParametricSurfaceTesselator	1600
Go::ParamFunctionInt	1602
Go::ParamGeomInt	1606
Go::ParamObjectInt	1612
Go::ParamPointInt	1621
Go::ParamSurface	1631
Go::ParamSurfaceInt	1647
Go::ParamVolume	1670
PathType	1676

Go::Plane	
Class that represents a plane. It is a subclass of ElementarySurface , and thus has a parametrization and is non-selfintersecting	1677
Go::PlaneInt	
Class representing planar algebraic intersection objects	1690
Go::Point	1693
Go::PointCloud< Dim >	1702
PointCloudPropertySheet	1707
Go::PointOnCurve	1709
Go::PointOnEdge	1711
Go::PointSequence	1712
PrBiCGStab	1715
PrCellStructure	1718
PrCG	1722
Go::preEvaluationSf	1724
Go::preEvaluationVol	1726
PrExplicitConnectivity	1728
PrFaber_F	1731
PrFastUnorganized_OP	1733
PrFilterbank	1738
PrHeap	1740
PrintCounter	1743
PrLevelTriangulation_OP	1744
PrMat	
This class implements a matrix	1749
PrMatrix	
This class implements a matrix	1752
PrMatSparse	1754
PrNestedNode	1758
PrNestedTriangulation	1762
PrNode	1765
NEWMAT::ProgramException	1768
Go::ProjectCurve	1770
Go::ProjectCurveAndCrossTan	1773
Go::ProjectIntersectionCurve	1777
PrOrganizedPoints	1780
PrParametrizeBdy	1786
PrParametrizeInt	1790
PrParametrizeMesh	1795
PrParamTriangulation	1798
PrPGNode	1802
PrPlanarGraph_OP	1805
PrPrewavelet_F	1809
PrPrmEDDHLS	1811
PrPrmExperimental	1813
PrPrmLeastSquare	1815
PrPrmMeanValue	1816
PrPrmShpPres	1818
PrPrmSurface	1820
PrPrmSymMeanValue	1821
PrPrmUniform	1823
PrPrmWachspress	1824
PrRectangularGrid_OP	1826
PrSubTriangulation	1834
PrThin	1839
PrThreshold	1842
PrTriangle	1845
PrTriangulation_NT	1849

PrTriangulation_OP	1853
PrUnorganized_OP	1860
PrVec	1864
Go::PtPtIntersector	
This class performs intersection between two points	1867
Go::QuadMesh	1871
Go::QualityResults	1874
NEWMAT::R1_Col_I_D	1876
RBD_COMMON::R1_R1	1878
RBD_COMMON::Range_error	1880
rank_info	1882
Go::Rational	1882
Go::raw_pointer_comp< T >	1885
RectangularSurfacePropertySheet	1886
Go::RectangularSurfaceTesselator	1887
RectangularVolumePropertySheet	1890
Go::RectangularVolumeTesselator	1892
Go::RectDomain	1894
Go::RectGrid	1899
Go::RectGridTesselator	1904
RectMatrixCol	1906
RectMatrixDiag	1908
RectMatrixRow	1910
RectMatrixRowCol	1912
Go::LRSplineSurface::Refinement2D	1915
Go::RegistrationInput	
Struct for input to registration, either raw, fine or combined	1916
Go::RegistrationResult	
Struct for result from registration process, either raw, fine or combined	1919
Go::Registrar< T >	1921
Go::RegularizeFace	
Split one face into a number of 4-sided domains without inner trimming. This class is intended for use in block structuring. One face with possible inner and outer trimming is split according to certain rules to result in a face set with 4 sided faces although faces with less than 4 sides can occur. A side is defined as a piece of the face boundary between two corners or between vertices where there are more than one adjacent face. The trimmed surfaces being output from this class can later be approximated by spline surfaces. The splitting procedure is recursive	1922
Go::RegularizeFaceSet	1925
Go::RegularMesh	1927
Go::RegularVolMesh	1931
NEWMAT::ReturnMatrixX	1935
NEWMAT::ReversedMatrix	1937
Go::RotatedBox	1939
Go::RotationInfo	1942
NEWMAT::RowedMatrix	1943
NEWMAT::RowVector	1946
RBD_COMMON::Runtime_error	1950
Go::SamplePointData	
Sample data related to one face. The struct contains one point in a point set with information about position, associated surface normal and curvature. Points lying on the face boundary know about its associated edge. If the surface normal and curvature information regarding boundary points is not unique, the associated data is set to be equal to MAX_DOUBLE	1952
NEWMAT::ScaledMatrix	1954
Go::ScratchVect< T, N >	1957
Go::SecondOrderProperties	1961
Go::SfBoundaryCondition	1963
Go::CompositeModelFileHandler::sfcvinfo	1965
Go::SfCvIntersector	1967

Go::SfPointBdCond	1971
Go::SfPtIntersector	1973
Go::SfSelfIntersector	
This class finds self-intersections for a parametric surface	1976
Go::SfSfIntersector	
This class performs intersection between two parametric surfaces	1981
Go::SfSolution	1987
NEWMAT::ShiftedMatrix	1993
Go::sideConstraint	1996
Go::sideConstraintSet	1997
NEWMAT::SimpleIntArray	1998
Go::SingBox	2001
NEWMAT::SingularException	2002
Go::SingularityInfo	2004
Go::SingUnion	2009
SISLbox	2010
SISLCurve	2011
SISLdir	2013
SISLEdge	2014
SISLIntcurve	2015
SISLIntdat	2017
SISLIntlist	2019
SISLIntpt	2020
SISLIntsurf	2024
SISLObject	2025
SISLPoint	2027
SISLPtedge	2028
SISLSurf	2029
SISLTrack	2032
SISLTrimpar	2034
Go::SmoothCurve	2034
Go::SmoothCurveSet	2037
Go::SmoothSurf	2039
Go::SmoothSurfSet	2050
Go::SmoothTransition	2055
Go::SmoothVolume	
This class modifies a NURBS volume with respect to smoothness, editing constraints and boundary conditions. Specified coefficients are adjusted to minimize a functional combining the different modification conditions	2059
RBD_COMMON::SolutionException	2061
Go::SolveBCG	2063
Go::SolveCG	2064
Go::SolveCGCO	2070
NEWMAT::SolvedMatrix	2071
Go::SpaceIntCrv	2074
Go::Sphere	
Class that represents a sphere. It is a subclass of ElementarySurface , and thus has a parametrization and is non-selfintersecting	2077
Go::SphereInt	
Class representing spherical algebraic intersection objects	2090
Go::SphereVolume	
Class that represents a solid sphere. It is a subclass of ElementaryVolume , and thus has a parametrization	2092
Go::Spline1FunctionInt	2099
Go::Spline2FunctionInt	2104
Go::SplineApproximator	2110

Go::SplineCurve	
SplineCurve	provides methodes for storing, reading and manipulating rational and non-rational
B-spline curves	2113
Go::SplineCurveInt	
Class representing the "intersection object" of a spline curve	2131
SplineCurvePropertySheet	2138
Go::SplineInterpolator	2140
Go::SplineSurface	
SplineSurface	provides methodes for storing, reading and manipulating rational and non-rational
B-spline surfaces	2145
Go::SplineSurfaceInt	2182
Go::SplineVolume	
SplineVolume	provides methodes for storing, reading and manipulating rational and non-rational
B-spline volumes	2193
NEWMAT::SPMatrix	2218
NEWMAT::StackedMatrix	2221
Go::Streamable	2224
NEWMAT::SubMatrixDimensionException	2226
Go::boxStructuring::SubSurfaceBoundingBox	2228
NEWMAT::SubtractedMatrix	2231
Go::LRSplineUtils::support_compare	2233
Go::SurfaceAssembly	2234
Go::boxStructuring::SurfaceData	2236
Go::SurfaceModel	2238
Go::SurfaceOfLinearExtrusion	2267
Go::SurfaceOfRevolution	
Class that represents a surface of revolution. A SurfaceOfRevolution is swept out by a SplineCurve that is rotated around an axis with a complete revolution, and is thereby a parametric surface	2279
Go::SurfaceOnVolume	
A surface living on a parametric volume. It either has got information about the surface in geometry space and in the parameter domain of the volume or both. The surface may have information on whether it is a constant parameter or boundary surface on the volume	2291
SurfaceResolutionSheet	2308
Go::SweepSurfaceCreator	
Functionality to create a spline surface by linear or rotational sweep	2309
Go::SweepVolumeCreator	
Class with static methods for volume creation by sweep methods	2311
NEWMAT::SymmetricBandMatrix	2312
NEWMAT::SymmetricEigenAnalysis	2319
NEWMAT::SymmetricMatrix	2320
Go::Tesselator	2324
TestClass	2325
Go::TestInDomain	
IntersectionPoolUtils. predicate for STL function	2326
time_lapse	2327
Go::Torus	
Class that represents a torus. It is a subclass of ElementarySurface , and thus has a parametrization. A torus may be degenerate. Then the minor radius is greater than the major radius, and it is in principle selfintersecting	2328
Go::TorusVolume	
Class that represents a solid torus, maybe with empty interior like a circular pipe, and/or a section (not full revolution along the main axis). It is a subclass of ElementaryVolume , and has a natural parametrization in terms of a radius u and two angles v and w : $\mathbf{p}(u, v, w) = \mathbf{C} + (R + u \cos w)((\cos v) \mathbf{x} + (\sin v) \mathbf{y}) + (u \sin w) \mathbf{z}$, where \mathbf{C} is a position vector, R is the major radius, and \mathbf{x} , \mathbf{y} and \mathbf{z} are the (local) axes. The parametrization for a full solid torus is bounded by: $0 \leq u \leq r$, $0 \leq v, w \leq 2\pi$, where r is the minor axis. A bended pipe will have a positive minimal limit for u , while a torus segment will have other limits for v . The dimension is 3	2341

Go::tpEdge	2349
Go::tpFace	2355
Go::tpMarchPoint	
Helper class for marching	2357
Go::tpTableEntry< edgeType >	2359
Go::tpTolerances	2360
Go::tpTopologicalInfo	2362
Go::tpTopologyTable< edgeType, faceType >	2363
RBD_COMMON::Tracer	2368
NEWMAT::TransposedMatrix	2370
Go::Triangle	2372
hetriang::Triangulation	
Triangulation class for the half-edge data structure with adaption to TTL	2373
hed::Triangulation	
Triangulation class for the half-edge data structure with adaption to TTL	2376
Go::TrimCurve	2381
Go::ttlPoint	2384
hed::TTLtraits	
Traits class (static struct) for the half-edge data structure	2386
hetriang::TTLtraits	
Traits class (static struct) for the half-edge data structure	2389
UnfNodeType	2393
Go::CoonsPatchGen::UnKnownError	
Exception class	2394
NEWMAT::UpperBandMatrix	2395
NEWMAT::UpperTriangularMatrix	2399
vector3t< T >	2403
NEWMAT::VectorException	2411
Go::Vertex	
The vertex class represents the vertex entity in a boundary represented solid or face set	2413
Go::VolBoundaryCondition	2420
Go::VolPointBdCond	2422
Go::VolSolution	2424
Go::VolumeAdjacency	
Adjacency analysis of volume models	2429
Go::VolumeAdjacencyInfo	
Struct to store information about adjacency relations between two bodies	2431
Go::VolumeModel	
A set of volumes including topology information	2434
Go::VolumeModelFileHandler	2444
Go::VolumeParameterCurve	
An evaluator based curve representing the parameter domain curve in a given volume which represents the same curve as a given space curve. Project a geometry curve into the parameter domain of a volume	2445
Go::VolumeSpaceCurve	
An evaluator based curve representing the space curve corresponding to parameter domain curve in a given volume. Compute the space curve corresponding to a curve in the parameter domain of a volume	2449
Go::Zero_Parameter_Span_Error	
Error object used internally in IntersectionCurve	2452

Chapter 26

File Index

26.1 File List

Here is a list of all files with brief descriptions:

compositemodel/include/GoTools/compositemodel/AdaptSurface.h	2453
compositemodel/include/GoTools/compositemodel/Body.h	2454
compositemodel/include/GoTools/compositemodel/CellDivision.h	2455
compositemodel/include/GoTools/compositemodel/cmUtils.h	2457
compositemodel/include/GoTools/compositemodel/CompleteEdgeNet.h	2458
compositemodel/include/GoTools/compositemodel/CompositeCurve.h	2459
compositemodel/include/GoTools/compositemodel/compositemodel-doxymain.h	2460
compositemodel/include/GoTools/compositemodel/CompositeModel.h	2460
compositemodel/include/GoTools/compositemodel/CompositeModelFactory.h	2461
compositemodel/include/GoTools/compositemodel/CompositeModelFileHandler.h	2462
compositemodel/include/GoTools/compositemodel/CurveModel.h	2463
compositemodel/include/GoTools/compositemodel/EdgeVertex.h	2464
compositemodel/include/GoTools/compositemodel/EvalOffsetSurface.h	2465
compositemodel/include/GoTools/compositemodel/examples_compositemodel_doxxygen.h	2465
compositemodel/include/GoTools/compositemodel/FaceUtilities.h	2465
compositemodel/include/GoTools/compositemodel/ftChartSurface.h	2467
compositemodel/include/GoTools/compositemodel/ftCurve.h	2467
compositemodel/include/GoTools/compositemodel/ftEdge.h	2469
compositemodel/include/GoTools/compositemodel/ftEdgeBase.h	2470
compositemodel/include/GoTools/compositemodel/ftFaceBase.h	2471
compositemodel/include/GoTools/compositemodel/ftLine.h	2471
compositemodel/include/GoTools/compositemodel/ftMessage.h	2473
compositemodel/include/GoTools/compositemodel/ftPlanarGraph.h	2474
compositemodel/include/GoTools/compositemodel/ftPlane.h	2476
compositemodel/include/GoTools/compositemodel/ftPoint.h	2477
compositemodel/include/GoTools/compositemodel/ftPointSet.h	2478
compositemodel/include/GoTools/compositemodel/ftSmoothSurf.h	2479
compositemodel/include/GoTools/compositemodel/ftSSfEdge.h	2479
compositemodel/include/GoTools/compositemodel/ftSurface.h	2480
compositemodel/include/GoTools/compositemodel/ftSurfaceSet.h	2481
compositemodel/include/GoTools/compositemodel/ftSurfaceSetPoint.h	2483
compositemodel/include/GoTools/compositemodel/HermiteApprEvalSurf.h	2484
compositemodel/include/GoTools/compositemodel/IntResultsCompCv.h	2484
compositemodel/include/GoTools/compositemodel/IntResultsModel.h	2485
compositemodel/include/GoTools/compositemodel/IntResultsSfModel.h	2487

compositemodel/include/GoTools/compositemodel/Loop.h	2487
compositemodel/include/GoTools/compositemodel/OffsetSurfaceUtils.h	2488
compositemodel/include/GoTools/compositemodel/Path.h	2489
compositemodel/include/GoTools/compositemodel/PointOnEdge.h	2490
compositemodel/include/GoTools/compositemodel/PointSetApp.h	2491
compositemodel/include/GoTools/compositemodel/RegularizeFace.h	2491
compositemodel/include/GoTools/compositemodel/RegularizeFaceSet.h	2492
compositemodel/include/GoTools/compositemodel/RegularizeUtils.h	2493
compositemodel/include/GoTools/compositemodel/SplitModelUtils.h	2494
compositemodel/include/GoTools/compositemodel/SurfaceModel.h	2495
compositemodel/include/GoTools/compositemodel/SurfaceModelUtils.h	2496
compositemodel/include/GoTools/compositemodel/ttlDart.h	2497
compositemodel/include/GoTools/compositemodel/ttlPoint.h	2498
compositemodel/include/GoTools/compositemodel/ttlTraits.h	2499
compositemodel/include/GoTools/compositemodel/ttlTriang.h	2500
compositemodel/include/GoTools/compositemodel/Vertex.h	2501
doc/html/_2dpoly_for_s2m_8h.js	2502
doc/html/_adapt_surface_8h.js	2502
doc/html/_array_8h.js	2502
doc/html/_bary_coord_system_8h.js	2503
doc/html/_bd_condition_type_8h.js	2503
doc/html/_bernstein_multi_8h.js	2504
doc/html/_bernstein_poly_8h.js	2504
doc/html/_bernstein_tetrahedral_poly_8h.js	2505
doc/html/_bernstein_triangular_poly_8h.js	2505
doc/html/_bernstein_utils_8h.js	2506
doc/html/_bezier_triangle_8h.js	2506
doc/html/_bounded_utils_8h.js	2507
doc/html/_bounding_box_8h.js	2507
doc/html/_bspline_basis_8h.js	2507
doc/html/_class_type_8h.js	2508
doc/html/_closest_point_8h.js	2508
doc/html/_closest_point_utils_8h.js	2508
doc/html/_coincidence_8h.js	2509
doc/html/_composite_model_8h.js	2510
doc/html/_constraint_definitions_8h.js	2510
doc/html/_coons_patch_gen_8h.js	2511
doc/html/_coons_patch_volume_gen_8h.js	2511
doc/html/_creators_offset_utils_8h.js	2512
doc/html/_creators_utils_8h.js	2512
doc/html/_curvature_8h.js	2512
doc/html/_curvature_analysis_8h.js	2513
doc/html/_curvature_utils_8h.js	2513
doc/html/_curve_creators_8h.js	2514
doc/html/_curve_interpolator_8h.js	2514
doc/html/_curve_loop_8h.js	2515
doc/html/_direction2_d_8h.js	2515
doc/html/_face_utilities_8h.js	2516
doc/html/_factory_8h.js	2516
doc/html/_gap_removal_8h.js	2516
doc/html/_gap_removal_volume_8h.js	2517
doc/html/_general_function_minimizer_8h.js	2517
doc/html/_general_function_minimizer_implementation_8h.js	2518
doc/html/_geom_object_8h.js	2518
doc/html/_geometry_tools_8h.js	2519
doc/html/_go_intersections_8h.js	2519
doc/html/_go_read_write_8cpp.js	2520
doc/html/_go_read_write_8h.js	2520

doc/html/_go_tools_version_8h.js	2521
doc/html/_hahns_surface_gen_8h.js	2521
doc/html/_he_triang_8h.js	2521
doc/html/_i_g_e_sconverter_8h.js	2522
doc/html/_implicit_utils_8h.js	2523
doc/html/_int_results_model_8h.js	2523
doc/html/_integrate_8h.js	2524
doc/html/_integration_8h.js	2524
doc/html/_intersection_curve_8h.js	2524
doc/html/_intersection_interface_8h.js	2525
doc/html/_intersection_point_utils_8h.js	2526
doc/html/_intersection_pool_utils_8h.js	2526
doc/html/_intersection_utils_8h.js	2527
doc/html/_l_r_approx_app_8h.js	2528
doc/html/_l_r_b_spline2_d_8h.js	2528
doc/html/_l_r_b_spline2_d_utils_8h.js	2528
doc/html/_l_r_benchmark_utils_8h.js	2529
doc/html/_l_r_spline_eval_grid_8h.js	2529
doc/html/_l_r_spline_m_b_a_8h.js	2530
doc/html/_l_r_spline_plot_utils_8h.js	2530
doc/html/_l_r_spline_utils_8h.js	2530
doc/html/_l_r_surf_smooth_l_s_8h.js	2531
doc/html/_l_r_surf_stitch_8h.js	2532
doc/html/_l_u_decomp_8h.js	2532
doc/html/_l_u_decomp_implementation_8h.js	2533
doc/html/_lin_dep_utils_8h.js	2533
doc/html/_link_type_8h.js	2534
doc/html/_loft_surface_creator_8h.js	2534
doc/html/_loft_volume_creator_8h.js	2534
doc/html/_loop_utils_8h.js	2535
doc/html/_matrix_x_d_8h.js	2535
doc/html/_mesh2_d_8h.js	2536
doc/html/_mesh2_d_utils_8h.js	2536
doc/html/_modify_surf_8h.js	2537
doc/html/_offset_surface_utils_8h.js	2537
doc/html/_param_surface_8h.js	2538
doc/html/_path_8h.js	2539
doc/html/_point_8h.js	2539
doc/html/_point_cloud_8h.js	2540
doc/html/_point_sequence_8h.js	2540
doc/html/_point_set_app_8h.js	2541
doc/html/_pr_dijkstra_8h.js	2541
doc/html/_pr_geodesics_8h.js	2541
doc/html/_pr_interface_8h.js	2542
doc/html/_pr_multi_dijkstra_8h.js	2543
doc/html/_pr_param_util_8h.js	2543
doc/html/_pr_parametrize_bdy_8h.js	2544
doc/html/_pr_parametrize_int_8h.js	2544
doc/html/_pr_path_triangle_seq_8h.js	2545
doc/html/_pr_texture_8h.js	2545
doc/html/_pr_wavelet_util_8h.js	2545
doc/html/_quality_utils_8h.js	2546
doc/html/_rational_8h.js	2546
doc/html/_registration_utils_8h.js	2547
doc/html/_regularize_utils_8h.js	2547
doc/html/_s_i_s_lconversion_8h.js	2548
doc/html/_singular_8h.js	2548
doc/html/_singularity_classification_8h.js	2549

doc/html/_singularity_type_8h.js	2549
doc/html/_smooth_volume_8h.js	2550
doc/html/_spline_debug_utils_8h.js	2550
doc/html/_spline_utils_8h.js	2551
doc/html/_spline_volume_8h.js	2552
doc/html/_split_model_utils_8h.js	2552
doc/html/_stream_utils_8h.js	2552
doc/html/_streamable_8h.js	2553
doc/html/_subdivision_classification_8h.js	2553
doc/html/_surface_creators_8h.js	2554
doc/html/_surface_interpolator_8h.js	2555
doc/html/_surface_model_utils_8h.js	2555
doc/html/_surface_on_volume_tools_8h.js	2555
doc/html/_surface_tools_8h.js	2556
doc/html/_tessellator_utils_8h.js	2556
doc/html/_utils_8h.js	2557
doc/html/_values_8h.js	2557
doc/html/_volume_interpolator_8h.js	2558
doc/html/_volume_model_creator_8h.js	2558
doc/html/_volume_model_file_handler_8h.js	2559
doc/html/_volume_tools_8h.js	2559
doc/html/_volumes_8h.js	2560
doc/html/annotated_dup.js	2560
doc/html/aux2_8cpp.js	2560
doc/html/aux2_8h.js	2561
doc/html/bandmat_8cpp.js	2561
doc/html/binom_8h.js	2562
doc/html/boolean_8h.js	2562
doc/html/brent_minimize_8h.js	2562
doc/html/checks_8h.js	2563
doc/html/cholesky_8cpp.js	2563
doc/html/class_control_word.js	2564
doc/html/class_curve_resolution_sheet.js	2564
doc/html/class_data_handler.js	2565
doc/html/class_data_handler_vol_and_l_r.js	2565
doc/html/class_default_data_handler.js	2566
doc/html/class_dijkstra.js	2566
doc/html/class_edge_type.js	2567
doc/html/class_g_a_r_c_h11_l_l.js	2567
doc/html/class_go_1_1_adapt_curve.js	2568
doc/html/class_go_1_1_alg_obj2_d_int.js	2568
doc/html/class_go_1_1_alg_obj3_d_int.js	2569
doc/html/class_go_1_1_alg_object_int.js	2569
doc/html/class_go_1_1_approx_crv_to_seqs.js	2570
doc/html/class_go_1_1_approx_curve.js	2570
doc/html/class_go_1_1_approx_surf.js	2571
doc/html/class_go_1_1_array.js	2571
doc/html/class_go_1_1_bary_coord_system.js	2572
doc/html/class_go_1_1_bary_coord_system_triangle3_d.js	2572
doc/html/class_go_1_1_bd_cond_func.js	2573
doc/html/class_go_1_1_bernstein_multi.js	2573
doc/html/class_go_1_1_bernstein_poly.js	2573
doc/html/class_go_1_1_bernstein_tetrahedral_poly.js	2574
doc/html/class_go_1_1_bernstein_triangular_poly.js	2574
doc/html/class_go_1_1_bezier_triangle.js	2575
doc/html/class_go_1_1_binomial.js	2576
doc/html/class_go_1_1_block_boundary_condition.js	2576
doc/html/class_go_1_1_block_point_bd_cond.js	2577

doc/html/class_go_1_1_block_solution.js	2577
doc/html/class_go_1_1_body.js	2578
doc/html/class_go_1_1_bounded_curve.js	2579
doc/html/class_go_1_1_bounded_surface.js	2579
doc/html/class_go_1_1_bounding_box.js	2579
doc/html/class_go_1_1_bspline_basis.js	2580
doc/html/class_go_1_1_cp_uclock.js	2580
doc/html/class_go_1_1_cell_division.js	2580
doc/html/class_go_1_1_circle.js	2581
doc/html/class_go_1_1_closest_point_calculator.js	2581
doc/html/class_go_1_1_complete_edge_net.js	2582
doc/html/class_go_1_1_complexity_info.js	2582
doc/html/class_go_1_1_composite_box.js	2583
doc/html/class_go_1_1_composite_curve.js	2583
doc/html/class_go_1_1_composite_model.js	2583
doc/html/class_go_1_1_composite_model_factory.js	2584
doc/html/class_go_1_1_composite_model_file_handler.js	2585
doc/html/class_go_1_1_composite_surface.js	2585
doc/html/class_go_1_1_concrete_creator.js	2586
doc/html/class_go_1_1_cone.js	2586
doc/html/class_go_1_1_cone_volume.js	2586
doc/html/class_go_1_1_connection_functor.js	2587
doc/html/class_go_1_1_coordinate_system.js	2588
doc/html/class_go_1_1_creator.js	2588
doc/html/class_go_1_1_cross_tan_off_dist.js	2589
doc/html/class_go_1_1_cross_tangent_offset.js	2589
doc/html/class_go_1_1_crosses_value.js	2590
doc/html/class_go_1_1_curve_bounded_domain.js	2590
doc/html/class_go_1_1_curve_loop.js	2591
doc/html/class_go_1_1_curve_model.js	2591
doc/html/class_go_1_1_curve_on_surface.js	2592
doc/html/class_go_1_1_curve_on_volume.js	2592
doc/html/class_go_1_1_curve_tesselator.js	2593
doc/html/class_go_1_1_cv_cv_intersector.js	2593
doc/html/class_go_1_1_cv_pt_intersector.js	2594
doc/html/class_go_1_1_cylinder.js	2594
doc/html/class_go_1_1_cylinder_int.js	2594
doc/html/class_go_1_1_cylinder_volume.js	2595
doc/html/class_go_1_1_degenerated_intersection_curve.js	2596
doc/html/class_go_1_1_direction_cone.js	2596
doc/html/class_go_1_1_disc.js	2597
doc/html/class_go_1_1_domain.js	2597
doc/html/class_go_1_1_edge_vertex.js	2597
doc/html/class_go_1_1_element2_d.js	2598
doc/html/class_go_1_1_elementary_curve.js	2598
doc/html/class_go_1_1_elementary_surface.js	2599
doc/html/class_go_1_1_elementary_volume.js	2600
doc/html/class_go_1_1_ellipse.js	2600
doc/html/class_go_1_1_eval_curve.js	2601
doc/html/class_go_1_1_eval_curve_set.js	2601
doc/html/class_go_1_1_eval_functor_curve.js	2602
doc/html/class_go_1_1_eval_functor_surface.js	2602
doc/html/class_go_1_1_eval_offset_surface.js	2603
doc/html/class_go_1_1_eval_param_curve.js	2603
doc/html/class_go_1_1_eval_surface.js	2604
doc/html/class_go_1_1_face_adjacency.js	2604
doc/html/class_go_1_1_face_connectivity_utils.js	2605
doc/html/class_go_1_1_face_set_quality.js	2605

doc/html/class_go_1_1_face_set_repair.js	2605
doc/html/class_go_1_1_factory.js	2606
doc/html/class_go_1_1_fun2_fun.js	2606
doc/html/class_go_1_1_function_minimizer.js	2607
doc/html/class_go_1_1_general_mesh.js	2607
doc/html/class_go_1_1_generic_tri_mesh.js	2608
doc/html/class_go_1_1_geo_tol.js	2609
doc/html/class_go_1_1_geom_object.js	2609
doc/html/class_go_1_1_geom_object_int.js	2610
doc/html/class_go_1_1_go_tools.js	2610
doc/html/class_go_1_1_gpu_matrix.js	2610
doc/html/class_go_1_1_hermite_app_c.js	2611
doc/html/class_go_1_1_hermite_app_s.js	2611
doc/html/class_go_1_1_hermite_appr_eval_surf.js	2612
doc/html/class_go_1_1_hermite_grid1_d.js	2612
doc/html/class_go_1_1_hermite_grid1_d_multi.js	2613
doc/html/class_go_1_1_hermite_grid2_d.js	2613
doc/html/class_go_1_1_hermite_interpolator.js	2614
doc/html/class_go_1_1_hyperbola.js	2614
doc/html/class_go_1_1_i_g_e_sconverter.js	2615
doc/html/class_go_1_1_identity.js	2615
doc/html/class_go_1_1_implicitize_curve_algo.js	2616
doc/html/class_go_1_1_implicitize_curve_and_vector_algo.js	2616
doc/html/class_go_1_1_implicitize_point_cloud_algo.js	2617
doc/html/class_go_1_1_implicitize_surface_algo.js	2618
doc/html/class_go_1_1_index_mesh2_d_iterator.js	2618
doc/html/class_go_1_1_int_crv_evaluator.js	2619
doc/html/class_go_1_1_int_results_comp_cv.js	2619
doc/html/class_go_1_1_int_results_model.js	2620
doc/html/class_go_1_1_int_results_sf_model.js	2620
doc/html/class_go_1_1_interpolated_intersection_curve.js	2621
doc/html/class_go_1_1_interpolator.js	2622
doc/html/class_go_1_1_intersection_curve.js	2622
doc/html/class_go_1_1_intersection_link.js	2623
doc/html/class_go_1_1_intersection_point.js	2623
doc/html/class_go_1_1_intersection_pool.js	2623
doc/html/class_go_1_1_intersector.js	2624
doc/html/class_go_1_1_intersector2_obj.js	2624
doc/html/class_go_1_1_intersector_alg_par.js	2624
doc/html/class_go_1_1_intersector_func_const.js	2625
doc/html/class_go_1_1_isogeometric_block.js	2626
doc/html/class_go_1_1_isogeometric_model.js	2627
doc/html/class_go_1_1_isogeometric_sf_block.js	2627
doc/html/class_go_1_1_isogeometric_sf_model.js	2628
doc/html/class_go_1_1_isogeometric_vol_block.js	2628
doc/html/class_go_1_1_isogeometric_vol_model.js	2628
doc/html/class_go_1_1_isoparametric_intersection_curve.js	2629
doc/html/class_go_1_1_l_r_b_spline2_d.js	2630
doc/html/class_go_1_1_l_r_spline_eval_grid.js	2630
doc/html/class_go_1_1_l_r_spline_surface.js	2631
doc/html/class_go_1_1_l_r_surf_approx.js	2631
doc/html/class_go_1_1_l_r_surf_smooth_l_s.js	2632
doc/html/class_go_1_1_lift_curve.js	2632
doc/html/class_go_1_1_line.js	2633
doc/html/class_go_1_1_line2_d_int.js	2633
doc/html/class_go_1_1_line_cloud.js	2633
doc/html/class_go_1_1_line_cloud_tesselator.js	2634
doc/html/class_go_1_1_line_strip.js	2634

doc/html/class_go_1_1_locked_dir_dist_func.js	2635
doc/html/class_go_1_1_loop.js	2635
doc/html/class_go_1_1_march_point.js	2636
doc/html/class_go_1_1_matrix_x_d.js	2636
doc/html/class_go_1_1_mesh2_d.js	2637
doc/html/class_go_1_1_mesh2_d_iterator.js	2637
doc/html/class_go_1_1_model_quality.js	2638
doc/html/class_go_1_1_model_repair.js	2638
doc/html/class_go_1_1_non_evaluable_intersection_curve.js	2639
doc/html/class_go_1_1_noop_tesselator.js	2639
doc/html/class_go_1_1_object_header.js	2640
doc/html/class_go_1_1_par0_func_intersector.js	2640
doc/html/class_go_1_1_par1_func_intersector.js	2641
doc/html/class_go_1_1_par2_func_intersector.js	2641
doc/html/class_go_1_1_parabola.js	2642
doc/html/class_go_1_1_parallelepiped.js	2642
doc/html/class_go_1_1_param0_function_int.js	2643
doc/html/class_go_1_1_param1_function_int.js	2643
doc/html/class_go_1_1_param2_function_int.js	2643
doc/html/class_go_1_1_param_curve.js	2643
doc/html/class_go_1_1_param_curve_int.js	2644
doc/html/class_go_1_1_param_function_int.js	2644
doc/html/class_go_1_1_param_geom_int.js	2645
doc/html/class_go_1_1_param_object_int.js	2645
doc/html/class_go_1_1_param_point_int.js	2646
doc/html/class_go_1_1_param_surface.js	2646
doc/html/class_go_1_1_param_surface_int.js	2646
doc/html/class_go_1_1_param_volume.js	2646
doc/html/class_go_1_1_parametric_surface_tesselator.js	2647
doc/html/class_go_1_1_plane.js	2648
doc/html/class_go_1_1_plane_int.js	2648
doc/html/class_go_1_1_point.js	2648
doc/html/class_go_1_1_point_cloud.js	2649
doc/html/class_go_1_1_point_on_curve.js	2649
doc/html/class_go_1_1_point_on_edge.js	2650
doc/html/class_go_1_1_point_sequence.js	2650
doc/html/class_go_1_1_project_curve.js	2651
doc/html/class_go_1_1_project_curve_and_cross_tan.js	2651
doc/html/class_go_1_1_project_intersection_curve.js	2652
doc/html/class_go_1_1_pt_pt_intersector.js	2652
doc/html/class_go_1_1_quad_mesh.js	2653
doc/html/class_go_1_1_quality_results.js	2654
doc/html/class_go_1_1_rational.js	2654
doc/html/class_go_1_1_rect_domain.js	2655
doc/html/class_go_1_1_rect_grid.js	2655
doc/html/class_go_1_1_rect_grid_tesselator.js	2656
doc/html/class_go_1_1_rectangular_surface_tesselator.js	2656
doc/html/class_go_1_1_rectangular_volume_tesselator.js	2657
doc/html/class_go_1_1_registrator.js	2657
doc/html/class_go_1_1_regular_mesh.js	2658
doc/html/class_go_1_1_regular_vol_mesh.js	2658
doc/html/class_go_1_1_regularize_face.js	2659
doc/html/class_go_1_1_regularize_face_set.js	2659
doc/html/class_go_1_1_rotated_box.js	2660
doc/html/class_go_1_1_scratch_vect.js	2660
doc/html/class_go_1_1_sf_boundary_condition.js	2661
doc/html/class_go_1_1_sf_cv_intersector.js	2661
doc/html/class_go_1_1_sf_point_bd_cond.js	2662

doc/html/class_go_1_1_sf_pt_intersector.js	2662
doc/html/class_go_1_1_sf_self_intersector.js	2663
doc/html/class_go_1_1_sf_sf_intersector.js	2664
doc/html/class_go_1_1_sf_solution.js	2664
doc/html/class_go_1_1_singularity_info.js	2664
doc/html/class_go_1_1_smooth_curve.js	2665
doc/html/class_go_1_1_smooth_curve_set.js	2666
doc/html/class_go_1_1_smooth_surf.js	2666
doc/html/class_go_1_1_smooth_surf_set.js	2666
doc/html/class_go_1_1_smooth_transition.js	2667
doc/html/class_go_1_1_smooth_volume.js	2668
doc/html/class_go_1_1_solve_b_c_g.js	2668
doc/html/class_go_1_1_solve_c_g.js	2669
doc/html/class_go_1_1_solve_c_g_c_o.js	2669
doc/html/class_go_1_1_space_int_crv.js	2670
doc/html/class_go_1_1_sphere.js	2670
doc/html/class_go_1_1_sphere_int.js	2670
doc/html/class_go_1_1_sphere_volume.js	2671
doc/html/class_go_1_1_spline1_function_int.js	2672
doc/html/class_go_1_1_spline2_function_int.js	2672
doc/html/class_go_1_1_spline_approximator.js	2673
doc/html/class_go_1_1_spline_curve.js	2673
doc/html/class_go_1_1_spline_curve_int.js	2674
doc/html/class_go_1_1_spline_interpolator.js	2674
doc/html/class_go_1_1_spline_surface.js	2675
doc/html/class_go_1_1_spline_surface_int.js	2676
doc/html/class_go_1_1_spline_volume.js	2676
doc/html/class_go_1_1_streamable.js	2676
doc/html/class_go_1_1_surface_assembly.js	2677
doc/html/class_go_1_1_surface_model.js	2677
doc/html/class_go_1_1_surface_of_linear_extrusion.js	2677
doc/html/class_go_1_1_surface_of_revolution.js	2678
doc/html/class_go_1_1_surface_on_volume.js	2678
doc/html/class_go_1_1_sweep_surface_creator.js	2678
doc/html/class_go_1_1_sweep_volume_creator.js	2679
doc/html/class_go_1_1_tesselator.js	2679
doc/html/class_go_1_1_test_in_domain.js	2680
doc/html/class_go_1_1_torus.js	2680
doc/html/class_go_1_1_torus_volume.js	2680
doc/html/class_go_1_1_triangle.js	2681
doc/html/class_go_1_1_trim_curve.js	2682
doc/html/class_go_1_1_vertex.js	2682
doc/html/class_go_1_1_vol_boundary_condition.js	2682
doc/html/class_go_1_1_vol_point_bd_cond.js	2683
doc/html/class_go_1_1_vol_solution.js	2684
doc/html/class_go_1_1_volume_adjacency.js	2684
doc/html/class_go_1_1_volume_model.js	2684
doc/html/class_go_1_1_volume_model_file_handler.js	2685
doc/html/class_go_1_1_volume_parameter_curve.js	2685
doc/html/class_go_1_1_volume_space_curve.js	2686
doc/html/class_go_1_1box_structuring_1_1_bounding_box_structure.js	2686
doc/html/class_go_1_1box_structuring_1_1_sub_surface_bounding_box.js	2687
doc/html/class_go_1_1box_structuring_1_1_surface_data.js	2688
doc/html/class_go_1_1ft_cell.js	2688
doc/html/class_go_1_1ft_cell_info.js	2689
doc/html/class_go_1_1ft_chart_surface.js	2689
doc/html/class_go_1_1ft_curve.js	2689
doc/html/class_go_1_1ft_curve_segment.js	2690

doc/html/class_go_1_1ft_edge.js	2690
doc/html/class_go_1_1ft_edge_base.js	2690
doc/html/class_go_1_1ft_face_base.js	2690
doc/html/class_go_1_1ft_graph_edge.js	2691
doc/html/class_go_1_1ft_group_geom.js	2692
doc/html/class_go_1_1ft_line.js	2692
doc/html/class_go_1_1ft_message.js	2693
doc/html/class_go_1_1ft_planar_graph.js	2693
doc/html/class_go_1_1ft_plane.js	2694
doc/html/class_go_1_1ft_point.js	2694
doc/html/class_go_1_1ft_point_set.js	2695
doc/html/class_go_1_1ft_s_sf_edge.js	2695
doc/html/class_go_1_1ft_sample_point.js	2695
doc/html/class_go_1_1ft_search_node.js	2696
doc/html/class_go_1_1ft_smooth_surf.js	2696
doc/html/class_go_1_1ft_surface.js	2697
doc/html/class_go_1_1ft_surface_set.js	2697
doc/html/class_go_1_1ft_surface_set_point.js	2697
doc/html/class_go_1_1ft_volume.js	2698
doc/html/class_go_1_1tp_edge.js	2698
doc/html/class_go_1_1tp_face.js	2698
doc/html/class_go_1_1tp_march_point.js	2699
doc/html/class_go_1_1tp_table_entry.js	2699
doc/html/class_go_1_1tp_topology_table.js	2700
doc/html/class_go_1_1ttl_point.js	2700
doc/html/class_handle.js	2701
doc/html/class_handle_id.js	2701
doc/html/class_heap_node.js	2702
doc/html/class_heap_node2.js	2702
doc/html/class_load_and_store_flag.js	2703
doc/html/class_matrix_col.js	2703
doc/html/class_matrix_col_x.js	2703
doc/html/class_matrix_row.js	2704
doc/html/class_matrix_row_col.js	2704
doc/html/class_model__3pe.js	2705
doc/html/class_multi_dijkstra.js	2705
doc/html/class_n_e_w_m_a_t_1_1_added_matrix.js	2706
doc/html/class_n_e_w_m_a_t_1_1_array_length_specifier.js	2706
doc/html/class_n_e_w_m_a_t_1_1_band_l_u_matrix.js	2706
doc/html/class_n_e_w_m_a_t_1_1_band_matrix.js	2707
doc/html/class_n_e_w_m_a_t_1_1_base_matrix.js	2707
doc/html/class_n_e_w_m_a_t_1_1_cannot_build_exception.js	2708
doc/html/class_n_e_w_m_a_t_1_1_coled_matrix.js	2708
doc/html/class_n_e_w_m_a_t_1_1_column_vector.js	2708
doc/html/class_n_e_w_m_a_t_1_1_concatenated_matrix.js	2709
doc/html/class_n_e_w_m_a_t_1_1_convergence_exception.js	2710
doc/html/class_n_e_w_m_a_t_1_1_cROUT_matrix.js	2710
doc/html/class_n_e_w_m_a_t_1_1_diaged_matrix.js	2711
doc/html/class_n_e_w_m_a_t_1_1_diagonal_matrix.js	2711
doc/html/class_n_e_w_m_a_t_1_1_find_maximum2.js	2711
doc/html/class_n_e_w_m_a_t_1_1_general_matrix.js	2712
doc/html/class_n_e_w_m_a_t_1_1_generic_matrix.js	2712
doc/html/class_n_e_w_m_a_t_1_1_get_sub_matrix.js	2713
doc/html/class_n_e_w_m_a_t_1_1_identity_matrix.js	2714
doc/html/class_n_e_w_m_a_t_1_1_incompatible_dimensions_exception.js	2714
doc/html/class_n_e_w_m_a_t_1_1_index_exception.js	2715
doc/html/class_n_e_w_m_a_t_1_1_internal_exception.js	2715
doc/html/class_n_e_w_m_a_t_1_1_inverted_matrix.js	2716

doc/html/class_n_e_w_m_a_t_1_1_k_p_matrix.js	2716
doc/html/class_n_e_w_m_a_t_1_1_l_l_d_f_i.js	2717
doc/html/class_n_e_w_m_a_t_1_1_linear_equation_solver.js	2717
doc/html/class_n_e_w_m_a_t_1_1_log_and_sign.js	2718
doc/html/class_n_e_w_m_a_t_1_1_lower_band_matrix.js	2718
doc/html/class_n_e_w_m_a_t_1_1_lower_triangular_matrix.js	2719
doc/html/class_n_e_w_m_a_t_1_1_m_l_e_d_f_i.js	2720
doc/html/class_n_e_w_m_a_t_1_1_mated_matrix.js	2721
doc/html/class_n_e_w_m_a_t_1_1_matrix.js	2721
doc/html/class_n_e_w_m_a_t_1_1_matrix_band_width.js	2722
doc/html/class_n_e_w_m_a_t_1_1_matrix_input.js	2722
doc/html/class_n_e_w_m_a_t_1_1_matrix_type.js	2723
doc/html/class_n_e_w_m_a_t_1_1_multi_radix_counter.js	2723
doc/html/class_n_e_w_m_a_t_1_1_multiplied_matrix.js	2723
doc/html/class_n_e_w_m_a_t_1_1_n_p_d_exception.js	2724
doc/html/class_n_e_w_m_a_t_1_1_neg_shifted_matrix.js	2724
doc/html/class_n_e_w_m_a_t_1_1_negated_matrix.js	2725
doc/html/class_n_e_w_m_a_t_1_1_non_linear_least_squares.js	2725
doc/html/class_n_e_w_m_a_t_1_1_not_defined_exception.js	2726
doc/html/class_n_e_w_m_a_t_1_1_not_square_exception.js	2726
doc/html/class_n_e_w_m_a_t_1_1_overflow_exception.js	2727
doc/html/class_n_e_w_m_a_t_1_1_program_exception.js	2727
doc/html/class_n_e_w_m_a_t_1_1_r1_col_id.js	2728
doc/html/class_n_e_w_m_a_t_1_1_return_matrix_x.js	2728
doc/html/class_n_e_w_m_a_t_1_1_reversed_matrix.js	2729
doc/html/class_n_e_w_m_a_t_1_1_row_vector.js	2729
doc/html/class_n_e_w_m_a_t_1_1_rowed_matrix.js	2730
doc/html/class_n_e_w_m_a_t_1_1_s_p_matrix.js	2731
doc/html/class_n_e_w_m_a_t_1_1_scaled_matrix.js	2731
doc/html/class_n_e_w_m_a_t_1_1_shifted_matrix.js	2732
doc/html/class_n_e_w_m_a_t_1_1_simple_int_array.js	2732
doc/html/class_n_e_w_m_a_t_1_1_singular_exception.js	2733
doc/html/class_n_e_w_m_a_t_1_1_solved_matrix.js	2733
doc/html/class_n_e_w_m_a_t_1_1_stacked_matrix.js	2733
doc/html/class_n_e_w_m_a_t_1_1_sub_matrix_dimension_exception.js	2734
doc/html/class_n_e_w_m_a_t_1_1_subtracted_matrix.js	2734
doc/html/class_n_e_w_m_a_t_1_1_symmetric_band_matrix.js	2735
doc/html/class_n_e_w_m_a_t_1_1_symmetric_eigen_analysis.js	2735
doc/html/class_n_e_w_m_a_t_1_1_symmetric_matrix.js	2736
doc/html/class_n_e_w_m_a_t_1_1_transposed_matrix.js	2736
doc/html/class_n_e_w_m_a_t_1_1_upper_band_matrix.js	2737
doc/html/class_n_e_w_m_a_t_1_1_upper_triangular_matrix.js	2738
doc/html/class_n_e_w_m_a_t_1_1_vector_exception.js	2739
doc/html/class_n_e_w_m_a_t_1_1nric_matrix.js	2739
doc/html/class_parametric_surface_property_sheet.js	2740
doc/html/class_path_type.js	2740
doc/html/class_point_cloud_property_sheet.js	2741
doc/html/class_pr_bi_c_g_stab.js	2741
doc/html/class_pr_c_g.js	2742
doc/html/class_pr_cell_structure.js	2742
doc/html/class_pr_explicit_connectivity.js	2743
doc/html/class_pr_faber_f.js	2743
doc/html/class_pr_fast_unorganized_o_p.js	2744
doc/html/class_pr_filterbank.js	2744
doc/html/class_pr_heap.js	2745
doc/html/class_pr_level_triangulation_o_p.js	2745
doc/html/class_pr_mat.js	2746
doc/html/class_pr_mat_sparse.js	2746

doc/html/class_pr_matrix.js	2747
doc/html/class_pr_nested_node.js	2748
doc/html/class_pr_nested_triangulation.js	2748
doc/html/class_pr_node.js	2749
doc/html/class_pr_organized_points.js	2749
doc/html/class_pr_p_g_node.js	2749
doc/html/class_pr_param_triangulation.js	2750
doc/html/class_pr_parametrize_bdy.js	2751
doc/html/class_pr_parametrize_int.js	2751
doc/html/class_pr_parametrize_mesh.js	2752
doc/html/class_pr_planar_graph__o_p.js	2753
doc/html/class_pr_prewavelet__f.js	2753
doc/html/class_pr_prm_e_d_d_h_l_s.js	2754
doc/html/class_pr_prm_experimental.js	2754
doc/html/class_pr_prm_least_square.js	2754
doc/html/class_pr_prm_mean_value.js	2755
doc/html/class_pr_prm_shp_pres.js	2755
doc/html/class_pr_prm_surface.js	2756
doc/html/class_pr_prm_sym_mean_value.js	2756
doc/html/class_pr_prm_uniform.js	2757
doc/html/class_pr_prm_wachspress.js	2757
doc/html/class_pr_rectangular_grid__o_p.js	2758
doc/html/class_pr_sub_triangulation.js	2758
doc/html/class_pr_thin.js	2759
doc/html/class_pr_threshold.js	2759
doc/html/class_pr_triangle.js	2760
doc/html/class_pr_triangulation__n_t.js	2760
doc/html/class_pr_triangulation__o_p.js	2760
doc/html/class_pr_unorganized__o_p.js	2761
doc/html/class_pr_vec.js	2761
doc/html/class_print_counter.js	2762
doc/html/class_r_b_d__c_o_m_m_o_n_1_1_bad_alloc.js	2762
doc/html/class_r_b_d__c_o_m_m_o_n_1_1_base_exception.js	2763
doc/html/class_r_b_d__c_o_m_m_o_n_1_1_domain_error.js	2763
doc/html/class_r_b_d__c_o_m_m_o_n_1_1_invalid_argument.js	2764
doc/html/class_r_b_d__c_o_m_m_o_n_1_1_janitor.js	2764
doc/html/class_r_b_d__c_o_m_m_o_n_1_1_length_error.js	2764
doc/html/class_r_b_d__c_o_m_m_o_n_1_1_logic_error.js	2765
doc/html/class_r_b_d__c_o_m_m_o_n_1_1_one_dim_solve.js	2765
doc/html/class_r_b_d__c_o_m_m_o_n_1_1_out_of_range.js	2766
doc/html/class_r_b_d__c_o_m_m_o_n_1_1_overflow_error.js	2766
doc/html/class_r_b_d__c_o_m_m_o_n_1_1_r1__r1.js	2766
doc/html/class_r_b_d__c_o_m_m_o_n_1_1_range_error.js	2767
doc/html/class_r_b_d__c_o_m_m_o_n_1_1_runtime_error.js	2767
doc/html/class_r_b_d__c_o_m_m_o_n_1_1_solution_exception.js	2768
doc/html/class_r_b_d__c_o_m_m_o_n_1_1_tracer.js	2768
doc/html/class_rect_matrix_col.js	2769
doc/html/class_rect_matrix_diag.js	2769
doc/html/class_rect_matrix_row.js	2770
doc/html/class_rect_matrix_row_col.js	2770
doc/html/class_rectangular_surface_property_sheet.js	2771
doc/html/class_rectangular_volume_property_sheet.js	2772
doc/html/class_spline_curve_property_sheet.js	2772
doc/html/class_surface_resolution_sheet.js	2772
doc/html/class_test_class.js	2773
doc/html/class_unf_node_type.js	2773
doc/html/classbool.js	2774
doc/html/classgv_action_sheet.js	2774

doc/html/classgv_application.js	2775
doc/html/classgv_application_vol_and_l_r.js	2775
doc/html/classgv_camera.js	2776
doc/html/classgv_curve_paintable.js	2776
doc/html/classgv_data.js	2776
doc/html/classgv_generic_tri_paintable.js	2777
doc/html/classgv_generic_tri_quad_mesh.js	2777
doc/html/classgv_generic_tri_quad_paintable.js	2778
doc/html/classgv_group.js	2778
doc/html/classgv_group_property_sheet.js	2779
doc/html/classgv_line_cloud_paintable.js	2779
doc/html/classgv_noop_paintable.js	2780
doc/html/classgv_object_list.js	2780
doc/html/classgv_observer.js	2780
doc/html/classgv_paintable.js	2781
doc/html/classgvPainter.js	2781
doc/html/classgv_parametric_surface_paintable.js	2782
doc/html/classgv_point_cloud_paintable.js	2782
doc/html/classgv_property_sheet.js	2783
doc/html/classgv_quads_paintable.js	2783
doc/html/classgv_rectangular_surface_paintable.js	2784
doc/html/classgv_rectangular_volume_paintable.js	2784
doc/html/classgv_resolution_dialog.js	2785
doc/html/classgv_standard_mouse_handler.js	2785
doc/html/classgv_texture.js	2786
doc/html/classgv_view.js	2787
doc/html/classhed_1_1_dart.js	2787
doc/html/classhed_1_1_edge.js	2787
doc/html/classhed_1_1_node.js	2788
doc/html/classhed_1_1_triangulation.js	2789
doc/html/classhetriang_1_1_dart.js	2789
doc/html/classhetriang_1_1_edge.js	2790
doc/html/classhetriang_1_1_node.js	2791
doc/html/classhetriang_1_1_triangulation.js	2791
doc/html/classmaterial__appearance.js	2792
doc/html/classlapse.js	2792
doc/html/classvector3t.js	2792
doc/html/cm_utils_8h.js	2793
doc/html/config_8h.js	2793
doc/html/construct_8c.js	2794
doc/html/controlw_8h.js	2794
doc/html/crvarctang_8c.js	2795
doc/html/crvcrvtang_8c.js	2795
doc/html/crvlintang_8c.js	2795
doc/html/destroy_8c.js	2796
doc/html/dir_0015c891e2eac5fd9e489e846b834700.js	2796
doc/html/dir_038b4c324d5349c9475465ffee024b1c.js	2797
doc/html/dir_04f3dbfb06d4e757244c31fc16142d05.js	2797
doc/html/dir_0a492bd6eb1e5e04630449d6b6b92955.js	2798
doc/html/dir_0df052b1ef5b53594ca9be5a6074277d.js	2798
doc/html/dir_15757c9e0b20f079697e78f9f294c1ee.js	2798
doc/html/dir_1c9e4414cb2b8eb9526a3aa2069941f8.js	2799
doc/html/dir_1ce88f9178ec82ce3d3b5366984a4a76.js	2799
doc/html/dir_26434c1759c56a49292c328c033e76d7.js	2800
doc/html/dir_27047e9e82ebbd34bcfea6195a1d237f.js	2800
doc/html/dir_287d50cea2dbd2d342307d58f17ae566.js	2800
doc/html/dir_28d725a1074140e3875acd99df863a9a.js	2801
doc/html/dir_2e9f58fed2f8cb33783712134eca6f54.js	2801

doc/html/dir_2ea40cee675de4bd87b06c9233dde18b.js	2801
doc/html/dir_34a54fc55cc4c76e2db47ca198785905.js	2802
doc/html/dir_34a7c6eb6b0d9060dd547ea008608408.js	2802
doc/html/dir_3706bca45f06e8690f2585ec746205ce.js	2802
doc/html/dir_37bdf26acb6f0e99095c72f4b261057d.js	2803
doc/html/dir_3a8e8a1c81bc0d4ea45d413753d6b9d6.js	2803
doc/html/dir_3b29025e4f1a16d99c0ac7281d29a749.js	2804
doc/html/dir_3cd14e887b3be1bb0699a0f4b9d7471b.js	2804
doc/html/dir_442358e1b75f159df2b4b10f68cb6669.js	2804
doc/html/dir_44eb60e2f3e103c46810fc3a6f6f0b38.js	2805
doc/html/dir_46fb2897033f53c047d4860af5672505.js	2805
doc/html/dir_4dc6ea3c21164f9c4718441280129d09.js	2806
doc/html/dir_4ed957b398f8a342f35e202c6dbeff82.js	2806
doc/html/dir_4f584829c2589cd0dd901c7bd8d4b8ca.js	2807
doc/html/dir_54c0807dc9dc58401cdf98175b301107.js	2807
doc/html/dir_5eaf107a44af8927c3e88d7693acb3ec.js	2808
doc/html/dir_61c47cbd77726dc04327162bb82f2acf.js	2808
doc/html/dir_61f4d2722e3a1702dec4dcefd169ded.js	2809
doc/html/dir_662c58e926d5c8a2e92c3a0efd184e27.js	2809
doc/html/dir_69009d91722e8158a6e1821a3d198fd0.js	2809
doc/html/dir_6917c6a4538152e6852497b387e02df2.js	2810
doc/html/dir_6958061072da6881959e38b0a737a4c6.js	2810
doc/html/dir_6a9f218ac24eb7b754dc250ffd508dae.js	2811
doc/html/dir_764e20a7fa93cafa3199ac45b6e6b986.js	2811
doc/html/dir_787e29cc9b8f49ef343dc9ff9fbd368.js	2811
doc/html/dir_79b3003bf1bdb319185970350806bf23.js	2812
doc/html/dir_7b91ef4997e28bdb148c447548446353.js	2812
doc/html/dir_7c72f061721a5863212d64c180702103.js	2813
doc/html/dir_7efa0b394c677aa9d932dff09cb14a2d.js	2813
doc/html/dir_7f10ec973083a3eaca908361800e49c7.js	2814
doc/html/dir_84497794bb37d283dbd68163bed8ef05.js	2814
doc/html/dir_85c45df6158cba585b9d394ec2ca8ba6.js	2815
doc/html/dir_892cdac2bffb6fa625049467eefe0a.js	2815
doc/html/dir_8946de0dbd7af8713b300be94ace6b21.js	2816
doc/html/dir_8c65b503da15ddfa217ea827bbddedc7.js	2817
doc/html/dir_8dcf46dfbb2e2ca3d570ad71ab7f5984.js	2817
doc/html/dir_90fbe926ef93e71fdbf5723e062c9da4.js	2817
doc/html/dir_981871de7611ee7a54f2d0f3790a0f82.js	2818
doc/html/dir_99c721c896ff1c980c5b18b3dcba94.js	2818
doc/html/dir_9df88d95c92a16255dc1e6f9b3433e60.js	2819
doc/html/dir_9e6bb4020941d0be38bb0d27bbe7cccb.js	2819
doc/html/dir_9e7f8af3df212ec4d479463341bb72a6.js	2819
doc/html/dir_a0fa69fab3be50a1b4c8c4ebaca5098c.js	2820
doc/html/dir_a39cd4dcd495fe456e1f661046bd5530.js	2820
doc/html/dir_a537b1e5a645cb4471f6768545d20c2b.js	2821
doc/html/dir_a6ecec081e9e6ebaf06ff4a52c078f1.js	2821
doc/html/dir_a702eb1d615fc8e494327e00bfd64e2d.js	2822
doc/html/dir_ab08e349df88865f421593b417138732.js	2822
doc/html/dir_b07f39590aec0282ad5866778d6f956e.js	2822
doc/html/dir_b168ed9c8c05331b63ad454a11f635f2.js	2823
doc/html/dir_b28ca551f00416d098dfb2d5d86cb85e.js	2823
doc/html/dir_b3f8b92e7ea1946157ca7a805e4d2267.js	2824
doc/html/dir_bf6a253d933d2fc523d4a21708f60d86.js	2824
doc/html/dir_ce2be4821eafb03cec507f192e45eceb.js	2824
doc/html/dir_ceb2d94ff415c9faa44cf694ce53cd4a.js	2825
doc/html/dir_cfbe617369797e2c088f7718a7b296e4.js	2825
doc/html/dir_d3b744872420378775da279872820ed2.js	2826
doc/html/dir_d3d5f8ee42e1e2dc7017f3f7157ca5bc.js	2826

doc/html/dir_dcc2523c85224d60248ee4ee7462cfa8.js	2827
doc/html/dir_e2ef5fb48cb01c9435f4dad81bf7eab.js	2827
doc/html/dir_e3e0722e0dd9c71678af19110aeec8f8.js	2828
doc/html/dir_e66b996f8b4be0681f8e9eb9869a3079.js	2828
doc/html/dir_e846bdbb82b84d132cb98df96cb07bf0.js	2828
doc/html/dir_e98c7b4edc7740dfc859667cd423ef3f.js	2829
doc/html/dir_eb531be94dc3f2da4c4acbf84e0e7dc.js	2829
doc/html/dir_f2117bb2bb20a60155ae701cb86e7351.js	2830
doc/html/dir_f2bf6ebad7a5cab5a643ce43ec016ae9.js	2830
doc/html/dir_f30d67bf21cfa80c94c2f56f2c05062d.js	2831
doc/html/dir_ffa09c8575bbfbed340eb641d2e0178f.js	2831
doc/html/dynsections.js	2831
doc/html/errormacros_8h.js	2832
doc/html/ev_cv_off_8c.js	2833
doc/html/eval_2_crv_8c.js	2833
doc/html/evalcrvarc_8c.js	2833
doc/html/evaluate_8cpp.js	2834
doc/html/example01_8cpp.js	2834
doc/html/example02_8cpp.js	2835
doc/html/example03_8cpp.js	2835
doc/html/example04_8cpp.js	2836
doc/html/example05_8cpp.js	2836
doc/html/example06_8cpp.js	2836
doc/html/example07_8cpp.js	2837
doc/html/example08_8cpp.js	2837
doc/html/example09_8cpp.js	2838
doc/html/example10_8cpp.js	2838
doc/html/example11_8cpp.js	2838
doc/html/example12_8cpp.js	2839
doc/html/example13_8cpp.js	2839
doc/html/example14_8cpp.js	2840
doc/html/example15_8cpp.js	2840
doc/html/example_8cpp.js	2840
doc/html/examples.js	2841
doc/html/extremal_pt_surf_surf_8h.js	2841
doc/html/fft_8cpp.js	2842
doc/html/files.js	2842
doc/html/ft_curve_8h.js	2843
doc/html/ft_message_8h.js	2843
doc/html/ft_planar_graph_8h.js	2844
doc/html/ft_point_set_8h.js	2844
doc/html/ft_surface_set_8h.js	2845
doc/html/ft_tang_priority_8h.js	2845
doc/html/ft_volume_tools_8h.js	2845
doc/html/functions_dup.js	2846
doc/html/functions_func.js	2847
doc/html/functions_vars.js	2847
doc/html/garch_8cpp.js	2848
doc/html/generic_graph_algorithms_8h.js	2849
doc/html/generic_graph_algorithms_implementation_8h.js	2849
doc/html/gl_aux_8cpp.js	2850
doc/html/gl_aux_8h.js	2850
doc/html/globals_defs.js	2851
doc/html/globals_dup.js	2851
doc/html/globals_func.js	2852
doc/html/glutils_8cpp.js	2853
doc/html/glutils_8h.js	2853
doc/html/gotools-core_2include_2_go_tools_2geometry_2copyright_8h.js	2854

doc/html/group__rbd__common.js	2854
doc/html/gv_application_8h.js	2855
doc/html/gv_parametric_surface_paintable_8h.js	2855
doc/html/gv_standard_mouse_handler_8h.js	2855
doc/html/gv_texture_8h.js	2856
doc/html/gv_utilities_8h.js	2857
doc/html/hholder_8cpp.js	2857
doc/html/hierarchy.js	2858
doc/html/hp__s1880_8c.js	2858
doc/html/igeslib__doxymain_8h.js	2858
doc/html/include_8h.js	2859
doc/html/intjoinper_8c.js	2859
doc/html/jacobi_8cpp.js	2859
doc/html/jquery.js	2860
doc/html/jquery_8js.js	2884
doc/html/main_8cpp.js	2885
doc/html/make3_d_8c.js	2885
doc/html/makecvkreg_8c.js	2885
doc/html/makesikreg_8c.js	2886
doc/html/maketracks_8c.js	2886
doc/html/mk__cv__cycl_8c.js	2887
doc/html/modules.js	2887
doc/html/mouse_8cpp.js	2887
doc/html/mouse_8h.js	2888
doc/html/myexcept_8cpp.js	2889
doc/html/myexcept_8h.js	2889
doc/html/namespace_go.js	2890
doc/html/namespace_go_1_1_coons_patch_gen.js	2890
doc/html/namespace_go_1_1_l_r_spline_utils.js	2891
doc/html/namespace_go_1_1_box_structuring.js	2891
doc/html/namespace_n_e_w_m_a_t.js	2891
doc/html/namespace_r_b_d__c_o_m_m_o_n.js	2892
doc/html/namespacehed.js	2892
doc/html/namespacehetriang.js	2893
doc/html/namespacemembers_dup.js	2893
doc/html/namespacemembers_eval.js	2894
doc/html/namespacemembers_func.js	2895
doc/html/namespaces.js	2895
doc/html/navtree.js	2896
doc/html/navtreedata.js	2900
doc/html/navtreeindex0.js	2900
doc/html/navtreeindex1.js	2901
doc/html/navtreeindex10.js	2901
doc/html/navtreeindex11.js	2901
doc/html/navtreeindex12.js	2901
doc/html/navtreeindex13.js	2902
doc/html/navtreeindex14.js	2902
doc/html/navtreeindex15.js	2902
doc/html/navtreeindex16.js	2903
doc/html/navtreeindex17.js	2903
doc/html/navtreeindex18.js	2903
doc/html/navtreeindex19.js	2903
doc/html/navtreeindex2.js	2904
doc/html/navtreeindex20.js	2904
doc/html/navtreeindex21.js	2904
doc/html/navtreeindex22.js	2905
doc/html/navtreeindex23.js	2905
doc/html/navtreeindex24.js	2905

doc/html/navtreeindex25.js	2905
doc/html/navtreeindex26.js	2906
doc/html/navtreeindex27.js	2906
doc/html/navtreeindex28.js	2906
doc/html/navtreeindex29.js	2907
doc/html/navtreeindex3.js	2907
doc/html/navtreeindex30.js	2907
doc/html/navtreeindex31.js	2907
doc/html/navtreeindex32.js	2908
doc/html/navtreeindex33.js	2908
doc/html/navtreeindex34.js	2908
doc/html/navtreeindex35.js	2909
doc/html/navtreeindex36.js	2909
doc/html/navtreeindex37.js	2909
doc/html/navtreeindex38.js	2909
doc/html/navtreeindex39.js	2910
doc/html/navtreeindex4.js	2910
doc/html/navtreeindex40.js	2910
doc/html/navtreeindex41.js	2911
doc/html/navtreeindex42.js	2911
doc/html/navtreeindex43.js	2911
doc/html/navtreeindex44.js	2911
doc/html/navtreeindex45.js	2912
doc/html/navtreeindex46.js	2912
doc/html/navtreeindex47.js	2912
doc/html/navtreeindex48.js	2913
doc/html/navtreeindex49.js	2913
doc/html/navtreeindex5.js	2913
doc/html/navtreeindex50.js	2913
doc/html/navtreeindex51.js	2914
doc/html/navtreeindex52.js	2914
doc/html/navtreeindex53.js	2914
doc/html/navtreeindex54.js	2915
doc/html/navtreeindex55.js	2915
doc/html/navtreeindex56.js	2915
doc/html/navtreeindex57.js	2915
doc/html/navtreeindex58.js	2916
doc/html/navtreeindex59.js	2916
doc/html/navtreeindex6.js	2916
doc/html/navtreeindex60.js	2917
doc/html/navtreeindex61.js	2917
doc/html/navtreeindex62.js	2917
doc/html/navtreeindex63.js	2917
doc/html/navtreeindex7.js	2918
doc/html/navtreeindex8.js	2918
doc/html/navtreeindex9.js	2918
doc/html/newfft_8cpp.js	2919
doc/html/newknots_8c.js	2919
doc/html/newmat1_8cpp.js	2919
doc/html/newmat2_8cpp.js	2920
doc/html/newmat3_8cpp.js	2920
doc/html/newmat4_8cpp.js	2921
doc/html/newmat5_8cpp.js	2921
doc/html/newmat6_8cpp.js	2921
doc/html/newmat7_8cpp.js	2922
doc/html/newmat8_8cpp.js	2922
doc/html/newmat9_8cpp.js	2923
doc/html/newmat_8h.js	2923

doc/html/newmatap_8h.js	2924
doc/html/newmatex_8cpp.js	2924
doc/html/newmatio_8h.js	2924
doc/html/newmatnl_8cpp.js	2925
doc/html/newmatnl_8h.js	2925
doc/html/newmatrc_8h.js	2925
doc/html/newmatrm_8cpp.js	2926
doc/html/newmatrm_8h.js	2926
doc/html/nl__ex_8cpp.js	2927
doc/html/orient_curves_8h.js	2927
doc/html/pickcrvsf_8c.js	2928
doc/html/pocrvtang_8c.js	2928
doc/html/precisio_8h.js	2929
doc/html/randomnoise_8h.js	2929
doc/html/raster_8h.js	2929
doc/html/refine__all_8c.js	2930
doc/html/resize.js	2930
doc/html/s1001_8c.js	2932
doc/html/s1011_8c.js	2932
doc/html/s1012_8c.js	2933
doc/html/s1013_8c.js	2933
doc/html/s1014_8c.js	2934
doc/html/s1015_8c.js	2934
doc/html/s1016_8c.js	2934
doc/html/s1017_8c.js	2935
doc/html/s1018_8c.js	2935
doc/html/s1021_8c.js	2936
doc/html/s1022_8c.js	2936
doc/html/s1023_8c.js	2936
doc/html/s1024_8c.js	2937
doc/html/s1025_8c.js	2937
doc/html/s1119_8c.js	2938
doc/html/s1161_8c.js	2938
doc/html/s1162_8c.js	2938
doc/html/s1172_8c.js	2939
doc/html/s1173_8c.js	2939
doc/html/s1174_8c.js	2940
doc/html/s1190_8c.js	2940
doc/html/s1192_8c.js	2940
doc/html/s1219_8c.js	2941
doc/html/s1220_8c.js	2941
doc/html/s1221_8c.js	2942
doc/html/s1222_8c.js	2942
doc/html/s1223_8c.js	2942
doc/html/s1224_8c.js	2943
doc/html/s1225_8c.js	2943
doc/html/s1226_8c.js	2944
doc/html/s1227_8c.js	2944
doc/html/s1231_8c.js	2944
doc/html/s1232_8c.js	2945
doc/html/s1233_8c.js	2945
doc/html/s1235_8c.js	2946
doc/html/s1236_8c.js	2946
doc/html/s1237_8c.js	2946
doc/html/s1238_8c.js	2947
doc/html/s1239_8c.js	2947
doc/html/s1240_8c.js	2948
doc/html/s1241_8c.js	2948

doc/html/s1243_8c.js	2948
doc/html/s1244_8c.js	2949
doc/html/s1245_8c.js	2949
doc/html/s1251_8c.js	2950
doc/html/s1252_8c.js	2950
doc/html/s1301_8c.js	2950
doc/html/s1302_8c.js	2951
doc/html/s1303_8c.js	2951
doc/html/s1304_8c.js	2952
doc/html/s1305_8c.js	2952
doc/html/s1306_8c.js	2952
doc/html/s1307_8c.js	2953
doc/html/s1308_8c.js	2953
doc/html/s1309_8c.js	2954
doc/html/s1310_8c.js	2954
doc/html/s1311_8c.js	2954
doc/html/s1312_8c.js	2955
doc/html/s1313_8c.js	2955
doc/html/s1314_8c.js	2956
doc/html/s1315_8c.js	2956
doc/html/s1316_8c.js	2956
doc/html/s1317_8c.js	2957
doc/html/s1318_8c.js	2957
doc/html/s1319_8c.js	2958
doc/html/s1320_8c.js	2958
doc/html/s1321_8c.js	2958
doc/html/s1322_8c.js	2959
doc/html/s1323_8c.js	2959
doc/html/s1324_8c.js	2960
doc/html/s1325_8c.js	2960
doc/html/s1327_8c.js	2960
doc/html/s1328_8c.js	2961
doc/html/s1329_8c.js	2961
doc/html/s1330_8c.js	2962
doc/html/s1331_8c.js	2962
doc/html/s1332_8c.js	2962
doc/html/s1333_8c.js	2963
doc/html/s1333count_8c.js	2963
doc/html/s1333cycli_8c.js	2964
doc/html/s1334_8c.js	2964
doc/html/s1340_8c.js	2964
doc/html/s1341_8c.js	2965
doc/html/s1342_8c.js	2965
doc/html/s1343_8c.js	2966
doc/html/s1345_8c.js	2966
doc/html/s1346_8c.js	2966
doc/html/s1347_8c.js	2967
doc/html/s1348_8c.js	2967
doc/html/s1349_8c.js	2968
doc/html/s1350_8c.js	2968
doc/html/s1351_8c.js	2968
doc/html/s1352_8c.js	2969
doc/html/s1353_8c.js	2969
doc/html/s1354_8c.js	2970
doc/html/s1355_8c.js	2970
doc/html/s1356_8c.js	2970
doc/html/s1357_8c.js	2971
doc/html/s1358_8c.js	2971

doc/html/s1359_8c.js	2972
doc/html/s1360_8c.js	2972
doc/html/s1361_8c.js	2972
doc/html/s1362_8c.js	2973
doc/html/s1363_8c.js	2973
doc/html/s1364_8c.js	2974
doc/html/s1365_8c.js	2974
doc/html/s1366_8c.js	2974
doc/html/s1367_8c.js	2975
doc/html/s1369_8c.js	2975
doc/html/s1370_8c.js	2976
doc/html/s1371_8c.js	2976
doc/html/s1372_8c.js	2976
doc/html/s1373_8c.js	2977
doc/html/s1374_8c.js	2977
doc/html/s1375_8c.js	2978
doc/html/s1376_8c.js	2978
doc/html/s1377_8c.js	2978
doc/html/s1378_8c.js	2979
doc/html/s1379_8c.js	2979
doc/html/s1380_8c.js	2980
doc/html/s1381_8c.js	2980
doc/html/s1382_8c.js	2980
doc/html/s1383_8c.js	2981
doc/html/s1384_8c.js	2981
doc/html/s1385_8c.js	2982
doc/html/s1386_8c.js	2982
doc/html/s1387_8c.js	2982
doc/html/s1388_8c.js	2983
doc/html/s1389_8c.js	2983
doc/html/s1390_8c.js	2984
doc/html/s1391_8c.js	2984
doc/html/s1393_8c.js	2984
doc/html/s1399_8c.js	2985
doc/html/s1401_8c.js	2985
doc/html/s1421_8c.js	2986
doc/html/s1422_8c.js	2986
doc/html/s1424_8c.js	2986
doc/html/s1425_8c.js	2987
doc/html/s1435_8c.js	2987
doc/html/s1436_8c.js	2988
doc/html/s1437_8c.js	2988
doc/html/s1438_8c.js	2988
doc/html/s1439_8c.js	2989
doc/html/s1440_8c.js	2989
doc/html/s1450_8c.js	2990
doc/html/s1451_8c.js	2990
doc/html/s1452_8c.js	2990
doc/html/s1500_8c.js	2991
doc/html/s1501_8c.js	2991
doc/html/s1502_8c.js	2992
doc/html/s1503_8c.js	2992
doc/html/s1504_8c.js	2992
doc/html/s1505_8c.js	2993
doc/html/s1506_8c.js	2993
doc/html/s1507_8c.js	2994
doc/html/s1508_8c.js	2994
doc/html/s1510_8c.js	2994

doc/html/s1511_8c.js	2995
doc/html/s1512_8c.js	2995
doc/html/s1513_8c.js	2996
doc/html/s1514_8c.js	2996
doc/html/s1515_8c.js	2996
doc/html/s1516_8c.js	2997
doc/html/s1517_8c.js	2997
doc/html/s1518_8c.js	2998
doc/html/s1520_8c.js	2998
doc/html/s1521_8c.js	2998
doc/html/s1522_8c.js	2999
doc/html/s1528_8c.js	2999
doc/html/s1529_8c.js	3000
doc/html/s1530_8c.js	3000
doc/html/s1531_8c.js	3000
doc/html/s1534_8c.js	3001
doc/html/s1535_8c.js	3001
doc/html/s1536_8c.js	3002
doc/html/s1537_8c.js	3002
doc/html/s1538_8c.js	3002
doc/html/s1539_8c.js	3003
doc/html/s1540_8c.js	3003
doc/html/s1541_8c.js	3004
doc/html/s1542_8c.js	3004
doc/html/s1600_8c.js	3004
doc/html/s1601_8c.js	3005
doc/html/s1602_8c.js	3005
doc/html/s1603_8c.js	3006
doc/html/s1604_8c.js	3006
doc/html/s1605_8c.js	3006
doc/html/s1606_8c.js	3007
doc/html/s1607_8c.js	3007
doc/html/s1608_8c.js	3008
doc/html/s1609_8c.js	3008
doc/html/s1611_8c.js	3008
doc/html/s1612_8c.js	3009
doc/html/s1613_8c.js	3009
doc/html/s1613bez_8c.js	3010
doc/html/s1614_8c.js	3010
doc/html/s1615_8c.js	3010
doc/html/s1616_8c.js	3011
doc/html/s1617_8c.js	3011
doc/html/s1618_8c.js	3012
doc/html/s1619_8c.js	3012
doc/html/s1620_8c.js	3012
doc/html/s1630_8c.js	3013
doc/html/s1631_8c.js	3013
doc/html/s1700_8c.js	3014
doc/html/s1701_8c.js	3014
doc/html/s1705_8c.js	3014
doc/html/s1706_8c.js	3015
doc/html/s1707_8c.js	3015
doc/html/s1708_8c.js	3016
doc/html/s1710_8c.js	3016
doc/html/s1711_8c.js	3016
doc/html/s1712_8c.js	3017
doc/html/s1713_8c.js	3017
doc/html/s1714_8c.js	3018

doc/html/s1715_8c.js	3018
doc/html/s1716_8c.js	3018
doc/html/s1720_8c.js	3019
doc/html/s1730_8c.js	3019
doc/html/s1731_8c.js	3020
doc/html/s1732_8c.js	3020
doc/html/s1733_8c.js	3020
doc/html/s1741_8c.js	3021
doc/html/s1750_8c.js	3021
doc/html/s1753_8c.js	3022
doc/html/s1754_8c.js	3022
doc/html/s1755_8c.js	3022
doc/html/s17702d_8c.js	3023
doc/html/s1770_8c.js	3023
doc/html/s1771_8c.js	3024
doc/html/s1772_8c.js	3024
doc/html/s1773_8c.js	3025
doc/html/s1774_8c.js	3025
doc/html/s1775_8c.js	3026
doc/html/s1780_8c.js	3026
doc/html/s1785_8c.js	3027
doc/html/s1786_8c.js	3027
doc/html/s1787_8c.js	3027
doc/html/s1788_8c.js	3028
doc/html/s1789_8c.js	3028
doc/html/s1790_8c.js	3029
doc/html/s1791_8c.js	3029
doc/html/s1792_8c.js	3029
doc/html/s1795_8c.js	3030
doc/html/s1796_8c.js	3030
doc/html/s1797_8c.js	3031
doc/html/s1830_8c.js	3031
doc/html/s1834_8c.js	3031
doc/html/s1839_8c.js	3032
doc/html/s1840_8c.js	3032
doc/html/s1850_8c.js	3033
doc/html/s1851_8c.js	3033
doc/html/s1852_8c.js	3033
doc/html/s1853_8c.js	3034
doc/html/s1854_8c.js	3034
doc/html/s1855_8c.js	3035
doc/html/s1856_8c.js	3035
doc/html/s1857_8c.js	3035
doc/html/s1858_8c.js	3036
doc/html/s1859_8c.js	3036
doc/html/s1860_8c.js	3037
doc/html/s1870_8c.js	3037
doc/html/s1871_8c.js	3037
doc/html/s1880_8c.js	3038
doc/html/s1890_8c.js	3038
doc/html/s1891_8c.js	3039
doc/html/s1893_8c.js	3039
doc/html/s1894_8c.js	3039
doc/html/s1896_8c.js	3040
doc/html/s1897_8c.js	3040
doc/html/s1900_8c.js	3041
doc/html/s1901_8c.js	3041
doc/html/s1902_8c.js	3041

doc/html/s1903_8c.js	3042
doc/html/s1904_8c.js	3042
doc/html/s1905_8c.js	3043
doc/html/s1906_8c.js	3043
doc/html/s1907_8c.js	3043
doc/html/s1908_8c.js	3044
doc/html/s1909_8c.js	3044
doc/html/s1910_8c.js	3045
doc/html/s1911_8c.js	3045
doc/html/s1912_8c.js	3045
doc/html/s1916_8c.js	3046
doc/html/s1917_8c.js	3046
doc/html/s1918_8c.js	3047
doc/html/s1919_8c.js	3047
doc/html/s1920_8c.js	3047
doc/html/s1921_8c.js	3048
doc/html/s1924_8c.js	3048
doc/html/s1925_8c.js	3049
doc/html/s1926_8c.js	3049
doc/html/s1927_8c.js	3049
doc/html/s1930_8c.js	3050
doc/html/s1931_8c.js	3050
doc/html/s1931unit_8c.js	3051
doc/html/s1932_8c.js	3051
doc/html/s1933_8c.js	3051
doc/html/s1934_8c.js	3052
doc/html/s1935_8c.js	3052
doc/html/s1936_8c.js	3053
doc/html/s1937_8c.js	3053
doc/html/s1938_8c.js	3053
doc/html/s1940_8c.js	3054
doc/html/s1941_8c.js	3054
doc/html/s1942_8c.js	3055
doc/html/s1943_8c.js	3055
doc/html/s1944_8c.js	3055
doc/html/s1945_8c.js	3056
doc/html/s1946_8c.js	3056
doc/html/s1947_8c.js	3057
doc/html/s1948_8c.js	3057
doc/html/s1949_8c.js	3057
doc/html/s1950_8c.js	3058
doc/html/s1951_8c.js	3058
doc/html/s1953_8c.js	3059
doc/html/s1954_8c.js	3059
doc/html/s1955_8c.js	3059
doc/html/s1956_8c.js	3060
doc/html/s1957_8c.js	3060
doc/html/s1958_8c.js	3061
doc/html/s1959_8c.js	3061
doc/html/s1960_8c.js	3061
doc/html/s1961_8c.js	3062
doc/html/s1962_8c.js	3062
doc/html/s1963_8c.js	3063
doc/html/s1965_8c.js	3063
doc/html/s1966_8c.js	3063
doc/html/s1967_8c.js	3064
doc/html/s1968_8c.js	3064
doc/html/s1986_8c.js	3065

doc/html/s1987_8c.js	3065
doc/html/s1988_8c.js	3065
doc/html/s1989_8c.js	3066
doc/html/s1990_8c.js	3066
doc/html/s1991_8c.js	3067
doc/html/s1992_8c.js	3067
doc/html/s1993_8c.js	3067
doc/html/s1994_8c.js	3068
doc/html/s2500_8c.js	3068
doc/html/s2501_8c.js	3069
doc/html/s2502_8c.js	3069
doc/html/s2503_8c.js	3069
doc/html/s2504_8c.js	3070
doc/html/s2505_8c.js	3070
doc/html/s2506_8c.js	3071
doc/html/s2507_8c.js	3071
doc/html/s2508_8c.js	3071
doc/html/s2509_8c.js	3072
doc/html/s2510_8c.js	3072
doc/html/s2511_8c.js	3073
doc/html/s2512_8c.js	3073
doc/html/s2513_8c.js	3073
doc/html/s2514_8c.js	3074
doc/html/s2515_8c.js	3074
doc/html/s2516_8c.js	3075
doc/html/s2532_8c.js	3075
doc/html/s2533_8c.js	3075
doc/html/s2534_8c.js	3076
doc/html/s2535_8c.js	3076
doc/html/s2536_8c.js	3077
doc/html/s2540_8c.js	3077
doc/html/s2541_8c.js	3077
doc/html/s2542_8c.js	3078
doc/html/s2543_8c.js	3078
doc/html/s2544_8c.js	3079
doc/html/s2545_8c.js	3079
doc/html/s2550_8c.js	3079
doc/html/s2551_8c.js	3080
doc/html/s2553_8c.js	3080
doc/html/s2554_8c.js	3081
doc/html/s2555_8c.js	3081
doc/html/s2556_8c.js	3081
doc/html/s2557_8c.js	3082
doc/html/s2558_8c.js	3082
doc/html/s2559_8c.js	3083
doc/html/s2560_8c.js	3083
doc/html/s2561_8c.js	3083
doc/html/s2562_8c.js	3084
doc/html/s6addcurve_8c.js	3084
doc/html/s6affdist_8c.js	3085
doc/html/s6ang_8c.js	3085
doc/html/s6angle_8c.js	3085
doc/html/s6castelja_8c.js	3086
doc/html/s6chpar_8c.js	3086
doc/html/s6crss_8c.js	3087
doc/html/s6crvature_8c.js	3087
doc/html/s6crvcheck_8c.js	3087
doc/html/s6curvrad_8c.js	3088

doc/html/s6decomp_8c.js	3088
doc/html/s6degnorm_8c.js	3089
doc/html/s6dertopt_8c.js	3089
doc/html/s6diff_8c.js	3089
doc/html/s6dist_8c.js	3090
doc/html/s6dline_8c.js	3090
doc/html/s6dplane_8c.js	3091
doc/html/s6drawseq_8c.js	3091
doc/html/s6equal_8c.js	3091
doc/html/s6err_8c.js	3092
doc/html/s6existbox_8c.js	3092
doc/html/s6findfac_8c.js	3093
doc/html/s6fndintv_8c.js	3093
doc/html/s6herm_8c.js	3093
doc/html/s6herm__bez_8c.js	3094
doc/html/s6idcon_8c.js	3094
doc/html/s6idcpt_8c.js	3095
doc/html/s6idedg_8c.js	3095
doc/html/s6identify_8c.js	3095
doc/html/s6idget_8c.js	3096
doc/html/s6idint_8c.js	3096
doc/html/s6idklist_8c.js	3097
doc/html/s6idkpt_8c.js	3097
doc/html/s6idlis_8c.js	3097
doc/html/s6idnpt_8c.js	3098
doc/html/s6idput_8c.js	3098
doc/html/s6inv4_8c.js	3099
doc/html/s6invert_8c.js	3099
doc/html/s6knotmult_8c.js	3099
doc/html/s6length_8c.js	3100
doc/html/s6line_8c.js	3100
doc/html/s6lprj_8c.js	3101
doc/html/s6lufacp_8c.js	3101
doc/html/s6lusolp_8c.js	3101
doc/html/s6metric_8c.js	3102
doc/html/s6move_8c.js	3102
doc/html/s6mulvec_8c.js	3103
doc/html/s6mvec_8c.js	3103
doc/html/s6newbox_8c.js	3103
doc/html/s6norm_8c.js	3104
doc/html/s6ratder_8c.js	3104
doc/html/s6rotax_8c.js	3105
doc/html/s6rotmat_8c.js	3105
doc/html/s6schoen_8c.js	3105
doc/html/s6scpr_8c.js	3106
doc/html/s6sortpar_8c.js	3106
doc/html/s6sratder_8c.js	3107
doc/html/s6strider_8c.js	3107
doc/html/s6takunion_8c.js	3107
doc/html/s6twonorm_8c.js	3108
doc/html/s9adsimp_8c.js	3108
doc/html/s9adstep_8c.js	3109
doc/html/s9boundimp_8c.js	3109
doc/html/s9boundit_8c.js	3109
doc/html/s9clipimp_8c.js	3110
doc/html/s9clipit_8c.js	3110
doc/html/s9conmarch_8c.js	3111
doc/html/s9iterate_8c.js	3111

doc/html/s9iterimp_8c.js	3111
doc/html/s9smpknot_8c.js	3112
doc/html/s9surmarch_8c.js	3112
doc/html/sh1260_8c.js	3271
doc/html/sh1261_8c.js	3271
doc/html/sh1262_8c.js	3272
doc/html/sh1263_8c.js	3272
doc/html/sh1365_8c.js	3272
doc/html/sh1369_8c.js	3273
doc/html/sh1371_8c.js	3273
doc/html/sh1372_8c.js	3274
doc/html/sh1373_8c.js	3274
doc/html/sh1374_8c.js	3274
doc/html/sh1375_8c.js	3275
doc/html/sh1460_8c.js	3275
doc/html/sh1461_8c.js	3276
doc/html/sh1462_8c.js	3276
doc/html/sh1463_8c.js	3276
doc/html/sh1464_8c.js	3277
doc/html/sh1465_8c.js	3277
doc/html/sh1466_8c.js	3278
doc/html/sh1467_8c.js	3278
doc/html/sh1502_8c.js	3279
doc/html/sh1503_8c.js	3279
doc/html/sh1510_8c.js	3279
doc/html/sh1511_8c.js	3280
doc/html/sh1761_8c.js	3280
doc/html/sh1762_8c.js	3281
doc/html/sh1779_8c.js	3281
doc/html/sh1779_at_8c.js	3281
doc/html/sh1780_8c.js	3282
doc/html/sh1780_at_8c.js	3282
doc/html/sh1781_8c.js	3283
doc/html/sh1781_at_8c.js	3283
doc/html/sh1782_8c.js	3283
doc/html/sh1783_8c.js	3284
doc/html/sh1784_8c.js	3284
doc/html/sh1786_8c.js	3285
doc/html/sh1787_8c.js	3285
doc/html/sh1790_8c.js	3285
doc/html/sh1830_8c.js	3286
doc/html/sh1831_8c.js	3286
doc/html/sh1834_8c.js	3287
doc/html/sh1839_8c.js	3287
doc/html/sh1850_8c.js	3287
doc/html/sh1851_8c.js	3288
doc/html/sh1852_8c.js	3288
doc/html/sh1853_8c.js	3289
doc/html/sh1854_8c.js	3289
doc/html/sh1855_8c.js	3289
doc/html/sh1856_8c.js	3290
doc/html/sh1857_8c.js	3290
doc/html/sh1858_8c.js	3291
doc/html/sh1859_8c.js	3291
doc/html/sh1860_8c.js	3291
doc/html/sh1870_8c.js	3292
doc/html/sh1871_8c.js	3292
doc/html/sh1922_8c.js	3293

doc/html/sh1923_8c.js	3293
doc/html/sh1924_8c.js	3293
doc/html/sh1925_8c.js	3294
doc/html/sh1926_8c.js	3294
doc/html/sh1927_8c.js	3295
doc/html/sh1928_8c.js	3295
doc/html/sh1929_8c.js	3295
doc/html/sh1930_8c.js	3296
doc/html/sh1992_8c.js	3296
doc/html/sh1993_8c.js	3297
doc/html/sh1994_8c.js	3297
doc/html/sh6clvert_8c.js	3297
doc/html/sh6comedg_8c.js	3298
doc/html/sh6connect_8c.js	3298
doc/html/sh6count_8c.js	3299
doc/html/sh6cvvert_8c.js	3299
doc/html/sh6degen_8c.js	3299
doc/html/sh6disconn_8c.js	3300
doc/html/sh6edgpnt_8c.js	3300
doc/html/sh6edgred_8c.js	3301
doc/html/sh6evalint_8c.js	3301
doc/html/sh6floop_8c.js	3301
doc/html/sh6fndsplt_8c.js	3302
doc/html/sh6getgeom_8c.js	3302
doc/html/sh6getlist_8c.js	3303
doc/html/sh6getmain_8c.js	3303
doc/html/sh6getnbrs_8c.js	3303
doc/html/sh6getnext_8c.js	3304
doc/html/sh6getothr_8c.js	3304
doc/html/sh6getprev_8c.js	3305
doc/html/sh6gettop_8c.js	3305
doc/html/sh6idaledg_8c.js	3305
doc/html/sh6idcon_8c.js	3306
doc/html/sh6idfcros_8c.js	3306
doc/html/sh6idget_8c.js	3307
doc/html/sh6idkpt_8c.js	3307
doc/html/sh6idlis_8c.js	3307
doc/html/sh6idnpt_8c.js	3308
doc/html/sh6idnwun_8c.js	3308
doc/html/sh6idput_8c.js	3309
doc/html/sh6idrcros_8c.js	3309
doc/html/sh6idsplit_8c.js	3309
doc/html/sh6idunite_8c.js	3310
doc/html/sh6insert_8c.js	3310
doc/html/sh6inspnt_8c.js	3311
doc/html/sh6iscnect_8c.js	3311
doc/html/sh6ishelp_8c.js	3311
doc/html/sh6isinsid_8c.js	3312
doc/html/sh6ismain_8c.js	3312
doc/html/sh6nmbhelp_8c.js	3313
doc/html/sh6nmbmain_8c.js	3313
doc/html/sh6ptobj_8c.js	3313
doc/html/sh6ptouchp_8c.js	3314
doc/html/sh6putsing_8c.js	3314
doc/html/sh6red_8c.js	3315
doc/html/sh6remcon_8c.js	3315
doc/html/sh6rempnt_8c.js	3315
doc/html/sh6sepcrv_8c.js	3316

doc/html/sh6setcnsd_8c.js	3316
doc/html/sh6setdir_8c.js	3317
doc/html/sh6settop_8c.js	3317
doc/html/sh6splgeo_8c.js	3317
doc/html/sh6tohelp_8c.js	3318
doc/html/sh6tomain_8c.js	3318
doc/html/sh6topohlp_8c.js	3319
doc/html/sh6trmlist_8c.js	3319
doc/html/sh__1d__div_8c.js	3319
doc/html/sh__div__crv_8c.js	3320
doc/html/sh__set__at_8c.js	3320
doc/html/shape_8c.js	3321
doc/html/shcheckput_8c.js	3321
doc/html/shchecktyp_8c.js	3321
doc/html/shcsfsing_8c.js	3322
doc/html/shdivsurf_8c.js	3322
doc/html/shevalc_8c.js	3323
doc/html/shmkhlppts_8c.js	3323
doc/html/shsing_8c.js	3323
doc/html/sisl_8h.js	3324
doc/html/sisl__aux_8cpp.js	3324
doc/html/sisl__aux_8h.js	3325
doc/html/sisl__file__io_8h.js	3325
doc/html/sisl__view__demo_8cpp.js	3325
doc/html/sisl_p_8h.js	3326
doc/html/sl__ex_8cpp.js	3326
doc/html/sleep_8h.js	3326
doc/html/solution_8cpp.js	3327
doc/html/sort_8cpp.js	3327
doc/html/spli__silh_8c.js	3328
doc/html/spline2mesh_8h.js	3328
doc/html/struct_go_1_1_adjacency_info.js	3329
doc/html/struct_go_1_1_alg2_d_elem.js	3329
doc/html/struct_go_1_1_alg3_d_elem.js	3330
doc/html/struct_go_1_1_basis_derivs.js	3330
doc/html/struct_go_1_1_basis_derivs2.js	3331
doc/html/struct_go_1_1_basis_derivs_sf.js	3331
doc/html/struct_go_1_1_basis_derivs_sf2.js	3332
doc/html/struct_go_1_1_basis_pts.js	3332
doc/html/struct_go_1_1_basis_pts_sf.js	3333
doc/html/struct_go_1_1_boundary_function_int.js	3333
doc/html/struct_go_1_1_boundary_geom_int.js	3334
doc/html/struct_go_1_1_boundary_intersection_data.js	3334
doc/html/struct_go_1_1_cached_interval.js	3335
doc/html/struct_go_1_1_composite_model_file_handler_1_1_sfcvinfo.js	3335
doc/html/struct_go_1_1_entity_list.js	3336
doc/html/struct_go_1_1_face_connectivity.js	3336
doc/html/struct_go_1_1_factorial.js	3337
doc/html/struct_go_1_1_factorial_3_011_01_4.js	3337
doc/html/struct_go_1_1_g_pos.js	3338
doc/html/struct_go_1_1_i_g_e_sdirentry.js	3338
doc/html/struct_go_1_1_i_g_e_sheader.js	3339
doc/html/struct_go_1_1_int_pt_info.js	3339
doc/html/struct_go_1_1_l_r_spline_surface_1_1_b_s_key.js	3340
doc/html/struct_go_1_1_l_r_spline_surface_1_1_elem_key.js	3340
doc/html/struct_go_1_1_l_r_spline_surface_1_1_refinement2_d.js	3341
doc/html/struct_go_1_1_l_r_spline_surface_1_1double__pair__hash.js	3341
doc/html/struct_go_1_1_l_r_spline_utils_1_1support__compare.js	3342

doc/html/struct_go_1_1_l_s_smooth_data.js	3342
doc/html/struct_go_1_1_param_surface_1_1degenerate__info.js	3343
doc/html/struct_go_1_1_registration_input.js	3343
doc/html/struct_go_1_1_registration_result.js	3344
doc/html/struct_go_1_1_rotation_info.js	3344
doc/html/struct_go_1_1_sample_point_data.js	3345
doc/html/struct_go_1_1_second_order_properties.js	3345
doc/html/struct_go_1_1_sing_box.js	3346
doc/html/struct_go_1_1_sing_union.js	3346
doc/html/struct_go_1_1_volume_adjacency_info.js	3346
doc/html/struct_go_1_1cv_set_constraint.js	3347
doc/html/struct_go_1_1go_iterator_traits.js	3348
doc/html/struct_go_1_1go_iterator_traits_3_01_t_01_5_01_4.js	3348
doc/html/struct_go_1_1go_iterator_traits_3_01const_01_t_01_5_01_4.js	3349
doc/html/struct_go_1_1pre_evaluation_sf.js	3349
doc/html/struct_go_1_1pre_evaluation_vol.js	3350
doc/html/struct_go_1_1raw__pointer__comp.js	3350
doc/html/struct_go_1_1side_constraint.js	3351
doc/html/struct_go_1_1side_constraint_set.js	3351
doc/html/struct_go_1_1tp_tolerances.js	3351
doc/html/struct_go_1_1tp_topological_info.js	3352
doc/html/struct_s_i_s_l_curve.js	3352
doc/html/struct_s_i_s_l_edge.js	3353
doc/html/struct_s_i_s_l_intcurve.js	3353
doc/html/struct_s_i_s_l_intdat.js	3354
doc/html/struct_s_i_s_l_intlist.js	3354
doc/html/struct_s_i_s_l_intpt.js	3355
doc/html/struct_s_i_s_l_intsurf.js	3356
doc/html/struct_s_i_s_l_object.js	3356
doc/html/struct_s_i_s_l_point.js	3356
doc/html/struct_s_i_s_l_ptedge.js	3357
doc/html/struct_s_i_s_l_surf.js	3357
doc/html/struct_s_i_s_l_track.js	3358
doc/html/struct_s_i_s_l_trimpar.js	3359
doc/html/struct_s_i_s_l_box.js	3359
doc/html/struct_s_i_s_l_dir.js	3359
doc/html/structgv_color.js	3360
doc/html/structhed_1_1_t_t_traits.js	3360
doc/html/structhetriang_1_1_t_t_traits.js	3361
doc/html/structrank__info.js	3361
doc/html/submat_8cpp.js	3362
doc/html/svd_8cpp.js	3362
doc/html/test_exc_8cpp.js	3362
doc/html/test_suite_8h.js	3363
doc/html/timeutils_8h.js	3363
doc/html/tmt1_8cpp.js	3364
doc/html/tmt2_8cpp.js	3364
doc/html/tmt3_8cpp.js	3364
doc/html/tmt4_8cpp.js	3365
doc/html/tmt5_8cpp.js	3365
doc/html/tmt6_8cpp.js	3366
doc/html/tmt7_8cpp.js	3366
doc/html/tmt8_8cpp.js	3366
doc/html/tmt9_8cpp.js	3367
doc/html/tmt_8cpp.js	3367
doc/html/tmt_8h.js	3368
doc/html/tmta_8cpp.js	3368
doc/html/tmtb_8cpp.js	3369

doc/html/tmtc_8cpp.js	3369
doc/html/tmtd_8cpp.js	3370
doc/html/tmte_8cpp.js	3370
doc/html/tmtf_8cpp.js	3370
doc/html/tmtg_8cpp.js	3371
doc/html/tmth_8cpp.js	3371
doc/html/tmti_8cpp.js	3372
doc/html/tmtj_8cpp.js	3372
doc/html/tmtk_8cpp.js	3372
doc/html/tmtl_8cpp.js	3373
doc/html/tmtm_8cpp.js	3373
doc/html/tp_joint_type_8h.js	3374
doc/html/tp_utils_8h.js	3374
doc/html/transfutils_8cpp.js	3375
doc/html/transfutils_8h.js	3375
doc/html/tstcyclknt_8c.js	3376
doc/html/ttl_8h.js	3376
doc/html/ttl_constr_8h.js	3377
doc/html/ttl_util_8h.js	3377
doc/html/search/all_0.js	3113
doc/html/search/all_1.js	3113
doc/html/search/all_10.js	3114
doc/html/search/all_11.js	3116
doc/html/search/all_12.js	3117
doc/html/search/all_13.js	3117
doc/html/search/all_14.js	3118
doc/html/search/all_15.js	3119
doc/html/search/all_16.js	3123
doc/html/search/all_17.js	3123
doc/html/search/all_18.js	3124
doc/html/search/all_19.js	3125
doc/html/search/all_1a.js	3125
doc/html/search/all_1b.js	3126
doc/html/search/all_1c.js	3127
doc/html/search/all_2.js	3127
doc/html/search/all_3.js	3129
doc/html/search/all_4.js	3129
doc/html/search/all_5.js	3132
doc/html/search/all_6.js	3133
doc/html/search/all_7.js	3135
doc/html/search/all_8.js	3136
doc/html/search/all_9.js	3139
doc/html/search/all_a.js	3141
doc/html/search/all_b.js	3141
doc/html/search/all_c.js	3142
doc/html/search/all_d.js	3143
doc/html/search/all_e.js	3145
doc/html/search/all_f.js	3146
doc/html/search/classes_0.js	3146
doc/html/search/classes_1.js	3147
doc/html/search/classes_10.js	3147
doc/html/search/classes_11.js	3148
doc/html/search/classes_12.js	3148
doc/html/search/classes_13.js	3148
doc/html/search/classes_14.js	3149
doc/html/search/classes_15.js	3149
doc/html/search/classes_16.js	3150
doc/html/search/classes_2.js	3150

doc/html/search/classes_3.js	3150
doc/html/search/classes_4.js	3151
doc/html/search/classes_5.js	3152
doc/html/search/classes_6.js	3152
doc/html/search/classes_7.js	3153
doc/html/search/classes_8.js	3153
doc/html/search/classes_9.js	3154
doc/html/search/classes_a.js	3154
doc/html/search/classes_b.js	3155
doc/html/search/classes_c.js	3155
doc/html/search/classes_d.js	3156
doc/html/search/classes_e.js	3157
doc/html/search/classes_f.js	3157
doc/html/search/defines_0.js	3158
doc/html/search/defines_1.js	3159
doc/html/search/defines_10.js	3159
doc/html/search/defines_11.js	3160
doc/html/search/defines_2.js	3160
doc/html/search/defines_3.js	3161
doc/html/search/defines_4.js	3161
doc/html/search/defines_5.js	3162
doc/html/search/defines_6.js	3162
doc/html/search/defines_7.js	3163
doc/html/search/defines_8.js	3163
doc/html/search/defines_9.js	3164
doc/html/search/defines_a.js	3164
doc/html/search/defines_b.js	3165
doc/html/search/defines_c.js	3166
doc/html/search/defines_d.js	3166
doc/html/search/defines_e.js	3167
doc/html/search/defines_f.js	3167
doc/html/search/enums_0.js	3168
doc/html/search/enums_1.js	3168
doc/html/search/enums_2.js	3168
doc/html/search/enums_3.js	3169
doc/html/search/enums_4.js	3169
doc/html/search/enums_5.js	3170
doc/html/search/enums_6.js	3170
doc/html/search/enums_7.js	3171
doc/html/search/enums_8.js	3171
doc/html/search/enums_9.js	3172
doc/html/search/enums_a.js	3172
doc/html/search/enums_b.js	3173
doc/html/search/enums_c.js	3173
doc/html/search/enums_d.js	3174
doc/html/search/enums_e.js	3174
doc/html/search/enums_f.js	3175
doc/html/search/enumvalues_0.js	3175
doc/html/search/enumvalues_1.js	3176
doc/html/search/enumvalues_10.js	3176
doc/html/search/enumvalues_11.js	3176
doc/html/search/enumvalues_12.js	3177
doc/html/search/enumvalues_13.js	3177
doc/html/search/enumvalues_14.js	3178
doc/html/search/enumvalues_15.js	3178
doc/html/search/enumvalues_16.js	3179
doc/html/search/enumvalues_17.js	3179
doc/html/search/enumvalues_18.js	3179

doc/html/search/enumvalues_2.js	3180
doc/html/search/enumvalues_3.js	3180
doc/html/search/enumvalues_4.js	3181
doc/html/search/enumvalues_5.js	3182
doc/html/search/enumvalues_6.js	3182
doc/html/search/enumvalues_7.js	3183
doc/html/search/enumvalues_8.js	3183
doc/html/search/enumvalues_9.js	3184
doc/html/search/enumvalues_a.js	3185
doc/html/search/enumvalues_b.js	3185
doc/html/search/enumvalues_c.js	3186
doc/html/search/enumvalues_d.js	3187
doc/html/search/enumvalues_e.js	3187
doc/html/search/enumvalues_f.js	3188
doc/html/search/files_0.js	3189
doc/html/search/files_1.js	3189
doc/html/search/files_10.js	3190
doc/html/search/files_11.js	3190
doc/html/search/files_12.js	3191
doc/html/search/files_13.js	3191
doc/html/search/files_14.js	3191
doc/html/search/files_15.js	3192
doc/html/search/files_2.js	3192
doc/html/search/files_3.js	3193
doc/html/search/files_4.js	3193
doc/html/search/files_5.js	3194
doc/html/search/files_6.js	3194
doc/html/search/files_7.js	3195
doc/html/search/files_8.js	3195
doc/html/search/files_9.js	3195
doc/html/search/files_a.js	3196
doc/html/search/files_b.js	3196
doc/html/search/files_c.js	3196
doc/html/search/files_d.js	3197
doc/html/search/files_e.js	3198
doc/html/search/files_f.js	3198
doc/html/search/functions_0.js	3198
doc/html/search/functions_1.js	3199
doc/html/search/functions_10.js	3200
doc/html/search/functions_11.js	3201
doc/html/search/functions_12.js	3201
doc/html/search/functions_13.js	3203
doc/html/search/functions_14.js	3204
doc/html/search/functions_15.js	3207
doc/html/search/functions_16.js	3207
doc/html/search/functions_17.js	3207
doc/html/search/functions_18.js	3209
doc/html/search/functions_19.js	3209
doc/html/search/functions_1a.js	3210
doc/html/search/functions_1b.js	3211
doc/html/search/functions_2.js	3211
doc/html/search/functions_3.js	3212
doc/html/search/functions_4.js	3217
doc/html/search/functions_5.js	3218
doc/html/search/functions_6.js	3220
doc/html/search/functions_7.js	3221
doc/html/search/functions_8.js	3224
doc/html/search/functions_9.js	3224

doc/html/search/functions_a.js	3225
doc/html/search/functions_b.js	3226
doc/html/search/functions_c.js	3227
doc/html/search/functions_d.js	3229
doc/html/search/functions_e.js	3230
doc/html/search/functions_f.js	3231
doc/html/search/groups_0.js	3232
doc/html/search/namespaces_0.js	3232
doc/html/search/namespaces_1.js	3233
doc/html/search/namespaces_2.js	3233
doc/html/search/namespaces_3.js	3233
doc/html/search/namespaces_4.js	3234
doc/html/search/namespaces_5.js	3234
doc/html/search/pages_0.js	3235
doc/html/search/pages_1.js	3235
doc/html/search/pages_2.js	3235
doc/html/search/pages_3.js	3236
doc/html/search/pages_4.js	3236
doc/html/search/related_0.js	3237
doc/html/search/related_1.js	3237
doc/html/search/related_2.js	3238
doc/html/search/related_3.js	3239
doc/html/search/related_4.js	3240
doc/html/search/related_5.js	3240
doc/html/search/related_6.js	3241
doc/html/search/related_7.js	3241
doc/html/search/related_8.js	3242
doc/html/search/related_9.js	3242
doc/html/search/related_a.js	3243
doc/html/search/related_b.js	3243
doc/html/search/related_c.js	3244
doc/html/search/related_d.js	3245
doc/html/search/related_e.js	3245
doc/html/search/related_f.js	3246
doc/html/search/search.js	3246
doc/html/search/searchdata.js	3248
doc/html/search/typedefs_0.js	3249
doc/html/search/typedefs_1.js	3249
doc/html/search/typedefs_2.js	3250
doc/html/search/typedefs_3.js	3250
doc/html/search/typedefs_4.js	3251
doc/html/search/typedefs_5.js	3251
doc/html/search/typedefs_6.js	3252
doc/html/search/typedefs_7.js	3252
doc/html/search/typedefs_8.js	3253
doc/html/search/typedefs_9.js	3253
doc/html/search/typedefs_a.js	3254
doc/html/search/typedefs_b.js	3254
doc/html/search/typedefs_c.js	3254
doc/html/search/typedefs_d.js	3255
doc/html/search/typedefs_e.js	3255
doc/html/search/typedefs_f.js	3256
doc/html/search/variables_0.js	3257
doc/html/search/variables_1.js	3257
doc/html/search/variables_10.js	3258
doc/html/search/variables_11.js	3258
doc/html/search/variables_12.js	3259
doc/html/search/variables_13.js	3259

doc/html/search/variables_14.js	3260
doc/html/search/variables_15.js	3260
doc/html/search/variables_16.js	3261
doc/html/search/variables_17.js	3262
doc/html/search/variables_18.js	3262
doc/html/search/variables_19.js	3263
doc/html/search/variables_2.js	3263
doc/html/search/variables_3.js	3264
doc/html/search/variables_4.js	3264
doc/html/search/variables_5.js	3264
doc/html/search/variables_6.js	3265
doc/html/search/variables_7.js	3266
doc/html/search/variables_8.js	3267
doc/html/search/variables_9.js	3267
doc/html/search/variables_a.js	3268
doc/html/search/variables_b.js	3268
doc/html/search/variables_c.js	3269
doc/html/search/variables_d.js	3269
doc/html/search/variables_e.js	3270
doc/html/search/variables_f.js	3270
gotools-core/doc/html/jquery.js	2872
gotools-core/include/GoTools/creators/AdaptCurve.h	3378
gotools-core/include/GoTools/creators/ApproxCrvToSeqs.h	3378
gotools-core/include/GoTools/creators/ApproxCurve.h	3379
gotools-core/include/GoTools/creators/ApproxSurf.h	3380
gotools-core/include/GoTools/creators/ConstraintDefinitions.h	3380
gotools-core/include/GoTools/creators/CoonsPatchGen.h	3381
gotools-core/include/GoTools/creators/CreatorsOffsetUtils.h	3383
gotools-core/include/GoTools/creators/CreatorsUtils.h	3384
gotools-core/include/GoTools/creators/CrossTangentOffset.h	3385
gotools-core/include/GoTools/creators/CrossTanOffDist.h	3385
gotools-core/include/GoTools/creators/CurveCreators.h	3386
gotools-core/include/GoTools/creators/EvalCurve.h	3387
gotools-core/include/GoTools/creators/EvalCurveSet.h	3388
gotools-core/include/GoTools/creators/EvalParamCurve.h	3388
gotools-core/include/GoTools/creators/EvalSurface.h	3389
gotools-core/include/GoTools/creators/HahnsSurfaceGen.h	3390
gotools-core/include/GoTools/creators/HermiteAppC.h	3391
gotools-core/include/GoTools/creators/HermiteAppS.h	3391
gotools-core/include/GoTools/creators/HermiteGrid1D.h	3392
gotools-core/include/GoTools/creators/HermiteGrid1DMulti.h	3393
gotools-core/include/GoTools/creators/HermiteGrid2D.h	3394
gotools-core/include/GoTools/creators/IntCrvEvaluator.h	3395
gotools-core/include/GoTools/creators/Integrate.h	3396
gotools-core/include/GoTools/creators/LiftCurve.h	3397
gotools-core/include/GoTools/creators/LoftSurfaceCreator.h	3397
gotools-core/include/GoTools/creators/ModifySurf.h	3398
gotools-core/include/GoTools/creators/ProjectCurve.h	3399
gotools-core/include/GoTools/creators/ProjectCurveAndCrossTan.h	3399
gotools-core/include/GoTools/creators/ProjectIntersectionCurve.h	3400
gotools-core/include/GoTools/creators/SmoothCurve.h	3401
gotools-core/include/GoTools/creators/SmoothCurveSet.h	3402
gotools-core/include/GoTools/creators/SmoothSurf.h	3402
gotools-core/include/GoTools/creators/SmoothSurfSet.h	3403
gotools-core/include/GoTools/creators/SmoothTransition.h	3403
gotools-core/include/GoTools/creators/SolveBCG.h	3404
gotools-core/include/GoTools/creators/SolveCG.h	3405
gotools-core/include/GoTools/creators/SolveCGCO.h	3406

gotools-core/include/GoTools/creators/SpaceIntCrv.h	3406
gotools-core/include/GoTools/creators/SurfaceCreators.h	3407
gotools-core/include/GoTools/creators/TrimCurve.h	3408
gotools-core/include/GoTools/geometry/BoundedCurve.h	3408
gotools-core/include/GoTools/geometry/BoundedSurface.h	3409
gotools-core/include/GoTools/geometry/BoundedUtils.h	3410
gotools-core/include/GoTools/geometry/BsplineBasis.h	3412
gotools-core/include/GoTools/geometry/Circle.h	3414
gotools-core/include/GoTools/geometry/ClassType.h	3415
gotools-core/include/GoTools/geometry/ClosestPoint.h	3415
gotools-core/include/GoTools/geometry/CompositeSurface.h	3417
gotools-core/include/GoTools/geometry/Cone.h	3418
gotools-core/include/GoTools/geometry/copyright.h	3419
gotools-core/include/GoTools/geometry/Curvature.h	3419
gotools-core/include/GoTools/geometry/CurvatureAnalysis.h	3420
gotools-core/include/GoTools/geometry/CurveBoundedDomain.h	3421
gotools-core/include/GoTools/geometry/CurveInterpolator.h	3421
gotools-core/include/GoTools/geometry/CurveLoop.h	3422
gotools-core/include/GoTools/geometry/CurveOnSurface.h	3423
gotools-core/include/GoTools/geometry/Cylinder.h	3424
gotools-core/include/GoTools/geometry/Disc.h	3425
gotools-core/include/GoTools/geometry/Domain.h	3426
gotools-core/include/GoTools/geometry/ElementaryCurve.h	3427
gotools-core/include/GoTools/geometry/ElementarySurface.h	3428
gotools-core/include/GoTools/geometry/Ellipse.h	3429
gotools-core/include/GoTools/geometry/examples_core_doxygen.h	3430
gotools-core/include/GoTools/geometry/examples_doxygen.h	3430
gotools-core/include/GoTools/geometry/extremalPtSurfSurf.h	3430
gotools-core/include/GoTools/geometry/Factory.h	3431
gotools-core/include/GoTools/geometry/GapRemoval.h	3431
gotools-core/include/GoTools/geometry/geometry_doxygen.h	3433
gotools-core/include/GoTools/geometry/GeometryTools.h	3433
gotools-core/include/GoTools/geometry/GeomObject.h	3435
gotools-core/include/GoTools/geometry/GoIntersections.h	3436
gotools-core/include/GoTools/geometry/GoTools.h	3438
gotools-core/include/GoTools/geometry/GoTools_version.h	3439
gotools-core/include/GoTools/geometry/HermiteInterpolator.h	3440
gotools-core/include/GoTools/geometry/Hyperbola.h	3440
gotools-core/include/GoTools/geometry/Interpolator.h	3441
gotools-core/include/GoTools/geometry/Line.h	3442
gotools-core/include/GoTools/geometry/LineCloud.h	3443
gotools-core/include/GoTools/geometry/LoopUtils.h	3444
gotools-core/include/GoTools/geometry/ObjectHeader.h	3445
gotools-core/include/GoTools/geometry/orientCurves.h	3447
gotools-core/include/GoTools/geometry/Parabola.h	3447
gotools-core/include/GoTools/geometry/ParamCurve.h	3448
gotools-core/include/GoTools/geometry/ParamSurface.h	3449
gotools-core/include/GoTools/geometry/Plane.h	3450
gotools-core/include/GoTools/geometry/PointCloud.h	3451
gotools-core/include/GoTools/geometry/PointOnCurve.h	3452
gotools-core/include/GoTools/geometry/PointSequence.h	3453
gotools-core/include/GoTools/geometry/RectDomain.h	3454
gotools-core/include/GoTools/geometry/RectGrid.h	3455
gotools-core/include/GoTools/geometry/SISL_code.h	3456
gotools-core/include/GoTools/geometry/sisl_file_io.h	3456
gotools-core/include/GoTools/geometry/SISLconversion.h	3457
gotools-core/include/GoTools/geometry/Sphere.h	3458
gotools-core/include/GoTools/geometry/SplineApproximator.h	3459

gotools-core/include/GoTools/geometry/SplineCurve.h	3460
gotools-core/include/GoTools/geometry/SplineDebugUtils.h	3460
gotools-core/include/GoTools/geometry/SplineInterpolator.h	3461
gotools-core/include/GoTools/geometry/SplineSurface.h	3462
gotools-core/include/GoTools/geometry/SplineUtils.h	3462
gotools-core/include/GoTools/geometry/Streamable.h	3464
gotools-core/include/GoTools/geometry/streamable_doxyman.h	3465
gotools-core/include/GoTools/geometry/SurfaceInterpolator.h	3465
gotools-core/include/GoTools/geometry/SurfaceOfLinearExtrusion.h	3465
gotools-core/include/GoTools/geometry/SurfaceOfRevolution.h	3466
gotools-core/include/GoTools/geometry/SurfaceTools.h	3468
gotools-core/include/GoTools/geometry/SweepSurfaceCreator.h	3469
gotools-core/include/GoTools/geometry/Torus.h	3470
gotools-core/include/GoTools/geometry/Utils.h	3471
gotools-core/include/GoTools/tesselator/2dpoly_for_s2m.h	3472
gotools-core/include/GoTools/tesselator/CurveTesselator.h	3474
gotools-core/include/GoTools/tesselator/GeneralMesh.h	3475
gotools-core/include/GoTools/tesselator/GenericTriMesh.h	3476
gotools-core/include/GoTools/tesselator/LineCloudTesselator.h	3477
gotools-core/include/GoTools/tesselator/LineStrip.h	3477
gotools-core/include/GoTools/tesselator/NoopTesselator.h	3479
gotools-core/include/GoTools/tesselator/ParametricSurfaceTesselator.h	3479
gotools-core/include/GoTools/tesselator/QuadMesh.h	3480
gotools-core/include/GoTools/tesselator/RectangularSurfaceTesselator.h	3482
gotools-core/include/GoTools/tesselator/RectGridTesselator.h	3483
gotools-core/include/GoTools/tesselator/RegularMesh.h	3483
gotools-core/include/GoTools/tesselator/spline2mesh.h	3485
gotools-core/include/GoTools/tesselator/Tesselator.h	3485
gotools-core/include/GoTools/tesselator/TesselatorUtils.h	3486
gotools-core/include/GoTools/utils/Array.h	3487
gotools-core/include/GoTools/utils/BaryCoordSystem.h	3489
gotools-core/include/GoTools/utils/BaryCoordSystemTriangle3D.h	3490
gotools-core/include/GoTools/utils/binom.h	3491
gotools-core/include/GoTools/utils/BoundingBox.h	3491
gotools-core/include/GoTools/utils/brent_minimize.h	3493
gotools-core/include/GoTools/utils/checks.h	3493
gotools-core/include/GoTools/utils/ClosestPointUtils.h	3495
gotools-core/include/GoTools/utils/CompositeBox.h	3496
gotools-core/include/GoTools/utils/config.h	3497
gotools-core/include/GoTools/utils/CoordinateSystem.h	3498
gotools-core/include/GoTools/utils/CPUclock.h	3499
gotools-core/include/GoTools/utils/CurvatureUtils.h	3499
gotools-core/include/GoTools/utils/DirectionCone.h	3500
gotools-core/include/GoTools/utils/errormacros.h	3501
gotools-core/include/GoTools/utils/Factorial.h	3503
gotools-core/include/GoTools/utils/GeneralFunctionMinimizer.h	3503
gotools-core/include/GoTools/utils/GeneralFunctionMinimizer_implementation.h	3504
gotools-core/include/GoTools/utils/gotools-doxyman.h	3506
gotools-core/include/GoTools/utils/Integration.h	3506
gotools-core/include/GoTools/utils/LUDecomp.h	3507
gotools-core/include/GoTools/utils/LUDecomp_implementation.h	3508
gotools-core/include/GoTools/utils/MatrixXD.h	3509
gotools-core/include/GoTools/utils/Point.h	3510
gotools-core/include/GoTools/utils/randomnoise.h	3511
gotools-core/include/GoTools/utils/Rational.h	3511
gotools-core/include/GoTools/utils/RegistrationUtils.h	3512
gotools-core/include/GoTools/utils/RotatedBox.h	3513
gotools-core/include/GoTools/utils/ScratchVect.h	3514

gotools-core/include/GoTools/utils/sleep.h	3515
gotools-core/include/GoTools/utils/StreamUtils.h	3516
gotools-core/include/GoTools/utils/timeutils.h	3518
gotools-core/include/GoTools/utils/Values.h	3518
gotools-core/include/GoTools/utils/Volumes.h	3520
igeslib/include/GoTools/igeslib/ftGroupGeom.h	3521
igeslib/include/GoTools/igeslib/ftTangPriority.h	3522
igeslib/include/GoTools/igeslib/IGESconverter.h	3523
igeslib/include/GoTools/igeslib/igeslib_doxyman.h	3524
implicitization/include/GoTools/implicitization/BernsteinMulti.h	3525
implicitization/include/GoTools/implicitization/BernsteinPoly.h	3526
implicitization/include/GoTools/implicitization/BernsteinTetrahedralPoly.h	3528
implicitization/include/GoTools/implicitization/BernsteinTriangularPoly.h	3529
implicitization/include/GoTools/implicitization/BernsteinUtils.h	3531
implicitization/include/GoTools/implicitization/BezierTriangle.h	3532
implicitization/include/GoTools/implicitization/Binomial.h	3533
implicitization/include/GoTools/implicitization/GpuMatrix.hpp	3534
implicitization/include/GoTools/implicitization/implicitization-doxyman.h	3535
implicitization/include/GoTools/implicitization/ImplicitizeCurveAlgo.h	3535
implicitization/include/GoTools/implicitization/ImplicitizeCurveAndVectorAlgo.h	3535
implicitization/include/GoTools/implicitization/ImplicitizePointCloudAlgo.h	3536
implicitization/include/GoTools/implicitization/ImplicitizeSurfaceAlgo.h	3537
implicitization/include/GoTools/implicitization/ImplicitUtils.h	3537
intersections/include/GoTools/intersections/AlgObj2DInt.h	3539
intersections/include/GoTools/intersections/AlgObj3DInt.h	3540
intersections/include/GoTools/intersections/AlgObjectInt.h	3541
intersections/include/GoTools/intersections/BoundaryFunctionInt.h	3541
intersections/include/GoTools/intersections/BoundaryGeomInt.h	3543
intersections/include/GoTools/intersections/Coincidence.h	3543
intersections/include/GoTools/intersections/ComplexityInfo.h	3545
intersections/include/GoTools/intersections/ConeInt.h	3545
intersections/include/GoTools/intersections/CvCvIntersector.h	3545
intersections/include/GoTools/intersections/CvPtIntersector.h	3546
intersections/include/GoTools/intersections/CylinderInt.h	3546
intersections/include/GoTools/intersections/generic_graph_algorithms.h	3547
intersections/include/GoTools/intersections/generic_graph_algorithms_implementation.h	3549
intersections/include/GoTools/intersections/GeomObjectInt.h	3551
intersections/include/GoTools/intersections/GeoTol.h	3552
intersections/include/GoTools/intersections/Identity.h	3553
intersections/include/GoTools/intersections/IntersectionCurve.h	3553
intersections/include/GoTools/intersections/IntersectionInterface.h	3555
intersections/include/GoTools/intersections/IntersectionLink.h	3556
intersections/include/GoTools/intersections/IntersectionPoint.h	3557
intersections/include/GoTools/intersections/IntersectionPointUtils.h	3558
intersections/include/GoTools/intersections/IntersectionPool.h	3560
intersections/include/GoTools/intersections/IntersectionPoolUtils.h	3561
intersections/include/GoTools/intersections/intersections-doxyman.h	3562
intersections/include/GoTools/intersections/IntersectionUtils.h	3562
intersections/include/GoTools/intersections/Intersector.h	3563
intersections/include/GoTools/intersections/Intersector2Obj.h	3564
intersections/include/GoTools/intersections/IntersectorAlgPar.h	3565
intersections/include/GoTools/intersections/IntersectorFuncConst.h	3565
intersections/include/GoTools/intersections/Line2DInt.h	3566
intersections/include/GoTools/intersections/LinkType.h	3567
intersections/include/GoTools/intersections/Par0FuncIntersector.h	3568
intersections/include/GoTools/intersections/Par1FuncIntersector.h	3568
intersections/include/GoTools/intersections/Par2FuncIntersector.h	3569
intersections/include/GoTools/intersections/Param0FunctionInt.h	3570

intersections/include/GoTools/intersections/Param1FunctionInt.h	3571
intersections/include/GoTools/intersections/Param2FunctionInt.h	3572
intersections/include/GoTools/intersections/ParamCurveInt.h	3573
intersections/include/GoTools/intersections/ParamFunctionInt.h	3574
intersections/include/GoTools/intersections/ParamGeomInt.h	3575
intersections/include/GoTools/intersections/ParamObjectInt.h	3575
intersections/include/GoTools/intersections/ParamPointInt.h	3576
intersections/include/GoTools/intersections/ParamSurfaceInt.h	3578
intersections/include/GoTools/intersections/PlaneInt.h	3578
intersections/include/GoTools/intersections/PtPtIntersector.h	3579
intersections/include/GoTools/intersections/SecondOrderProperties.h	3580
intersections/include/GoTools/intersections/SfCvIntersector.h	3581
intersections/include/GoTools/intersections/SfPtIntersector.h	3582
intersections/include/GoTools/intersections/SfSelfIntersector.h	3583
intersections/include/GoTools/intersections/SfSfIntersector.h	3583
intersections/include/GoTools/intersections/Singular.h	3584
intersections/include/GoTools/intersections/SingularityClassification.h	3585
intersections/include/GoTools/intersections/SingularityInfo.h	3585
intersections/include/GoTools/intersections/SingularityType.h	3587
intersections/include/GoTools/intersections/SphereInt.h	3588
intersections/include/GoTools/intersections/Spline1FunctionInt.h	3588
intersections/include/GoTools/intersections/Spline2FunctionInt.h	3589
intersections/include/GoTools/intersections/SplineCurveInt.h	3590
intersections/include/GoTools/intersections/SplineSurfaceInt.h	3590
intersections/include/GoTools/intersections/SubdivisionClassification.h	3591
intersections/include/GoTools/intersections/SurfaceAssembly.h	3591
intersections/include/GoTools/intersections/TorusInt.h	3592
isogeometric_model/include/GoTools/isogeometric_model/BdCondFuncor.h	3592
isogeometric_model/include/GoTools/isogeometric_model/BdConditionType.h	3593
isogeometric_model/include/GoTools/isogeometric_model/BlockBoundaryCondition.h	3594
isogeometric_model/include/GoTools/isogeometric_model/BlockPointBdCond.h	3595
isogeometric_model/include/GoTools/isogeometric_model/BlockSolution.h	3596
isogeometric_model/include/GoTools/isogeometric_model/EvalFuncorCurve.h	3597
isogeometric_model/include/GoTools/isogeometric_model/EvalFuncorSurface.h	3598
isogeometric_model/include/GoTools/isogeometric_model/isogeometric_model-doxymain.h	3599
isogeometric_model/include/GoTools/isogeometric_model/IsogeometricBlock.h	3599
isogeometric_model/include/GoTools/isogeometric_model/IsogeometricModel.h	3600
isogeometric_model/include/GoTools/isogeometric_model/IsogeometricSfBlock.h	3601
isogeometric_model/include/GoTools/isogeometric_model/IsogeometricSfModel.h	3602
isogeometric_model/include/GoTools/isogeometric_model/IsogeometricVolBlock.h	3603
isogeometric_model/include/GoTools/isogeometric_model/IsogeometricVolModel.h	3604
isogeometric_model/include/GoTools/isogeometric_model/SfBoundaryCondition.h	3604
isogeometric_model/include/GoTools/isogeometric_model/SfPointBdCond.h	3606
isogeometric_model/include/GoTools/isogeometric_model/SfSolution.h	3607
isogeometric_model/include/GoTools/isogeometric_model/VolBoundaryCondition.h	3609
isogeometric_model/include/GoTools/isogeometric_model/VolPointBdCond.h	3610
isogeometric_model/include/GoTools/isogeometric_model/VolSolution.h	3611
lrsplines2D/include/GoTools/lrsplines2D/Direction2D.h	3613
lrsplines2D/include/GoTools/lrsplines2D/Element2D.h	3613
lrsplines2D/include/GoTools/lrsplines2D/IndexMesh2DIterator.h	3614
lrsplines2D/include/GoTools/lrsplines2D/LinDepUtils.h	3615
lrsplines2D/include/GoTools/lrsplines2D/LRApproxApp.h	3616
lrsplines2D/include/GoTools/lrsplines2D/LRBenchmarkUtils.h	3617
lrsplines2D/include/GoTools/lrsplines2D/LRBSpline2D.h	3617
lrsplines2D/include/GoTools/lrsplines2D/LRBSpline2DUtils.h	3618
lrsplines2D/include/GoTools/lrsplines2D/LRSplineEvalGrid.h	3619
lrsplines2D/include/GoTools/lrsplines2D/LRSplineMBA.h	3620
lrsplines2D/include/GoTools/lrsplines2D/LRSplinePlotUtils.h	3621

Irsplines2D/include/GoTools/Irsplines2D/Irsplines2d-doxymain.h	3621
Irsplines2D/include/GoTools/Irsplines2D/LRSplineSurface.h	3621
Irsplines2D/include/GoTools/Irsplines2D/LRSplineSurface_impl.h	3622
Irsplines2D/include/GoTools/Irsplines2D/LRSplineUtils.h	3623
Irsplines2D/include/GoTools/Irsplines2D/LRSurfApprox.h	3624
Irsplines2D/include/GoTools/Irsplines2D/LRSurfSmoothLS.h	3625
Irsplines2D/include/GoTools/Irsplines2D/LRSurfStitch.h	3626
Irsplines2D/include/GoTools/Irsplines2D/Mesh2D.h	3627
Irsplines2D/include/GoTools/Irsplines2D/Mesh2DIterator.h	3628
Irsplines2D/include/GoTools/Irsplines2D/Mesh2DUtils.h	3629
newmat/app/example.cpp	3630
newmat/app/garch.cpp	3633
newmat/app/nl_ex.cpp	3634
newmat/app/sl_ex.cpp	3636
newmat/app/test_exc.cpp	3637
newmat/app/tmt.cpp	3639
newmat/app/tmt1.cpp	3642
newmat/app/tmt2.cpp	3643
newmat/app/tmt3.cpp	3644
newmat/app/tmt4.cpp	3645
newmat/app/tmt5.cpp	3646
newmat/app/tmt6.cpp	3648
newmat/app/tmt7.cpp	3649
newmat/app/tmt8.cpp	3650
newmat/app/tmt9.cpp	3651
newmat/app/tmta.cpp	3652
newmat/app/tmtb.cpp	3653
newmat/app/tmtc.cpp	3654
newmat/app/tmtd.cpp	3655
newmat/app/tmte.cpp	3656
newmat/app/tmtf.cpp	3657
newmat/app/tmtg.cpp	3658
newmat/app/tmth.cpp	3660
newmat/app/tmti.cpp	3661
newmat/app/tmtj.cpp	3663
newmat/app/tmtk.cpp	3664
newmat/app/tmtl.cpp	3665
newmat/app/tmtm.cpp	3666
newmat/include/boolean.h	3667
newmat/include/controlw.h	3668
newmat/include/include.h	3669
newmat/include/myexcept.h	3670
newmat/include/newmat.h	3673
newmat/include/newmatap.h	3676
newmat/include/newmatio.h	3678
newmat/include/newmatnl.h	3679
newmat/include/newmatrc.h	3680
newmat/include/newmatrm.h	3681
newmat/include/precisio.h	3683
newmat/include/solution.h	3684
newmat/include/tmt.h	3685
newmat/src/bandmat.cpp	3689
newmat/src/cholesky.cpp	3690
newmat/src/evaluate.cpp	3692
newmat/src/fft.cpp	3693
newmat/src/hholder.cpp	3696
newmat/src/jacobi.cpp	3698
newmat/src/myexcept.cpp	3700

newmat/src/newfft.cpp	3702
newmat/src/newmat1.cpp	3703
newmat/src/newmat2.cpp	3704
newmat/src/newmat3.cpp	3706
newmat/src/newmat4.cpp	3707
newmat/src/newmat5.cpp	3708
newmat/src/newmat6.cpp	3709
newmat/src/newmat7.cpp	3710
newmat/src/newmat8.cpp	3712
newmat/src/newmat9.cpp	3713
newmat/src/newmatex.cpp	3715
newmat/src/newmatnl.cpp	3716
newmat/src/newmatrm.cpp	3717
newmat/src/solution.cpp	3718
newmat/src/sort.cpp	3720
newmat/src/submat.cpp	3722
newmat/src/svd.cpp	3722
parametrization/include/GoTools/parametrization/copyright.h	3419
parametrization/include/GoTools/parametrization/parametrization-doxymain.h	3724
parametrization/include/GoTools/parametrization/PrBiCGStab.h	3724
parametrization/include/GoTools/parametrization/PrCellStructure.h	3725
parametrization/include/GoTools/parametrization/PrCG.h	3726
parametrization/include/GoTools/parametrization/PrDijkstra.h	3726
parametrization/include/GoTools/parametrization/PrExplicitConnectivity.h	3729
parametrization/include/GoTools/parametrization/PrFaber_F.h	3730
parametrization/include/GoTools/parametrization/PrFastUnorganized_OP.h	3730
parametrization/include/GoTools/parametrization/PrFilterbank.h	3731
parametrization/include/GoTools/parametrization/PrGeodesics.h	3732
parametrization/include/GoTools/parametrization/PrHeap.h	3736
parametrization/include/GoTools/parametrization/PrInterface.h	3737
parametrization/include/GoTools/parametrization/PrLevelTriangulation_OP.h	3740
parametrization/include/GoTools/parametrization/PrMat.h	3741
parametrization/include/GoTools/parametrization/PrMatrix.h	3742
parametrization/include/GoTools/parametrization/PrMatSparse.h	3743
parametrization/include/GoTools/parametrization/PrMultiDijkstra.h	3745
parametrization/include/GoTools/parametrization/PrNestedNode.h	3746
parametrization/include/GoTools/parametrization/PrNestedTriangulation.h	3747
parametrization/include/GoTools/parametrization/PrNode.h	3748
parametrization/include/GoTools/parametrization/PrOrganizedPoints.h	3749
parametrization/include/GoTools/parametrization/PrParametrizeBdy.h	3750
parametrization/include/GoTools/parametrization/PrParametrizeInt.h	3752
parametrization/include/GoTools/parametrization/PrParametrizeMesh.h	3753
parametrization/include/GoTools/parametrization/PrParamTriangulation.h	3753
parametrization/include/GoTools/parametrization/PrParamUtil.h	3755
parametrization/include/GoTools/parametrization/PrPathTriangleSeq.h	3756
parametrization/include/GoTools/parametrization/PrPGNode.h	3764
parametrization/include/GoTools/parametrization/PrPlanarGraph_OP.h	3765
parametrization/include/GoTools/parametrization/PrPrewavelet_F.h	3765
parametrization/include/GoTools/parametrization/PrPrmEDDHLs.h	3766
parametrization/include/GoTools/parametrization/PrPrmExperimental.h	3767
parametrization/include/GoTools/parametrization/PrPrmLeastSquare.h	3767
parametrization/include/GoTools/parametrization/PrPrmMeanValue.h	3768
parametrization/include/GoTools/parametrization/PrPrmShpPres.h	3769
parametrization/include/GoTools/parametrization/PrPrmSurface.h	3770
parametrization/include/GoTools/parametrization/PrPrmSymMeanValue.h	3771
parametrization/include/GoTools/parametrization/PrPrmUniform.h	3772
parametrization/include/GoTools/parametrization/PrPrmWachspress.h	3773
parametrization/include/GoTools/parametrization/PrRectangularGrid_OP.h	3774

parametrization/include/GoTools/parametrization/PrSubTriangulation.h	3775
parametrization/include/GoTools/parametrization/PrTexture.h	3777
parametrization/include/GoTools/parametrization/PrThin.h	3777
parametrization/include/GoTools/parametrization/PrThreshold.h	3778
parametrization/include/GoTools/parametrization/PrTriangle.h	3778
parametrization/include/GoTools/parametrization/PrTriangulation_NT.h	3779
parametrization/include/GoTools/parametrization/PrTriangulation_OP.h	3780
parametrization/include/GoTools/parametrization/PrUnorganized_OP.h	3781
parametrization/include/GoTools/parametrization/PrVec.h	3781
parametrization/include/GoTools/parametrization/PrWaveletUtil.h	3782
qualitymodule/include/GoTools/qualitymodule/FaceSetQuality.h	3783
qualitymodule/include/GoTools/qualitymodule/FaceSetRepair.h	3783
qualitymodule/include/GoTools/qualitymodule/ModelQuality.h	3784
qualitymodule/include/GoTools/qualitymodule/ModelRepair.h	3785
qualitymodule/include/GoTools/qualitymodule/qualitymodule-doxymain.h	3786
qualitymodule/include/GoTools/qualitymodule/QualityResults.h	3786
qualitymodule/include/GoTools/qualitymodule/QualityUtils.h	3787
qualitymodule/include/GoTools/qualitymodule/testSuite.h	3788
sisl/examples/example01.cpp	3789
sisl/examples/example02.cpp	3790
sisl/examples/example03.cpp	3790
sisl/examples/example04.cpp	3791
sisl/examples/example05.cpp	3792
sisl/examples/example06.cpp	3793
sisl/examples/example07.cpp	3793
sisl/examples/example08.cpp	3794
sisl/examples/example09.cpp	3795
sisl/examples/example10.cpp	3796
sisl/examples/example11.cpp	3796
sisl/examples/example12.cpp	3797
sisl/examples/example13.cpp	3798
sisl/examples/example14.cpp	3799
sisl/examples/example15.cpp	3799
sisl/examples/streaming/include/GoReadWrite.h	3800
sisl/examples/streaming/src/GoReadWrite.cpp	3801
sisl/examples/viewer/sisl_view_demo.cpp	3820
sisl/examples/viewer/include/aux2.h	3803
sisl/examples/viewer/include/gl_aux.h	3808
sisl/examples/viewer/include/glutils.h	3811
sisl/examples/viewer/include/jonvec.h	3813
sisl/examples/viewer/include/mouse.h	3814
sisl/examples/viewer/include/sisl_aux.h	3816
sisl/examples/viewer/include/transfutils.h	3817
sisl/examples/viewer/src/aux2.cpp	3825
sisl/examples/viewer/src/gl_aux.cpp	3826
sisl/examples/viewer/src/glutils.cpp	3829
sisl/examples/viewer/src/mouse.cpp	3830
sisl/examples/viewer/src/sisl_aux.cpp	3833
sisl/examples/viewer/src/transfutils.cpp	3834
sisl/include/sisl-copyright.h	3837
sisl/include/sisl.h	3837
sisl/include/sislPh	3850
sisl/src/construct.c	3883
sisl/src/crvrctang.c	3886
sisl/src/crvcrvtang.c	3887
sisl/src/crvlintang.c	3888
sisl/src/destruct.c	3889
sisl/src/ev_cv_off.c	3891

sisl/src/eval_2_crv.c	3892
sisl/src/evalcrvarc.c	3893
sisl/src/hp_s1880.c	3894
sisl/src/intjoinper.c	3895
sisl/src/make3D.c	3896
sisl/src/makecvkreg.c	3897
sisl/src/makesfkreg.c	3898
sisl/src/maketracks.c	3899
sisl/src/mk_cv_cycl.c	3900
sisl/src/newknots.c	3901
sisl/src/pickcrvsf.c	3902
sisl/src/pocrvtang.c	3903
sisl/src/refine_all.c	3904
sisl/src/s1001.c	3905
sisl/src/s1011.c	3906
sisl/src/s1012.c	3907
sisl/src/s1013.c	3908
sisl/src/s1014.c	3909
sisl/src/s1015.c	3910
sisl/src/s1016.c	3911
sisl/src/s1017.c	3912
sisl/src/s1018.c	3913
sisl/src/s1021.c	3914
sisl/src/s1022.c	3915
sisl/src/s1023.c	3916
sisl/src/s1024.c	3917
sisl/src/s1025.c	3918
sisl/src/s1119.c	3919
sisl/src/s1161.c	3920
sisl/src/s1162.c	3921
sisl/src/s1172.c	3922
sisl/src/s1173.c	3923
sisl/src/s1174.c	3924
sisl/src/s1190.c	3925
sisl/src/s1192.c	3926
sisl/src/s1219.c	3927
sisl/src/s1220.c	3928
sisl/src/s1221.c	3929
sisl/src/s1222.c	3930
sisl/src/s1223.c	3931
sisl/src/s1224.c	3932
sisl/src/s1225.c	3933
sisl/src/s1226.c	3934
sisl/src/s1227.c	3935
sisl/src/s1231.c	3936
sisl/src/s1232.c	3937
sisl/src/s1233.c	3938
sisl/src/s1235.c	3939
sisl/src/s1236.c	3940
sisl/src/s1237.c	3941
sisl/src/s1238.c	3942
sisl/src/s1239.c	3943
sisl/src/s1240.c	3944
sisl/src/s1241.c	3945
sisl/src/s1243.c	3946
sisl/src/s1244.c	3947
sisl/src/s1245.c	3948
sisl/src/s1251.c	3949

sisl/src/s1252.c	3950
sisl/src/s1301.c	3951
sisl/src/s1302.c	3952
sisl/src/s1303.c	3953
sisl/src/s1304.c	3954
sisl/src/s1305.c	3955
sisl/src/s1306.c	3956
sisl/src/s1307.c	3957
sisl/src/s1308.c	3958
sisl/src/s1309.c	3959
sisl/src/s1310.c	3960
sisl/src/s1311.c	3961
sisl/src/s1312.c	3962
sisl/src/s1313.c	3963
sisl/src/s1314.c	3964
sisl/src/s1315.c	3965
sisl/src/s1316.c	3966
sisl/src/s1317.c	3967
sisl/src/s1318.c	3968
sisl/src/s1319.c	3969
sisl/src/s1320.c	3970
sisl/src/s1321.c	3971
sisl/src/s1322.c	3972
sisl/src/s1323.c	3973
sisl/src/s1324.c	3974
sisl/src/s1325.c	3975
sisl/src/s1327.c	3976
sisl/src/s1328.c	3977
sisl/src/s1329.c	3978
sisl/src/s1330.c	3979
sisl/src/s1331.c	3980
sisl/src/s1332.c	3981
sisl/src/s1333.c	3982
sisl/src/s1333count.c	3983
sisl/src/s1333cycli.c	3984
sisl/src/s1334.c	3985
sisl/src/s1340.c	3986
sisl/src/s1341.c	3987
sisl/src/s1342.c	3988
sisl/src/s1343.c	3989
sisl/src/s1345.c	3990
sisl/src/s1346.c	3991
sisl/src/s1347.c	3992
sisl/src/s1348.c	3993
sisl/src/s1349.c	3994
sisl/src/s1350.c	3995
sisl/src/s1351.c	3996
sisl/src/s1352.c	3997
sisl/src/s1353.c	3998
sisl/src/s1354.c	3999
sisl/src/s1355.c	4000
sisl/src/s1356.c	4001
sisl/src/s1357.c	4002
sisl/src/s1358.c	4003
sisl/src/s1359.c	4004
sisl/src/s1360.c	4005
sisl/src/s1361.c	4006
sisl/src/s1362.c	4007

sisl/src/s1363.c	4008
sisl/src/s1364.c	4009
sisl/src/s1365.c	4010
sisl/src/s1366.c	4011
sisl/src/s1367.c	4012
sisl/src/s1369.c	4013
sisl/src/s1370.c	4014
sisl/src/s1371.c	4015
sisl/src/s1372.c	4016
sisl/src/s1373.c	4017
sisl/src/s1374.c	4018
sisl/src/s1375.c	4019
sisl/src/s1376.c	4020
sisl/src/s1377.c	4021
sisl/src/s1378.c	4022
sisl/src/s1379.c	4023
sisl/src/s1380.c	4024
sisl/src/s1381.c	4025
sisl/src/s1382.c	4026
sisl/src/s1383.c	4027
sisl/src/s1384.c	4028
sisl/src/s1385.c	4029
sisl/src/s1386.c	4030
sisl/src/s1387.c	4031
sisl/src/s1388.c	4032
sisl/src/s1389.c	4033
sisl/src/s1390.c	4034
sisl/src/s1391.c	4035
sisl/src/s1393.c	4036
sisl/src/s1399.c	4037
sisl/src/s1401.c	4038
sisl/src/s1421.c	4039
sisl/src/s1422.c	4040
sisl/src/s1424.c	4041
sisl/src/s1425.c	4042
sisl/src/s1435.c	4043
sisl/src/s1436.c	4044
sisl/src/s1437.c	4045
sisl/src/s1438.c	4046
sisl/src/s1439.c	4047
sisl/src/s1440.c	4048
sisl/src/s1450.c	4049
sisl/src/s1451.c	4050
sisl/src/s1452.c	4051
sisl/src/s1500.c	4052
sisl/src/s1501.c	4053
sisl/src/s1502.c	4054
sisl/src/s1503.c	4055
sisl/src/s1504.c	4056
sisl/src/s1505.c	4057
sisl/src/s1506.c	4058
sisl/src/s1507.c	4059
sisl/src/s1508.c	4060
sisl/src/s1510.c	4061
sisl/src/s1511.c	4062
sisl/src/s1512.c	4063
sisl/src/s1513.c	4064
sisl/src/s1514.c	4065

sisl/src/s1515.c	4066
sisl/src/s1516.c	4067
sisl/src/s1517.c	4068
sisl/src/s1518.c	4069
sisl/src/s1520.c	4070
sisl/src/s1521.c	4071
sisl/src/s1522.c	4072
sisl/src/s1528.c	4073
sisl/src/s1529.c	4074
sisl/src/s1530.c	4075
sisl/src/s1531.c	4076
sisl/src/s1534.c	4077
sisl/src/s1535.c	4078
sisl/src/s1536.c	4079
sisl/src/s1537.c	4080
sisl/src/s1538.c	4081
sisl/src/s1539.c	4082
sisl/src/s1540.c	4083
sisl/src/s1541.c	4084
sisl/src/s1542.c	4085
sisl/src/s1600.c	4086
sisl/src/s1601.c	4087
sisl/src/s1602.c	4088
sisl/src/s1603.c	4089
sisl/src/s1604.c	4090
sisl/src/s1605.c	4091
sisl/src/s1606.c	4092
sisl/src/s1607.c	4093
sisl/src/s1608.c	4094
sisl/src/s1609.c	4095
sisl/src/s1611.c	4096
sisl/src/s1612.c	4097
sisl/src/s1613.c	4098
sisl/src/s1613bez.c	4099
sisl/src/s1614.c	4100
sisl/src/s1615.c	4101
sisl/src/s1616.c	4102
sisl/src/s1617.c	4103
sisl/src/s1618.c	4104
sisl/src/s1619.c	4105
sisl/src/s1620.c	4106
sisl/src/s1630.c	4107
sisl/src/s1631.c	4108
sisl/src/s1700.c	4109
sisl/src/s1701.c	4110
sisl/src/s1705.c	4111
sisl/src/s1706.c	4112
sisl/src/s1707.c	4113
sisl/src/s1708.c	4114
sisl/src/s1710.c	4115
sisl/src/s1711.c	4116
sisl/src/s1712.c	4117
sisl/src/s1713.c	4118
sisl/src/s1714.c	4119
sisl/src/s1715.c	4120
sisl/src/s1716.c	4121
sisl/src/s1720.c	4122
sisl/src/s1730.c	4123

sisl/src/s1731.c	4124
sisl/src/s1732.c	4125
sisl/src/s1733.c	4126
sisl/src/s1741.c	4127
sisl/src/s1750.c	4128
sisl/src/s1753.c	4129
sisl/src/s1754.c	4130
sisl/src/s1755.c	4131
sisl/src/s1770.c	4132
sisl/src/s17702d.c	4133
sisl/src/s1771.c	4135
sisl/src/s1772.c	4136
sisl/src/s1773.c	4137
sisl/src/s1774.c	4138
sisl/src/s1775.c	4139
sisl/src/s1780.c	4140
sisl/src/s1785.c	4141
sisl/src/s1786.c	4142
sisl/src/s1787.c	4143
sisl/src/s1788.c	4144
sisl/src/s1789.c	4145
sisl/src/s1790.c	4146
sisl/src/s1791.c	4147
sisl/src/s1792.c	4148
sisl/src/s1795.c	4149
sisl/src/s1796.c	4150
sisl/src/s1797.c	4151
sisl/src/s1830.c	4152
sisl/src/s1834.c	4153
sisl/src/s1839.c	4154
sisl/src/s1840.c	4155
sisl/src/s1850.c	4156
sisl/src/s1851.c	4157
sisl/src/s1852.c	4158
sisl/src/s1853.c	4159
sisl/src/s1854.c	4160
sisl/src/s1855.c	4161
sisl/src/s1856.c	4162
sisl/src/s1857.c	4163
sisl/src/s1858.c	4164
sisl/src/s1859.c	4165
sisl/src/s1860.c	4166
sisl/src/s1870.c	4167
sisl/src/s1871.c	4168
sisl/src/s1880.c	4169
sisl/src/s1890.c	4170
sisl/src/s1891.c	4171
sisl/src/s1893.c	4172
sisl/src/s1894.c	4173
sisl/src/s1896.c	4174
sisl/src/s1897.c	4175
sisl/src/s1900.c	4176
sisl/src/s1901.c	4177
sisl/src/s1902.c	4179
sisl/src/s1903.c	4180
sisl/src/s1904.c	4181
sisl/src/s1905.c	4182
sisl/src/s1906.c	4183

sisl/src/s1907.c	4184
sisl/src/s1908.c	4185
sisl/src/s1909.c	4186
sisl/src/s1910.c	4187
sisl/src/s1911.c	4188
sisl/src/s1912.c	4189
sisl/src/s1916.c	4190
sisl/src/s1917.c	4191
sisl/src/s1918.c	4192
sisl/src/s1919.c	4193
sisl/src/s1920.c	4194
sisl/src/s1921.c	4195
sisl/src/s1924.c	4196
sisl/src/s1925.c	4197
sisl/src/s1926.c	4198
sisl/src/s1927.c	4199
sisl/src/s1930.c	4200
sisl/src/s1931.c	4201
sisl/src/s1931unit.c	4202
sisl/src/s1932.c	4203
sisl/src/s1933.c	4204
sisl/src/s1934.c	4205
sisl/src/s1935.c	4206
sisl/src/s1936.c	4207
sisl/src/s1937.c	4208
sisl/src/s1938.c	4209
sisl/src/s1940.c	4210
sisl/src/s1941.c	4211
sisl/src/s1942.c	4212
sisl/src/s1943.c	4213
sisl/src/s1944.c	4214
sisl/src/s1945.c	4215
sisl/src/s1946.c	4216
sisl/src/s1947.c	4217
sisl/src/s1948.c	4218
sisl/src/s1949.c	4219
sisl/src/s1950.c	4220
sisl/src/s1951.c	4221
sisl/src/s1953.c	4222
sisl/src/s1954.c	4223
sisl/src/s1955.c	4224
sisl/src/s1956.c	4225
sisl/src/s1957.c	4226
sisl/src/s1958.c	4227
sisl/src/s1959.c	4228
sisl/src/s1960.c	4229
sisl/src/s1961.c	4230
sisl/src/s1962.c	4231
sisl/src/s1963.c	4232
sisl/src/s1965.c	4233
sisl/src/s1966.c	4234
sisl/src/s1967.c	4235
sisl/src/s1968.c	4236
sisl/src/s1986.c	4237
sisl/src/s1987.c	4238
sisl/src/s1988.c	4239
sisl/src/s1989.c	4240
sisl/src/s1990.c	4241

sisl/src/s1991.c	4242
sisl/src/s1992.c	4243
sisl/src/s1993.c	4244
sisl/src/s1994.c	4245
sisl/src/s2500.c	4246
sisl/src/s2501.c	4247
sisl/src/s2502.c	4248
sisl/src/s2503.c	4249
sisl/src/s2504.c	4250
sisl/src/s2505.c	4251
sisl/src/s2506.c	4252
sisl/src/s2507.c	4253
sisl/src/s2508.c	4254
sisl/src/s2509.c	4255
sisl/src/s2510.c	4256
sisl/src/s2511.c	4257
sisl/src/s2512.c	4258
sisl/src/s2513.c	4259
sisl/src/s2514.c	4260
sisl/src/s2515.c	4261
sisl/src/s2516.c	4262
sisl/src/s2532.c	4263
sisl/src/s2533.c	4264
sisl/src/s2534.c	4265
sisl/src/s2535.c	4266
sisl/src/s2536.c	4267
sisl/src/s2540.c	4268
sisl/src/s2541.c	4269
sisl/src/s2542.c	4270
sisl/src/s2543.c	4271
sisl/src/s2544.c	4272
sisl/src/s2545.c	4273
sisl/src/s2550.c	4274
sisl/src/s2551.c	4275
sisl/src/s2553.c	4276
sisl/src/s2554.c	4277
sisl/src/s2555.c	4278
sisl/src/s2556.c	4279
sisl/src/s2557.c	4280
sisl/src/s2558.c	4281
sisl/src/s2559.c	4282
sisl/src/s2560.c	4283
sisl/src/s2561.c	4284
sisl/src/s2562.c	4285
sisl/src/s6addcurve.c	4286
sisl/src/s6affdist.c	4287
sisl/src/s6ang.c	4288
sisl/src/s6angle.c	4289
sisl/src/s6castelja.c	4290
sisl/src/s6chpar.c	4291
sisl/src/s6crss.c	4292
sisl/src/s6crvature.c	4293
sisl/src/s6crvcheck.c	4294
sisl/src/s6curvrad.c	4295
sisl/src/s6decomp.c	4296
sisl/src/s6degnorm.c	4297
sisl/src/s6dertopt.c	4298
sisl/src/s6diff.c	4299

sisl/src/s6dist.c	4300
sisl/src/s6dline.c	4301
sisl/src/s6dplane.c	4302
sisl/src/s6drawseq.c	4303
sisl/src/s6equal.c	4304
sisl/src/s6err.c	4305
sisl/src/s6existbox.c	4306
sisl/src/s6findfac.c	4307
sisl/src/s6fndintv.c	4308
sisl/src/s6herm.c	4309
sisl/src/s6herm_bez.c	4310
sisl/src/s6idcon.c	4311
sisl/src/s6idcpt.c	4312
sisl/src/s6idedg.c	4313
sisl/src/s6identify.c	4314
sisl/src/s6idget.c	4315
sisl/src/s6idint.c	4316
sisl/src/s6idklist.c	4317
sisl/src/s6idkpt.c	4318
sisl/src/s6idlis.c	4319
sisl/src/s6idnpt.c	4320
sisl/src/s6idput.c	4321
sisl/src/s6inv4.c	4322
sisl/src/s6invert.c	4323
sisl/src/s6knotmult.c	4324
sisl/src/s6length.c	4325
sisl/src/s6line.c	4326
sisl/src/s6lprj.c	4327
sisl/src/s6lufacp.c	4328
sisl/src/s6lusolp.c	4329
sisl/src/s6metric.c	4330
sisl/src/s6move.c	4331
sisl/src/s6mulvec.c	4332
sisl/src/s6mvec.c	4333
sisl/src/s6newbox.c	4334
sisl/src/s6norm.c	4335
sisl/src/s6ratder.c	4336
sisl/src/s6rotax.c	4337
sisl/src/s6rotmat.c	4338
sisl/src/s6schoen.c	4339
sisl/src/s6scpr.c	4340
sisl/src/s6sortpar.c	4341
sisl/src/s6sradter.c	4342
sisl/src/s6strider.c	4343
sisl/src/s6takunion.c	4344
sisl/src/s6twonorm.c	4345
sisl/src/s9adsimp.c	4346
sisl/src/s9adstep.c	4347
sisl/src/s9boundimp.c	4348
sisl/src/s9boundit.c	4349
sisl/src/s9clipimp.c	4350
sisl/src/s9clipit.c	4351
sisl/src/s9conmarch.c	4352
sisl/src/s9iterate.c	4353
sisl/src/s9iterimp.c	4354
sisl/src/s9smplknot.c	4355
sisl/src/s9surmarch.c	4356
sisl/src/sh1260.c	4357

sisl/src/sh1261.c	4358
sisl/src/sh1262.c	4359
sisl/src/sh1263.c	4360
sisl/src/sh1365.c	4361
sisl/src/sh1369.c	4362
sisl/src/sh1371.c	4363
sisl/src/sh1372.c	4364
sisl/src/sh1373.c	4365
sisl/src/sh1374.c	4366
sisl/src/sh1375.c	4367
sisl/src/sh1460.c	4368
sisl/src/sh1461.c	4369
sisl/src/sh1462.c	4371
sisl/src/sh1463.c	4372
sisl/src/sh1464.c	4373
sisl/src/sh1465.c	4374
sisl/src/sh1466.c	4375
sisl/src/sh1467.c	4376
sisl/src/sh1502.c	4377
sisl/src/sh1503.c	4378
sisl/src/sh1510.c	4379
sisl/src/sh1511.c	4380
sisl/src/sh1761.c	4381
sisl/src/sh1762.c	4382
sisl/src/sh1779.c	4383
sisl/src/sh1779_at.c	4384
sisl/src/sh1780.c	4385
sisl/src/sh1780_at.c	4386
sisl/src/sh1781.c	4387
sisl/src/sh1781_at.c	4388
sisl/src/sh1782.c	4389
sisl/src/sh1783.c	4390
sisl/src/sh1784.c	4391
sisl/src/sh1786.c	4393
sisl/src/sh1787.c	4394
sisl/src/sh1790.c	4395
sisl/src/sh1830.c	4396
sisl/src/sh1831.c	4397
sisl/src/sh1834.c	4398
sisl/src/sh1839.c	4399
sisl/src/sh1850.c	4400
sisl/src/sh1851.c	4401
sisl/src/sh1852.c	4402
sisl/src/sh1853.c	4403
sisl/src/sh1854.c	4404
sisl/src/sh1855.c	4405
sisl/src/sh1856.c	4406
sisl/src/sh1857.c	4407
sisl/src/sh1858.c	4408
sisl/src/sh1859.c	4409
sisl/src/sh1860.c	4410
sisl/src/sh1870.c	4411
sisl/src/sh1871.c	4412
sisl/src/sh1922.c	4413
sisl/src/sh1923.c	4414
sisl/src/sh1924.c	4415
sisl/src/sh1925.c	4416
sisl/src/sh1926.c	4417

sisl/src/sh1927.c	4418
sisl/src/sh1928.c	4419
sisl/src/sh1929.c	4420
sisl/src/sh1930.c	4421
sisl/src/sh1992.c	4422
sisl/src/sh1993.c	4423
sisl/src/sh1994.c	4424
sisl/src/sh6clvert.c	4425
sisl/src/sh6comedg.c	4426
sisl/src/sh6connect.c	4427
sisl/src/sh6count.c	4428
sisl/src/sh6cvvert.c	4429
sisl/src/sh6degen.c	4430
sisl/src/sh6disconn.c	4431
sisl/src/sh6edgpnt.c	4432
sisl/src/sh6edgred.c	4433
sisl/src/sh6evalint.c	4434
sisl/src/sh6floop.c	4435
sisl/src/sh6fndsplt.c	4436
sisl/src/sh6getgeom.c	4437
sisl/src/sh6getlist.c	4438
sisl/src/sh6getmain.c	4439
sisl/src/sh6getnbrs.c	4440
sisl/src/sh6getnext.c	4441
sisl/src/sh6getothr.c	4442
sisl/src/sh6getprev.c	4443
sisl/src/sh6gettop.c	4444
sisl/src/sh6idaledg.c	4445
sisl/src/sh6idcon.c	4446
sisl/src/sh6idfcros.c	4447
sisl/src/sh6idget.c	4448
sisl/src/sh6idkpt.c	4449
sisl/src/sh6idlis.c	4450
sisl/src/sh6idnpt.c	4451
sisl/src/sh6idnwun.c	4452
sisl/src/sh6idput.c	4453
sisl/src/sh6idrcros.c	4454
sisl/src/sh6idsplit.c	4455
sisl/src/sh6idunite.c	4456
sisl/src/sh6insert.c	4457
sisl/src/sh6inspnt.c	4458
sisl/src/sh6iscnect.c	4459
sisl/src/sh6ishelp.c	4460
sisl/src/sh6isinsid.c	4461
sisl/src/sh6ismain.c	4462
sisl/src/sh6nmbhelp.c	4463
sisl/src/sh6nmbmain.c	4464
sisl/src/sh6ptobj.c	4465
sisl/src/sh6ptouchp.c	4466
sisl/src/sh6putsing.c	4467
sisl/src/sh6red.c	4468
sisl/src/sh6remcon.c	4469
sisl/src/sh6rempnt.c	4470
sisl/src/sh6sepcrv.c	4471
sisl/src/sh6setcnsd.c	4472
sisl/src/sh6setdir.c	4473
sisl/src/sh6settop.c	4474
sisl/src/sh6spltgeo.c	4475

sisl/src/sh6tohelp.c	4476
sisl/src/sh6tomain.c	4477
sisl/src/sh6topohlp.c	4478
sisl/src/sh6trmlist.c	4479
sisl/src/sh_1d_div.c	4480
sisl/src/sh_div_crv.c	4481
sisl/src/sh_set_at.c	4482
sisl/src/shape.c	4483
sisl/src/shcheckput.c	4484
sisl/src/shchecktyp.c	4485
sisl/src/shcsfsing.c	4486
sisl/src/shdivsurf.c	4487
sisl/src/shevalc.c	4488
sisl/src/shmklppts.c	4489
sisl/src/shsing.c	4490
sisl/src/spli_silh.c	4491
sisl/src/tstcyclknt.c	4492
topology/include/GoTools/topology/FaceAdjacency.h	4493
topology/include/GoTools/topology/FaceConnectivity.h	4494
topology/include/GoTools/topology/FaceConnectivityUtils.h	4495
topology/include/GoTools/topology/topology_doxyman.h	4495
topology/include/GoTools/topology/tpEdge.h	4495
topology/include/GoTools/topology/tpFace.h	4496
topology/include/GoTools/topology/tpJointType.h	4497
topology/include/GoTools/topology/tpTolerances.h	4498
topology/include/GoTools/topology/tpTopologyTable.h	4498
topology/include/GoTools/topology/tpUtils.h	4499
trivariate/include/GoTools/trivariate/ConeVolume.h	4500
trivariate/include/GoTools/trivariate/CoonsPatchVolumeGen.h	4501
trivariate/include/GoTools/trivariate/CurveOnVolume.h	4502
trivariate/include/GoTools/trivariate/CylinderVolume.h	4502
trivariate/include/GoTools/trivariate/ElementaryVolume.h	4503
trivariate/include/GoTools/trivariate/examples_trivariate_doxyman.h	4504
trivariate/include/GoTools/trivariate/GapRemovalVolume.h	4504
trivariate/include/GoTools/trivariate/LoftVolumeCreator.h	4505
trivariate/include/GoTools/trivariate/Parallelepiped.h	4506
trivariate/include/GoTools/trivariate/ParamVolume.h	4507
trivariate/include/GoTools/trivariate/RectangularVolumeTesselator.h	4507
trivariate/include/GoTools/trivariate/RegularVolMesh.h	4509
trivariate/include/GoTools/trivariate/SmoothVolume.h	4510
trivariate/include/GoTools/trivariate/SphereVolume.h	4511
trivariate/include/GoTools/trivariate/SplineVolume.h	4512
trivariate/include/GoTools/trivariate/SurfaceOnVolume.h	4513
trivariate/include/GoTools/trivariate/SurfaceOnVolumeTools.h	4514
trivariate/include/GoTools/trivariate/SweepVolumeCreator.h	4515
trivariate/include/GoTools/trivariate/TorusVolume.h	4516
trivariate/include/GoTools/trivariate/trivariate_doxyman.h	4516
trivariate/include/GoTools/trivariate/VolumeInterpolator.h	4516
trivariate/include/GoTools/trivariate/VolumeParameterCurve.h	4517
trivariate/include/GoTools/trivariate/VolumeSpaceCurve.h	4518
trivariate/include/GoTools/trivariate/VolumeTools.h	4519
trivariatemodel/include/GoTools/trivariatemodel/examples_trivariatemodel_doxyman.h	4520
trivariatemodel/include/GoTools/trivariatemodel/ftVolume.h	4520
trivariatemodel/include/GoTools/trivariatemodel/ftVolumeTools.h	4521
trivariatemodel/include/GoTools/trivariatemodel/trivariatemodel_doxyman.h	4522
trivariatemodel/include/GoTools/trivariatemodel/VolumeAdjacency.h	4523
trivariatemodel/include/GoTools/trivariatemodel/VolumeModel.h	4523
trivariatemodel/include/GoTools/trivariatemodel/VolumeModelCreator.h	4524

trivariatemodel/include/GoTools/trivariatemodel/VolumeModelFileHandler.h	4525
tfl/examples/hesimplest/main.cpp	4526
tfl/include/tfl/api.h	4527
tfl/include/tfl/mainpage_tfl.h	4531
tfl/include/tfl/tfl.h	4531
tfl/include/tfl/tfl_constr.h	4534
tfl/include/tfl/tfl_util.h	4535
tfl/include/tfl/halfedge/HeDart.h	4527
tfl/include/tfl/halfedge/HeTraits.h	4528
tfl/include/tfl/halfedge/HeTriang.h	4529
tfl/include/tfl/halfedge/main_he_ref.h	4531
tfl/include/tfl/halfedge/mainpage_hed.h	4531
tfl/include/tfl/utlis/Handle.h	4536
tfl/include/tfl/utlis/HandleId.h	4538
tfl/src/halfedge/HeTriang.cpp	4539
tfl/src/utlis/HandleId.cpp	4539
viewlib/include/GoTools/viewlib/CurveResolutionSheet.h	4540
viewlib/include/GoTools/viewlib/DataHandler.h	4540
viewlib/include/GoTools/viewlib/DefaultDataHandler.h	4541
viewlib/include/GoTools/viewlib/gvActionSheet.h	4543
viewlib/include/GoTools/viewlib/gvApplication.h	4543
viewlib/include/GoTools/viewlib/gvCamera.h	4545
viewlib/include/GoTools/viewlib/gvColor.h	4546
viewlib/include/GoTools/viewlib/gvCurvePaintable.h	4547
viewlib/include/GoTools/viewlib/gvData.h	4548
viewlib/include/GoTools/viewlib/gvGenericTriPaintable.h	4549
viewlib/include/GoTools/viewlib/gvGenericTriQuadMesh.h	4550
viewlib/include/GoTools/viewlib/gvGenericTriQuadPaintable.h	4551
viewlib/include/GoTools/viewlib/gvGroup.h	4552
viewlib/include/GoTools/viewlib/gvGroupPropertySheet.h	4553
viewlib/include/GoTools/viewlib/gvLineCloudPaintable.h	4553
viewlib/include/GoTools/viewlib/gvNoopPaintable.h	4554
viewlib/include/GoTools/viewlib/gvObjectList.h	4554
viewlib/include/GoTools/viewlib/gvObserver.h	4555
viewlib/include/GoTools/viewlib/gvPaintable.h	4555
viewlib/include/GoTools/viewlib/gvPainter.h	4556
viewlib/include/GoTools/viewlib/gvParametricSurfacePaintable.h	4558
viewlib/include/GoTools/viewlib/gvPointCloudPaintable.h	4559
viewlib/include/GoTools/viewlib/gvPropertySheet.h	4559
viewlib/include/GoTools/viewlib/gvQuadsPaintable.h	4560
viewlib/include/GoTools/viewlib/gvRectangularSurfacePaintable.h	4560
viewlib/include/GoTools/viewlib/gvResolutionDialog.h	4561
viewlib/include/GoTools/viewlib/gvStandardMouseHandler.h	4562
viewlib/include/GoTools/viewlib/gvTexture.h	4562
viewlib/include/GoTools/viewlib/gvUtilities.h	4564
viewlib/include/GoTools/viewlib/gvView.h	4566
viewlib/include/GoTools/viewlib/ParametricSurfacePropertySheet.h	4567
viewlib/include/GoTools/viewlib/PointCloudPropertySheet.h	4567
viewlib/include/GoTools/viewlib/raster.h	4568
viewlib/include/GoTools/viewlib/RectangularSurfacePropertySheet.h	4569
viewlib/include/GoTools/viewlib/SplineCurvePropertySheet.h	4569
viewlib/include/GoTools/viewlib/SurfaceResolutionSheet.h	4570
viewlib/include/GoTools/viewlib/viewlib_doxymain.h	4570
viewlib/include/GoTools/viewlib/vol_and_lr/DataHandlerVolAndLR.h	4570
viewlib/include/GoTools/viewlib/vol_and_lr/gvApplicationVolAndLR.h	4571
viewlib/include/GoTools/viewlib/vol_and_lr/gvRectangularVolumePaintable.h	4572
viewlib/include/GoTools/viewlib/vol_and_lr/RectangularVolumePropertySheet.h	4574

Chapter 27

Module Documentation

27.1 RBD common library

Files

- file [include.h](#)
- file [myexcept.h](#)
- file [myexcept.cpp](#)

Namespaces

- [RBD_COMMON](#)
- [RBD_LIBRARIES](#)

Macros

- `#define` [use_namespace](#)
- `#define` [UseExceptions](#)
- `#define` [USING_DOUBLE](#)
- `#define` [bool_LIB](#) 0
- `#define` [TypeDefException](#)
- `#define` [DEFAULT_HEADER](#)
- `#define` [ATandT](#)

27.1.1 Detailed Description

27.1.2 Macro Definition Documentation

27.1.2.1 `#define` ATandT

Definition at line 277 of file [include.h](#).

27.1.2.2 #define bool_LIB 0

Definition at line 34 of file include.h.

27.1.2.3 #define DEFAULT_HEADER

Definition at line 105 of file include.h.

27.1.2.4 #define TypeDefException

Definition at line 47 of file include.h.

27.1.2.5 #define use_namespace

Definition at line 10 of file include.h.

27.1.2.6 #define UseExceptions

Definition at line 19 of file include.h.

27.1.2.7 #define USING_DOUBLE

Definition at line 31 of file include.h.

Chapter 28

Namespace Documentation

28.1 Go Namespace Reference

Namespaces

- [AdaptSurface](#)
- [BoundedUtils](#)
- [boxStructuring](#)
- [ClosestPoint](#)

Namespace for computing closest points.

- [cmUtils](#)

Various utility functions for the compositemodel module.

- [CoonsPatchGen](#)
- [CoonsPatchVolumeGen](#)

This namespace contains functions used to create a Coons Patch volume.

- [CreatorsUtils](#)

Related to the generation of cross tangent curves.

- [Curvature](#)

Curvature analysis related to curves.

- [CurvatureAnalysis](#)
- [CurveCreators](#)
- [CurveInterpolator](#)

Curve interpolation functionality not adapted to the [Interpolator](#) class.

- [FaceUtilities](#)
- [ftVolumeTools](#)

This namespace contains service functions related to [ftVolume](#).

- [GapRemoval](#)

Functionality for removal of gaps between two adjacent surfaces of various types.

- [GeometryTools](#)
- [HahnsSurfaceGen](#)
- [IntersectionUtils](#)

Various functions related to the intersection algorithms.

- [LinDepUtils](#)
- [LoftSurfaceCreator](#)

This namespace contains functions used to create lofted surfaces.

- [LoftVolumeCreator](#)

This namespace contains functions used to create lofted volumes.

- [LoopUtils](#)
- [LRApproxApp](#)
- [LRBSpline2DUtills](#)
- [LRSplineMBA](#)
- [LRSplineUtills](#)
- [LRSurfStitch](#)
- [Mesh2DUtills](#)
- [ModifySurf](#)
- [OffsetSurfaceUtills](#)
- [OffsetUtills](#)

Related to the generation of cross tangent curves.

- [orientCurves](#)

Sorts and orients a set of curves.

- [Path](#)

Functions related to sequence of edges.

- [PointSetApp](#)

Point set and triangulation applications.

- [qualityUtills](#)
- [RegularizeUtills](#)

Utility functionality for [RegularizeFace](#) and [RegularizeFaceSet](#).

- [Singular](#)
- [SplineDebugUtills](#)
- [SplineUtills](#)
- [SplitModelUtills](#)

Utility functionality for splitting of surface models.

- [SurfaceCreators](#)
- [SurfaceInterpolator](#)

Functionality for creating interpolating surfaces.

- [SurfaceModelUtills](#)

Utility functionality for splitting of surface models.

- [SurfaceOnVolumeTools](#)
- [SurfaceTools](#)

Free functions operating on parametric surfaces.

- [TesselatorUtills](#)

Related to the relative resolution of tessellation.

- [tpUtills](#)
- [Utills](#)

Namespace for some utility functions.

- [VolumeInterpolator](#)

This namespace contains functions used to interpolate a set of points.

- [VolumeModelCreator](#)
- [VolumeTools](#)

This namespace contains free functions operating on parametric volumes.

Classes

- class [AdaptCurve](#)
- struct [AdjacencyInfo](#)
 - Struct to store information about adjacency relations between two faces.*
- struct [Alg2DElem](#)
- struct [Alg3DElem](#)
- class [AlgObj2DInt](#)
 - Class for 2-dimensional algebraic intersection objects.*
- class [AlgObj3DInt](#)
- class [AlgObjectInt](#)
- class [ApproxCrvToSeqs](#)
- class [ApproxCurve](#)
- class [ApproxSurf](#)
- class [Array](#)
- class [BaryCoordSystem](#)
- class [BaryCoordSystemTriangle3D](#)
- class [BaseSurface](#)
- struct [BasisDerivs](#)
- struct [BasisDerivs2](#)
- struct [BasisDerivsSf](#)
- struct [BasisDerivsSf2](#)
- struct [BasisPts](#)
- struct [BasisPtsSf](#)
- class [BdCondFunctor](#)
- class [BernsteinMulti](#)
- class [BernsteinPoly](#)
- class [BernsteinTetrahedralPoly](#)
- class [BernsteinTriangularPoly](#)
- class [BezierTriangle](#)
 - Not documented.*
- class [Binomial](#)
- class [BlockBoundaryCondition](#)
- class [BlockPointBdCond](#)
- class [BlockSolution](#)
- class [Body](#)
 - A boundary represented solid.*
- struct [BoundaryFunctionInt](#)
- struct [BoundaryGeomInt](#)
- struct [BoundaryIntersectionData](#)
- class [BoundedCurve](#)
 - A bounded curve. Both parameter values and end points may be given to define the boundaries. Assuming that both points prefer parameter, or both points prefer points. Typically used to bound infinite curves, for instance lines.*
- class [BoundedSurface](#)
- class [BoundingBox](#)
- class [BsplineBasis](#)
- struct [CachedInterval](#)
 - Helper struct for saving bracketed bounds of influence areas.*
- class [CellDivision](#)
- class [Circle](#)
 - Class that represents a circle. It is a subclass of [ElementaryCurve](#) and thus has a parametrization.*
- class [ClosestPointCalculator](#)
- class [CompleteEdgeNet](#)

Complete the edge net of a [SurfaceModel](#). Fetches the wire frame model corresponding to a surface model, and extends it such that the extended wire frame will become the wire frame model corresponding to a volume model have the given surface model as its outer boundary. The extension to the wireframe is represented as pairs of vertices where these vertices lie at the endpoints of the missing edges. NB! This solution is currently not expected to handle all configurations.

- class [ComplexityInfo](#)
- class [CompositeBox](#)
- class [CompositeCurve](#)
- class [CompositeModel](#)
- class [CompositeModelFactory](#)
- class [CompositeModelFileHandler](#)
- class [CompositeSurface](#)
- class [ConcreteCreator](#)
- class [Cone](#)

Class that represents a cone. It is a subclass of [ElementarySurface](#), and thus has a parametrization and is non-selfintersecting.

- class [ConeVolume](#)

Class that represents a solid cone. It is a subclass of [ElementaryVolume](#), and has a natural parametrization in terms of a radius u , an angle v , and distance w : $\mathbf{p}(u, v, w) = \mathbf{C} + u(R + w \tan \alpha)((\cos v) \mathbf{x} + (\sin v) \mathbf{y}) + w \mathbf{z}$, where \mathbf{C} is the cone apex, R is the radius when $w = 0$, α is the cone angle, and \mathbf{x} , \mathbf{y} and \mathbf{z} are the (local) axes. The parametrization is bounded by: $0 \leq u \leq 1, 0 \leq v \leq 2\pi, -\infty < w < \infty$. The dimension is 3.

- class [ConnectionFunctor](#)
- class [CoordinateSystem](#)

Defines a Cartesian coordinate system.

- class [CPUclock](#)

A class for measuring CPU time in programs.

- class [Creator](#)
- class [CrossesValue](#)

[IntersectionPoolUtils](#). predicate for STL function.

- class [CrossTangentOffset](#)
- class [CrossTanOffDist](#)
- class [CurveBoundedDomain](#)
- class [CurveLoop](#)
- class [CurveModel](#)
- class [CurveOnSurface](#)

A curve living on a parametric surface. It either has got information about the curve in geometry space and in the parameter domain of the surface or the ability to compute the other representation given one. The curve may have information on whether it is a constant parameter or boundary curve on the surface.

- class [CurveOnVolume](#)

A curve living on a parametric volume. It either has got information about the curve in geometry space and in the parameter domain of the volume or the ability to compute the other representation given one.

- class [CurveTesselator](#)
- class [CvCvIntersector](#)

Class that performs intersection between two parametric curves.

- class [CvPtIntersector](#)
- struct [cvSetConstraint](#)
- class [Cylinder](#)

Class that represents a cylinder. It is a subclass of [ElementarySurface](#), and thus has a parametrization and is non-selfintersecting.

- class [CylinderInt](#)

Class for cylindrical algebraic intersection objects.

- class [CylinderVolume](#)

Class that represents a solid cylinder, maybe with empty interior like a straight tube. It is a subclass of [ElementaryVolume](#), and has a natural parametrization in terms of a radius u , an angle v , and distance w : $\mathbf{p}(u, v, w) = \mathbf{C} + u(\cos(v) \mathbf{x} + \sin(v) \mathbf{y}) + w \mathbf{z}$, where \mathbf{C} is a position vector, and \mathbf{x} , \mathbf{y} and \mathbf{z} are the (local) axes. The parametrization is bounded by: $R \leq u \leq S, 0 \leq v \leq 2\pi, -\infty < w < \infty$, where R and S are the inner and outer radius. The dimension is 3.

- class [DegeneratedIntersectionCurve](#)
IntersectionCurve that is degenerated into a single point.
- class [DirectionCone](#)
- class [Disc](#)
Class that represents a circular disc. It is a subclass of [ElementarySurface](#), and has a natural parametrization by polar coordinates in terms of a radius r and angle v : $\mathbf{p}(r, v) = \mathbf{C} + r((\cos v) \mathbf{x} + (\sin v) \mathbf{y})$, where \mathbf{C} is the centre position vector and \mathbf{x} and \mathbf{y} are the (local) axes. The parametrization is bounded by: $0 \leq r \leq R$ and $0 \leq v \leq 2\pi$, where R is the disc radius. The dimension is 2 or 3.
- class [Domain](#)
- class [EdgeVertex](#)
- class [Element2D](#)
- class [ElementaryCurve](#)
ElementaryCurve is a base class for elementary curves like lines and circles. Such curves have natural parametrizations and *ElementaryCurve* is therefore a subclass of [ParamCurve](#).
- class [ElementarySurface](#)
ElementarySurface is a base class for elementary surfaces like planes and cylinders. Such surfaces have natural parametrizations and *ElementarySurface* is therefore a subclass of [ParamSurface](#). These surfaces are non-self-intersecting.
- class [ElementaryVolume](#)
ElementaryVolume is a base class for elementary volumes like boxes and solid cylinders. Such volumes have natural parametrizations and *ElementaryVolume* is therefore a subclass of [ParamVolume](#). These volumes are non-self-intersecting.
- class [Ellipse](#)
Class that represents an ellipse. It is a subclass of [ElementaryCurve](#) and thus has a parametrization.
- struct [EntityList](#)
The entity number of all supported IGES entites.
- class [EvalCurve](#)
- class [EvalCurveSet](#)
- class [EvalFunctorCurve](#)
- class [EvalFunctorSurface](#)
- class [EvalOffsetSurface](#)
- class [EvalParamCurve](#)
- class [EvalSurface](#)
- class [FaceAdjacency](#)
Compute adjacency information for a set of surfaces.
- struct [FaceConnectivity](#)
- class [FaceConnectivityUtils](#)
Utilities used in adjacency computations of face sets.
- class [FaceSetQuality](#)
- class [FaceSetRepair](#)
- struct [Factorial](#)
Compile-time factorial calculations.
- struct [Factorial< 1 >](#)
- class [Factory](#)
- class [ftCell](#)
Helper class handling one of the cells in class [CellDivision](#).
- class [ftCellInfo](#)
Function object for sorting [ftCells](#).
- class [ftChartSurface](#)
- class [ftCurve](#)
- class [ftCurveSegment](#)
- class [ftEdge](#)
The [ftEdge](#) is a half-edge implementation of a topological data structure.
- class [ftEdgeBase](#)

Base class for edges. Defines the interface used in topology analysis.

- class [ftFaceBase](#)
- class [ftGraphEdge](#)
- class [ftGroupGeom](#)

A group of geometrical objects.

- class [ftLine](#)
- class [ftMessage](#)
- class [ftPlanarGraph](#)
- class [ftPlane](#)
- class [ftPoint](#)
- class [ftPointSet](#)
- class [ftSamplePoint](#)
- class [ftSearchNode](#)
- class [ftSmoothSurf](#)
- class [ftSSfEdge](#)
- class [ftSurface](#)
- class [ftSurfaceSet](#)
- class [ftSurfaceSetPoint](#)
- class [ftVolume](#)

A topological solid with a trivariate geometry description.

- class [Fun2Fun](#)
- class [FunctionMinimizer](#)
- class [GeneralMesh](#)
- class [GenericTriMesh](#)
- class [GeomObject](#)
- class [GeomObjectInt](#)
- class [GeoTol](#)

Class handling various tolerances.

- struct [go_iterator_traits](#)
- struct [go_iterator_traits< const T * >](#)
- struct [go_iterator_traits< T * >](#)
- class [GoTools](#)

Class containing some service functions for the [GoTools](#) "system".

- struct [GPos](#)
- class [GpuMatrix](#)
- class [HermiteAppC](#)
- class [HermiteApprEvalSurf](#)
- class [HermiteAppS](#)
- class [HermiteGrid1D](#)
- class [HermiteGrid1DMulti](#)
- class [HermiteGrid2D](#)
- class [HermiteInterpolator](#)
- class [Hyperbola](#)

Class that represents a hyperbola. It is a subclass of [ElementaryCurve](#) and thus has a parametrization.

- class [Identity](#)

Check coincidence.

- class [IGESconverter](#)
- struct [IGESdirentry](#)

Storage of all data contained in an IGES directory entity.

- struct [IGESheader](#)

Storage of all data contained in the IGES header.

- class [ImplicitizeCurveAlgo](#)
- class [ImplicitizeCurveAndVectorAlgo](#)

- class [ImplicitizePointCloudAlgo](#)
- class [ImplicitizeSurfaceAlgo](#)
- class [IndexMesh2DIterator](#)
- class [IntCrvEvaluator](#)
- class [InterpolatedIntersectionCurve](#)
- class [Interpolator](#)

Base class for spline interpolators or approximators.

- class [IntersectionCurve](#)
- class [IntersectionLink](#)
- class [IntersectionPoint](#)
- class [IntersectionPool](#)
- class [Intersector](#)
- class [Intersector2Obj](#)
- class [IntersectorAlgPar](#)
- class [IntersectorFuncConst](#)
- struct [IntPtInfo](#)
- class [IntResultsCompCv](#)
- class [IntResultsModel](#)
- class [IntResultsSfModel](#)
- struct [InverseFactorial](#)
- class [IsogeometricBlock](#)
- class [IsogeometricModel](#)
- class [IsogeometricSfBlock](#)
- class [IsogeometricSfModel](#)
- class [IsogeometricVolBlock](#)
- class [IsogeometricVolModel](#)
- class [IsoparametricIntersectionCurve](#)
- class [LiftCurve](#)
- class [Line](#)

Class that represents a line. It is a subclass of [ElementaryCurve](#) and thus has a parametrization.

- class [Line2DInt](#)

Class representing an algebraic line in 2-dimensional space.

- class [LineCloud](#)
- class [LineCloudTesselator](#)
- class [LineStrip](#)
- class [LockedDirDistFunc](#)
- class [Loop](#)

Primarily a loop connected to a face in a boundary represented solid or face set. May also be used to represent a general closed sequence of edges.

- class [LRBSpline2D](#)
- class [LRSplineEvalGrid](#)
- class [LRSplineSurface](#)
- class [LRSurfApprox](#)
- class [LRSurfSmoothLS](#)
- struct [LSSmoothData](#)
- class [MarchPoint](#)

Helper class for marching.

- class [MatrixXD](#)
- class [Mesh2D](#)
- class [Mesh2DIterator](#)
- class [ModelQuality](#)
- class [ModelRepair](#)
- class [NonEvaluableIntersectionCurve](#)

[IntersectionCurve](#) that cannot be evaluated.

- class [NoopTesselator](#)
- class [ObjectHeader](#)
- class [Par0FuncIntersector](#)

This class is performing intersections between two constants.

- class [Par1FuncIntersector](#)
- class [Par2FuncIntersector](#)
- class [Parabola](#)

Class that represents a parabola. It is a subclass of [ElementaryCurve](#) and thus has a parametrization.

- class [Parallelepiped](#)

Class that represents a solid parallelepiped. It is a subclass of [ElementaryVolume](#), and thus has a parametrization.

- class [Param0FunctionInt](#)
- class [Param1FunctionInt](#)
- class [Param2FunctionInt](#)
- class [ParamCurve](#)
- class [ParamCurveInt](#)
- class [ParametricSurfaceTesselator](#)
- class [ParamFunctionInt](#)
- class [ParamGeomInt](#)
- class [ParamObjectInt](#)
- class [ParamPointInt](#)
- class [ParamSurface](#)
- class [ParamSurfaceInt](#)
- class [ParamVolume](#)
- class [Plane](#)

Class that represents a plane. It is a subclass of [ElementarySurface](#), and thus has a parametrization and is non-selfintersecting.

- class [PlaneInt](#)

Class representing planar algebraic intersection objects.

- class [Point](#)
- class [PointCloud](#)
- class [PointOnCurve](#)
- class [PointOnEdge](#)
- class [PointSequence](#)
- struct [preEvaluationSf](#)
- struct [preEvaluationVol](#)
- class [ProjectCurve](#)
- class [ProjectCurveAndCrossTan](#)
- class [ProjectIntersectionCurve](#)
- class [PtPtIntersector](#)

This class performs intersection between two points.

- class [QuadMesh](#)
- class [QualityResults](#)
- class [Rational](#)
- struct [raw_pointer_comp](#)
- class [RectangularSurfaceTesselator](#)
- class [RectangularVolumeTesselator](#)
- class [RectDomain](#)
- class [RectGrid](#)
- class [RectGridTesselator](#)
- struct [RegistrationInput](#)

Struct for input to registration, either raw, fine or combined.

- struct [RegistrationResult](#)

Struct for result from registration process, either raw, fine or combined.

- class [Registrar](#)
- class [RegularizeFace](#)

Split one face into a number of 4-sided domains without inner trimming. This class is intended for use in block structuring. One face with possible inner and outer trimming is split according to certain rules to result in a face set with 4 sided faces although faces with less than 4 sides can occur. A side is defined as a piece of the face boundary between two corners or between vertices where there are more than one adjacent face. The trimmed surfaces being output from this class can later be approximated by spline surfaces. The splitting procedure is recursive.
- class [RegularizeFaceSet](#)
- class [RegularMesh](#)
- class [RegularVolMesh](#)
- class [RotatedBox](#)
- struct [RotationInfo](#)
- struct [SamplePointData](#)

Sample data related to one face. The struct contains one point in a point set with information about position, associated surface normal and curvature. Points lying on the face boundary know about its associated edge. If the surface normal and curvature information regarding boundary points is not unique, the associated data is set to be equal to MAX_DOUBLE.
- class [ScratchVect](#)
- struct [SecondOrderProperties](#)
- class [SfBoundaryCondition](#)
- class [SfCvIntersector](#)
- class [SfPointBdCond](#)
- class [SfPtIntersector](#)
- class [SfSelfIntersector](#)

This class finds self-intersections for a parametric surface.
- class [SfSfIntersector](#)

This class performs intersection between two parametric surfaces.
- class [SfSolution](#)
- struct [sideConstraint](#)
- struct [sideConstraintSet](#)
- struct [SingBox](#)
- class [SingularityInfo](#)
- struct [SingUnion](#)
- class [SmoothCurve](#)
- class [SmoothCurveSet](#)
- class [SmoothSurf](#)
- class [SmoothSurfSet](#)
- class [SmoothTransition](#)
- class [SmoothVolume](#)

This class modifies a NURBS volume with respect to smoothness, editing constraints and boundary conditions. Specified coefficients are adjusted to minimize a functional combining the different modification conditions.
- class [SolveBCG](#)
- class [SolveCG](#)
- class [SolveCGCO](#)
- class [SpaceIntCrv](#)
- class [Sphere](#)

Class that represents a sphere. It is a subclass of [ElementarySurface](#), and thus has a parametrization and is non-selfintersecting.
- class [SphereInt](#)

Class representing spherical algebraic intersection objects.
- class [SphereVolume](#)

Class that represents a solid sphere. It is a subclass of [ElementaryVolume](#), and thus has a parametrization.
- class [Spline1FunctionInt](#)
- class [Spline2FunctionInt](#)
- class [SplineApproximator](#)

- class [SplineCurve](#)
SplineCurve provides methodes for storing, reading and manipulating rational and non-rational B-spline curves.
- class [SplineCurveInt](#)
Class representing the "intersection object" of a spline curve.
- class [SplineInterpolator](#)
- class [SplineSurface](#)
SplineSurface provides methodes for storing, reading and manipulating rational and non-rational B-spline surfaces.
- class [SplineSurfaceInt](#)
- class [SplineVolume](#)
SplineVolume provides methodes for storing, reading and manipulating rational and non-rational B-spline volumes.
- class [Streamable](#)
- class [SurfaceAssembly](#)
- class [SurfaceModel](#)
- class [SurfaceOfLinearExtrusion](#)
- class [SurfaceOfRevolution](#)
Class that represents a surface of revolution. A [SurfaceOfRevolution](#) is swept out by a [SplineCurve](#) that is rotated around an axis with a complete revolution, and is thereby a parametric surface.
- class [SurfaceOnVolume](#)
A surface living on a parametric volume. It either has got information about the surface in geometry space and in the parameter domain of the volume or both. The surface may have information on whether it is a constant parameter or boundary surface on the volume.
- class [SweepSurfaceCreator](#)
Functionality to create a spline surface by linear or rotational sweep.
- class [SweepVolumeCreator](#)
Class with static methods for volume creation by sweep methods.
- class [Tesselator](#)
- class [TestInDomain](#)
IntersectionPoolUtils. predicate for STL function.
- class [Torus](#)
Class that represents a torus. It is a subclass of [ElementarySurface](#), and thus has a parametrization. A torus may be degenerate. Then the minor radius is greater than the major radius, and it is in principle selfintersecting.
- class [TorusVolume](#)
Class that represents a solid torus, maybe with empty interior like a circular pipe, and/or a section (not full revolution along the main axis). It is a subclass of [ElementaryVolume](#), and has a natural parametrization in terms of a radius u and two angles v and w : $\mathbf{p}(u, v, w) = \mathbf{C} + (R + u \cos w)((\cos v) \mathbf{x} + (\sin v) \mathbf{y}) + (u \sin w) \mathbf{z}$, where \mathbf{C} is a position vector, R is the major radius, and \mathbf{x} , \mathbf{y} and \mathbf{z} are the (local) axes. The parametrization for a full solid torus is bounded by: $0 \leq u \leq r$, $0 \leq v, w \leq 2\pi$, where r is the minor axis. A bended pipe will have a positive minimal limit for u , while a torus segment will have other limits for v . The dimension is 3.
- class [tpEdge](#)
- class [tpFace](#)
- class [tpMarchPoint](#)
Helper class for marching.
- class [tpTableEntry](#)
- struct [tpTolerances](#)
- struct [tpTopologicalInfo](#)
- class [tpTopologyTable](#)
- class [Triangle](#)
- class [TrimCurve](#)
- class [ttlPoint](#)
- class [Vertex](#)
The vertex class represents the vertex entity in a boundary represented solid or face set.
- class [VolBoundaryCondition](#)
- class [VolPointBdCond](#)
- class [VolSolution](#)

- class [VolumeAdjacency](#)
Adjacency analysis of volume models.
- struct [VolumeAdjacencyInfo](#)
Struct to store information about adjacency relations between two bodies.
- class [VolumeModel](#)
A set of volumes including topology information.
- class [VolumeModelFileHandler](#)
- class [VolumeParameterCurve](#)
An evaluator based curve representing the parameter domain curve in a given volume which represents the same curve as a given space curve. Project a geometry curve into the parameter domain of a volume.
- class [VolumeSpaceCurve](#)
An evaluator based curve representing the space curve corresponding to parameter domain curve in a given volume. Compute the space curve corresponding to a curve in the parameter domain of a volume.
- class [Zero_Parameter_Span_Error](#)
Error object used internally in [IntersectionCurve](#).

Typedefs

- typedef `std::vector< ftSearchNode >::iterator` [GraphIter](#)
- typedef `std::list< shared_ptr< ftSamplePoint > >` [PointList](#)
- typedef `ftSamplePoint *` [PointIter](#)
- typedef `std::pair< std::pair< double, double >, std::pair< ftFaceBase *, int > >` [BoundaryPiece](#)
- typedef struct [Go::sideConstraint](#) [sideConstraint](#)
- typedef struct [Go::sideConstraintSet](#) [sideConstraintSet](#)
- typedef `PointCloud< 3 >` [PointCloud3D](#)
- typedef `PointCloud< 4 >` [PointCloud4D](#)
- typedef `Array< double, 2 >` [Vector2D](#)
Typedef for ease of use in frequently used case.
- typedef `Array< double, 3 >` [Vector3D](#)
Typedef for ease of use in frequently used case.
- typedef `Array< double, 4 >` [Vector4D](#)
Typedef for ease of use in frequently used case.
- typedef `BaryCoordSystem< 2 >` [BaryCoordSystem2D](#)
- typedef `BaryCoordSystem< 3 >` [BaryCoordSystem3D](#)
- typedef `Element2D simpleElement`
- typedef `std::map< LRBSpline2D *, size_t >` [BsplineIndexMap](#)

Enumerations

- enum `closestPointLevel` { LOCAL_SEARCH = 1, SEMI_LOCAL_SEARCH, GLOBAL_SEARCH }
- enum `ftCurveType` { CURVE_NOTYPE = -1, CURVE_INTERSECTION, CURVE_EDGE, CURVE_KINK, CURVE_CORNER, CURVE_GAP, CURVE_SINGULAR }
- enum `ftmessages` { FT_NOT_IMPLEMENTED = -99, FT_NO_DATA, FT_BAD_INPUT_FILE, FT_ERROR_IN_READ_IGES, FT_DISCONTINUITY, FT_UNEXPECTED_DATA_TYPE, FT_NON_4_SIDED_SURF, FT_WRONG_NO_OF_BOUNDARIES, FT_INSUFFICIENT_MESH_CURVES, FT_ERROR_IN_TOPOLOGY_ANALYSIS, FT_TOPOLOGY_PROBLEM, FT_ERROR_IN_SURFACE_CREATION, FT_ERROR_IN_PARAMETERIZE, FT_ERROR_IN_SURFACE_TRIMMING, FT_NOT_SUPPORTED, FT_NOT_SPLINE_SURF, FT_ERROR_IN_SURFACE_GRIDING, FT_NEIGHBOUR_GRID_SIZE_DIFFER, FT_OK = 0, FT_APPROX_ERROR_TOO_LARGE, FT_COULD_NOT_REFINE_SURFACE, FT_FEATURE_ERROR_TOO_LARGE, FT_DEGENERATE_PATCH_CREATED, FT_SURFACE_ALREADY_CREATED, FT_DEGENERATE_SURFACE, FT_UNEXPECTED_INPUT_OBJECT_IGNORED, FT_FACES_OVERLAP,

- ```

FT_COULD_NOT_SMOOTH_SURFACE,
FT_COULD_NOT_SMOOTH_SURFACESET, FT_NO_SMOOTHING, FT_FAILED_CREATING_GRAPH,
FT_GRID_NOT_CREATED,
FT_SURFACE_NOT_MODIFIED }

```
- enum `IntersectionType` {  
`No_Type` = 0, `SurfaceModel_SurfaceModel`, `SurfaceModel_Plane`, `SurfaceModel_Line`,  
`CompositeCurve_CompositeCurve`, `CompositeCurve_Plane`, `CompositeCurve_Line` }
  - enum `ClassType` {  
`Class_Unknown` = 0, `Class_SplineCurve` = 100, `Class_CurveOnSurface` = 110, `Class_CurveOnVolume` = 111,  
`Class_Line` = 120, `Class_Circle` = 130, `Class_Ellipse` = 140, `Class_BoundedCurve` = 150,  
`Class_Hyperbola` = 160, `Class_Parabola` = 170, `Class_SplineSurface` = 200, `Class_BoundedSurface` = 210,  
`Class_SurfaceOnVolume` = 211, `Class_GoBaryPolSurface` = 220, `Class_GoHBSplineParamSurface` = 230,  
`Class_CompositeSurface` = 240,  
`Class_Plane` = 250, `Class_Cylinder` = 260, `Class_SurfaceOfLinearExtrusion` = 261, `Class_Sphere` = 270,  
`Class_Cone` = 280, `Class_Torus` = 290, `Class_SurfaceOfRevolution` = 291, `Class_Disc` = 292,  
`Class_LRSplineSurface` = 293, `Class_TSplineSurface` = 294, `Class_Go3dsObject` = 300, `Class_GoHeTriang`  
= 310,  
`Class_GoSdTriang` = 320, `Class_GoQuadMesh` = 330, `Class_GoHybridMesh` = 340, `Class_ParamTriang` = 350,  
`Class_GoVrmlGeometry` = 360, `Class_PointCloud` = 400, `Class_LineCloud` = 410, `Class_GoTriangleSets` = 500,  
`Class_RectGrid` = 510, `Class_SplineVolume` = 700, `Class_BoundedVolume` = 710, `Class_Parallelepiped` = 720,  
`Class_SphereVolume` = 721, `Class_CylinderVolume` = 722, `Class_ConeVolume` = 723, `Class_TorusVolume`  
= 724,  
`Class_LRSplineVolume` = 793 }
  - enum `AlgorithmChoice` { `GEOMETRICAL`, `FUNCTIONAL` }
  - enum {  
`pretop_UNDEF`, `pretop_IN`, `pretop_OUT`, `pretop_ON`,  
`pretop_AT` }
  - enum `IteratorType` { `Iterator_parametric` = 0, `Iterator_geometric` = 1 }
  - enum `PointSequenceType` { `PSTPoint`, `PSTPointTangent`, `PSTScattered` }
- Type of point.*
- enum `RegistrationReturnType` {  
`RegistrationOK`, `TooFewPoints`, `PointSetSizeDiff`, `AreaTooSmall`,  
`SolveFailed` }
  - enum `ftTangPriority` { `ftNoType` = 0, `ftMaster`, `ftSlave` }
  - enum `FileFormat` { `go`, `disp`, `IGES` }
  - enum `IGESSection` {  
`S`, `G`, `D`, `P`,  
`T`, `E` }
  - enum `EvalKind` { `SPACECURVE`, `PARAMCURVE_1`, `PARAMCURVE_2` }
  - enum `TangentDomain` { `GEOM`, `PARAM1`, `PARAM2` }
  - enum `EstimateDirection` { `FORWARDS`, `BACKWARDS` }
  - enum `IntPtClassification` {  
`DIR_UNDEF` = 0, `DIR_IN`, `DIR_OUT`, `DIR_PARALLEL`,  
`DIR_PERPENDICULAR`, `DIR_HIGHLY_SINGULAR`, `DIR_TOUCH` }
  - enum `IntPtLocation` {  
`LOC_OUTSIDE_BOTH` = 0, `LOC_INSIDE_BOTH`, `LOC_EDGE_ONE`, `LOC_CORNER_ONE`,  
`LOC_CORNER_BOTH`, `LOC_EDGE_BOTH` }
  - enum `LinkType` {  
`LINK_UNDEFINED` = 0, `SIMPLE_CONE`, `SIMPLE_MONOTONE`, `SIMPLE_IMPLICIT`,  
`SIMPLE_TWO_POINTS`, `COINCIDENCE_CVPT`, `COINCIDENCE_CVCV`, `COINCIDENCE_SFCV`,  
`COINCIDENCE_SFPT`, `MICRO_CVCV`, `MICRO_SFPT`, `MICRO_SFCV`,  
`MICRO_SFSF`, `MICRO_PAR1FUNC`, `MICRO_PAR2FUNC`, `LINEAR_CVCV`,  
`LINEAR_SFCV`, `LINEAR_SFSF`, `MERGED_UNDEFINED`, `MERGED_SIMPLE_CONE`,  
`MERGED_SIMPLE_MONOTONE`, `MERGED_SIMPLE_CONE_MONOTONE`, `MERGED_COINCIDENCE`↵



- `_SFCV_SFCV`, `DEG_TRIANGLE`,  
`POST_ITERATE`, `BRANCH_CONNECTION`, `COMPLEX_SFSF`, `SPLIT_LINK`,  
`REPAIRED_MISSING_LINK`, `INSIDE_OUTSIDE_SINGULARITY_BOX` }
- enum `SingularityClassification` { `NO_SING` = 0, `INIT_SING`, `DIVIDED_SING`, `KEEP_SING` }  
*This enum classifies the type of singularity.*
- enum `SingularityType` {  
`ORDINARY_POINT` = 0, `TANGENTIAL_POINT`, `ISOLATED_POINT`, `BRANCH_POINT`,  
`HIGHER_ORDER_POINT` }
- enum `SubdivisionClassification` {  
`CANNOT_SUBDIVIDE` = 0, `DIVIDE_DEG`, `DIVIDE_CRITICAL`, `DIVIDE_HIGH_SING`,  
`DIVIDE_SING`, `DIVIDE_KNOT`, `DIVIDE_INT`, `DIVIDE_PAR` }  
*This enum classifies the type of subdivision.*
- enum `BdConditionType` {  
`UNKNOWN` = 0, `ZERO_DIRICHLET`, `CONSTANT_DIRICHLET`, `DIRICHLET`,  
`ZERO_NEUMANN`, `NEUMANN`, `SYMMETRY` }
- enum `Direction2D` { `XFIXED` = 0, `YFIXED` = 1 }
- enum `testSuite` {  
`IDENTICAL_VERTICES` = 0, `IDENTICAL_EDGES` = 1, `EMBEDDED_EDGES` = 2, `IDENTICAL_FACES` = 3,  
`EMBEDDED_FACES` = 4, `MINI_CURVE` = 5, `MINI_SURFACE` = 6, `MINI_EDGE` = 7,  
`MINI_FACE` = 8, `SLIVER_FACE` = 9, `NARROW_REGION` = 10, `DEGEN_SRF_BD` = 11,  
`DEGEN_SRF_CORNER` = 12, `VANISHING_TANGENT` = 13, `VANISHING_NORMAL` = 14, `EDGE_VERTEX`↵  
`EX_DISTANCE` = 15,  
`FACE_VERTEX_DISTANCE` = 16, `FACE_EDGE_DISTANCE` = 17, `EDGE_POSITION_DISCONT` = 18, `E`↵  
`DGE_TANGENTIAL_DISCONT` = 19,  
`FACE_POSITION_DISCONT` = 20, `FACE_TANGENTIAL_DISCONT` = 21, `LOOP_CONSISTENCY` = 22, `L`↵  
`OOO_ORIENTATION` = 23,  
`FACE_ORIENTATION` = 24, `CV_G1DISCONT` = 25, `CV_C1DISCONT` = 26, `SF_G1DISCONT` = 27,  
`SF_C1DISCONT` = 28, `CV_CURVATURE_RADIUS` = 29, `SF_CURVATURE_RADIUS` = 30, `EDGE_ACU`↵  
`TE_ANGLE` = 31,  
`FACE_ACUTE_ANGLE` = 32, `LOOP_INTERSECTION` = 33, `LOOP_SELF_INTERSECTION` = 34, `INDIST`↵  
`INCT_KNOTS` = 35 }
- enum `tpJointType` {  
`JOINT_G1` = 0, `JOINT_KINK` = 1, `JOINT_G0` = 2, `JOINT_GAP` = 3,  
`JOINT_DISC` = 4, `JOINT_NONE` = 5 }
- enum `CoefStatus` { `CoefFree`, `CoefKnown`, `CoefAvoid`, `CoefOther` }

## Functions

- `double areaTriangle` (`const Vector2D` &corner1, `const Vector2D` &corner2, `const Vector2D` &corner3)
- `void GaussQuadValues` (`const BsplineBasis` &basis, `std::vector`< `double` > &parameters, `std::vector`< `double` > &par\_weights)  
*Functions used to compute integrals of inner products of B-splines.*
- `void GaussQuadInner` (`const BsplineBasis` &basis, int ider, `double` lim1, `double` lim2, `double` \*\*\*integral)
- `void GaussQuadInnerFlat` (`const BsplineBasis` &basis, int derivs, int start\_der, int gap, `double` lim1, `double` lim2, `std::vector`< `double` > &integral)
- `void GaussQuadInner2` (`const BsplineBasis` &basis, int ider, `double` lim1, `double` lim2, `double` \*\*integral)
- `void GaussQuadInnerRational` (`const BsplineBasis` &basis, int ider, `double` lim1, `double` lim2, `shared_ptr`< `SplineCurve` > bspline\_curve, `double` \*\*\*integral)
- `template`<class `PtrToCurveType` >  
`double computeLoopGap` (`const` `std::vector`< `PtrToCurveType` > &curves)  
*Computes the largest gap in the loop specified by the vector of curves.*
- `int extremalPtSurfSurf` (`ParamSurface` \*psurf1, `ParamSurface` \*psurf2, int constraints[2], `double` constraints↵  
\_par[2], `double` limit[], `double` enext[], `double` gpos[], `double` angle\_tol)
- `template`<class `T` >  
`void Register` ()

- void `intersectCurvePoint` (`const ParamCurve *crv`, `Point pnt`, `double epsge`, `std::vector< double >` &intersections, `std::vector< std::pair< double, double >` > &int\_crvs)
- void `intersect2Dcurves` (`const ParamCurve *cv1`, `const ParamCurve *cv2`, `double epsge`, `std::vector< std::pair< double, double >` > &intersections, `std::vector< int >` &pretopology, `std::vector< std::pair< std::pair< double, double >`, `std::pair< double, double >` > > &int\_crvs)
- void `intersectcurves` (`SplineCurve *cv1`, `SplineCurve *cv2`, `double epsge`, `std::vector< std::pair< double, double >` > &intersections)
- void `intersectCurveSurf` (`const SplineCurve *cv`, `const SplineSurface *sf`, `double epsge`, `std::vector< std::pair< double, Point >` > &int\_pts, `std::vector< int >` &pretopology, `std::vector< std::pair< std::pair< double, Point >`, `std::pair< double, Point >` > > &int\_crvs)
  - Intersect a spline curve and a spline surface.*
- void `closestPtCurves` (`SplineCurve *cv1`, `SplineCurve *cv2`, `double epsge`, `double &par1`, `double &par2`, `double &dist`)
- `SISLCurve GO_API * Curve2SISL` (`const SplineCurve &cv`, `bool copy=true`)
- `SISLCurve GO_API * Curve2SISL_rat` (`const SplineCurve &cv`)
- `SplineCurve GO_API * SISLCurve2Go` (`const SISLCurve *const cv`)
- `SISLSurf GO_API * GoSurf2SISL` (`const SplineSurface &sf`, `bool copy=true`)
- `SplineSurface GO_API * SISLSurf2Go` (`SISLSurf *sf`)
- `std::istream & operator>>` (`std::istream &is`, `Go::Streamable &obj`)
- `std::ostream & operator<<` (`std::ostream &os`, `const Go::Streamable &obj`)
- `bool point_inside_contour` (`const double x0`, `const double y0`, `const double *const vertices`, `const std::vector< int >` &contour)
- `bool segment_contour_intersection_for_s2m` (`const double x0`, `const double y0`, `const double x1`, `const double y1`, `const double *const vertices`, `const std::vector< int >` &contour, `double &x`, `double &y`, `double &s`, `const bool snap_ends=false`)
- `std::vector< short_list_short_list > sort_2dpoly_segments` (`const double *const vertices`, `const std::vector< int >` &contour, `const bool transposed=false`)
- `int is_inside` (`const std::vector< Go::Vector3D >` &trim\_curve\_p, `const std::vector< int >` &contour, `const double u`, `const double v`)
- `bool is_on_corner` (`const std::vector< Go::Vector3D >` &trim\_curve\_p, `const std::vector< int >` &contour, `const double u`, `const double v`)
- `int is_on_contour` (`const std::vector< Go::Vector3D >` &trim\_curve\_p, `const std::vector< int >` &contour, `const double u`, `const double v`)
- `bool degenerate_triangle` (`const Vector2D &c1`, `const Vector2D &c2`, `const Vector2D &c3`)
- void `make_trimmed_mesh` (`shared_ptr< ParamSurface >` srf, `std::vector< shared_ptr< ParamCurve >` > &crv\_set, `std::vector< Vector3D >` &vert, `std::vector< Vector2D >` &vert\_p, `std::vector< int >` &bd, `std::vector< Vector3D >` &norm, `std::vector< int >` &mesh, `std::vector< Vector3D >` &trim\_curve, `std::vector< Vector3D >` &trim\_curve\_p, `const int dn`, `const int dm`, `double bd_res_ratio`)
- `template<typename T, int Dim>`  
`Go::Array< T, Dim > operator*` (`T d`, `const Go::Array< T, Dim >` &v)
  - The product of a vector and a scalar.*
- `template<typename T, int Dim>`  
`std::istream & operator>>` (`std::istream &is`, `Go::Array< T, Dim >` &v)
  - Stream extraction for Array.*
- `template<typename T, int Dim>`  
`std::ostream & operator<<` (`std::ostream &os`, `const Go::Array< T, Dim >` &v)
  - Stream insertion for Array.*
- `template<class T, int Dim>`  
`Array< T, Dim > operator*` (`const Array< double, Dim >` &a, `const T b`)
- `double binom` (`int n`, `int i`)
  - Computes the binomial coefficient:  $n! / (i! (n-i)!)$*
- `double factorial` (`int n`)
  - computes  $n!$  ( $n$  factorial)*
- `double trinomial` (`int n`, `int i`, `int j`)
  - computes the trinomial coefficient:  $n! / (i! j! (n-i-j)!)$*

- `double quadrimonial` (int n, int i, int j, int k)  
*computes the quadrimonial coefficient:  $n! / (i! j! k! (n-i-j-k)!)$*
- `template<class Functor >`  
`double brent_minimize` (const Functor &f, double a, double b, double c, double &parmin, const double rel\_← tolerance=std::sqrt(std::numeric\_limits< double >::epsilon()))
- `template<typename Iterator >`  
`bool strictly_increasing` (Iterator begin, Iterator end)
- `template<typename Array >`  
`bool strictly_increasing` (const Array &A)
- `template<typename Iterator >`  
`bool weakly_increasing` (Iterator begin, Iterator end)
- `template<typename Array >`  
`bool weakly_increasing` (const Array &A)
- `template<typename Iterator >`  
`bool strictly_decreasing` (Iterator begin, Iterator end)
- `template<typename Array >`  
`bool strictly_decreasing` (const Array &A)
- `template<typename Iterator >`  
`bool weakly_decreasing` (Iterator begin, Iterator end)
- `template<typename Array >`  
`bool weakly_decreasing` (const Array &A)
- `template<typename ValueType >`  
`bool nondecreasing` (ValueType a, ValueType b, ValueType c)
- `template<typename ValueType >`  
`bool nonincreasing` (ValueType a, ValueType b, ValueType c)
- `template<typename ValueType >`  
`bool strictly_decreasing` (ValueType a, ValueType b, ValueType c)
- `template<typename ValueType >`  
`bool strictly_increasing` (ValueType a, ValueType b, ValueType c)
- `template<typename ValueType >`  
`bool interval_overlap` (ValueType front1, ValueType back1, ValueType front2, ValueType back2)
- `template<typename Iterator >`  
`int compare_seq` (Iterator begin\_1, Iterator end\_1, Iterator begin\_2, Iterator end\_2)
- `shared_ptr< boxStructuring::BoundingBoxStructure > preprocessClosestVectors` (const std::vector< shared\_ptr< GeomObject > > &surfaces, double par\_len\_el)
- `void closestPointSingleCalculation` (int pt\_idx, int start\_idx, int skip, const std::vector< float > &inPoints, const std::vector< std::vector< double > > &rotationMatrix, const Point &translation, const shared\_ptr< box← Structuring::BoundingBoxStructure > &boxStructure, std::vector< float > &result, std::vector< std::vector< int > > &lastBoxCall, int return\_type, int search\_extend)
- `std::vector< float > closestPointCalculations` (const std::vector< float > &pts, const shared\_ptr< box← Structuring::BoundingBoxStructure > &structure, const std::vector< std::vector< double > > &rotation← Matrix, const Point &translation, int return\_type, int start\_idx, int skip, int max\_idx, int search\_extend=3, bool m\_core=true)
- `std::vector< float > closestVectorsOld` (const std::vector< float > &inPoints, const shared\_ptr< box← Structuring::BoundingBoxStructure > &boxStructure, const std::vector< std::vector< double > > &rotation← Matrix, const Point &translation, int return\_type, int start\_idx, int skip, int max\_idx, int search\_extend=3)  
*Calculates the closest points of a point cloud to a surface model, by not using the inside polygons in closestVectors()*
- `std::vector< float > closestPointCalculations` (const std::vector< float > &pts, const shared\_ptr< box← Structuring::BoundingBoxStructure > &structure, const std::vector< std::vector< double > > &rotation← Matrix, const Point &translation, int return\_type)
- `std::vector< float > closestDistances` (const std::vector< float > &pts, const shared\_ptr< boxStructuring← ::BoundingBoxStructure > &structure, const std::vector< std::vector< double > > &rotationMatrix, const Point &translation)
- `std::vector< float > closestSignedDistances` (const std::vector< float > &pts, const shared\_ptr< box← Structuring::BoundingBoxStructure > &structure, const std::vector< std::vector< double > > &rotation← Matrix, const Point &translation)

- `std::vector< float > closestPoints` (`const std::vector< float > &pts`, `const shared_ptr< boxStructuring::BoundingBoxStructure > &structure`, `const std::vector< std::vector< double > > &rotationMatrix`, `const Point &translation`)
- `double curvatureRadius` (`const std::vector< Point > &der`, `std::vector< Point > &unitder`)
- `double stepLenFromRadius` (`double radius`, `double aepsge`)
- `double tanLenFromRadius` (`double radius`, `double angle`)
- `void getHermiteData` (`const std::vector< Point > &der1`, `const std::vector< Point > &der2`, `double &parint`, `double &len1`, `double &len2`)
- `template<class Functor >`  
`void minimise_conjugated_gradient` (`FunctionMinimizer< Functor > &dfmin`)
- `template<typename Functor >`  
`void trapezoidal` (`Functor &f`, `double a`, `double b`, `double &s`, `int n`)
- `template<typename Functor >`  
`double simpsons_rule` (`Functor &f`, `double a`, `double b`, `const double eps=1.0e-6`, `const int max_iter=20`)
- `template<typename Functor >`  
`double gaussian_quadrature` (`Functor &f`, `double a`, `double b`)
- `template<typename Functor2D >`  
`double simpsons_rule2D` (`Functor2D &f`, `double ax`, `double bx`, `double ay`, `double by`, `const double eps=1.0e-6`, `const int max_iter=20`)
- `template<typename Functor2D >`  
`double gaussian_quadrature2D` (`Functor2D &f`, `double ax`, `double bx`, `double ay`, `double by`)
- `template<typename SquareMatrix >`  
`void LUDecomp` (`SquareMatrix &mat`, `int num_rows`, `int *perm`, `bool &parity`)
- `template<typename SquareMatrix , typename T >`  
`void LUsolveSystem` (`SquareMatrix &A`, `int num_unknowns`, `T *vec`)
- `template<typename SquareMatrix , typename T >`  
`void forwardSubstitution` (`const SquareMatrix &L`, `T *x`, `int num_unknowns`)
- `template<typename SquareMatrix , typename T >`  
`void backwardSubstitution` (`const SquareMatrix &U`, `T *x`, `int num_unknowns`)
- `template<typename SquareMatrix >`  
`void forwardSubstitution` (`const SquareMatrix &L`, `std::vector< double > *x`, `int num_unknowns`)
- `template<typename SquareMatrix >`  
`void backwardSubstitution` (`const SquareMatrix &U`, `std::vector< double > *x`, `int num_unknowns`)
- `template<typename T , int Dim>`  
`std::ostream & operator<<` (`std::ostream &os`, `const MatrixXD< T, Dim > &m`)  
  
*output operator*
- `Point operator*` (`double d`, `const Point &p`)  
  
*The product of a vector and a scalar.*
- `std::istream & operator>>` (`std::istream &is`, `Go::Point &v`)  
  
*Stream extraction for Point.*
- `std::ostream & operator<<` (`std::ostream &os`, `const Go::Point &v`)  
  
*Stream insertion for Point.*
- `bool operator<` (`const Point &p1`, `const Point &p2`)  
  
*Less than operator.*
- `bool operator==` (`const Point &p1`, `const Point &p2`)  
  
*Equal operator.*
- `void normalNoise` (`double *res`, `double mean_err`, `int num_samples`)
- `void uniformNoise` (`double *res`, `double lval`, `double uval`, `int num_samples`)
- `Rational operator+` (`const Rational &r1`, `const Rational r2`)
- `Rational operator-` (`const Rational &r1`, `const Rational r2`)
- `Rational operator*` (`const Rational &r1`, `const Rational r2`)
- `Rational operator/` (`const Rational &r1`, `const Rational r2`)
- `std::ostream & operator<<` (`std::ostream &os`, `const Rational &p`)
- `RegistrationResult rawRegistration` (`const std::vector< Point > &points_fixed`, `const std::vector< Point > &points_transform`, `bool allow_rescaling`, `RegistrationInput params`)

- void `addToLinearSystem` (int pt\_idx, const std::vector< Point > &points\_fixed, const std::vector< Point > &points\_transform, bool allow\_rescaling, const std::vector< std::vector< double > > &id, const Point &fine←\_R, const Point &fine\_T, double fine\_s, const std::vector< std::vector< double > > &m\_rot\_R, double s2, double R2, bool zero\_R, const std::vector< std::vector< std::vector< double > > > &lhs\_matrix, const std::vector< std::vector< double > > &rhs\_matrix)
  - *Add the contribution from one single point to the linear system coefficients used during fine registration.*
- `RegistrationResult fineRegistration` (const std::vector< Point > &points\_fixed, const std::vector< Point > &points\_transform, bool allow\_rescaling, RegistrationInput params)
- `RegistrationResult registration` (const std::vector< Point > &points\_fixed, const std::vector< Point > &points\_transform, bool allow\_rescaling, RegistrationInput params)
- `double getCurrentTime` ()
  - *Number of seconds since some (probably system-dependent) epoch.*
- void `systemSleep` (double sleep\_time)
  - *Sleep for sleep\_time seconds.*
- template<typename T >
  - `T determinantOf` (const Array< T, 2 > \*a)
- template<typename T >
  - `T determinantOf` (const Array< T, 3 > \*a)
- template<typename T, int Dim>
  - `T simplex_volume` (const Array< T, Dim > \*a)
- template<typename T >
  - `T area` (const Array< T, 2 > \*c)
- template<typename T >
  - `T area` (const Array< T, 3 > \*c)
- template<typename T >
  - `T volume` (const Array< T, 3 > \*c)
    - *Computes the volume of a 3D simplex (embedded i 3D space).*
- template<typename T >
  - `T signed_area` (const Array< T, 3 > \*c, const Array< T, 3 > &normal)
- `BernsteinMulti operator*` (const BernsteinMulti &m1, const BernsteinMulti &m2)
  - *Multiplication of two polynomials.*
- `BernsteinMulti operator*` (const BernsteinMulti &m1, double c)
  - *Multiplication of a polynomial with a scalar.*
- `BernsteinMulti operator*` (double c, const BernsteinMulti &m1)
  - *Multiplication of a scalar with a polynomial.*
- `BernsteinMulti operator+` (const BernsteinMulti &m1, const BernsteinMulti &m2)
  - *Addition of two polynomials.*
- `BernsteinMulti operator+` (const BernsteinMulti &m, double c)
  - *Addition of a polynomial with a scalar.*
- `BernsteinMulti operator+` (double c, const BernsteinMulti &m)
  - *Addition of a scalar with a polynomial.*
- `BernsteinMulti operator-` (const BernsteinMulti &m1, const BernsteinMulti &m2)
  - *Subtraction of two polynomials.*
- `BernsteinMulti operator-` (const BernsteinMulti &m, double c)
  - *Subtraction of a scalar from a polynomial.*
- `BernsteinMulti operator-` (double c, const BernsteinMulti &m)
  - *Subtraction of a polynomial from a scalar.*
- `BernsteinMulti operator/` (const BernsteinMulti &m, double c)
  - *Division of a polynomial with a scalar.*
- `BernsteinPoly operator*` (const BernsteinPoly &p1, const BernsteinPoly &p2)
  - *Multiplication of two polynomials.*
- `BernsteinPoly operator*` (const BernsteinPoly &p, double c)
  - *Multiplication of a polynomial with a scalar.*

- [BernsteinPoly operator\\*](#) (double c, const BernsteinPoly &p)  
*Multiplication of a scalar with a polynomial.*
- [BernsteinPoly operator+](#) (const BernsteinPoly &p1, const BernsteinPoly &p2)  
*Addition of two polynomials.*
- [BernsteinPoly operator+](#) (double c, const BernsteinPoly &p)  
*Addition of a scalar with a polynomial.*
- [BernsteinPoly operator+](#) (const BernsteinPoly &p, double c)  
*Addition of a polynomial with a scalar.*
- [BernsteinPoly operator-](#) (const BernsteinPoly &p1, const BernsteinPoly &p2)  
*Subtraction of two polynomials.*
- [BernsteinPoly operator-](#) (double c, const BernsteinPoly &p)  
*Subtraction of a polynomial from a scalar.*
- [BernsteinPoly operator-](#) (const BernsteinPoly &p, double c)  
*Subtraction of a scalar from a polynomial.*
- [BernsteinPoly operator/](#) (const BernsteinPoly &p, double c)  
*Division of a polynomial with a scalar.*
- [BernsteinTetrahedralPoly operator\\*](#) (const BernsteinTetrahedralPoly &p1, const BernsteinTetrahedralPoly &p2)  
*Multiplication of two polynomials.*
- [BernsteinTetrahedralPoly operator\\*](#) (const BernsteinTetrahedralPoly &p, double c)  
*Multiplication of a polynomial with a scalar.*
- [BernsteinTetrahedralPoly operator\\*](#) (double c, const BernsteinTetrahedralPoly &p)  
*Multiplication of a scalar with a polynomial.*
- [BernsteinTetrahedralPoly operator+](#) (const BernsteinTetrahedralPoly &p1, const BernsteinTetrahedralPoly &p2)  
*Addition of two polynomials.*
- [BernsteinTetrahedralPoly operator+](#) (double c, const BernsteinTetrahedralPoly &p)  
*Addition of a scalar with a polynomial.*
- [BernsteinTetrahedralPoly operator+](#) (const BernsteinTetrahedralPoly &p, double c)  
*Addition of a polynomial with a scalar.*
- [BernsteinTetrahedralPoly operator-](#) (const BernsteinTetrahedralPoly &p1, const BernsteinTetrahedralPoly &p2)  
*Subtraction of two polynomials.*
- [BernsteinTetrahedralPoly operator-](#) (double c, const BernsteinTetrahedralPoly &p)  
*Subtraction of a polynomial from a scalar.*
- [BernsteinTetrahedralPoly operator-](#) (const BernsteinTetrahedralPoly &p, double c)  
*Subtraction of a scalar from a polynomial.*
- [BernsteinTetrahedralPoly operator/](#) (const BernsteinTetrahedralPoly &p, double c)  
*Division of a polynomial with a scalar.*
- `std::istream & operator>>` (std::istream &is, Go::BernsteinTetrahedralPoly &p)  
*Read [BernsteinTetrahedralPoly](#) from input stream.*
- `std::ostream & operator<<` (std::ostream &os, const Go::BernsteinTetrahedralPoly &p)  
*Write [BernsteinTetrahedralPoly](#) to output stream.*
- [BernsteinTriangularPoly operator\\*](#) (const BernsteinTriangularPoly &p1, const BernsteinTriangularPoly &p2)  
*Multiplication of two polynomials.*
- [BernsteinTriangularPoly operator\\*](#) (const BernsteinTriangularPoly &p, double c)  
*Multiplication of a polynomial with a scalar.*
- [BernsteinTriangularPoly operator\\*](#) (double c, const BernsteinTriangularPoly &p)  
*Multiplication of a scalar with a polynomial.*
- [BernsteinTriangularPoly operator+](#) (const BernsteinTriangularPoly &p1, const BernsteinTriangularPoly &p2)  
*Addition of two polynomials.*

- [BernsteinTriangularPoly operator+](#) (double c, const BernsteinTriangularPoly &p)  
*Addition of a scalar with a polynomial.*
- [BernsteinTriangularPoly operator+](#) (const BernsteinTriangularPoly &p, double c)  
*Addition of a polynomial with a scalar.*
- [BernsteinTriangularPoly operator-](#) (const BernsteinTriangularPoly &p1, const BernsteinTriangularPoly &p2)  
*Subtraction of two polynomials.*
- [BernsteinTriangularPoly operator-](#) (double c, const BernsteinTriangularPoly &p)  
*Subtraction of a polynomial from a scalar.*
- [BernsteinTriangularPoly operator-](#) (const BernsteinTriangularPoly &p, double c)  
*Subtraction of a scalar from a polynomial.*
- [BernsteinTriangularPoly operator/](#) (const BernsteinTriangularPoly &p, double c)  
*Division of a polynomial with a scalar.*
- `std::istream & operator>>` (std::istream &is, [Go::BernsteinTriangularPoly](#) &p)  
*Read [BernsteinTriangularPoly](#) from input stream.*
- `std::ostream & operator<<` (std::ostream &os, const [Go::BernsteinTriangularPoly](#) &p)  
*Write [BernsteinTriangularPoly](#) to output stream.*
- void [spline\\_to\\_bernstein](#) (const [SplineCurve](#) &seg, int dd, [BernsteinPoly](#) &bp)
- void [spline\\_to\\_bernstein](#) (const [SplineSurface](#) &pat, int dd, [BernsteinMulti](#) &bm)
- void [spline\\_to\\_bernstein](#) (const [SplineCurve](#) &seg, std::vector< [BernsteinPoly](#) > &seg\_bp)
- void [spline\\_to\\_bernstein](#) (const [SplineSurface](#) &pat, std::vector< [BernsteinMulti](#) > &pat\_bm)
- template<int Ndim>  
void [splineToBernstein](#) (const [SplineCurve](#) &segment, [Array](#)< [BernsteinPoly](#), Ndim > &curve\_bp)
- template<int Ndim>  
void [splineToBernstein](#) (const [SplineSurface](#) &patch, [Array](#)< [BernsteinMulti](#), Ndim > &surface\_bm)
- template<int Ndim>  
void [bernsteinToSpline](#) (const [Array](#)< [BernsteinPoly](#), Ndim > &curve\_bp, bool rational, [SplineCurve](#) &segment)
- template<int Ndim>  
void [bernsteinToSpline](#) (const [Array](#)< [BernsteinMulti](#), Ndim > &surface\_bm, bool rational, [SplineSurface](#) &patch)
- void [create\\_bary\\_coord\\_system2D](#) (const [SplineCurve](#) &curve, [BaryCoordSystem2D](#) &bc)  
*Creates a barycentric coordinate system from a given spline curve.*
- void [create\\_bary\\_coord\\_system3D](#) (const [SplineCurve](#) &curve, [BaryCoordSystem3D](#) &bc)  
*Creates a barycentric coordinate system from a given 3D spline curve.*
- void [create\\_bary\\_coord\\_system3D](#) (const [SplineSurface](#) &surface, [BaryCoordSystem3D](#) &bc)  
*Creates a barycentric coordinate system from a given spline surface.*
- void [create\\_bary\\_coord\\_system3D](#) (const [PointCloud3D](#) &cloud, [BaryCoordSystem3D](#) &bc)  
*Creates a barycentric coordinate system from a point cloud.*
- void [create\\_bary\\_coord\\_system3D](#) (const [BoundingBox](#) &box, [BaryCoordSystem3D](#) &bc)  
*Creates a barycentric coordinate system from a bounding box.*
- void [cart\\_to\\_bary](#) (const [SplineCurve](#) &cv, const [BaryCoordSystem2D](#) &bc, [SplineCurve](#) &cv\_bc)
- void [cart\\_to\\_bary](#) (const [SplineCurve](#) &cv, const [BaryCoordSystem3D](#) &bc, [SplineCurve](#) &cv\_bc)
- void [cart\\_to\\_bary](#) (const [SplineSurface](#) &sf, const [BaryCoordSystem3D](#) &bc, [SplineSurface](#) &sf\_bc)
- void [cart\\_to\\_bary](#) (const [PointCloud3D](#) &cloud, const [BaryCoordSystem3D](#) &bc, [PointCloud4D](#) &cloud\_bc)
- void [make\\_matrix](#) (const [SplineCurve](#) &curve, int deg, std::vector< std::vector< double > > &mat)  
*Make the matrix D.*
- void [make\\_matrix](#) (const [SplineSurface](#) &surf, int deg, std::vector< std::vector< double > > &mat)  
*Make the matrix D.*
- void [make\\_matrix](#) (const [PointCloud4D](#) &cloud, int deg, std::vector< std::vector< double > > &mat)  
*Make the matrix D.*
- void [make\\_implicit\\_svd](#) (std::vector< std::vector< double > > &mat, std::vector< double > &b, double &sigma\_min)



- void `make_implicit_gauss` (std::vector< std::vector< double > > &mat, std::vector< double > &b)
- int `checkCoincide` (ParamCurveInt \*curve, double start, double end, shared\_ptr< GeoTol > tol, ParamCurveInt \*other, double other\_start, double other\_end)
- int `checkCoincide` (ParamCurveInt \*curve, double start, double end, ParamSurfaceInt \*surf, Point su\_start, Point su\_end, shared\_ptr< GeoTol > tol)
- int `checkCoincide` (ParamCurveInt \*curve, double start, double end, SplineSurfaceInt \*surf, const Point &su\_start, const Point &su\_end, shared\_ptr< GeoTol > tol)
- int `checkCoincide` (Param1FunctionInt \*func1, double start, double end, Param0FunctionInt \*C, shared\_ptr< GeoTol > tol)
- int `checkCoincide` (ParamSurfaceInt \*surf1, ParamSurfaceInt \*surf2, std::vector< double > &par\_loop, const shared\_ptr< GeoTol > tol)
  - by the parameter values of the intersection points representing it
- int `stepLength` (const std::vector< Point > &ft, const std::vector< Point > &gs, double delta, bool forward, double &delta\_t)
- double `stepLength` (const std::vector< Point > &ft, const std::vector< Point > &gs, bool forward, std::vector< Point > &cvder, double min\_step, double max\_step, double aepsge)
- int `evalProjectedCurve` (const std::vector< Point > &ft, const std::vector< Point > &gs, std::vector< Point > &res)
- void `evalDistCurve` (const std::vector< Point > &marching\_curve, const std::vector< Point > &other\_curve, std::vector< Point > &dist\_curve)
- bool `determineCoincidenceRegion` (const ParamObjectInt \*obj1, const ParamObjectInt \*obj2, shared\_ptr< const GeoTol > tol, const double \*current\_params, int dir, bool forward, double &last\_param\_val\_inside, double &first\_param\_val\_outside)
- bool `determineCoincidenceRegion` (const ParamFunctionInt \*obj\_1d, double C, shared\_ptr< const GeoTol > tol, const double \*current\_params, int dir, bool forward, double &last\_param\_val\_inside, double &first\_param\_val\_outside)
- template<class iterator >
  - shared\_ptr< IntersectionCurve > `constructIntersectionCurve` (const iterator begin, const iterator end)
- void `intersectCurves` (shared\_ptr< ParamCurve > crv1, shared\_ptr< ParamCurve > crv2, double tol, std::vector< std::pair< double, double > > &intersection\_points)
  - Intersection between two parametric curves.
- void `debug_write_line` (const Point &p1, const Point &p2, const char \*fname)
- void `debug_write_point` (const Point &p1, const char \*fname)
- bool `no_parent` (const shared\_ptr< IntersectionPoint > &p)
- IntersectionPoint \* `flip_intersecting_objects` (IntersectionPoint \*p)
- bool `link_is_iso_in` (shared\_ptr< IntersectionLink > link, int dir)
- bool `link_is_iso` (shared\_ptr< IntersectionLink > link)
- bool `link_is_iso_in_other_than` (shared\_ptr< IntersectionLink > link, int dir)
- template<class T1 , class T2 >
  - bool `compare_first` (const std::pair< T1, T2 > &x, const std::pair< T1, T2 > &y)
- void `add_reachables_from` (const IntersectionPoint \*pt, const IntersectionPoint \*last\_pt, std::set< IntersectionPoint \* > &result)
- void `estimate_seed_by_interpolation` (const IntersectionPoint \*p1, const IntersectionPoint \*p2, int fixed\_dir, double fixed\_value, double \*result)
- void `determine_seed` (double \*par, int par\_start, int par\_end, const Point &pt, const IntersectionPoint \*const ip1, const IntersectionPoint \*const ip2)
- int `find_point_in` (IntersectionPoint \*p, const std::vector< shared\_ptr< IntersectionPoint > > &vec)
- shared\_ptr< const CurveOnSurface > `make_curve_on_surface` (shared\_ptr< const ParamSurface > surf, double u\_start, double v\_start, double u\_end, double v\_end, double p\_start, double p\_end)
- void `extract_chains` (const std::vector< shared\_ptr< IntersectionPoint > > pts, std::vector< std::vector< shared\_ptr< IntersectionPoint > > > &chains)
- double `benchmarkSfRefinement` (LRSplineSurface &lr\_sf, const std::vector< LRSplineSurface::Refinement2D > &refs, bool single\_insertions=false)
- std::ostream & `operator<<` (std::ostream &os, const LRBSpline2D &b)
- std::istream & `operator>>` (std::istream &is, LRBSpline2D &b)
- void `writePostscriptMesh` (Go::LRSplineSurface &lr\_spline\_sf, std::ostream &out, const bool close=true)



- [Direction2D flip](#) ([Direction2D](#) d)
- `std::ostream & operator<<` (`std::ostream &os`, `const GPos &g`)
- `std::istream & operator>>` (`std::istream &is`, `GPos &g`)
- `std::ostream & operator<<` (`std::ostream &os`, `const Mesh2D &m`)
- `std::istream & operator>>` (`std::istream &is`, `Mesh2D &m`)

## Variables

- `const int MAJOR_VERSION = 1`
- `const int MINOR_VERSION = 0`

### 28.1.1 Detailed Description

The [Go](#) namespace is the common namespace for all [GoTools](#) modules.

### 28.1.2 Typedef Documentation

#### 28.1.2.1 `typedef BaryCoordSystem<2> Go::BaryCoordSystem2D`

Definition at line 183 of file `BaryCoordSystem.h`.

#### 28.1.2.2 `typedef BaryCoordSystem<3> Go::BaryCoordSystem3D`

Definition at line 184 of file `BaryCoordSystem.h`.

#### 28.1.2.3 `typedef std::pair<std::pair<double, double>, std::pair<ftFaceBase*, int> > Go::BoundaryPiece`

Definition at line 45 of file `ftSurfaceSet.h`.

#### 28.1.2.4 `typedef std::map<LRBSpline2D*, size_t> Go::BsplineIndexMap`

This class modifies a LR spline surface with respect to conditions on smoothness, data points and boundary conditions.

Definition at line 70 of file `LRSurfSmoothLS.h`.

#### 28.1.2.5 `typedef std::vector<ftSearchNode>::iterator Go::GraphIter`

Definition at line 29 of file `ftPlanarGraph.h`.

#### 28.1.2.6 `typedef PointCloud<3> Go::PointCloud3D`

Definition at line 261 of file `PointCloud.h`.

**28.1.2.7 typedef PointCloud<4> Go::PointCloud4D**

Definition at line 262 of file PointCloud.h.

**28.1.2.8 typedef ftSamplePoint\* Go::PointIter**

Definition at line 63 of file ftPointSet.h.

**28.1.2.9 typedef std::list<shared\_ptr<ftSamplePoint> > Go::PointList**

Definition at line 59 of file ftPointSet.h.

**28.1.2.10 typedef struct Go::sideConstraint Go::sideConstraint**

Struct defining linear side constraints between control points in a surface. The elements of `factor_` correspond to a linear combination of control points on the left side of the equation, whilst `constant_term_` denotes the right side of the equation.

**28.1.2.11 typedef struct Go::sideConstraintSet Go::sideConstraintSet**

Struct defining linear side constraints between control points in a set of surfaces. The elements of `factor_` correspond to a linear combination of control points on the left side of the equation, whilst `constant_term_` denotes the right side of the equation.

**28.1.2.12 typedef Element2D Go::simpleElement**

Definition at line 66 of file LRSplineEvalGrid.h.

**28.1.2.13 typedef Array<double, 2> Go::Vector2D**

Typedef for ease of use in frequently used case.

Definition at line 431 of file Array.h.

**28.1.2.14 typedef Array<double, 3> Go::Vector3D**

Typedef for ease of use in frequently used case.

Definition at line 433 of file Array.h.

**28.1.2.15 typedef Array<double, 4> Go::Vector4D**

Typedef for ease of use in frequently used case.

Definition at line 435 of file Array.h.

### 28.1.3 Enumeration Type Documentation

#### 28.1.3.1 anonymous enum

Enumeration of various pretopologies that can be associated with intersections.

Enumerator

***pretop\_UNDEF***  
***pretop\_IN***  
***pretop\_OUT***  
***pretop\_ON***  
***pretop\_AT***

Definition at line 56 of file GoIntersections.h.

#### 28.1.3.2 enum Go::AlgorithmChoice

This enumeration denotes whether the 'closestPtSurfSurfPlane' function should base itself upon a geometrical algorithm (marching on the surfaces) or a pure functional one (constrained minimization of a cost function). The default has been set to FUNCTIONAL, as there are flaws in the GEOMETRICAL one when it comes to close-to-degenerate cases.

Enumerator

***GEOMETRICAL***  
***FUNCTIONAL***

Definition at line 56 of file ClosestPoint.h.

#### 28.1.3.3 enum Go::BdConditionType

Enumerator

***UNKNOWN***  
***ZERO\_DIRICHLET***  
***CONSTANT\_DIRICHLET***  
***DIRICHLET***  
***ZERO\_NEUMANN***  
***NEUMANN***  
***SYMMETRY***

Definition at line 48 of file BdConditionType.h.

#### 28.1.3.4 enum Go::ClassType

All concrete classes that inherit [GeomObject](#) should have a matching enum in ClassType. It is necessary to have a central location for the type identification numbers, so that two classes in different modules (for example) does not try to use the same number. At the same time, it is something we'd like to avoid, because it forces you to update this file every time you add a new GeomObject- inheriting class.

If you need to add a new type, you may also need to add a "Registrar", see the '[GoTools](#)' class. If the new class is part of 'geometry' in 'gotools-core', do this in the init() function of the [GoTools](#) class itself. If it's outside the core, then write another class with its own static init() function containing the [Registrar](#).

Also, if you need the name of that added type, i.e. a ClassType-to-string conversion, then you should add that as well. See [GoTools::className\(\)](#).

#### Enumerator

- Class\_Unknown***
- Class\_SplineCurve***
- Class\_CurveOnSurface***
- Class\_CurveOnVolume***
- Class\_Line***
- Class\_Circle***
- Class\_Ellipse***
- Class\_BoundedCurve***
- Class\_Hyperbola***
- Class\_Parabola***
- Class\_SplineSurface***
- Class\_BoundedSurface***
- Class\_SurfaceOnVolume***
- Class\_GoBaryPolSurface***
- Class\_GoHBSplineParamSurface***
- Class\_CompositeSurface***
- Class\_Plane***
- Class\_Cylinder***
- Class\_SurfaceOfLinearExtrusion***
- Class\_Sphere***
- Class\_Cone***
- Class\_Torus***
- Class\_SurfaceOfRevolution***
- Class\_Disc***
- Class\_LRSplineSurface***
- Class\_TSplineSurface***
- Class\_Go3dsObject***
- Class\_GoHeTriang***
- Class\_GoSdTriang***
- Class\_GoQuadMesh***
- Class\_GoHybridMesh***
- Class\_ParamTriang***

*Class\_GoVrmlGeometry*  
*Class\_PointCloud*  
*Class\_LineCloud*  
*Class\_GoTriangleSets*  
*Class\_RectGrid*  
*Class\_SplineVolume*  
*Class\_BoundedVolume*  
*Class\_Parallelepiped*  
*Class\_SphereVolume*  
*Class\_CylinderVolume*  
*Class\_ConeVolume*  
*Class\_TorusVolume*  
*Class\_LRSplineVolume*

Definition at line 63 of file ClassType.h.

#### 28.1.3.5 enum Go::closestPointLevel

Enumerator

*LOCAL\_SEARCH*  
*SEMI\_LOCAL\_SEARCH*  
*GLOBAL\_SEARCH*

Definition at line 56 of file CompositeModel.h.

#### 28.1.3.6 enum Go::CoefStatus

Enumerator

*CoefFree*  
*CoefKnown*  
*CoefAvoid*  
*CoefOther*

Definition at line 56 of file SmoothVolume.h.

#### 28.1.3.7 enum Go::Direction2D

Enumerator

*XFIXED*  
*YFIXED*

Definition at line 46 of file Direction2D.h.

## 28.1.3.8 enum Go::EstimateDirection

Enumerator

**FORWARDS**  
**BACKWARDS**

Definition at line 57 of file IntersectionCurve.h.

## 28.1.3.9 enum Go::EvalKind

Enumerator

**SPACECURVE**  
**PARAMCURVE\_1**  
**PARAMCURVE\_2**

Definition at line 55 of file IntersectionCurve.h.

## 28.1.3.10 enum Go::FileFormat

Enumerator

**go**  
**disp**  
**IGES**

Definition at line 74 of file IGESconverter.h.

## 28.1.3.11 enum Go::ftCurveType

Indicates type of curve described by a [ftCurve](#).

Enumerator

**CURVE\_NOTYPE** Untyped curve.  
**CURVE\_INTERSECTION** Curve mostly in interior of surfaces.  
**CURVE\_EDGE** Curve lying on the edge of surfaces.  
**CURVE\_KINK** Between surfaces joined by kink.  
**CURVE\_CORNER** Between surfaces meeting at a corner line.  
**CURVE\_GAP** Between surfaces joined by gap.  
**CURVE\_SINGULAR** Curve mostly in interior of one surface.

Definition at line 57 of file ftCurve.h.

## 28.1.3.12 enum Go::ftmessages

Enumerator

*FT\_NOT\_IMPLEMENTED*  
*FT\_NO\_DATA*  
*FT\_BAD\_INPUT\_FILE*  
*FT\_ERROR\_IN\_READ\_IGES*  
*FT\_DISCONTINUITY*  
*FT\_UNEXPECTED\_DATA\_TYPE*  
*FT\_NON\_4\_SIDED\_SURF*  
*FT\_WRONG\_NO\_OF\_BOUNDARIES*  
*FT\_INSUFFICIENT\_MESH\_CURVES*  
*FT\_ERROR\_IN\_TOPOLOGY\_ANALYSIS*  
*FT\_TOPOLOGY\_PROBLEM*  
*FT\_ERROR\_IN\_SURFACE\_CREATION*  
*FT\_ERROR\_IN\_PARAMETERIZE*  
*FT\_ERROR\_IN\_SURFACE\_TRIMMING*  
*FT\_NOT\_SUPPORTED*  
*FT\_NOT\_SPLINE\_SURF*  
*FT\_ERROR\_IN\_SURFACE\_GRIDDING*  
*FT\_NEIGHBOUR\_GRID\_SIZE\_DIFFER*  
*FT\_OK*  
*FT\_APPROX\_ERROR\_TOO\_LARGE*  
*FT\_COULD\_NOT\_REFINE\_SURFACE*  
*FT\_FEATURE\_ERROR\_TOO\_LARGE*  
*FT\_DEGENERATE\_PATCH\_CREATED*  
*FT\_SURFACE\_ALREADY\_CREATED*  
*FT\_DEGENERATE\_SURFACE*  
*FT\_UNEXPECTED\_INPUT\_OBJECT\_IGNORED*  
*FT\_FACES\_OVERLAP*  
*FT\_COULD\_NOT\_SMOOTH\_SURFACE*  
*FT\_COULD\_NOT\_SMOOTH\_SURFACESET*  
*FT\_NO\_SMOOTHING*  
*FT\_FAILED\_CREATING\_GRAPH*  
*FT\_GRID\_NOT\_CREATED*  
*FT\_SURFACE\_NOT\_MODIFIED*

Definition at line 51 of file ftMessage.h.

## 28.1.3.13 enum Go::ftTangPriority

Whether the group of objects (surfaces) are a master, a slave or not specified. Related to tangent plane continuity between groups of surfaces

Enumerator

*ftNoType*  
*ftMaster*  
*ftSlave*

Definition at line 49 of file ftTangPriority.h.

## 28.1.3.14 enum Go::IGESSection

Enumerator

**S**  
**G**  
**D**  
**P**  
**T**  
**E**

Definition at line 75 of file IGESconverter.h.

## 28.1.3.15 enum Go::IntersectionType

Enumerator

**No\_Type**  
**SurfaceModel\_SurfaceModel**  
**SurfaceModel\_Plane**  
**SurfaceModel\_Line**  
**CompositeCurve\_CompositeCurve**  
**CompositeCurve\_Plane**  
**CompositeCurve\_Line**

Definition at line 54 of file IntResultsModel.h.

## 28.1.3.16 enum Go::IntPtClassification

Enumeration used to classify the direction of an intersection curve passing through an intersection point with respect to surface boundaries

Enumerator

**DIR\_UNDEF** Not defined.  
**DIR\_IN** Tangent of the intersection curve pointing into the domain.  
**DIR\_OUT** Tangent of the intersection curve pointing out from the domain.  
**DIR\_PARALLEL** Tangent of the intersection curve parallel to the domain.  
**DIR\_PERPENDICULAR** Tangent of the intersection curve perpendicular to the domain.  
**DIR\_HIGHLY\_SINGULAR** No defined tangent.  
**DIR\_TOUCH** Tangent touching the domain at a corner.

Definition at line 54 of file IntersectionPointUtils.h.



## 28.1.3.17 enum Go::IntPtLocation

Enumeration used to classify the location of an intersection point in the parameter domain of the object.

Enumerator

**LOC\_OUTSIDE\_BOTH** [Point](#) is outside domain for both objects.  
**LOC\_INSIDE\_BOTH** [Point](#) is inside domains (inner point) for both objects.  
**LOC\_EDGE\_ONE** [Point](#) is on the edge of at least one object.  
**LOC\_CORNER\_ONE** [Point](#) is on the corner of at least one object.  
**LOC\_CORNER\_BOTH** [Point](#) is on the corner of both objects.  
**LOC\_EDGE\_BOTH** [Point](#) is on the edge of both objects.

Definition at line 77 of file IntersectionPointUtils.h.

## 28.1.3.18 enum Go::IteratorType

Enumerator

**Iterator\_parametric**  
**Iterator\_geometric**

Definition at line 54 of file ParamSurface.h.

## 28.1.3.19 enum Go::LinkType

A classification of [IntersectionLink](#) objects. The classification is strictly based on how it was made within the algorithm.

Enumerator

**LINK\_UNDEFINED** 0  
**SIMPLE\_CONE** 1  
**SIMPLE\_MONOTONE** 2  
**SIMPLE\_IMPLICIT** 3  
**SIMPLE\_TWO\_POINTS** 4  
**COINCIDENCE\_CVPT** 5  
**COINCIDENCE\_CVCV** 6  
**COINCIDENCE\_SFCV** 7  
**COINCIDENCE\_SFPT** 8  
**MICRO\_CVCV** 9  
**MICRO\_SFPT** 10  
**MICRO\_SFCV** 11  
**MICRO\_SFSF** 12  
**MICRO\_PAR1FUNC** 13  
**MICRO\_PAR2FUNC** 14  
**LINEAR\_CVCV** 15

**LINEAR\_SFCV** 16  
**LINEAR\_SFSF** 17  
**MERGED\_UNDEFINED** 18  
**MERGED\_SIMPLE\_CONE** 19  
**MERGED\_SIMPLE\_MONOTONE** 20  
**MERGED\_SIMPLE\_CONE\_MONOTONE** 21  
**MERGED\_COINCIDENCE\_SFCV\_SFCV** 22  
**DEG\_TRIANGLE** 23  
**POST\_ITERATE** 24  
**BRANCH\_CONNECTION** 25  
**COMPLEX\_SFSF** 26  
**SPLIT\_LINK** 27  
**REPAIRED\_MISSING\_LINK** 28  
**INSIDE\_OUTSIDE\_SINGULARITY\_BOX** 29

Definition at line 50 of file LinkType.h.

#### 28.1.3.20 enum Go::PointSequenceType

Type of point.

Enumerator

**PSTPoint** Each grid position holds a point.  
**PSTPointTangent** Each grid position holds a point, then a tangent.  
**PSTScattered** Regard this an unordered point cloud.

Definition at line 53 of file PointSequence.h.

#### 28.1.3.21 enum Go::RegistrationReturnType

Enumerator for registration error reasons  
**RegistrationOK** No error found  
**TooFewPoints** Less than three points are given  
**PointSetSizeDiff** The two point sets given have different length  
**AreaTooSmall** Failed to find a tripple of points that are far enough from being almost colinear  
**SolveFailed** Solving the linear system in the Newtons approach method failed. Error code is stored in registrationData.solve\_code

Enumerator

**RegistrationOK**  
**TooFewPoints**  
**PointSetSizeDiff**  
**AreaTooSmall**  
**SolveFailed**

Definition at line 57 of file RegistrationUtils.h.

### 28.1.3.22 enum Go::SingularityClassification

This enum classifies the type of singularity.

Enumerator

***NO\_SING***  
***INIT\_SING***  
***DIVIDED\_SING***  
***KEEP\_SING***

Definition at line 49 of file SingularityClassification.h.

### 28.1.3.23 enum Go::SingularityType

A classification of an [IntersectionPoint](#), based on the nature of the intersection on which it lies.

Enumerator

***ORDINARY\_POINT*** [IntersectionPoint](#) lies on normal, transversal intersection.  
***TANGENTIAL\_POINT*** The two intersecting objects are tangent to each other at the location of this [IntersectionPoint](#).  
***ISOLATED\_POINT*** This is an isolated [IntersectionPoint](#) (does not lie on an [IntersectionCurve](#) or on a partial coincidence area).  
***BRANCH\_POINT*** This is an [IntersectionPoint](#) where flere interection curves meet.  
***HIGHER\_ORDER\_POINT*** The two objects intersecting at this point have same tangent plane AND curvature.

Definition at line 46 of file SingularityType.h.

### 28.1.3.24 enum Go::SubdivisionClassification

This enum classifies the type of subdivision.

Enumerator

***CANNOT\_SUBDIVIDE***  
***DIVIDE\_DEG***  
***DIVIDE\_CRITICAL***  
***DIVIDE\_HIGH\_SING***  
***DIVIDE\_SING***  
***DIVIDE\_KNOT***  
***DIVIDE\_INT***  
***DIVIDE\_PAR***

Definition at line 49 of file SubdivisionClassification.h.

## 28.1.3.25 enum Go::TangentDomain

Enumerator

***GEOM***  
***PARAM1***  
***PARAM2***

Definition at line 56 of file IntersectionCurve.h.

## 28.1.3.26 enum Go::testSuite

Enumerator

***IDENTICAL\_VERTICES***  
***IDENTICAL\_EDGES***  
***EMBEDDED\_EDGES***  
***IDENTICAL\_FACES***  
***EMBEDDED\_FACES***  
***MINI\_CURVE***  
***MINI\_SURFACE***  
***MINI\_EDGE***  
***MINI\_FACE***  
***SLIVER\_FACE***  
***NARROW\_REGION***  
***DEGEN\_SRF\_BD***  
***DEGEN\_SRF\_CORNER***  
***VANISHING\_TANGENT***  
***VANISHING\_NORMAL***  
***EDGE\_VERTEX\_DISTANCE***  
***FACE\_VERTEX\_DISTANCE***  
***FACE\_EDGE\_DISTANCE***  
***EDGE\_POSITION\_DISCONT***  
***EDGE\_TANGENTIAL\_DISCONT***  
***FACE\_POSITION\_DISCONT***  
***FACE\_TANGENTIAL\_DISCONT***  
***LOOP\_CONSISTENCY***  
***LOOP\_ORIENTATION***  
***FACE\_ORIENTATION***  
***CV\_G1DISCONT***  
***CV\_C1DISCONT***  
***SF\_G1DISCONT***  
***SF\_C1DISCONT***  
***CV\_CURVATURE\_RADIUS***  
***SF\_CURVATURE\_RADIUS***  
***EDGE\_ACUTE\_ANGLE***  
***FACE\_ACUTE\_ANGLE***  
***LOOP\_INTERSECTION***  
***LOOP\_SELF\_INTERSECTION***  
***INDISTINCT\_KNOTS***

Definition at line 51 of file testSuite.h.

## 28.1.3.27 enum Go::tpJointType

Enumerator

**JOINT\_G1**  $G^1$  continuous (angle diff. less than kink tolerance) Continuous (angle diff. between kink and bend tolerance)  
**JOINT\_KINK** Continuous (angle diff. greater than bend tolerance)  
**JOINT\_G0** Discontinuous (distance between gap and neighbour tol.)  
**JOINT\_GAP** Discontinuous (distance greater than neighbour tolerance)  
**JOINT\_DISC** No joint, this was the last segment.  
**JOINT\_NONE**

Definition at line 46 of file tpJointType.h.

## 28.1.4 Function Documentation

28.1.4.1 void Go::add\_reachables\_from ( const IntersectionPoint \* *pt*, const IntersectionPoint \* *last\_pt*, std::set< IntersectionPoint \* > & *result* )

Definition at line 426 of file IntersectionPoolUtils.h.

28.1.4.2 void Go::addToLinearSystem ( int *pt\_idx*, const std::vector< Point > & *points\_fixed*, const std::vector< Point > & *points\_transform*, bool *allow\_rescaling*, const std::vector< std::vector< double > > & *id*, const Point & *fine\_R*, const Point & *fine\_T*, double *fine\_s*, const std::vector< std::vector< double > > & *m\_rot\_R*, double *s2*, double *R2*, bool *zero\_R*, const std::vector< std::vector< std::vector< double > > > & *lhs\_matrix*, const std::vector< std::vector< double > > & *rhs\_matrix* )

Add the contribution from one single point to the linear system coefficients used during fine registration.

28.1.4.3 template<typename T> T Go::area ( const Array< T, 2 > \* *c* ) [inline]

Computes the area of a 2-dimensional triangle. Input is an array of corner points. Same function also exists for 3-dimensional triangles.

Definition at line 91 of file Volumes.h.

28.1.4.4 template<typename T> T Go::area ( const Array< T, 3 > \* *c* ) [inline]

Computes the area of a 2-dimensional triangle. Input is an array of corner points. Same function also exists for 2-dimensional triangles.

Definition at line 98 of file Volumes.h.

28.1.4.5 double Go::areaTriangle ( const Vector2D & *corner1*, const Vector2D & *corner2*, const Vector2D & *corner3* )

28.1.4.6 template<typename SquareMatrix, typename T> void Go::backwardSubstitution ( const SquareMatrix & *U*, T \* *x*, int *num\_unknowns* )

Using backward substitution to calculate  $x$  on the system  $Ux = b$ , where  $U$  is an upper triangular matrix with unitary diagonal.

## Parameters

|                     |                                                                                                                                                                                          |
|---------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>U</i>            | The upper triangular matrix. The actually used class must support the operation <code>[][]</code> and return 'double'.                                                                   |
| <i>x</i>            | At function invocation, <i>x</i> should point to an array of T, containing the vector 'b'. On successful completion of the function, this array will contain the solution for <i>x</i> . |
| <i>num_unknowns</i> | The system size (number of unknowns).                                                                                                                                                    |

Definition at line 178 of file LUDecomp\_implementation.h.

```
28.1.4.7 template<typename SquareMatrix > void Go::backwardSubstitution (const SquareMatrix & U, std::vector<
double > * x, int num_unknowns)
```

Using backward substitution to calculate *x* on the system  $Ux = b$ , where *U* is an upper triangular matrix with unitary diagonal.

## Parameters

|                     |                                                                                                                                                                                                 |
|---------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>U</i>            | The upper triangular matrix. The actually used class must support the operation <code>[][]</code> and return 'double'.                                                                          |
| <i>x</i>            | At function invocation, <i>x</i> should point to a vector of doubles, containing the vector 'b'. On successful completion of the function, this vector will contain the solution for <i>x</i> . |
| <i>num_unknowns</i> | The system size (number of unknowns).                                                                                                                                                           |

Definition at line 192 of file LUDecomp\_implementation.h.

```
28.1.4.8 double Go::benchmarkSfRefinement (LRSplineSurface & lr_sf, const std::vector<
LRSplineSurface::Refinement2D > & refs, bool single_insertions = false)
```

```
28.1.4.9 template<int Ndim> void Go::bernsteinToSpline (const Array< BernsteinPoly, Ndim > & curve_bp, bool
rational, SplineCurve & segment)
```

Converts an [Array](#) of *Ndim* BernsteinPolys to a [SplineCurve](#) segment. If the wanted [SplineCurve](#) is rational, *Ndim* is equal to  $dim+1$ , where *dim* is the dimension of space. Furthermore, in *curve\_bp*, the weights are included in the space part of the coefficients, i.e. they have the form  $wP1, wP2, \dots, wPdim, w$ .

## Parameters

|                 |                                                        |
|-----------------|--------------------------------------------------------|
| <i>curve_bp</i> | an <a href="#">Array</a> of <i>Ndim</i> BernsteinPolys |
| <i>rational</i> | flag to indicate if the objects are rational           |
| <i>segment</i>  | the resulting spline curve segment                     |

Definition at line 184 of file BernsteinUtils.h.

```
28.1.4.10 template<int Ndim> void Go::bernsteinToSpline (const Array< BernsteinMulti, Ndim > & surface_bm, bool
rational, SplineSurface & patch)
```

Converts an [Array](#) of *Ndim* BernsteinMultis to a [SplineSurface](#) patch. If the wanted [SplineSurface](#) is rational, *Ndim* is equal to  $dim+1$ , where *dim* is the dimension of space. Furthermore, in *surface\_bm*, the weights are included in

the space part of the coefficients, i.e. they have the form  $wP_1, wP_2, \dots, wP_{dim}, w$ .

#### Parameters

|                   |                                                         |
|-------------------|---------------------------------------------------------|
| <i>surface_bm</i> | an <a href="#">Array</a> of <i>Ndim</i> BernsteinMultis |
| <i>rational</i>   | flag to indicate if the objects are rational            |
| <i>patch</i>      | the resulting spline surface patch                      |

Definition at line 220 of file BernsteinUtils.h.

28.1.4.11 `double Go::binom ( int n, int i ) [inline]`

Computes the binomial coefficient:  $n! / (i! (n-i)!)$

Definition at line 50 of file binom.h.

28.1.4.12 `template<class Functor > double Go::brent_minimize ( const Functor & f, double a, double b, double c, double & parmin, const double rel_tolerance = std::sqrt (std::numeric_limits<double>::epsilon()) ) [inline]`

Definition at line 78 of file brent\_minimize.h.

28.1.4.13 `void Go::cart_to_bary ( const SplineCurve & cv, const BaryCoordSystem2D & bc, SplineCurve & cv_bc )`

Creates a new curve with control points in 3 barycentric coordinates from the 2D input curve.

28.1.4.14 `void Go::cart_to_bary ( const SplineCurve & cv, const BaryCoordSystem3D & bc, SplineCurve & cv_bc )`

Creates a new curve with control points in 4 barycentric coordinates from the 3D input curve.

28.1.4.15 `void Go::cart_to_bary ( const SplineSurface & sf, const BaryCoordSystem3D & bc, SplineSurface & sf_bc )`

Creates a new surface with control points in 4 barycentric coordinates from the 3D input surface.

28.1.4.16 `void Go::cart_to_bary ( const PointCloud3D & cloud, const BaryCoordSystem3D & bc, PointCloud4D & cloud_bc )`

Creates a new point cloud in 4 barycentric coordinates from the 3D input cloud.

28.1.4.17 `int Go::checkCoincide ( ParamCurveInt * curve, double start, double end, shared_ptr< GeoTol > tol, ParamCurveInt * other, double other_start, double other_end )`

Check if two curves are coinciding between parameters start and end on this curve, and other\_start and other\_end on the other curve.

28.1.4.18 `int Go::checkCoincide ( ParamCurveInt * curve, double start, double end, ParamSurfaceInt * surf, Point su_start, Point su_end, shared_ptr< GeoTol > tol )`

Check if this curve is coinciding between parameters start and end, with the [ParamSurface](#) between parameters su\_start and su\_end.

28.1.4.19 `int Go::checkCoincide ( ParamCurveInt * curve, double start, double end, SplineSurfaceInt * surf, const Point & su_start, const Point & su_end, shared_ptr< GeoTol > tol )`

Check if this curve is coinciding between parameters start and end, with the [SplineSurface](#) between parameters su\_start and su\_end.

28.1.4.20 `int Go::checkCoincide ( Param1FunctionInt * func1, double start, double end, Param0FunctionInt * C, shared_ptr< GeoTol > tol )`

Check if this function is coinciding between parameters start and end, with the [Point](#).

28.1.4.21 `int Go::checkCoincide ( ParamSurfaceInt * surf1, ParamSurfaceInt * surf2, std::vector< double > & par_loop, const shared_ptr< GeoTol > tol )`

by the parameter values of the intersection points representing it

Check if the two given surfaces coincide within the loop given

28.1.4.22 `std::vector<float> Go::closestDistances ( const std::vector< float > & pts, const shared_ptr< boxStructuring::BoundingBoxStructure > & structure, const std::vector< std::vector< double > > & rotationMatrix, const Point & translation )`

Calculates the distance for each point in a point cloud to a given surface model, after a SO(3)-rotation and translation is applied on the point cloud. pts - The point cloud, of length 3N where N is the number of points, on format p[0][0], p[0][1], p[0][2], p[1][0], ... structure - the preprocessed structure used to improve the calculation speed. This also holds the surface model. rotationMatrix - An orthogonal 3x3 matrix describing the rotation to be applied in the point cloud before starting the calculations translation - A translation vector to be added to the point cloud (after the orthogonal rotation) before starting the calculations returns a vector of length pts.size()/3, holding the distances in the same order as the input points

28.1.4.23 `std::vector<float> Go::closestPointCalculations ( const std::vector< float > & pts, const shared_ptr< boxStructuring::BoundingBoxStructure > & structure, const std::vector< std::vector< double > > & rotationMatrix, const Point & translation, int return_type, int start_idx, int skip, int max_idx, int search_extend = 3, bool m_core = true )`

Calculates the closest points of a point cloud to a surface model, after a SO(3)-rotation and translation is applied on the point cloud. The method uses polygons inside the bounding curves on parameter domains to help determining if parameter pairs are inside the parameter domain. NB! The creation of the polygons is not yet proven to guarantee inside polygons, thus there is a theoretical risk of not getting the closest point in every case. pts - The point cloud, of length 3N where N is the number of points, on format p[0][0], p[0][1], p[0][2], p[1][0], ... structure - the preprocessed structure used to improve the calculation speed. This also holds the surface model. rotationMatrix - An orthogonal 3x3 matrix describing the rotation to be applied in the point cloud before starting the calculations translation - A translation vector to be added to the point cloud (after the orthogonal rotation) before starting the calculations



return\_type - Tell whether the distances (0), signed distances (1) or closest points (2) should be returned start\_idx - Used for defining the subset of the points on which the calculation should be performed, see below skip - Used for defining the subset of the points on which the calculation should be performed, see below max\_idx - Used for defining the subset of the points on which the calculation should be performed. The subset consists of the points with index  $idx = skip + N * step$  such that  $start\_idx \leq idx < max\_idx$ . For calculations on the entire point cloud, use  $start\_idx = 0$ ,  $skip = 1$  and  $max\_idx \geq$  number of points search\_extend - Used to define the number of segments to be added in each direction when defining the parameter subset on which the closest point functions should be performed. Will be removed. m\_core - Whether the calculations should be performed in parallel on multiple cores (only if OPENMP is included) returns A vector of the distances to the closest points for the subset on which the calculations are performed.

28.1.4.24 `std::vector<float> Go::closestPointCalculations ( const std::vector< float > & pts, const shared_ptr< boxStructuring::BoundingBoxStructure > & structure, const std::vector< std::vector< double > > & rotationMatrix, const Point & translation, int return_type )`

Make closest point calculations on the entire set of a point cloud. For return\_type == 0, the distances are returned For return\_type == 1, the signed distances are returned For return\_type == 2, the points are returned

28.1.4.25 `std::vector<float> Go::closestPoints ( const std::vector< float > & pts, const shared_ptr< boxStructuring::BoundingBoxStructure > & structure, const std::vector< std::vector< double > > & rotationMatrix, const Point & translation )`

Calculates the closest point for each point in a point cloud to a given surface model, after a SO(3)-rotation and translation is applied on the point cloud. pts - The point cloud, of length 3N where N is the number of points, on format  $p[0][0]$ ,  $p[0][1]$ ,  $p[0][2]$ ,  $p[1][0]$ , ... structure - the preprocessed structure used to improve the calculation speed. This also holds the surface model. rotationMatrix - An orthogonal 3x3 matrix describing the rotation to be applied in the point cloud before starting the calculations translation - A translation vector to be added to the point cloud (after the orthogonal rotation) before starting the calculations returns a vector of same length as pts, holding the closest point coordinates in the same order as the input points

28.1.4.26 `void Go::closestPointSingleCalculation ( int pt_idx, int start_idx, int skip, const std::vector< float > & inPoints, const std::vector< std::vector< double > > & rotationMatrix, const Point & translation, const shared_ptr< boxStructuring::BoundingBoxStructure > & boxStructure, std::vector< float > & result, std::vector< std::vector< int > > & lastBoxCall, int return_type, int search_extend )`

28.1.4.27 `void Go::closestPtCurves ( SplineCurve * cv1, SplineCurve * cv2, double epsge, double & par1, double & par2, double & dist )`

Compute the closest point between two curves

#### Parameters

|              |                             |
|--------------|-----------------------------|
| <i>cv1</i>   | pointer to the first curve  |
| <i>cv2</i>   | pointer to the second curve |
| <i>epsge</i> | geometrical tolerance       |

#### Return values

|             |                                                    |
|-------------|----------------------------------------------------|
| <i>par1</i> | parameter of the closest point in the first curve  |
| <i>par2</i> | parameter of the closest point in the second curve |

## Return values

|             |                                                  |
|-------------|--------------------------------------------------|
| <i>dist</i> | distance between the curves at the closest point |
|-------------|--------------------------------------------------|

28.1.4.28 `std::vector<float> Go::closestSignedDistances ( const std::vector< float > & pts, const shared_ptr< boxStructuring::BoundingBoxStructure > & structure, const std::vector< std::vector< double > > & rotationMatrix, const Point & translation )`

Calculates the signed distance for each point in a point cloud to a given surface model, after a SO(3)-rotation and translation is applied on the point cloud. For every point the signed distance

- has absolute value equal to the distance between the point and the closest point on the surface model
- is positive if the point is outside the model
- is negative if the point is inside the model  
 pts - The point cloud, of length 3N where N is the number of points, on format p[0][0], p[0][1], p[0][2], p[1][0], ...  
 structure - the preprocessed structure used to improve the calculation speed. This also holds the surface model.  
 rotationMatrix - An orthogonal 3x3 matrix describing the rotation to be applied in the point cloud before starting the calculations  
 translation - A translation vector to be added to the point cloud (after the orthogonal rotation) before starting the calculations  
 returns a vector of length pts.size()/3, holding the signed distances in the same order as the input points

28.1.4.29 `std::vector<float> Go::closestVectorsOld ( const std::vector< float > & inPoints, const shared_ptr< boxStructuring::BoundingBoxStructure > & boxStructure, const std::vector< std::vector< double > > & rotationMatrix, const Point & translation, int return_type, int start_idx, int skip, int max_idx, int search_extend = 3 )`

Calculates the closest points of a point cloud to a surface model, by not using the inside polygons in closestVectors()

28.1.4.30 `template<class T1, class T2> bool Go::compare_first ( const std::pair< T1, T2 > & x, const std::pair< T1, T2 > & y )`

Definition at line 254 of file IntersectionPoolUtils.h.

28.1.4.31 `template<typename Iterator > int Go::compare_seq ( Iterator begin_1, Iterator end_1, Iterator begin_2, Iterator end_2 )`

Definition at line 161 of file checks.h.

28.1.4.32 `template<class PtrToCurveType > double Go::computeLoopGap ( const std::vector< PtrToCurveType > & curves ) [inline]`

Computes the largest gap in the loop specified by the vector of curves.

Definition at line 55 of file CurveLoop.h.

28.1.4.33 `template<class iterator > shared_ptr<IntersectionCurve> Go::constructIntersectionCurve ( const iterator begin, const iterator end )`

Definition at line 601 of file IntersectionCurve.h.

28.1.4.34 `void Go::create_bary_coord_system2D ( const SplineCurve & curve, BaryCoordSystem2D & bc )`

Creates a barycentric coordinate system from a given spline curve.

28.1.4.35 `void Go::create_bary_coord_system3D ( const SplineCurve & curve, BaryCoordSystem3D & bc )`

Creates a barycentric coordinate system from a given 3D spline curve.

28.1.4.36 `void Go::create_bary_coord_system3D ( const SplineSurface & surface, BaryCoordSystem3D & bc )`

Creates a barycentric coordinate system from a given spline surface.

28.1.4.37 `void Go::create_bary_coord_system3D ( const PointCloud3D & cloud, BaryCoordSystem3D & bc )`

Creates a barycentric coordinate system from a point cloud.

28.1.4.38 `void Go::create_bary_coord_system3D ( const BoundingBox & box, BaryCoordSystem3D & bc )`

Creates a barycentric coordinate system from a bounding box.

28.1.4.39 `double Go::curvatureRadius ( const std::vector< Point > & der, std::vector< Point > & unitder )`

Help functions in related to curvature. Given position, first and second derivative of a curve passing through a point, compute the unit tangent, curvature vector and curvature radius of this curve.

28.1.4.40 `SISLCurve GO_API* Go::Curve2SISL ( const SplineCurve & cv, bool copy = true )`

Convert a [SplineCurve](#) to a [SISLCurve](#)

#### Parameters

|             |                                                                                                                                                                                                                                                                         |
|-------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>cv</i>   | the <a href="#">SplineCurve</a> to convert                                                                                                                                                                                                                              |
| <i>copy</i> | if 'true', then the generated <a href="#">SISLCurve</a> will have its own copy of the coefficient information contained in 'cv'. Otherwise, it will share this information with 'cv' (ie. it will only contain a pointer into the corresponding storage array in 'cv'). |

**Returns**

A newly generated [SISLCurve](#) that describes the same curve as 'cv'. The user assumes ownership and is responsible for cleaning up (which means calling the SISL function `freeCurve(...)` on the pointer when it should be destroyed).

**28.1.4.41** `SISLCurve GO_API* Go::Curve2SISL_rat ( const SplineCurve & cv )`

Convert a [SplineCurve](#) to a rational [SISLCurve](#) Arrays are copied

**28.1.4.42** `void Go::debug_write_line ( const Point & p1, const Point & p2, const char * fname )`

Definition at line 70 of file `IntersectionPoolUtils.h`.

**28.1.4.43** `void Go::debug_write_point ( const Point & p1, const char * fname )`

Definition at line 86 of file `IntersectionPoolUtils.h`.

**28.1.4.44** `bool Go::degenerate_triangle ( const Vector2D & c1, const Vector2D & c2, const Vector2D & c3 )`

**28.1.4.45** `template<typename T> T Go::determinantOf ( const Array< T, 2 > * a ) [inline]`

Calculates the determinant of a 2 x 2 matrix, represented in memory as a sequence of 2 [Array](#) s of length 2. Same function also exists for 3 x 3 matrices.

Definition at line 55 of file `Volumes.h`.

**28.1.4.46** `template<typename T> T Go::determinantOf ( const Array< T, 3 > * a ) [inline]`

Calculates the determinant of a 3 x 3 matrix, represented in memory as a sequence of 3 [Array](#) s of length 3. Same function also exists for 2 x 2 matrices.

Definition at line 65 of file `Volumes.h`.

**28.1.4.47** `void Go::determine_seed ( double * par, int par_start, int par_end, const Point & pt, const IntersectionPoint *const ip1, const IntersectionPoint *const ip2 )`

Definition at line 468 of file `IntersectionPoolUtils.h`.

- 28.1.4.48 `bool Go::determineCoincidenceRegion ( const ParamObjectInt * obj1, const ParamObjectInt * obj2, shared_ptr< const GeoTol > tol, const double * current_params, int dir, bool forward, double & last_param_val_inside, double & first_param_val_outside )`
- 28.1.4.49 `bool Go::determineCoincidenceRegion ( const ParamFunctionInt * obj_1d, double C, shared_ptr< const GeoTol > tol, const double * current_params, int dir, bool forward, double & last_param_val_inside, double & first_param_val_outside )`
- 28.1.4.50 `void Go::estimate_seed_by_interpolation ( const IntersectionPoint * p1, const IntersectionPoint * p2, int fixed_dir, double fixed_value, double * result )`

Definition at line 444 of file IntersectionPoolUtils.h.

- 28.1.4.51 `void Go::evalDistCurve ( const std::vector< Point > & marching_curve, const std::vector< Point > & other_curve, std::vector< Point > & dist_curve )`
- 28.1.4.52 `int Go::evalProjectedCurve ( const std::vector< Point > & ft, const std::vector< Point > & gs, std::vector< Point > & res )`
- 28.1.4.53 `void Go::extract_chains ( const std::vector< shared_ptr< IntersectionPoint > > pts, std::vector< std::vector< shared_ptr< IntersectionPoint > > > & chains )`

Definition at line 537 of file IntersectionPoolUtils.h.

- 28.1.4.54 `int Go::extremalPtSurfSurf ( ParamSurface * psurf1, ParamSurface * psurf2, int constraints[2], double constraints_par[2], double limit[,], double enext[,], double gpos[,], double angle_tol )`

Finds one point in each surface where the two normals are parallel to each other and to the difference vector between the two points. The edge info makes it possible to constrain the search to an ISO-curve in one of or both surfaces. Ported from the sisl-function `shsing_ext`.

METHOD : - Start with a guess value (u,v) in domain of surface 1 (S(u,v))  
 (a) - Find domain value (r,t) of closest point (to S(u,v) in surface 2 (Q(r,t))  
 - If  $vf1(u,v) = \langle Su, Normal(Q) \rangle$  and  $vf2(u,v) = \langle Sv, Normal(Q) \rangle$  is small enough stop  
 ( $\langle, \rangle$  means scalar prod.)  
 - Find du and dv by taylorizing vf1 and vf2.  
 This include finding the derivatives of the closest point function (r(u,v),t(u,v)) with respect to u and v. (called h(u,v) in article, see comments in `shsing_s9dir`)  
 -  $u := u + du$   $v := v + dv$ , goto (a)

REFERENCES : Solutions of tangential surface and curve intersections.  
 R P Markot and R L Magedson  
 Computer-Aided Design; vol. 21, no 7 sept. 1989, page 421-429

#### Parameters

|                          |                                                                                                                                                                                                                                         |
|--------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>psurf1</code>      | Pointer to the first surface.                                                                                                                                                                                                           |
| <code>psurf2</code>      | Pointer to the second surface                                                                                                                                                                                                           |
| <code>constraints</code> | Constraints flag. <ul style="list-style-type: none"> <li>• constraints           <ol style="list-style-type: none"> <li>1. constraints[0] Constraint 1. surface</li> <li>2. constraints[1] Constraint 2. surface</li> </ol> </li> </ul> |
| Generated by Doxygen     | <ul style="list-style-type: none"> <li>• constraints values           <ol style="list-style-type: none"> <li>1. -1 : No constraints.</li> <li>2. 0 : Constant in 1. par direction</li> </ol> </li> </ul>                                |

----- or -----

- `constraints[0]` Constraint 1. surface
- `constraints[1]` Constraint 2. surface
- `-1` No constraints.
- `0` Constant in 1. par direction
- `1` Constant in 2. par direction

#### Parameters

|                        |                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>constraints_par</i> | Constraints parameters.<br>1. <code>constraints_par[0]</code> Constraints parameter 1. surface.<br>2. <code>constraints_par[1]</code> Constraints parameter 2. surface.                                                                                                                                                                                                                                                                   |
| <i>limit</i>           | - Parameter borders of both surfaces.<br>1. <code>limit[0]</code> - <code>limit[1]</code> Parameter interval. 1. surface 1. direction.<br>2. <code>limit[2]</code> - <code>limit[3]</code> Parameter interval. 1. surface 2. direction.<br>3. <code>limit[4]</code> - <code>limit[5]</code> Parameter interval. 2. surface 1. direction.<br>4. <code>limit[6]</code> - <code>limit[7]</code> Parameter interval. 2. surface 2. direction. |
| <i>enext</i>           | Parameter start value for iteration(4 values).                                                                                                                                                                                                                                                                                                                                                                                            |
| <i>angle_tol</i>       | The angular tolerance for success.                                                                                                                                                                                                                                                                                                                                                                                                        |
| <i>gpos</i>            | Parameter values of the found singularity(4 values).                                                                                                                                                                                                                                                                                                                                                                                      |

#### Returns

Status message. =1 Extremum found. = 0 Extremum NOT found.

**28.1.4.55** `double Go::factorial ( int n ) [inline]`

computes  $n!$  (n factorial)

Definition at line 82 of file `binom.h`.

**28.1.4.56** `int Go::find_point_in ( IntersectionPoint * p, const std::vector< shared_ptr< IntersectionPoint > > & vec )`

Definition at line 491 of file `IntersectionPoolUtils.h`.

**28.1.4.57** `RegistrationResult Go::fineRegistration ( const std::vector< Point > & points_fixed, const std::vector< Point > & points_transform, bool allow_rescaling, RegistrationInput params )`

Given two sequences of points in 3D, get the rotation, rescaling (optional) and translation that sends the second point set as close as possible to the first (i.e. that minimizes the sum of the square distances). The sequences must be of same length (at least three), and the points will be matched in the order they come in the vectors, i.e. `points_transform[i]` should, after the transformation, be close to `points_fixed[i]`. The method works best if the point sets already are close to each other, but not necessarily in optimal position.

28.1.4.58 `Direction2D Go::flip ( Direction2D d ) [inline]`

Definition at line 430 of file Mesh2D.h.

28.1.4.59 `IntersectionPoint* Go::flip_intersecting_objects ( IntersectionPoint * p )`

Definition at line 106 of file IntersectionPoolUtils.h.

28.1.4.60 `template<typename SquareMatrix , typename T > void Go::forwardSubstitution ( const SquareMatrix & L, T * x, int num_unknowns )`

Using forward substitution to calculate  $x$  on the system  $Lx = b$ , where  $L$  is a lower triangular matrix with unitary diagonal.

#### Parameters

|                 |                                                                                                                                                                                   |
|-----------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| $L$             | The lower triangular matrix. The actually used class must support the operation <code>[][]</code> and return 'double'.                                                            |
| $x$             | At function invocation, $x$ should point to an array of $T$ , containing the vector 'b'. On successful completion of the function, this array will contain the solution for $x$ . |
| $num\_unknowns$ | The system size (number of unknowns).                                                                                                                                             |

Definition at line 151 of file LUDecomp\_implementation.h.

28.1.4.61 `template<typename SquareMatrix > void Go::forwardSubstitution ( const SquareMatrix & L, std::vector< double > * x, int num_unknowns )`

Using forward substitution to calculate  $x$  on the system  $Lx = b$ , where  $L$  is a lower triangular matrix with unitary diagonal.

#### Parameters

|                 |                                                                                                                                                                                       |
|-----------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| $L$             | The lower triangular matrix. The actually used class must support the operation <code>[][]</code> and return 'double'.                                                                |
| $x$             | At function invocation, $x$ should point to a vector of doubles, containing the vector 'b'. On successful completion of the function, this vector will contain the solution for $x$ . |
| $num\_unknowns$ | The system size (number of unknowns).                                                                                                                                                 |

Definition at line 163 of file LUDecomp\_implementation.h.

28.1.4.62 `template<typename Functor > double Go::gaussian_quadrature ( Functor & f, double a, double b )`

Routine to calculate the integral of the functor  $f$  from  $a$  to  $b$  using Gaussian quadrature with  $W=1$  and  $N=10$ .

Definition at line 110 of file Integration.h.

28.1.4.63 `template<typename Functor2D > double Go::gaussian_quadrature2D ( Functor2D & f, double ax, double bx, double ay, double by )`

Routine to integrate the two-dimensional functor *f* over the rectangle defined by *ax*, *bx*, *ay* and *by*. Uses Gaussian quadrature.

Definition at line 173 of file *Integration.h*.

28.1.4.64 `void Go::GaussQuadInner ( const BsplineBasis & basis, int ider, double lim1, double lim2, double *** integral )`

Compute all definite integrals of inner products of derivatives of B-splines up to a given order where the differentiation is of the same order for both B-splines. The interval of integration are equal to the parameter intervals of the surface in the current par. dir.

#### Parameters

|                 |                                   |
|-----------------|-----------------------------------|
| <i>basis</i>    | B-spline basis.                   |
| <i>ider</i>     | Number of derivatives to compute. |
| <i>lim1</i>     | Start of parameter interval.      |
| <i>lim2</i>     | End of parameter interval.        |
| <i>integral</i> | Computed integrals.               |

28.1.4.65 `void Go::GaussQuadInner2 ( const BsplineBasis & basis, int ider, double lim1, double lim2, double ** integral )`

Compute all definite integrals of inner products of derivatives of B-splines up to a given order where the differentiation is of the same order for both B-splines. The interval of integration are equal to the parameter intervals of the surface in the current par. dir.

#### Parameters

|                 |                                   |
|-----------------|-----------------------------------|
| <i>basis</i>    | B-spline basis.                   |
| <i>ider</i>     | Number of derivatives to compute. |
| <i>lim1</i>     | Start of parameter interval.      |
| <i>lim2</i>     | End of parameter interval.        |
| <i>integral</i> | Computed integrals.               |

28.1.4.66 `void Go::GaussQuadInnerFlat ( const BsplineBasis & basis, int derivs, int start_der, int gap, double lim1, double lim2, std::vector< double > & integral )`

Compute all definite integrals of inner products of derivatives of B-splines up to a given order where the gap between the order of differentiation on the first and second B-spline is constant. The interval of integration are equal to the parameter intervals of the surface in the current par. dir.

#### Parameters

|              |                 |
|--------------|-----------------|
| <i>basis</i> | B-spline basis. |
|--------------|-----------------|



## Parameters

|                  |                                      |
|------------------|--------------------------------------|
| <i>derivs</i>    | Number of derivatives to compute.    |
| <i>gap</i>       | Difference between derivation order. |
| <i>start_der</i> | First derivative to compute.         |
| <i>lim1</i>      | Start of parameter interval.         |
| <i>lim2</i>      | End of parameter interval.           |
| <i>integral</i>  | Computed integrals.                  |

28.1.4.67 `void Go::GaussQuadInnerRational ( const BsplineBasis & basis, int ider, double lim1, double lim2, shared_ptr< SplineCurve > bspline_curve, double *** integral )`

Compute all definite integrals of inner products of derivatives of rational B-splines up to a given order where the differentiation is of the same order for both B-splines. The interval of integration are equal to the parameter intervals of the surface in the current par. dir.

## Parameters

|                      |                                            |
|----------------------|--------------------------------------------|
| <i>basis</i>         | B-spline basis.                            |
| <i>ider</i>          | Number of derivatives to compute.          |
| <i>lim1</i>          | Start of parameter interval.               |
| <i>lim2</i>          | End of parameter interval.                 |
| <i>bspline_curve</i> | 1-dim rational 0-function defining weights |
| <i>coefs</i>         | Wiegths for denominator function           |
| <i>integral</i>      | Computed integrals.                        |

28.1.4.68 `void Go::GaussQuadValues ( const BsplineBasis & basis, std::vector< double > & parameters, std::vector< double > & par_weights )`

Functions used to compute integrals of inner products of B-splines.

Store parameters and weights used for numerical integration by Gauss quadrature. All parameter values for all Bezier segments are stored in parameters, while the weights within an interval is stored only once in weights. Thus, the lenght of weights gives the number of samples for each Bezier segment.

## Parameters

|                    |                                 |
|--------------------|---------------------------------|
| <i>basis</i>       | B-spline basis                  |
| <i>parameters</i>  | Parameter values for all points |
| <i>par_weights</i> | Weight for each parameter       |

28.1.4.69 `double Go::getCurrentTime ( )`

Number of seconds since some (probably system-dependent) epoch.

28.1.4.70 `void Go::getHermiteData ( const std::vector< Point > & der1, const std::vector< Point > & der2, double & parint, double & len1, double & len2 )`

Given position, first and second derivative in both ends of an Hermite segment, compute parameter interval and tangent lengths in order to stay close to a circular segment.

28.1.4.71 `SISLSurf GO_API* Go::GoSurf2SISL ( const SplineSurface & sf, bool copy = true )`

Convert a [SplineSurface](#) to a SISLSurface

#### Parameters

|             |                                                                                                                                                                                                                                                                        |
|-------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>sf</i>   | the <a href="#">SplineSurface</a> to convert                                                                                                                                                                                                                           |
| <i>copy</i> | if 'true', then the generated <a href="#">SISLSurf</a> will have its own copy of the coefficient information contained in 'sf'. Otherwise, it will share this information with 'sf' (ie. it will only contain a pointer into the corresponding storage array in 'sf'). |

#### Returns

a newly generated [SISLSurf](#) that describes the same surface as 'sf'. The user assumes ownership and is responsible for cleaning up (which means calling the SISL function `freeSurf(...)` on the pointer when it should be destroyed).

28.1.4.72 `void Go::intersect2Dcurves ( const ParamCurve * cv1, const ParamCurve * cv2, double epsge, std::vector< std::pair< double, double > > & intersections, std::vector< int > & pretopology, std::vector< std::pair< std::pair< double, double >, std::pair< double, double > > > & int_crvs )`

Intersect two 2D spline curves. Collect intersection parameters and pretopology information.

#### Parameters

|             |                                |
|-------------|--------------------------------|
| <i>cv1</i>  | pointer to the first 2D curve  |
| <i>cv2</i>  | pointer to the second 2D curve |
| <i>epsg</i> | geometrical tolerance          |

#### Return values

|                      |                                                                                                                                                                                                                                  |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>intersections</i> | this vector will contain the parameter pairs of the found intersections (one vector entry per intersection. The two parameters in the pair<> correspond to the parameter value in 'cv1' and 'cv2' for a particular intersection. |
| <i>pretopology</i>   | vector containing a pretopology indicator for each detected intersection point. There is one entry per intersection point.                                                                                                       |

28.1.4.73 `void Go::intersectCurvePoint ( const ParamCurve * crv, Point pnt, double epsge, std::vector< double > & intersections, std::vector< std::pair< double, double > > & int_crvs )`

28.1.4.74 `void Go::intersectCurves ( shared_ptr< ParamCurve > crv1, shared_ptr< ParamCurve > crv2, double tol, std::vector< std::pair< double, double > > & intersection_points )`

Intersection between two parametric curves.

28.1.4.75 `void Go::intersectcurves ( SplineCurve * cv1, SplineCurve * cv2, double epsge, std::vector< std::pair< double, double > > & intersections )`

Intersect two spline curves. Collect intersection parameters.

#### Parameters

|              |                                    |
|--------------|------------------------------------|
| <i>cv1</i>   | pointer to the first spline curve  |
| <i>cv2</i>   | pointer to the second spline curve |
| <i>epsge</i> | geometrical tolerance              |

#### Return values

|                      |                                                                                                                                                                                                                                  |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>intersections</i> | this vector will contain the parameter pairs of the found intersections (one vector entry per intersection. The two parameters in the pair<> correspond to the parameter value in 'cv1' and 'cv2' for a particular intersection. |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

28.1.4.76 `void Go::intersectCurveSurf ( const SplineCurve * cv, const SplineSurface * sf, double epsge, std::vector< std::pair< double, Point > > & int_pts, std::vector< int > & pretopology, std::vector< std::pair< std::pair< double, Point >, std::pair< double, Point > > > & int_crvs )`

Intersect a spline curve and a spline surface.

28.1.4.77 `template<typename ValueType > bool Go::interval_overlap ( ValueType front1, ValueType back1, ValueType front2, ValueType back2 )`

Definition at line 151 of file checks.h.

28.1.4.78 `int Go::is_inside ( const std::vector< Go::Vector3D > & trim_curve_p, const std::vector< int > & contour, const double u, const double v )`

28.1.4.79 `int Go::is_on_contour ( const std::vector< Go::Vector3D > & trim_curve_p, const std::vector< int > & contour, const double u, const double v )`

28.1.4.80 `bool Go::is_on_corner ( const std::vector< Go::Vector3D > & trim_curve_p, const std::vector< int > & contour, const double u, const double v )`

28.1.4.81 `bool Go::link_is_iso ( shared_ptr< IntersectionLink > link )`

Definition at line 231 of file IntersectionPoolUtils.h.

28.1.4.82 `bool Go::link_is_iso_in ( shared_ptr< IntersectionLink > link, int dir )`

Definition at line 223 of file IntersectionPoolUtils.h.

28.1.4.83 `bool Go::link_is_iso_in_other_than ( shared_ptr< IntersectionLink > link, int dir )`

Definition at line 239 of file IntersectionPoolUtils.h.

28.1.4.84 `template<typename SquareMatrix > void Go::LUDecomp ( SquareMatrix & mat, int num_rows, int * perm, bool & parity )`

LU decomposition algorithm, based on Crout's algorithm

#### Parameters

|                 |                                                                                                                        |
|-----------------|------------------------------------------------------------------------------------------------------------------------|
| <i>mat</i>      | The matrix to be decomposed. The actually used class must support the operation <code>[][]</code> and return 'double'. |
| <i>num_rows</i> | Number of rows (which is also equal to the number of columns).                                                         |
| <i>perm</i>     | Should point to an n-sized array where the permutation is written.                                                     |
| <i>parity</i>   | Value upon function completion is 'true' if the number of row interchanges was pair. 'false' otherwise.                |

Definition at line 52 of file LUDecomp\_implementation.h.

28.1.4.85 `template<typename SquareMatrix , typename T > void Go::LUSolveSystem ( SquareMatrix & A, int num_unknowns, T * vec )`

Solve the system  $Ax = b$  for  $x$ , using LU decomposition of the matrix  $A$ . Upon successful completion of the function, the matrix  $A$  will be LU decomposed, and the solution  $x$  will be computed and stored at the memory area pointed to by the last argument.

#### Parameters

|                     |                                                                                                                                                                                     |
|---------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>A</i>            | The system matrix $A$ . The actually used class must support the operation <code>[][]</code> and return 'double'.                                                                   |
| <i>num_unknowns</i> | Number of unknowns in the equation system.                                                                                                                                          |
| <i>vec</i>          | At function invocation, 'vec' should point to an array of $T$ , containing the vector 'b'. On successful completion of the function, this array will contain the solution for $x$ . |

Definition at line 132 of file LUDecomp\_implementation.h.

28.1.4.86 `shared_ptr<const CurveOnSurface> Go::make_curve_on_surface ( shared_ptr< const ParamSurface > surf, double u_start, double v_start, double u_end, double v_end, double p_start, double p_end )`

Definition at line 508 of file IntersectionPoolUtils.h.

28.1.4.87 `void Go::make_implicit_gauss ( std::vector< std::vector< double > > & mat, std::vector< double > & b )`

Performs implicitization using Gaussian elimination. This method is suitable when the implicitization is exact. If the implicitization is approximate, [make\\_implicit\\_svd\(\)](#) is better.

28.1.4.88 `void Go::make_implicit_svd ( std::vector< std::vector< double > > & mat, std::vector< double > & b, double & sigma_min )`

Performs implicitization using SVD. This method is suitable when the implicitization is approximate. If the implicitization is exact, [make\\_implicit\\_gauss\(\)](#) is better. Based on the function [SVD\(\)](#) from the newmat matrix library.

28.1.4.89 `void Go::make_matrix ( const SplineCurve & curve, int deg, std::vector< std::vector< double > > & mat )`

Make the matrix D.

28.1.4.90 `void Go::make_matrix ( const SplineSurface & surf, int deg, std::vector< std::vector< double > > & mat )`

Make the matrix D.

28.1.4.91 `void Go::make_matrix ( const PointCloud4D & cloud, int deg, std::vector< std::vector< double > > & mat )`

Make the matrix D.

28.1.4.92 `void Go::make_trimmed_mesh ( shared_ptr< ParamSurface > srf, std::vector< shared_ptr< ParamCurve > > & crv_set, std::vector< Vector3D > & vert, std::vector< Vector2D > & vert_p, std::vector< int > & bd, std::vector< Vector3D > & norm, std::vector< int > & mesh, std::vector< Vector3D > & trim_curve, std::vector< Vector3D > & trim_curve_p, const int dn, const int dm, double bd_res_ratio )`

28.1.4.93 `template<class Functor > void Go::minimise_conjugated_gradient ( FunctionMinimizer< Functor > & dfmin )`

This is the algorithm for minimising a function taking multiple parameters, using the conjugated gradient method. It is used in conjunction with the [FunctionMinimizer](#) class. Documentation for both can be found here: [FunctionMinimizer](#)  
Definition at line 78 of file `GeneralFunctionMinimizer_implementation.h`.

28.1.4.94 `bool Go::no_parent ( const shared_ptr< IntersectionPoint > & p )`

Definition at line 98 of file `IntersectionPoolUtils.h`.

28.1.4.95 `template<typename ValueType > bool Go::nondecreasing ( ValueType a, ValueType b, ValueType c )`

Definition at line 117 of file `checks.h`.

28.1.4.96 `template<typename ValueType > bool Go::nonincreasing ( ValueType a, ValueType b, ValueType c )`

Definition at line 125 of file `checks.h`.

28.1.4.97 `void Go::normalNoise ( double * res, double mean_err, int num_samples )`

Gives a certain number of random samples drawn from the normal distribution.

## Parameters

|                    |                                                                                                 |
|--------------------|-------------------------------------------------------------------------------------------------|
| <i>res</i>         | a pointer to the array where the resulting samples should be written.                           |
| <i>mean_err</i>    | the sigma parameter to the normal distribution.                                                 |
| <i>num_samples</i> | the desired number of samples (should also be the size of the array pointed to by <i>res</i> ). |

**28.1.4.98 Rational** `Go::operator* ( const Rational & r1, const Rational r2 )`

Definition at line 157 of file Rational.h.

**28.1.4.99** `template<class T, int Dim> Array<T, Dim> Go::operator* ( const Array< double, Dim > & a, const T b )`  
`[inline]`

Definition at line 171 of file BaryCoordSystem.h.

**28.1.4.100 BernsteinPoly** `Go::operator* ( const BernsteinPoly & p1, const BernsteinPoly & p2 )` `[inline]`

Multiplication of two polynomials.

Definition at line 200 of file BernsteinPoly.h.

**28.1.4.101 BernsteinPoly** `Go::operator* ( const BernsteinPoly & p, double c )` `[inline]`

Multiplication of a polynomial with a scalar.

Definition at line 210 of file BernsteinPoly.h.

**28.1.4.102 BernsteinPoly** `Go::operator* ( double c, const BernsteinPoly & p )` `[inline]`

Multiplication of a scalar with a polynomial.

Definition at line 220 of file BernsteinPoly.h.

**28.1.4.103 BernsteinTriangularPoly** `Go::operator* ( const BernsteinTriangularPoly & p1, const BernsteinTriangularPoly & p2 )` `[inline]`

Multiplication of two polynomials.

Definition at line 225 of file BernsteinTriangularPoly.h.

**28.1.4.104 BernsteinTriangularPoly** `Go::operator* ( const BernsteinTriangularPoly & p, double c )` `[inline]`

Multiplication of a polynomial with a scalar.

Definition at line 235 of file BernsteinTriangularPoly.h.

28.1.4.105 **BernsteinTriangularPoly** `Go::operator*( double c, const BernsteinTriangularPoly & p )` `[inline]`

Multiplication of a scalar with a polynomial.

Definition at line 245 of file BernsteinTriangularPoly.h.

28.1.4.106 **BernsteinTetrahedralPoly** `Go::operator*( const BernsteinTetrahedralPoly & p1, const BernsteinTetrahedralPoly & p2 )` `[inline]`

Multiplication of two polynomials.

Definition at line 252 of file BernsteinTetrahedralPoly.h.

28.1.4.107 **BernsteinTetrahedralPoly** `Go::operator*( const BernsteinTetrahedralPoly & p, double c )` `[inline]`

Multiplication of a polynomial with a scalar.

Definition at line 263 of file BernsteinTetrahedralPoly.h.

28.1.4.108 **BernsteinTetrahedralPoly** `Go::operator*( double c, const BernsteinTetrahedralPoly & p )` `[inline]`

Multiplication of a scalar with a polynomial.

Definition at line 273 of file BernsteinTetrahedralPoly.h.

28.1.4.109 **BernsteinMulti** `Go::operator*( const BernsteinMulti & m1, const BernsteinMulti & m2 )` `[inline]`

Multiplication of two polynomials.

Definition at line 300 of file BernsteinMulti.h.

28.1.4.110 **BernsteinMulti** `Go::operator*( const BernsteinMulti & m1, double c )` `[inline]`

Multiplication of a polynomial with a scalar.

Definition at line 310 of file BernsteinMulti.h.

28.1.4.111 **BernsteinMulti** `Go::operator*( double c, const BernsteinMulti & m1 )` `[inline]`

Multiplication of a scalar with a polynomial.

Definition at line 320 of file BernsteinMulti.h.

28.1.4.112 `template<typename T, int Dim> Go::Array<T, Dim> Go::operator* ( T d, const Go::Array< T, Dim > & v )`  
[inline]

The product of a vector and a scalar.

Definition at line 440 of file Array.h.

28.1.4.113 `Point Go::operator* ( double d, const Point & p )` [inline]

The product of a vector and a scalar.

Definition at line 539 of file Point.h.

28.1.4.114 `Rational Go::operator+ ( const Rational & r1, const Rational r2 )`

Definition at line 143 of file Rational.h.

28.1.4.115 `BernsteinPoly Go::operator+ ( const BernsteinPoly & p1, const BernsteinPoly & p2 )` [inline]

Addition of two polynomials.

Definition at line 230 of file BernsteinPoly.h.

28.1.4.116 `BernsteinPoly Go::operator+ ( double c, const BernsteinPoly & p )` [inline]

Addition of a scalar with a polynomial.

Definition at line 240 of file BernsteinPoly.h.

28.1.4.117 `BernsteinPoly Go::operator+ ( const BernsteinPoly & p, double c )` [inline]

Addition of a polynomial with a scalar.

Definition at line 250 of file BernsteinPoly.h.

28.1.4.118 `BernsteinTriangularPoly Go::operator+ ( const BernsteinTriangularPoly & p1, const BernsteinTriangularPoly & p2 )` [inline]

Addition of two polynomials.

Definition at line 255 of file BernsteinTriangularPoly.h.

28.1.4.119 `BernsteinTriangularPoly Go::operator+ ( double c, const BernsteinTriangularPoly & p )` [inline]

Addition of a scalar with a polynomial.

Definition at line 265 of file BernsteinTriangularPoly.h.



28.1.4.120 **BernsteinTriangularPoly** `Go::operator+ ( const BernsteinTriangularPoly & p, double c )` `[inline]`

Addition of a polynomial with a scalar.

Definition at line 275 of file BernsteinTriangularPoly.h.

28.1.4.121 **BernsteinTetrahedralPoly** `Go::operator+ ( const BernsteinTetrahedralPoly & p1, const BernsteinTetrahedralPoly & p2 )` `[inline]`

Addition of two polynomials.

Definition at line 283 of file BernsteinTetrahedralPoly.h.

28.1.4.122 **BernsteinTetrahedralPoly** `Go::operator+ ( double c, const BernsteinTetrahedralPoly & p )` `[inline]`

Addition of a scalar with a polynomial.

Definition at line 294 of file BernsteinTetrahedralPoly.h.

28.1.4.123 **BernsteinTetrahedralPoly** `Go::operator+ ( const BernsteinTetrahedralPoly & p, double c )` `[inline]`

Addition of a polynomial with a scalar.

Definition at line 304 of file BernsteinTetrahedralPoly.h.

28.1.4.124 **BernsteinMulti** `Go::operator+ ( const BernsteinMulti & m1, const BernsteinMulti & m2 )` `[inline]`

Addition of two polynomials.

Definition at line 330 of file BernsteinMulti.h.

28.1.4.125 **BernsteinMulti** `Go::operator+ ( const BernsteinMulti & m, double c )` `[inline]`

Addition of a polynomial with a scalar.

Definition at line 340 of file BernsteinMulti.h.

28.1.4.126 **BernsteinMulti** `Go::operator+ ( double c, const BernsteinMulti & m )` `[inline]`

Addition of a scalar with a polynomial.

Definition at line 350 of file BernsteinMulti.h.

28.1.4.127 **Rational** Go::operator- ( const Rational & r1, const Rational r2 )

Definition at line 150 of file Rational.h.

28.1.4.128 **BernsteinPoly** Go::operator- ( const BernsteinPoly & p1, const BernsteinPoly & p2 ) [inline]

Subtraction of two polynomials.

Definition at line 260 of file BernsteinPoly.h.

28.1.4.129 **BernsteinPoly** Go::operator- ( double c, const BernsteinPoly & p ) [inline]

Subtraction of a polynomial from a scalar.

Definition at line 270 of file BernsteinPoly.h.

28.1.4.130 **BernsteinPoly** Go::operator- ( const BernsteinPoly & p, double c ) [inline]

Subtraction of a scalar from a polynomial.

Definition at line 280 of file BernsteinPoly.h.

28.1.4.131 **BernsteinTriangularPoly** Go::operator- ( const BernsteinTriangularPoly & p1, const BernsteinTriangularPoly & p2 ) [inline]

Subtraction of two polynomials.

Definition at line 285 of file BernsteinTriangularPoly.h.

28.1.4.132 **BernsteinTriangularPoly** Go::operator- ( double c, const BernsteinTriangularPoly & p ) [inline]

Subtraction of a polynomial from a scalar.

Definition at line 295 of file BernsteinTriangularPoly.h.

28.1.4.133 **BernsteinTriangularPoly** Go::operator- ( const BernsteinTriangularPoly & p, double c ) [inline]

Subtraction of a scalar from a polynomial.

Definition at line 305 of file BernsteinTriangularPoly.h.

28.1.4.134 **BernsteinTetrahedralPoly** Go::operator- ( const BernsteinTetrahedralPoly & p1, const BernsteinTetrahedralPoly & p2 ) [inline]

Subtraction of two polynomials.

Definition at line 314 of file BernsteinTetrahedralPoly.h.

28.1.4.135 **BernsteinTetrahedralPoly** Go::operator- ( double *c*, const BernsteinTetrahedralPoly & *p* )  
[inline]

Subtraction of a polynomial from a scalar.

Definition at line 325 of file BernsteinTetrahedralPoly.h.

28.1.4.136 **BernsteinTetrahedralPoly** Go::operator- ( const BernsteinTetrahedralPoly & *p*, double *c* )  
[inline]

Subtraction of a scalar from a polynomial.

Definition at line 335 of file BernsteinTetrahedralPoly.h.

28.1.4.137 **BernsteinMulti** Go::operator- ( const BernsteinMulti & *m1*, const BernsteinMulti & *m2* ) [inline]

Subtraction of two polynomials.

Definition at line 360 of file BernsteinMulti.h.

28.1.4.138 **BernsteinMulti** Go::operator- ( const BernsteinMulti & *m*, double *c* ) [inline]

Subtraction of a scalar from a polynomial.

Definition at line 370 of file BernsteinMulti.h.

28.1.4.139 **BernsteinMulti** Go::operator- ( double *c*, const BernsteinMulti & *m* ) [inline]

Subtraction of a polynomial from a scalar.

Definition at line 380 of file BernsteinMulti.h.

28.1.4.140 **Rational** Go::operator/ ( const Rational & *r1*, const Rational *r2* )

Definition at line 164 of file Rational.h.

28.1.4.141 **BernsteinPoly** Go::operator/ ( const BernsteinPoly & *p*, double *c* ) [inline]

Division of a polynomial with a scalar.

Definition at line 290 of file BernsteinPoly.h.

28.1.4.142 **BernsteinTriangularPoly** Go::operator/ ( const BernsteinTriangularPoly & *p*, double *c* ) [inline]

Division of a polynomial with a scalar.

Definition at line 315 of file BernsteinTriangularPoly.h.

**28.1.4.143 BernsteinTetrahedralPoly** `Go::operator/ ( const BernsteinTetrahedralPoly & p, double c )`  
[inline]

Division of a polynomial with a scalar.

Definition at line 345 of file BernsteinTetrahedralPoly.h.

**28.1.4.144 BernsteinMulti** `Go::operator/ ( const BernsteinMulti & m, double c )` [inline]

Division of a polynomial with a scalar.

Definition at line 390 of file BernsteinMulti.h.

**28.1.4.145 bool** `Go::operator< ( const Point & p1, const Point & p2 )` [inline]

Less than operator.

Definition at line 551 of file Point.h.

**28.1.4.146 std::ostream&** `Go::operator<< ( std::ostream & os, const Go::Streamable & obj )` [inline]

Definition at line 79 of file Streamable.h.

**28.1.4.147 std::ostream&** `Go::operator<< ( std::ostream & os, const Rational & p )`

Definition at line 171 of file Rational.h.

**28.1.4.148 std::ostream&** `Go::operator<< ( std::ostream & os, const Go::BernsteinTriangularPoly & p )`  
[inline]

Write [BernsteinTriangularPoly](#) to output stream.

Definition at line 333 of file BernsteinTriangularPoly.h.

**28.1.4.149 std::ostream&** `Go::operator<< ( std::ostream & os, const Go::BernsteinTetrahedralPoly & p )`  
[inline]

Write [BernsteinTetrahedralPoly](#) to output stream.

Definition at line 363 of file BernsteinTetrahedralPoly.h.

**28.1.4.150 std::ostream&** `Go::operator<< ( std::ostream & os, const LRBSpline2D & b )` [inline]

Definition at line 374 of file LRBSpline2D.h.

28.1.4.151 `std::ostream& Go::operator<< ( std::ostream & os, const GPos & g )` `[inline]`

Definition at line 437 of file Mesh2D.h.

28.1.4.152 `std::ostream& Go::operator<< ( std::ostream & os, const Mesh2D & m )` `[inline]`

Definition at line 439 of file Mesh2D.h.

28.1.4.153 `template<typename T, int Dim> std::ostream& Go::operator<< ( std::ostream & os, const Go::Array< T, Dim > & v )` `[inline]`

Stream insertion for [Array](#).

Definition at line 450 of file Array.h.

28.1.4.154 `template<typename T, int Dim> std::ostream& Go::operator<< ( std::ostream & os, const MatrixXD< T, Dim > & m )` `[inline]`

output operator

Specialization of the determinant function for the 1x1 case. Terminates the [det\(\)](#) template recursion.

Definition at line 462 of file MatrixXD.h.

28.1.4.155 `std::ostream& Go::operator<< ( std::ostream & os, const Go::Point & v )` `[inline]`

Stream insertion for [Point](#).

Definition at line 547 of file Point.h.

28.1.4.156 `bool Go::operator==( const Point & p1, const Point & p2 )` `[inline]`

Equal operator.

Definition at line 563 of file Point.h.

28.1.4.157 `std::istream& Go::operator>> ( std::istream & is, Go::Streamable & obj )` `[inline]`

Definition at line 72 of file Streamable.h.

28.1.4.158 `std::istream& Go::operator>> ( std::istream & is, Go::BernsteinTriangularPoly & p )` `[inline]`

Read [BernsteinTriangularPoly](#) from input stream.

Definition at line 325 of file BernsteinTriangularPoly.h.

28.1.4.159 `std::istream& Go::operator>> ( std::istream & is, Go::BernsteinTetrahedralPoly & p )` `[inline]`

Read [BernsteinTetrahedralPoly](#) from input stream.

Definition at line 354 of file BernsteinTetrahedralPoly.h.

28.1.4.160 `std::istream& Go::operator>> ( std::istream & is, LRBSpline2D & b )` `[inline]`

Definition at line 375 of file LRBSpline2D.h.

28.1.4.161 `std::istream& Go::operator>> ( std::istream & is, GPos & g )` `[inline]`

Definition at line 438 of file Mesh2D.h.

28.1.4.162 `std::istream& Go::operator>> ( std::istream & is, Mesh2D & m )` `[inline]`

Definition at line 440 of file Mesh2D.h.

28.1.4.163 `template<typename T, int Dim> std::istream& Go::operator>> ( std::istream & is, Go::Array< T, Dim > & v )`  
`[inline]`

Stream extraction for [Array](#).

Definition at line 445 of file Array.h.

28.1.4.164 `std::istream& Go::operator>> ( std::istream & is, Go::Point & v )` `[inline]`

Stream extraction for [Point](#).

Definition at line 543 of file Point.h.

28.1.4.165 `bool Go::point_inside_contour ( const double x0, const double y0, const double *const vertices, const std::vector< int > & contour )`

28.1.4.166 `shared_ptr<boxStructuring::BoundingBoxStructure> Go::preProcessClosestVectors ( const std::vector< shared_ptr< GeomObject > > & surfaces, double par_len_el )`

Create preprocessing data for the closest vector calculations on a surface. `surfaces` - a collection of the parametric surfaces defining the surface model. Only instances of the [ParamSurface](#) subclass hierarchy are used `par_len_el` - a guiding for the side lengths of the segments in geometry space, used to determine the number of segments for elementary surfaces returns the preprocessing structures used as input for the closest point calculations

28.1.4.167 `double Go::quadrinomial ( int n, int i, int j, int k )` `[inline]`

computes the quadrinomial coefficient:  $n! / (i! j! k! (n-i-j-k)!)$

Definition at line 102 of file binom.h.

**28.1.4.168** `RegistrationResult Go::rawRegistration ( const std::vector< Point > & points_fixed, const std::vector< Point > & points_transform, bool allow_rescaling, RegistrationInput params )`

Given two sequences of points in 3D, get an approximate rotation, rescaling (optional) and translation that sends the second point set close to the first. The point sets should be close (but not necessarily identical) in shape, with matching points coming in the same order (i.e. point number *i* in the two sequences should be close to each other after performing the transformation on the second point set), but the original coordinate representation of the two sets might be totally different.

**28.1.4.169** `template<class T > void Go::Register ( )`

This function is used to register a class derived from [GeomObject](#) with the global [Factory](#). By using this function rather than [Factory::registerClass\(\)](#), the user does not have to worry about the details of the [Creator](#) class. To register a class

`DerivedClass`

, it should be sufficient to run:

```
Register<DerivedClass> ()
```

.

Definition at line 154 of file [Factory.h](#).

**28.1.4.170** `RegistrationResult Go::registration ( const std::vector< Point > & points_fixed, const std::vector< Point > & points_transform, bool allow_rescaling, RegistrationInput params )`

Given two sequences of points in 3D, get the rotation, rescaling (optional) and translation that sends the second point set as close as possible to the first (i.e. that minimizes the sum of the square distances). The sequences must be of same length (at least three), and the points will be matched in the order they come in the vectors, i.e. `points_transform[i]` should, after the transformation, be close to `points_fixed[i]`.

**28.1.4.171** `bool Go::segment_contour_intersection_for_s2m ( const double x0, const double y0, const double x1, const double y1, const double *const vertices, const std::vector< int > & contour, double & x, double & y, double & s, const bool snap_ends = false )`

**28.1.4.172** `template<typename T > T Go::signed_area ( const Array< T, 3 > * c, const Array< T, 3 > & normal )`

Computes the signed area of a triangle embedded in 3D space. Input is an array of corner points and a normal to determine the sign.

Definition at line 117 of file [Volumes.h](#).

**28.1.4.173** `template<typename T, int Dim> T Go::simplex_volume ( const Array< T, Dim > * a ) [inline]`

Computes the volume of a simplex consisting of (Dim+1) vertices embedded in Euclidean space of dimension (Dim)

Definition at line 77 of file [Volumes.h](#).

28.1.4.174 `template<typename Functor > double Go::simpsons_rule ( Functor & f, double a, double b, const double eps = 1.0e-6, const int max_iter = 20 )`

Routine to calculate the integral of the functor f from a to b using Simpson's rule.

Definition at line 84 of file Integration.h.

28.1.4.175 `template<typename Functor2D > double Go::simpsons_rule2D ( Functor2D & f, double ax, double bx, double ay, double by, const double eps = 1.0e-6, const int max_iter = 20 )`

Help functor to simpsons\_rule2D and gaussian\_quadrature2DRoutine to integrate the two-dimensional functor f over the rectangle defined by ax, bx, ay and by. Uses Simpson's rule.

Definition at line 161 of file Integration.h.

28.1.4.176 `SplineCurve GO_API* Go::SISLCurve2Go ( const SISLCurve *const cv )`

Convert a [SISLCurve](#) to a [SplineCurve](#)

Parameters

|                 |                          |
|-----------------|--------------------------|
| <code>cv</code> | the SISLCurve to convert |
|-----------------|--------------------------|

Returns

A newly generated [SplineCurve](#) that describes the same curve as 'cv'. The user assumes ownership and is responsible for cleaning up by calling the `delete` function.

28.1.4.177 `SplineSurface GO_API* Go::SISLSurf2Go ( SISLSurf * sf )`

Convert a SISLSurface to a [SplineSurface](#)

Parameters

|                 |                         |
|-----------------|-------------------------|
| <code>sf</code> | the SISLSurf to convert |
|-----------------|-------------------------|

Returns

A newly generated [SplineSurface](#) that describes the same surface as 'sf'. The user assumes ownership and is responsible for cleaning up by calling the `delete` function.

28.1.4.178 `std::vector< short_list_short_list > Go::sort_2dpoly_segments ( const double *const vertices, const std::vector< int > & contour, const bool transposed = false )`

28.1.4.179 `void Go::spline_to_bernstein ( const SplineCurve & seg, int dd, BernsteinPoly & bp )`

Takes a segment (defined as having numCoefs() == order() and an order()-regular knot vector) and returns a bernstein polynomial equal to the curve's coordinate number dd. Allowed values for dd are in [0, dimension()-1] for



nonrational segments, and in  $[0, \text{dimension}())$  for rational (NURBS) segments. The polynomials returned in the rational case are the weighted coordinate functions for  $dd$  in  $[0, \text{dimension}()-1]$  and to the weight polynomial for  $dd$  equal to  $\text{dimension}()$ .

28.1.4.180 `void Go::spline_to_bernstein ( const SplineSurface & pat, int dd, BernsteinMulti & bm )`

Converts the  $dd$ -component of a surface patch on [SplineSurface](#) form to a [BernsteinMulti](#)

28.1.4.181 `void Go::spline_to_bernstein ( const SplineCurve & seg, std::vector< BernsteinPoly > & seg_bp )`

Converts a curve segment on [SplineCurve](#) form to a vector of BernsteinPolys

28.1.4.182 `void Go::spline_to_bernstein ( const SplineSurface & pat, std::vector< BernsteinMulti > & pat_bm )`

Converts a surface patch on [SplineSurface](#) form to a vector of BernsteinMultis

28.1.4.183 `template<int Ndim> void Go::splineToBernstein ( const SplineCurve & segment, Array< BernsteinPoly, Ndim > & curve_bp )`

Converts a Bezier curve segment on [SplineCurve](#) form to an [Array](#) of  $Ndim$  BernsteinPolys. If the [SplineCurve](#) is rational,  $Ndim$  is equal to  $dim+1$ , where  $dim$  is the dimension of space. Furthermore, in  $curve\_bp$ , the weights are included in the space part of the coefficients, i.e. they have the form  $wP1, wP2, \dots, wPdim, w$ .

#### Parameters

|                 |                                                              |
|-----------------|--------------------------------------------------------------|
| <i>segment</i>  | the Bezier curve segment on <a href="#">SplineCurve</a> form |
| <i>curve_bp</i> | the resulting <a href="#">Array</a> of BernsteinPolys        |

Definition at line 96 of file BernsteinUtils.h.

28.1.4.184 `template<int Ndim> void Go::splineToBernstein ( const SplineSurface & patch, Array< BernsteinMulti, Ndim > & surface_bm )`

Converts a Bezier surface patch on [SplineSurface](#) form to an [Array](#) of  $Ndim$  BernsteinMultis. If the [SplineSurface](#) is rational,  $Ndim$  is equal to  $dim+1$ , where  $dim$  is the dimension of space. Furthermore, in  $surface\_bm$ , the weights are included in the space part of the coefficients, i.e. they have the form  $wP1, wP2, \dots, wPdim, w$ .

#### Parameters

|                   |                                                          |
|-------------------|----------------------------------------------------------|
| <i>patch</i>      | the Bezier surface on <a href="#">SplineSurface</a> form |
| <i>surface_bm</i> | the resulting <a href="#">Array</a> of BernsteinMultis   |

Definition at line 137 of file BernsteinUtils.h.

28.1.4.185 **double** Go::stepLenFromRadius ( *double radius*, *double aepsge* )

Computes the step length along a curve based on radius of curvature at a point on the curve, and an absolute tolerance.

28.1.4.186 **int** Go::stepLength ( *const std::vector< Point > & ft*, *const std::vector< Point > & gs*, *double delta*, *bool forward*, *double & delta\_t* )

28.1.4.187 **double** Go::stepLength ( *const std::vector< Point > & ft*, *const std::vector< Point > & gs*, *bool forward*, *std::vector< Point > & cvder*, *double min\_step*, *double max\_step*, *double aepsge* )

28.1.4.188 **template**<typename Iterator > **bool** Go::strictly\_decreasing ( Iterator *begin*, Iterator *end* )

Definition at line 88 of file checks.h.

28.1.4.189 **template**<typename Array > **bool** Go::strictly\_decreasing ( *const Array & A* )

Definition at line 95 of file checks.h.

28.1.4.190 **template**<typename ValueType > **bool** Go::strictly\_decreasing ( ValueType *a*, ValueType *b*, ValueType *c* )

Definition at line 133 of file checks.h.

28.1.4.191 **template**<typename Iterator > **bool** Go::strictly\_increasing ( Iterator *begin*, Iterator *end* )

Definition at line 58 of file checks.h.

28.1.4.192 **template**<typename Array > **bool** Go::strictly\_increasing ( *const Array & A* )

Definition at line 66 of file checks.h.

28.1.4.193 **template**<typename ValueType > **bool** Go::strictly\_increasing ( ValueType *a*, ValueType *b*, ValueType *c* )

Definition at line 141 of file checks.h.

28.1.4.194 **void** Go::systemSleep ( *double sleep\_time* )

Sleep for sleep\_time seconds.

28.1.4.195 **double** Go::tanLenFromRadius ( *double radius*, *double angle* )

To create the tangent length for interpolating a circular arc with an almost equi-oscillating Hermit cubic.

28.1.4.196 `template<typename Functor > void Go::trapezoidal ( Functor & f, double a, double b, double & s, int n )`

This routine computes the n'th stage of refinement of an extended trapezoidal rule. Used as a help routine for 'simpsons\_rule', found further down in this file. NB: It only carries out the computation for ONE stage of the procedure, so if you want to use it to integrate a function, you must call it successively for  $n = 1$ ,  $n = 2$ , ....

Definition at line 60 of file Integration.h.

28.1.4.197 `double Go::trinomial ( int n, int i, int j ) [inline]`

computes the trinomial coefficient:  $n! / (i! j! (n-i-j)!)$

Definition at line 92 of file binom.h.

28.1.4.198 `void Go::uniformNoise ( double * res, double lval, double uval, int num_samples )`

Gives a certain number of random samples drawn from the uniform distribution.

#### Parameters

|                    |                                                                                                 |
|--------------------|-------------------------------------------------------------------------------------------------|
| <i>res</i>         | a pointer to the array where the resulting samples should be written.                           |
| <i>lval</i>        | lower bound of the range from which the samples can take their values.                          |
| <i>uval</i>        | upper bound of the range from which the samples can take their values.                          |
| <i>num_samples</i> | the desired number of samples (should also be the size of the array pointed to by <i>res</i> ). |

28.1.4.199 `template<typename T > T Go::volume ( const Array< T, 3 > * c ) [inline]`

Computes the volume of a 3D simplex (embedded in 3D space).

Definition at line 110 of file Volumes.h.

28.1.4.200 `template<typename Iterator > bool Go::weakly_decreasing ( Iterator begin, Iterator end )`

Definition at line 102 of file checks.h.

28.1.4.201 `template<typename Array > bool Go::weakly_decreasing ( const Array & A )`

Definition at line 109 of file checks.h.

28.1.4.202 `template<typename Iterator > bool Go::weakly_increasing ( Iterator begin, Iterator end )`

Definition at line 74 of file checks.h.

28.1.4.203 `template<typename Array > bool Go::weakly_increasing ( const Array & A )`

Definition at line 81 of file checks.h.

28.1.4.204 `void Go::writePostscriptMesh ( Go::LRSplineSurface & lr_spline_sf, std::ostream & out, const bool close = true )`

## 28.1.5 Variable Documentation

28.1.5.1 `const int Go::MAJOR_VERSION = 1`

Definition at line 51 of file GeomObject.h.

28.1.5.2 `const int Go::MINOR_VERSION = 0`

Definition at line 52 of file GeomObject.h.

## 28.2 Go::AdaptSurface Namespace Reference

### Functions

- `shared_ptr< SplineSurface > approxInSplineSpace (shared_ptr< ParamSurface > surf, shared_ptr< SplineSurface > surf2, double tol)`
- `std::vector< shared_ptr< SplineSurface > > expressInSameSplineSpace (shared_ptr< ParamSurface > surf1, shared_ptr< ParamSurface > surf2, double tol)`
- `bool getCornerCorrespondance (shared_ptr< ParamSurface > surf1, shared_ptr< ParamSurface > surf2, int &idx, bool &turned)`
- `std::vector< shared_ptr< SplineCurve > > curveApprox (shared_ptr< ParamCurve > cvs[], int nmb_cvs, const BsplineBasis &init_basis, double tol)`
- `std::vector< shared_ptr< SplineCurve > > curveApprox (shared_ptr< ParamCurve > cvs[], int nmb_cvs, double tol, double degree=3)`
- `shared_ptr< SplineSurface > adaptSurface (shared_ptr< ParamSurface > surf, shared_ptr< SplineSurface > init_surf, double tol)`
- `double parameterizePoints (shared_ptr< SplineSurface > init_surf, shared_ptr< ftPointSet > points, std::vector< int > corner)`
- `double projectPoints (shared_ptr< SplineSurface > surf, shared_ptr< ftPointSet > points)`  
*Parameterize by projection.*
- `shared_ptr< SplineSurface > doApprox (shared_ptr< SplineSurface > init_surf, int max_iter, shared_ptr< ftPointSet > points, double tol, double &max_error, double &mean_error)`
- `void createTriangulation (shared_ptr< ParamSurface > surf, const RectDomain &dom, shared_ptr< ftPointSet > &points, vector< int > &corner, bool consider_joint=true, int nmb=-1)`  
*Fetch data and create triangulation.*
- `void getBoundaryData (shared_ptr< ParamSurface > surf, const RectDomain &dom, int nmb_sample, shared_ptr< ftPointSet > points, std::vector< int > &corner)`
- `void getInnerData (shared_ptr< ParamSurface > surf, const RectDomain &dom, int nmb_sample, shared_ptr< ftPointSet > points, bool consider_joint=true)`
- `void updatePointTopology (shared_ptr< ParamSurface > surf, ftPointSet &points)`  
*Compute point set topology.*

### 28.2.1 Detailed Description

This namespace contains functions for generating SplineSurfaces by approximation, etc.

### 28.2.2 Function Documentation

**28.2.2.1** `shared_ptr<SplineSurface> Go::AdaptSurface::adaptSurface ( shared_ptr< ParamSurface > surf, shared_ptr< SplineSurface > init_surf, double tol )`

Adapt the initial surface, `init_surf`, to the surface, `surf`, within the given tolerance, `tol`. The spline space of the initial surface may be refined

**28.2.2.2** `shared_ptr<SplineSurface> Go::AdaptSurface::approxInSplineSpace ( shared_ptr< ParamSurface > surf, shared_ptr< SplineSurface > surf2, double tol )`

Approximate `surf1` within the tolerance `tol` in the spline space of `surf2` or in a refinement of this spline space.

**28.2.2.3** `void Go::AdaptSurface::createTriangulation ( shared_ptr< ParamSurface > surf, const RectDomain & dom, shared_ptr< ftPointSet > & points, vector< int > & corner, bool consider_joint = true, int nmb = -1 )`

Fetch data and create triangulation.

**28.2.2.4** `std::vector<shared_ptr<SplineCurve>> Go::AdaptSurface::curveApprox ( shared_ptr< ParamCurve > cvs[], int nmb_cvs, const BsplineBasis & init_basis, double tol )`

Approximate a number of curves in the same spline space given an initial spline space that may be refined. The approximation tolerance is `tol`. This function is an interface to the function [CurveCreators::curveApprox](#) in `gotools-core` creators

**28.2.2.5** `std::vector<shared_ptr<SplineCurve>> Go::AdaptSurface::curveApprox ( shared_ptr< ParamCurve > cvs[], int nmb_cvs, double tol, double degree = 3 )`

Approximate a number of curves in the same spline space. The approximation tolerance is `tol` This function is an interface to the function [CurveCreators::curveApprox](#) in `gotools-core` creators

**28.2.2.6** `shared_ptr<SplineSurface> Go::AdaptSurface::doApprox ( shared_ptr< SplineSurface > init_surf, int max_iter, shared_ptr< ftPointSet > points, double tol, double & max_error, double & mean_error )`

Given an initial spline space and a point set, perform maximum `max_iter` iterations to approximate the points within the tolerance `tol`. The maximum and average error in the given points are returned.

**28.2.2.7** `std::vector<shared_ptr<SplineSurface>> Go::AdaptSurface::expressInSameSplineSpace ( shared_ptr< ParamSurface > surf1, shared_ptr< ParamSurface > surf2, double tol )`

Approximate the surfaces `surf1` and `surf2` in the same spline space with approximation errors less than `tol`

28.2.2.8 `void Go::AdaptSurface::getBoundaryData ( shared_ptr< ParamSurface > surf, const RectDomain & dom, int nmb_sample, shared_ptr< ftPointSet > points, std::vector< int > & corner )`

Fetch data points at the boundaries a the surface surf. To be used in surface approximation

28.2.2.9 `bool Go::AdaptSurface::getCornerCorrespondance ( shared_ptr< ParamSurface > surf1, shared_ptr< ParamSurface > surf2, int & idx, bool & turned )`

Compute the correspondance between corners in the surfaces surf1 and surf2. This function is used from approxInSplineSpace and expressInSameSplineSpace

28.2.2.10 `void Go::AdaptSurface::getInnerData ( shared_ptr< ParamSurface > surf, const RectDomain & dom, int nmb_sample, shared_ptr< ftPointSet > points, bool consider_joint = true )`

Fetch data points in the inner of the surface surf. To be used in surface approximation

28.2.2.11 `double Go::AdaptSurface::parameterizePoints ( shared_ptr< SplineSurface > init_surf, shared_ptr< ftPointSet > points, std::vector< int > corner )`

Parameterize the point set points with respect to the surface init\_surf. To be used in surface approximation.

28.2.2.12 `double Go::AdaptSurface::projectPoints ( shared_ptr< SplineSurface > surf, shared_ptr< ftPointSet > points )`

Parameterize by projection.

28.2.2.13 `void Go::AdaptSurface::updatePointTopology ( shared_ptr< ParamSurface > surf, ftPointSet & points )`

Compute point set topology.

## 28.3 Go::BoundedUtils Namespace Reference

### Functions

- `std::vector< shared_ptr< CurveOnSurface > > intersectWithSurface ( CurveOnSurface &curve, BoundedSurface &bounded_surf, double epsge)`
- `void intersectWithSurfaces (std::vector< shared_ptr< CurveOnSurface > > &curves1, shared_ptr< BoundedSurface > &bd_sf1, std::vector< shared_ptr< CurveOnSurface > > &curves2, shared_ptr< BoundedSurface > &bd_sf2, double epsge)`
- `std::vector< shared_ptr< CurveOnSurface > > getPlaneIntersections (const shared_ptr< ParamSurface > &surf, Point point, Point normal, double epsge, shared_ptr< BoundedSurface > &bounded_sf)`
- `std::vector< shared_ptr< CurveOnSurface > > getCylinderIntersections (const shared_ptr< ParamSurface > &surf, Point point, Point axis, double radius, double epsge, shared_ptr< BoundedSurface > &bounded_sf)`

- void [getSurfaceIntersections](#) (const shared\_ptr< ParamSurface > &surf1, const shared\_ptr< ParamSurface > &surf2, double epsge, std::vector< shared\_ptr< CurveOnSurface > > &int\_cv1, shared\_ptr< BoundedSurface > &bounded\_sf1, std::vector< shared\_ptr< CurveOnSurface > > &int\_cv2, shared\_ptr< BoundedSurface > &bounded\_sf2)
- std::vector< shared\_ptr< BoundedSurface > > [splitWithPlane](#) (const shared\_ptr< ParamSurface > &surf, Point point, Point normal, double epsge)
- std::vector< shared\_ptr< BoundedSurface > > [splitBetweenParams](#) (const shared\_ptr< ParamSurface > &surf, Point parval1, Point parval2, double epsge)
  - Split a parametric surface between specified parameter values.*
- std::vector< shared\_ptr< BoundedSurface > > [splitBetweenParPairs](#) (const shared\_ptr< ParamSurface > &surf, std::vector< std::pair< Point, Point > > parvals, double epsge)
- std::vector< shared\_ptr< CurveOnSurface > > [getTrimCrvsParam](#) (const shared\_ptr< ParamSurface > &surf, Point parval1, Point parval2, double epsge, shared\_ptr< BoundedSurface > &bounded\_sf)
  - Get the split curves between specified parameter values.*
- std::vector< shared\_ptr< CurveOnSurface > > [getTrimCrvsPcrv](#) (const shared\_ptr< ParamSurface > &surf, shared\_ptr< ParamCurve > &pcurve, double epsge, shared\_ptr< BoundedSurface > &bounded\_sf)
- std::vector< shared\_ptr< BoundedSurface > > [trimWithPlane](#) (const shared\_ptr< ParamSurface > &surf, Point point, Point normal, double epsge)
- std::vector< shared\_ptr< BoundedSurface > > [trimSurfWithSurf](#) (const shared\_ptr< ParamSurface > &sf1, const shared\_ptr< ParamSurface > &sf2, double epsge)
- std::vector< std::vector< shared\_ptr< BoundedSurface > > > [trimSurfsWithSurfs](#) (const std::vector< shared\_ptr< ParamSurface > > &sfs1, const std::vector< shared\_ptr< ParamSurface > > &sfs2, double epsge)
- BoundedSurface \* [convertToBoundedSurface](#) (const SplineSurface &surf, double space\_epsilon)
- std::vector< std::vector< shared\_ptr< CurveOnSurface > > > [getBoundaryLoops](#) (const BoundedSurface &sf, std::vector< shared\_ptr< CurveOnSurface > > &part\_bnd\_cvs, double eps, int last\_split=-1)
- int [checkCurveCoincidence](#) (shared\_ptr< CurveOnSurface > cv1, shared\_ptr< CurveOnSurface > cv2, double tol, bool same\_orient)
  - Help function for getBoundaryLoops.*
- std::vector< shared\_ptr< BoundedSurface > > [createTrimmedSurfs](#) (std::vector< std::vector< shared\_ptr< CurveOnSurface > > > &loops, shared\_ptr< ParamSurface > under\_sf, double epsgeo)
- std::vector< shared\_ptr< BoundedSurface > > [splitWithTrimSegments](#) (shared\_ptr< BoundedSurface > surf, std::vector< shared\_ptr< CurveOnSurface > > &bnd\_cvs, double eps)
- std::vector< shared\_ptr< BoundedSurface > > [subtractSfPart](#) (shared\_ptr< BoundedSurface > surf, std::vector< shared\_ptr< CurveOnSurface > > &bnd\_cvs, double eps)
- std::vector< shared\_ptr< CurveOnSurface > > [intersectWithPlane](#) (shared\_ptr< ParamSurface > &surf, Point pnt, Point normal, double geom\_tol)
- std::vector< shared\_ptr< CurveOnSurface > > [intersectWithCylinder](#) (shared\_ptr< ParamSurface > &surf, Point pnt, Point vec, double radius, double geom\_tol)
- void [getIntersectionCurve](#) (shared\_ptr< ParamSurface > &sf1, shared\_ptr< ParamSurface > &sf2, std::vector< shared\_ptr< CurveOnSurface > > &int\_segments1, std::vector< shared\_ptr< CurveOnSurface > > &int\_segments2, double epsge)
- void [translateBoundedSurf](#) (Point trans\_vec, BoundedSurface &bd\_sf, double deg\_eps)
- void [rotateBoundedSurf](#) (Point rot\_axis, double alpha, BoundedSurface &bf\_sf, double deg\_eps)
- void [trimSurfaceKinks](#) (const BoundedSurface &sf, double max\_normal\_angle, std::vector< double > &g1\_disc\_u, std::vector< double > &g1\_disc\_v, bool compute\_g1\_disc=true)
- int [checkAndFixLoopOrientation](#) (shared\_ptr< BoundedSurface > surf)
- shared\_ptr< Go::SplineSurface > [makeTrimmedPlane](#) (shared\_ptr< Go::Plane > &plane, std::vector< shared\_ptr< Go::ParamCurve > > &space\_crvs)
- void [translatePlaneToCurves](#) (shared\_ptr< Go::Plane > &plane, std::vector< shared\_ptr< Go::ParamCurve > > &space\_crvs)
- void [fixInvalidBoundedSurface](#) (shared\_ptr< Go::BoundedSurface > &bd\_sf, double max\_tol\_mult=1.0)
  - Fix the boundary loops of a surface in case of inconsistencies.*
- bool [loopsIsDegenerate](#) (std::vector< shared\_ptr< CurveOnSurface > > &loop, double epsgeo)
- bool [createMissingParCvs](#) (Go::BoundedSurface &bd\_sf)
- bool [createMissingParCvs](#) (std::vector< Go::CurveLoop > &bd\_loops)

### 28.3.1 Detailed Description

Functions related to the trimming of surfaces, etc. Also contains functions for spatial transformations of such surfaces (rotation, translation, etc.)

### 28.3.2 Function Documentation

28.3.2.1 `int Go::BoundedUtils::checkAndFixLoopOrientation ( shared_ptr< BoundedSurface > surf )`

Check loop orientation and fix if necessary. NB! It is assumed that the loops has got the correct sequence

#### Parameters

|             |                                             |
|-------------|---------------------------------------------|
| <i>surf</i> | the <a href="#">BoundedSurface</a> to check |
|-------------|---------------------------------------------|

#### Returns

`true` if the loop orientation was fixed, `false` otherwise

28.3.2.2 `int Go::BoundedUtils::checkCurveCoincidence ( shared_ptr< CurveOnSurface > cv1, shared_ptr< CurveOnSurface > cv2, double tol, bool same_orient )`

Help function for `getBoundaryLoops`.

28.3.2.3 `BoundedSurface* Go::BoundedUtils::convertToBoundedSurface ( const SplineSurface & surf, double space_epsilon )`

If `surf` already is a [BoundedSurface](#), return clone. If [SplineSurface](#), convert. Otherwise, Error. Return surface is created inside function. Convert a [SplineSurface](#) to a [BoundedSurface](#). All information is copied, nothing is shared.

#### Parameters

|                      |                                                                            |
|----------------------|----------------------------------------------------------------------------|
| <i>surf</i>          | the <a href="#">SplineSurface</a> to convert                               |
| <i>space_epsilon</i> | the tolerance assigned to the newly created <a href="#">BoundedSurface</a> |

#### Returns

(pointer to) a [BoundedSurface](#) that represent the same surface as 'surf'. The user assumes ownership of the object.

28.3.2.4 `bool Go::BoundedUtils::createMissingParCvs ( Go::BoundedSurface & bd_sf )`

28.3.2.5 `bool Go::BoundedUtils::createMissingParCvs ( std::vector< Go::CurveLoop > & bd_loops )`



28.3.2.6 `std::vector<shared_ptr<BoundedSurface>> Go::BoundedUtils::createTrimmedSurfs ( std::vector<std::vector<shared_ptr<CurveOnSurface>>> & loops, shared_ptr<ParamSurface> under_sf, double epsgeo )`

All input loops are expected to be simple, lying on surface. They are sorted based on orientation. No pair of curves with the same orientation may lie inside/outside each other. If they do the outer/inner (ccw/cw) loop(s) will be erased. Furthermore assuming no pair of loops intersect (may touch tangentially). Define the surfaces that result from trimming a given [SplineSurface](#) with a set of boundary loops. Counterclockwise loops define the interior of a parameter domain, while clockwise loops define holes in the domain. Method assumes all parameter curves exist.

#### Parameters

|                 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|-----------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>loops</i>    | each entry in the outermost vector represent a vector of curves that together specify a loop in 2D parametrical space of the surface 'under_sf'. These are the trim curves. The curves are expected to be simple, lying on the surface 'under_sf'. No pair of curve loops with the same orientation may lie inside/outside of each other; in that case, the irrelevant loop will be erased. Furthermore, we assume that no pair of loops intersect transversally (they are still allowed to touch tangentially). |
| <i>under_sf</i> | the underlying <a href="#">SplineSurface</a> that we are going to trim with the curves given in 'loops'.                                                                                                                                                                                                                                                                                                                                                                                                         |
| <i>epsgeo</i>   | geometrical tolerance used in computations                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |

#### Returns

a vector containing [BoundedSurface](#) s that each represent a trimmed part of the 'under\_sf' surface.

28.3.2.7 `void Go::BoundedUtils::fixInvalidBoundedSurface ( shared_ptr<Go::BoundedSurface> & bd_sf, double max_tol_mult = 1.0 )`

Fix the boundary loops of a surface in case of inconsistencies.

28.3.2.8 `std::vector<std::vector<shared_ptr<CurveOnSurface>>> Go::BoundedUtils::getBoundaryLoops ( const BoundedSurface & sf, std::vector<shared_ptr<CurveOnSurface>> & part_bnd_cvs, double eps, int last_split = -1 )`

Given input of partial boundary curves, extract parts of boundary making it a boundary loop (or more). Input segments expected to be ordered, going in the same direction. *part\_bnd\_cvs* should lie on *sf* (as opposed to only parts of *cvs*). This function tries to complete "partial" boundary loops by filling out the missing parts using fragments from the domain boundary of a [BoundedSurface](#).

#### Parameters

|                     |                                                                                                                                     |
|---------------------|-------------------------------------------------------------------------------------------------------------------------------------|
| <i>sf</i>           | the surface whose domain boundary will be used                                                                                      |
| <i>part_bnd_cvs</i> | a vector of (shared pointers to) curve segments that represent incomplete loops. Upon function return, this vector will be emptied. |

#### Returns

a vector contained the loops that the function was able to completely close using curve segments from '*part\_bnd\_cvs*' and the domain boundaries of '*sf*'.

28.3.2.9 `std::vector<shared_ptr<CurveOnSurface>> Go::BoundedUtils::getCylinderIntersections ( const shared_ptr< ParamSurface > & surf, Point point, Point axis, double radius, double epsge, shared_ptr< BoundedSurface > & bounded_sf )`

Intersect a parametric surface with a cylinder and fetch the intersections curves represented as curve on surface

28.3.2.10 `void Go::BoundedUtils::getIntersectionCurve ( shared_ptr< ParamSurface > & sf1, shared_ptr< ParamSurface > & sf2, std::vector< shared_ptr< CurveOnSurface >> & int_segments1, std::vector< shared_ptr< CurveOnSurface >> & int_segments2, double epsge )`

Find the intersction curve(s) between two spline surfaces

#### Parameters

|                  |                                 |
|------------------|---------------------------------|
| <code>sf1</code> | the first surface to intersect  |
| <code>sf2</code> | the second surface to intersect |

#### Return values

|                            |                                                                                                       |
|----------------------------|-------------------------------------------------------------------------------------------------------|
| <code>int_segments1</code> | a vector of <a href="#">CurveOnSurface</a> s, representing the intersection curves as lying on 'sf1'. |
| <code>int_segments2</code> | a vector of <a href="#">CurveOnSurface</a> s, representing the intersection curves as lying on 'sf2'. |

#### Parameters

|                    |                                                                |
|--------------------|----------------------------------------------------------------|
| <code>epsge</code> | geometrical tolerance to be used for intersection computations |
|--------------------|----------------------------------------------------------------|

28.3.2.11 `std::vector<shared_ptr<CurveOnSurface>> Go::BoundedUtils::getPlaneIntersections ( const shared_ptr< ParamSurface > & surf, Point point, Point normal, double epsge, shared_ptr< BoundedSurface > & bounded_sf )`

Intersect a parametric surface with a plane and fetch the intersections curves represented as curve on surface

28.3.2.12 `void Go::BoundedUtils::getSurfaceIntersections ( const shared_ptr< ParamSurface > & surf1, const shared_ptr< ParamSurface > & surf2, double epsge, std::vector< shared_ptr< CurveOnSurface >> & int_cv1, shared_ptr< BoundedSurface > & bounded_sf1, std::vector< shared_ptr< CurveOnSurface >> & int_cv2, shared_ptr< BoundedSurface > & bounded_sf2 )`

Intersect two parametric surfaces and fetch the resulting intersection curves represented as curve on surface

28.3.2.13 `std::vector<shared_ptr<CurveOnSurface>> Go::BoundedUtils::getTrimCrvsParam ( const shared_ptr< ParamSurface > & surf, Point parval1, Point parval2, double epsge, shared_ptr< BoundedSurface > & bounded_sf )`

Get the split curves between specified parameter values.

28.3.2.14 `std::vector<shared_ptr<CurveOnSurface>> Go::BoundedUtils::getTrimCrvsPcrv ( const shared_ptr<ParamSurface> & surf, shared_ptr<ParamCurve> & pcurve, double epsge, shared_ptr<BoundedSurface> & bounded_sf )`

28.3.2.15 `std::vector<shared_ptr<CurveOnSurface>> Go::BoundedUtils::intersectWithCylinder ( shared_ptr<ParamSurface> & surf, Point pnt, Point vec, double radius, double geom_tol )`

Find the intersection curve(s) between a [SplineSurface](#) and a given cylinder. The cylinder is defined by a point on the axis, the axis direction and the radius

#### Parameters

|                 |                                                                  |
|-----------------|------------------------------------------------------------------|
| <i>surf</i>     | the <a href="#">SplineSurface</a> to intersect with the cylinder |
| <i>pnt</i>      | a point lying on the cylinder axis                               |
| <i>vec</i>      | is the direction of the cylinder axis is the cylinder radius     |
| <i>geom_tol</i> | geometrical tolerance to be used for intersection computations   |

#### Returns

a vector with (shared pointers to) [CurveOnSurface](#) s, which represent the intersection curves found.

28.3.2.16 `std::vector<shared_ptr<CurveOnSurface>> Go::BoundedUtils::intersectWithPlane ( shared_ptr<ParamSurface> & surf, Point pnt, Point normal, double geom_tol )`

Find the intersection curve(s) between a [SplineSurface](#) and a given plane. The plane is defined by its normal and a point on the plane.

#### Parameters

|                 |                                                                |
|-----------------|----------------------------------------------------------------|
| <i>surf</i>     | the <a href="#">SplineSurface</a> to intersect with the plane  |
| <i>pnt</i>      | a point lying on the plane                                     |
| <i>normal</i>   | the normal of the plane                                        |
| <i>geom_tol</i> | geometrical tolerance to be used for intersection computations |

#### Returns

a vector with (shared pointers to) [CurveOnSurface](#) s, which represent the intersection curves found.

28.3.2.17 `std::vector<shared_ptr<CurveOnSurface>> Go::BoundedUtils::intersectWithSurface ( CurveOnSurface & curve, BoundedSurface & bounded_surf, double epsge )`

Extract those parts of a given [CurveOnSurface](#) where its parameter curve lies inside the parameter domain of a [BoundedSurface](#).

#### Parameters

|                     |                                                                                 |
|---------------------|---------------------------------------------------------------------------------|
| <i>curve</i>        | the <a href="#">CurveOnSurface</a> that we want to check                        |
| <i>bounded_surf</i> | the <a href="#">BoundedSurface</a> whose parameter domain we will check against |
| <i>epsge</i>        | geometric tolerance used in calculations                                        |

**Returns**

a vector containing those segments of 'curve' that have parameter descriptions inside the parameter domain of the 'bounded\_surf'.

```
28.3.2.18 void Go::BoundedUtils::intersectWithSurfaces (std::vector< shared_ptr< CurveOnSurface > > & curves1,
shared_ptr< BoundedSurface > & bd_sf1, std::vector< shared_ptr< CurveOnSurface > > & curves2,
shared_ptr< BoundedSurface > & bd_sf2, double epsge)
```

We have two set of [CurveOnSurface](#) s, 'curves1' and 'curves2', and two [BoundedSurface](#) s, 'bd\_sf1' and 'bd\_sf2'. We extract those segments of curves in 'curves1' and 'curves2' that have parameter curves in the parameter domains of respective 'bd\_sf1' and 'bd\_sf2'. Then we compare the resulting segments from the two sets against each others, and keep those that overlap spatially. Segments are split in two if necessary in order to make start and end points from one set coincide with those from the other set. 'curves1' and 'curves2' are then cleared and filled with the resulting curve segments.

**Parameters**

|                |                                                                                                                  |
|----------------|------------------------------------------------------------------------------------------------------------------|
| <i>curves1</i> | see above, we suppose that the curves are NOT self-intersecting                                                  |
| <i>bd_sf1</i>  | see above, we suppose that the underlying surface is the same as the one referred to by the curves in 'curves1'. |
| <i>curves2</i> | see above, we suppose that the curves are NOT self-intersecting                                                  |
| <i>bd_sf2</i>  | see above, we suppose that the underlying surface is the same as the one referred to by the curves in 'curves2'. |
| <i>epsge</i>   | geometric epsilon used in closest point calculations, checks for coincidence, etc.                               |

```
28.3.2.19 bool Go::BoundedUtils::loopsDegenerate (std::vector< shared_ptr< CurveOnSurface > > & loop, double
epsgeo)
```

Check if the distance between two curves in the loop is less than the tolerance epsgeo. Common endpoints between curves are not counted.

```
28.3.2.20 shared_ptr<Go::SplineSurface> Go::BoundedUtils::makeTrimmedPlane (shared_ptr< Go::Plane > & plane,
std::vector< shared_ptr< Go::ParamCurve > > & space_crvs)
```

Create spline surface description of plane. The plane should be bounded according to the space\_crvs (which should lie in the plane).

**Parameters**

|                   |                             |
|-------------------|-----------------------------|
| <i>plane</i>      |                             |
| <i>space_crvs</i> | Curves lying in the plane,. |

**Returns**

Spline surface description of a bounded plane.

28.3.2.21 void Go::BoundedUtils::rotateBoundedSurf ( Point *rot\_axis*, double *alpha*, BoundedSurface & *bf\_sf*, double *deg\_eps* )

## Parameters

|                 |                                                                  |
|-----------------|------------------------------------------------------------------|
| <i>rot_axis</i> | a vector specifying the axis of rotation                         |
| <i>alpha</i>    | the angle of rotation (in radians)                               |
| <i>bf_sf</i>    | the surface to rotate                                            |
| <i>deg_eps</i>  | an epsilon value used when determining degenerate boundary loops |

28.3.2.22 std::vector<shared\_ptr<BoundedSurface>> Go::BoundedUtils::splitBetweenParams ( const shared\_ptr<ParamSurface> & *surf*, Point *parval1*, Point *parval2*, double *epsge* )

Split a parametric surface between specified parameter values.

28.3.2.23 std::vector<shared\_ptr<BoundedSurface>> Go::BoundedUtils::splitBetweenParPairs ( const shared\_ptr<ParamSurface> & *surf*, std::vector<std::pair<Point, Point>> *parvals*, double *epsge* )

28.3.2.24 std::vector<shared\_ptr<BoundedSurface>> Go::BoundedUtils::splitWithPlane ( const shared\_ptr<ParamSurface> & *surf*, Point *point*, Point *normal*, double *epsge* )

Split a parametric surface by intersecting it with a plane and split along intersection curves

28.3.2.25 std::vector<shared\_ptr<BoundedSurface>> Go::BoundedUtils::splitWithTrimSegments ( shared\_ptr<BoundedSurface> & *surf*, std::vector<shared\_ptr<CurveOnSurface>> & *bnd\_cvs*, double *eps* )

Split a given bounded surface according to given trimming curves in this surface Notice that the function expects the given curves to actually split the surface

28.3.2.26 std::vector<shared\_ptr<BoundedSurface>> Go::BoundedUtils::subtractSfPart ( shared\_ptr<BoundedSurface> & *surf*, std::vector<shared\_ptr<CurveOnSurface>> & *bnd\_cvs*, double *eps* )

Subtract the part of a trimmed surface corresponding to a given boundary loop from the surface and return the remaining pieces

28.3.2.27 void Go::BoundedUtils::translateBoundedSurf ( Point *trans\_vec*, BoundedSurface & *bd\_sf*, double *deg\_eps* )

Translate a given [BoundedSurface](#)

## Parameters

|                  |                                                                  |
|------------------|------------------------------------------------------------------|
| <i>trans_vec</i> | the vector specifying the translation to apply to the surface    |
| <i>bd_sf</i>     | the surface to translate                                         |
| <i>deg_eps</i>   | an epsilon value used when determining degenerate boundary loops |

28.3.2.28 `void Go::BoundedUtils::translatePlaneToCurves ( shared_ptr< Go::Plane > & plane, std::vector< shared_ptr< Go::ParamCurve > > & space_crvs )`

Change the position of a plane to adapt it to curves supposed to lie in this plane. Geometry fix related to external models

28.3.2.29 `void Go::BoundedUtils::trimSurfaceKinks ( const BoundedSurface & sf, double max_normal_angle, std::vector< double > & g1_disc_u, std::vector< double > & g1_disc_v, bool compute_g1_disc = true )`

Surface assumed to be continuous. Return parameter values failing to achieve G1-continuity.

28.3.2.30 `std::vector<std::vector<shared_ptr<BoundedSurface>>> Go::BoundedUtils::trimSurfsWithSurfs ( const std::vector< shared_ptr< ParamSurface > > & sfs1, const std::vector< shared_ptr< ParamSurface > > & sfs2, double epsge )`

28.3.2.31 `std::vector<shared_ptr<BoundedSurface>> Go::BoundedUtils::trimSurfWithSurf ( const shared_ptr< ParamSurface > & sf1, const shared_ptr< ParamSurface > & sf2, double epsge )`

must be BoundedSurfaces or [SplineSurface](#) underlying surfaces must be of type [SplineSurface](#) If the argument surfaces intersect, and if the intersection curves result in new boundary loops being defined, then the new surface parts defined within these domains will be returned. Otherwise, the return vector will be empty.

#### Parameters

|              |                                                       |
|--------------|-------------------------------------------------------|
| <i>sf1</i>   | the first surface to participate in the intersection  |
| <i>sf2</i>   | the second surface to participate in the intersection |
| <i>epsge</i> | geometrical tolerance to be used in computations      |

#### Returns

a vector with the surfaces representing the parts of the original surfaces enclosed by new parametrical loops arising when combining existing loops with the curves defined by the intersection.

28.3.2.32 `std::vector<shared_ptr<BoundedSurface>> Go::BoundedUtils::trimWithPlane ( const shared_ptr< ParamSurface > & surf, Point point, Point normal, double epsge )`

We intersect a parametric surface with a plane, and return the surface(s) consisting only of the part(s) of the surface that were located on the positive side of the intersection. If there was no intersection, an empty stl-vector is returned. The plane is defined by its normal and a point located on it.

#### Parameters

|               |                                                                                                                 |
|---------------|-----------------------------------------------------------------------------------------------------------------|
| <i>surf</i>   | the parametric surface. It must be either a <a href="#">BoundedSurface</a> or a <a href="#">SplineSurface</a> . |
| <i>point</i>  | a point on the plane to intersect against                                                                       |
| <i>normal</i> | normal of the plane to intersect against                                                                        |
| <i>epsge</i>  | geometric tolerance                                                                                             |

**Returns**

the surface(s) consisting of the part(s) of 'surf' that were located on the positive side of the intersection.

## 28.4 Go::boxStructuring Namespace Reference

**Classes**

- class [BoundingBoxStructure](#)
- class [SubSurfaceBoundingBox](#)
- class [SurfaceData](#)

### 28.4.1 Detailed Description

Namespace for preprocessing structures on a surface model used to speed up closest point calculations. It consists of three classes:

- [BoundingBoxStructure](#), the overall class where one instance holds the entire preprocess structure data
- [SurfaceData](#), one instance for each surface in the surface model
- [SubSurfaceBoundingBox](#), holding preprocessed data for one specific rectangular segment in a tensor mesh sub structure of the parameter space on a surface (the underlying surface in case of a [BoundedSurface](#))

## 28.5 Go::ClosestPoint Namespace Reference

Namespace for computing closest points.

**Functions**

- void [closestPtCurves2D](#) ([ParamCurve](#) \*cv1, [ParamCurve](#) \*cv2, [double](#) aepsge, [double](#) tmin1, [double](#) tmax1, [double](#) tmin2, [double](#) tmax2, [double](#) seed1, [double](#) seed2, int method, [bool](#) quick, [double](#) &par1, [double](#) &par2, [double](#) &dist, [Point](#) &ptc1, [Point](#) &ptc2, int &istat)
- void [closestPtCurves](#) ([const ParamCurve](#) \*cv1, [const ParamCurve](#) \*cv2, [double](#) &par1, [double](#) &par2, [double](#) &dist, [Point](#) &ptc1, [Point](#) &ptc2)
- void [closestPtCurves](#) ([const ParamCurve](#) \*cv1, [const ParamCurve](#) \*cv2, [double](#) tmin1, [double](#) tmax1, [double](#) tmin2, [double](#) tmax2, [double](#) seed1, [double](#) seed2, [double](#) &par1, [double](#) &par2, [double](#) &dist, [Point](#) &ptc1, [Point](#) &ptc2)
- void [closestPtCurveSurf](#) ([ParamCurve](#) \*pcurve, [ParamSurface](#) \*psurf, [double](#) aepsge, [double](#) astart1, [double](#) aend1, [RectDomain](#) \*domain, [double](#) anext1, [double](#) enext2[], [double](#) &cpos1, [double](#) gpos2[], [double](#) &dist, [Point](#) &pt\_cv, [Point](#) &pt\_su, [bool](#) second\_order=false)
- void [closestPtCurveSurf](#) ([ParamCurve](#) \*pcurve, [ParamSurface](#) \*psurf, [double](#) aepsge, [double](#) astart1, [double](#) estart2[], [double](#) aend1, [double](#) eend2[], [double](#) anext1, [double](#) enext2[], [double](#) &cpos1, [double](#) gpos2[], [double](#) &dist, [Point](#) &pt\_cv, [Point](#) &pt\_su, int &istat, [bool](#) second\_order=false)
- void [closestPtSurfSurfPlane](#) ([const std::vector< Point >](#) &epoint, [const std::vector< Point >](#) &epnt1, [const std::vector< Point >](#) &epnt2, [const Point](#) &epar1, [const Point](#) &epar2, [const ParamSurface](#) \*psurf1, [const ParamSurface](#) \*psurf2, [double](#) aepsge, [std::vector< Point >](#) &gpnt1, [std::vector< Point >](#) &gpnt2, [Point](#) &gpar1, [Point](#) &gpar2, int &jstat, [AlgorithmChoice](#) algo=FUNCTIONAL)
- void [closestPtSurfSurfPlaneGeometrical](#) ([const std::vector< Point >](#) &epoint, [const std::vector< Point >](#) &epnt1, [const std::vector< Point >](#) &epnt2, [const Point](#) &epar1, [const Point](#) &epar2, [const ParamSurface](#) \*psurf1, [const ParamSurface](#) \*psurf2, [double](#) aepsge, [std::vector< Point >](#) &gpnt1, [std::vector< Point >](#) &gpnt2, [Point](#) &gpar1, [Point](#) &gpar2, int &jstat)
- void [closestPtSurfSurfPlaneFunctional](#) ([const std::vector< Point >](#) &epoint, [const std::vector< Point >](#) &epnt1, [const std::vector< Point >](#) &epnt2, [const Point](#) &epar1, [const Point](#) &epar2, [const ParamSurface](#) \*psurf1, [const ParamSurface](#) \*psurf2, [double](#) aepsge, [std::vector< Point >](#) &gpnt1, [std::vector< Point >](#) &gpnt2, [Point](#) &gpar1, [Point](#) &gpar2, int &jstat)

## 28.5.1 Detailed Description

Namespace for computing closest points.

## 28.5.2 Function Documentation

**28.5.2.1** void Go::ClosestPoint::closestPtCurves ( const ParamCurve \* *cv1*, const ParamCurve \* *cv2*, double & *par1*, double & *par2*, double & *dist*, Point & *ptc1*, Point & *ptc2* )

Compute a closest point or an intersection point between two curves. Ported from the sisl-functions s1770, s1770↔\_s9corr, s1770\_s9dir

### Parameters

|            |                   |
|------------|-------------------|
| <i>cv1</i> | Curve number one. |
| <i>cv2</i> | Curve number two. |

### Return values

|             |                                              |
|-------------|----------------------------------------------|
| <i>par1</i> | Parameter value of the first curve's point.  |
| <i>par2</i> | Parameter value of the second curve's point. |
| <i>dist</i> | Distance between the points.                 |
| <i>ptc1</i> | <a href="#">Point</a> on curve number one.   |
| <i>ptc2</i> | <a href="#">Point</a> on curve number two.   |

**28.5.2.2** void Go::ClosestPoint::closestPtCurves ( const ParamCurve \* *cv1*, const ParamCurve \* *cv2*, double *tmin1*, double *tmax1*, double *tmin2*, double *tmax2*, double *seed1*, double *seed2*, double & *par1*, double & *par2*, double & *dist*, Point & *ptc1*, Point & *ptc2* )

Newton iteration on the distance function between two curves to find a closest point or an intersection point. The solution is restricted to the intervals [*tmin1*,*tmax1*] and [*tmin2*,*tmax2*] in the first and second curves's parameter domain. Ported from the sisl function s1770.

### Parameters

|              |                                                   |
|--------------|---------------------------------------------------|
| <i>cv1</i>   | Curve number one.                                 |
| <i>cv2</i>   | Curve number two.                                 |
| <i>tmin1</i> | Min. parameter value. First curve                 |
| <i>tmax1</i> | Max. parameter value. First curve                 |
| <i>tmin2</i> | Min. parameter value. Second curve                |
| <i>tmax2</i> | Max. parameter value. Second curve                |
| <i>seed1</i> | Start point for iteration along curve number one. |
| <i>seed2</i> | Start point for iteration along curve number two. |

### Return values

|             |                                             |
|-------------|---------------------------------------------|
| <i>par1</i> | Parameter value of the first curve's point. |
|-------------|---------------------------------------------|



## Return values

|             |                                              |
|-------------|----------------------------------------------|
| <i>par2</i> | Parameter value of the second curve's point. |
| <i>dist</i> | Distance between the points.                 |
| <i>ptc1</i> | <a href="#">Point</a> on curve number one.   |
| <i>ptc2</i> | <a href="#">Point</a> on curve number two.   |

28.5.2.3 void Go::ClosestPoint::closestPtCurves2D ( ParamCurve \* *cv1*, ParamCurve \* *cv2*, double *aepsge*, double *tmin1*, double *tmax1*, double *tmin2*, double *tmax2*, double *seed1*, double *seed2*, int *method*, bool *quick*, double & *par1*, double & *par2*, double & *dist*, Point & *ptc1*, Point & *ptc2*, int & *istat* )

Newton iteration on the distance function between two curves in 2D to find a closest point or an intersection point. The solution is restricted to the intervals [*tmin1*,*tmax1*] and [*tmin2*,*tmax2*] in the first and second curves's parameter domain. Ported from the sisl-function `s1770_2D`.

## Parameters

|               |                                                                                                                                                                    |
|---------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>cv1</i>    | Curve number one.                                                                                                                                                  |
| <i>cv2</i>    | Curve number two.                                                                                                                                                  |
| <i>aepsge</i> | Geometry resolution.                                                                                                                                               |
| <i>tmin1</i>  | Min. parameter value. First curve                                                                                                                                  |
| <i>tmax1</i>  | Max. parameter value. First curve                                                                                                                                  |
| <i>tmin2</i>  | Min. parameter value. Second curve                                                                                                                                 |
| <i>tmax2</i>  | Max. parameter value. Second curve                                                                                                                                 |
| <i>seed1</i>  | Start point for iteration along curve number one.                                                                                                                  |
| <i>seed2</i>  | Start point for iteration along curve number two.                                                                                                                  |
| <i>method</i> | 1 or 2. Using the the method in <code>s1770_2D</code> , starting with order one or two. <code>method=3</code> uses the method from <code>s1770</code> (order one). |
| <i>quick</i>  | Reduce requirement on exactness.                                                                                                                                   |

## Return values

|              |                                                      |
|--------------|------------------------------------------------------|
| <i>par1</i>  | Parameter value of the first curve's point.          |
| <i>par2</i>  | Parameter value of the second curve's point.         |
| <i>dist</i>  | Distance in space between the points.                |
| <i>ptc1</i>  | <a href="#">Point</a> on curve number one.           |
| <i>ptc2</i>  | <a href="#">Point</a> on curve number two.           |
| <i>istat</i> | =1 Intersection. =2 Minimum value. =3 Nothing found. |

28.5.2.4 void Go::ClosestPoint::closestPtCurveSurf ( ParamCurve \* *pcurve*, ParamSurface \* *psurf*, double *aepsge*, double *astart1*, double *aend1*, RectDomain \* *domain*, double *anext1*, double *enext2*[], double & *cpos1*, double *gpos2*[], double & *dist*, Point & *pt\_cv*, Point & *pt\_su*, bool *second\_order* = false )

Newton iteration on the distance function between a curve and a surface to find a closest point or an intersection point. The solution is restricted to the interval [*astart1*,*aend1*] in the curves's parameter domain and [*estart2*[],*eend2*[][]] in the surface's parameter domain. Ported from the sisl-function `s1772`.

## Parameters

|                     |                                                                     |
|---------------------|---------------------------------------------------------------------|
| <i>pcurve</i>       | Pointer to the curve in the intersection.                           |
| <i>psurf</i>        | Pointer to the surface in the intersection.                         |
| <i>aepsge</i>       | Geometry resolution.                                                |
| <i>astart1</i>      | Start parameter value of the curve.                                 |
| <i>aend1</i>        | End parameter value of the curve.                                   |
| <i>domain</i>       | the parametric domain of interest of the surface                    |
| <i>anext1</i>       | Start point for iteration along curve number one.                   |
| <i>enext2[]</i>     | Start parameter values of the iteration on the surface.             |
| <i>second_order</i> | if 'true' the function will attempt a second-order method directly. |

## Return values

|                |                                                            |
|----------------|------------------------------------------------------------|
| <i>cpos1</i>   | Parameter value of the curve in the intersection point.    |
| <i>gpos2[]</i> | Parameter values of the surface in the intersection point. |
| <i>dist</i>    | Distance between the points.                               |
| <i>pt_cv</i>   | The point on the curve.                                    |
| <i>pt_su</i>   | The point on the surface.                                  |

28.5.2.5 void Go::ClosestPoint::closestPtCurveSurf ( ParamCurve \* *pcurve*, ParamSurface \* *psurf*, double *aepsge*, double *astart1*, double *estart2[]*, double *aend1*, double *eend2[]*, double *anext1*, double *enext2[]*, double & *cpos1*, double *gpos2[]*, double & *dist*, Point & *pt\_cv*, Point & *pt\_su*, int & *istat*, bool *second\_order* = false )

Newton iteration on the distance function between a curve and a surface to find a closest point or an intersection point. The solution is restricted to the interval [*astart1*,*aend1*] in the curves's parameter domain and [*estart2[]*,*eend2[]*] in the surface's parameter domain. Ported from the sisl-function `s1772`.

## Parameters

|                     |                                                                     |
|---------------------|---------------------------------------------------------------------|
| <i>pcurve</i>       | Pointer to the curve in the intersection.                           |
| <i>psurf</i>        | Pointer to the surface in the intersection.                         |
| <i>aepsge</i>       | Geometry resolution.                                                |
| <i>astart1</i>      | Start parameter value of the curve.                                 |
| <i>estart2[]</i>    | Start parameter values of surface.                                  |
| <i>aend1</i>        | End parameter value of the curve.                                   |
| <i>eend2[]</i>      | End parameter values of the surface.                                |
| <i>anext1</i>       | Start point for iteration along curve number one.                   |
| <i>enext2[]</i>     | Start parameter values of the iteration on the surface.             |
| <i>second_order</i> | if 'true' the function will attempt a second-order method directly. |

## Return values

|                |                                                            |
|----------------|------------------------------------------------------------|
| <i>cpos1</i>   | Parameter value of the curve in the intersection point.    |
| <i>gpos2[]</i> | Parameter values of the surface in the intersection point. |
| <i>dist</i>    | Distance between the points.                               |
| <i>pt_cv</i>   | The point on the curve.                                    |
| <i>pt_su</i>   | The point on the surface.                                  |

## Return values

|              |                                                      |
|--------------|------------------------------------------------------|
| <i>istat</i> | =1 Intersection. =2 Minimum value. =3 Nothing found. |
|--------------|------------------------------------------------------|

28.5.2.6 void Go::ClosestPoint::closestPtSurfSurfPlane ( const std::vector< Point > & *epoint*, const std::vector< Point > & *epnt1*, const std::vector< Point > & *epnt2*, const Point & *epar1*, const Point & *epar2*, const ParamSurface \* *psurf1*, const ParamSurface \* *psurf2*, double *aepsge*, std::vector< Point > & *gpnt1*, std::vector< Point > & *gpnt2*, Point & *gpar1*, Point & *gpar2*, int & *jstat*, AlgorithmChoice *algo* = FUNCTIONAL )

Newton iteration on the distance function between two surfaces and a plane to find a closest point or an intersection point. Ported from the sisl-function `s9iterate`.

## Parameters

|               |                                                                                                                                                                       |
|---------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>epoint</i> | - Vector of Points containing parts of plane description. <i>epoint</i> [0] contains a point in the plane. <i>epoint</i> [1] contains the normal vector to the plane. |
| <i>epnt1</i>  | 0-2 Derivatives + normal of start point for iteration in first surface.                                                                                               |
| <i>epnt2</i>  | 0-2 Derivatives + normal of start point for iteration in second surface.                                                                                              |
| <i>epar1</i>  | Parameter pair of start point in first surface.                                                                                                                       |
| <i>epar2</i>  | Parameter pair of start point in second surface.                                                                                                                      |
| <i>psurf1</i> | Pointer to the first surface.                                                                                                                                         |
| <i>psurf2</i> | Pointer to the second surface.                                                                                                                                        |
| <i>aepsge</i> | Absolute tolerance.                                                                                                                                                   |
| <i>gpnt1</i>  | 0-2 Derivatives + normal of result of iteration in first surface.                                                                                                     |
| <i>gpnt2</i>  | 0-2 Derivatives + normal of result of iteration in second surface.                                                                                                    |
| <i>gpar1</i>  | Parameter pair of result of iteration in first surface.                                                                                                               |
| <i>gpar2</i>  | Parameter pair of result of iteration in second surface.                                                                                                              |
| <i>jstat</i>  | =1 Intersection. =2 Minimum value. =3 Nothing found.                                                                                                                  |
| <i>algo</i>   | choose whether the implementation of the algorithm should be based on general function minimization, or a geometric approach                                          |

28.5.2.7 void Go::ClosestPoint::closestPtSurfSurfPlaneFunctional ( const std::vector< Point > & *epoint*, const std::vector< Point > & *epnt1*, const std::vector< Point > & *epnt2*, const Point & *epar1*, const Point & *epar2*, const ParamSurface \* *psurf1*, const ParamSurface \* *psurf2*, double *aepsge*, std::vector< Point > & *gpnt1*, std::vector< Point > & *gpnt2*, Point & *gpar1*, Point & *gpar2*, int & *jstat* )

This is the functional-based implementation of `closestPtSurfSurfPlane`. It is called from the function above (`closestPtSurfSurfPlane()`) if it is called with `algo = FUNCTIONAL`. The user can also choose to call this function directly. The parameter description is the same as in `closestPtSurfSurfPlane()`.

28.5.2.8 void Go::ClosestPoint::closestPtSurfSurfPlaneGeometrical ( const std::vector< Point > & *epoint*, const std::vector< Point > & *epnt1*, const std::vector< Point > & *epnt2*, const Point & *epar1*, const Point & *epar2*, const ParamSurface \* *psurf1*, const ParamSurface \* *psurf2*, double *aepsge*, std::vector< Point > & *gpnt1*, std::vector< Point > & *gpnt2*, Point & *gpar1*, Point & *gpar2*, int & *jstat* )

This is the geometrically-based implementation of `closestPtSurfSurfPlane`. It is called from the function above (`closestPtSurfSurfPlane()`) if it is called with `algo = GEOMETRICAL`. The user can also choose to call this function directly. The parameter description is the same as in `closestPtSurfSurfPlane()`.

## 28.6 Go::cmUtils Namespace Reference

Various utility functions for the compositemodel module.

### Functions

- [double estimatedCurveLength](#) ([ftEdgeBase](#) \*edge, int nmb\_samples=4)  
*Estimate the curve length corresponding to an edge.*
- [RectDomain geometricParamDomain](#) ([ParamSurface](#) \*sf)  
*Domain of surface is rescaled according to surface lengths in geometry space.*
- [bool abovePlane](#) ([Go::Point](#) pt, [Go::Point](#) plane\_pt, [Go::Point](#) normal)  
*Return true if pt lies above plane as defined.*
- void [extendWithDegBd](#) (std::vector< int > &corner, std::vector< shared\_ptr< [Go::ParamCurve](#) > > &bd\_↔  
curves, std::vector< shared\_ptr< [Go::ParamCurve](#) > > &cross\_curves, int idxmin)
- void [updateWithNewCorners](#) (const std::vector< [Go::ftEdgeBase](#) \* > &outer\_loop, std::vector< int > &cor-  
ners, const std::vector< [Go::Point](#) > &add\_corner\_pts, double gap)  
*Assuming input corner pts lie inside gap of outer\_loop, and not in the middle of a segment.*
- [bool insideBdTrain](#) (const [Go::Vector2D](#) &sample\_pt, const std::vector< [Go::Vector2D](#) > &bd\_train)  
*bd\_train must be a simple loop, direction may be either.*
- void [reparametrizeBdCvs](#) (const std::vector< shared\_ptr< [Go::SplineCurve](#) > > &bd\_cvs, double appr\_tol,  
std::vector< shared\_ptr< [Go::SplineCurve](#) > > &new\_bd\_cvs)
- void [reparametrizeBdCvs2](#) (const std::vector< shared\_ptr< [Go::SplineCurve](#) > > &bd\_cvs, double appr\_tol,  
std::vector< shared\_ptr< [Go::SplineCurve](#) > > &new\_bd\_cvs)
- void [splitEdgesInCorners](#) (std::vector< shared\_ptr< [Go::ftEdgeBase](#) > > &edges, const std::vector< [Go](#)↔  
::Point > &corner\_pts, double gap)  
*interior of an edge. To be called prior to connection with twin.*
- void [cwOrientation](#) (std::vector< [Go::ftEdgeBase](#) \* > &meeting\_edges, std::vector< bool > &start, double  
angle\_tol=1e-05)
- void [cwOrientation2](#) (std::vector< [Go::ftEdgeBase](#) \* > &meeting\_edges, std::vector< bool > &start)
- std::vector< std::pair< shared\_ptr< [Go::ParamCurve](#) >, [Go::ftFaceBase](#) \* > > [getG1FaceCurves](#) ([Go::ft](#)↔  
[Curve](#) &bd\_curve)  
*Divide the input curve into G1 segments lying on one surface only (input expected to lie on bd).*
- [double ccwAngle](#) ([Go::Point](#) first\_leg, [Go::Point](#) second\_leg, [Go::Point](#) \*normal=NULL)
- std::vector< int > [removeInnerCorners](#) (const std::vector< [Go::ftEdgeBase](#) \* > &outer\_loop, std::vector<  
int > &corners)

### 28.6.1 Detailed Description

Various utility functions for the compositemodel module.

### 28.6.2 Function Documentation

#### 28.6.2.1 bool Go::cmUtils::abovePlane ( Go::Point pt, Go::Point plane\_pt, Go::Point normal )

Return true if pt lies above plane as defined.

28.6.2.2 `double Go::cmUtils::ccwAngle ( Go::Point first_leg, Go::Point second_leg, Go::Point * normal = NULL )`

Return angle from first\_leg to second\_leg, in ccw direction. Dimension is 2 or 3. For dimension 3 the input normal is used to define plane. Returned angle lies in  $[0, 2\pi)$ . normal assumed to exist if dim of legs equals 3, need not be normal to legs. Only defines ccw.

28.6.2.3 `void Go::cmUtils::cwOrientation ( std::vector< Go::ftEdgeBase * > & meeting_edges, std::vector< bool > & start, double angle_tol = 1e-05 )`

Given input of edges, meeting in a common vertex (as given by start), make sure that the edges after the first element is sorted based on angle, in cw direction. In addition we also make sure that edges with common sf\_id are given in sequence. Note: Expecting that all sfs have consistent orientation (which they'll have if topology has been built)!!!

28.6.2.4 `void Go::cmUtils::cwOrientation2 ( std::vector< Go::ftEdgeBase * > & meeting_edges, std::vector< bool > & start )`

Method does not compute tangents and angles but is based on next\_, prev\_ & twin\_. We're assuming that at most two input edges are without twin.

28.6.2.5 `double Go::cmUtils::estimatedCurveLength ( ftEdgeBase * edge, int nmb_samples = 4 )`

Estimate the curve length corresponding to an edge.

28.6.2.6 `void Go::cmUtils::extendWithDegBd ( std::vector< int > & corner, std::vector< shared_ptr< Go::ParamCurve > > & bd_curves, std::vector< shared_ptr< Go::ParamCurve > > & cross_curves, int idxmin )`

Extend the set of boundary curves with degenerate curves if a degenerate surface is to be created. Handles 2 or 3 bnd\_curves.

28.6.2.7 `RectDomain Go::cmUtils::geometricParamDomain ( ParamSurface * sf )`

**Domain** of surface is rescaled according to surface lengths in geometry space.

28.6.2.8 `std::vector<std::pair<shared_ptr<Go::ParamCurve>, Go::ftFaceBase*> > Go::cmUtils::getG1FaceCurves ( Go::ftCurve & bd_curve )`

Divide the input curve into G1 segments lying on one surface only (input expected to lie on bd).

28.6.2.9 `bool Go::cmUtils::insideBdTrain ( const Go::Vector2D & sample_pt, const std::vector< Go::Vector2D > & bd_train )`

bd\_train must be a simple loop, direction may be either.

We decide whether sample\_pt lies inside polygonal domain defined by bd\_train.

- 28.6.2.10 `std::vector<int> Go::cmUtils::removeInnerCorners ( const std::vector< Go::ftEdgeBase * > & outer_loop, std::vector< int > & corners )`
- 28.6.2.11 `void Go::cmUtils::reparametrizeBdCvs ( const std::vector< shared_ptr< Go::SplineCurve > > & bd_cvs, double appr_tol, std::vector< shared_ptr< Go::SplineCurve > > & new_bd_cvs )`
- 28.6.2.12 `void Go::cmUtils::reparametrizeBdCvs2 ( const std::vector< shared_ptr< Go::SplineCurve > > & bd_cvs, double appr_tol, std::vector< shared_ptr< Go::SplineCurve > > & new_bd_cvs )`
- 28.6.2.13 `void Go::cmUtils::splitEdgesInCorners ( std::vector< shared_ptr< Go::ftEdgeBase > > & edges, const std::vector< Go::Point > & corner_pts, double gap )`

interior of an edge. To be called prior to connection with twin.

We make sure that all `corner_pts` (within gap from an edge) do not lie in the

- 28.6.2.14 `void Go::cmUtils::updateWithNewCorners ( const std::vector< Go::ftEdgeBase * > & outer_loop, std::vector< int > & corners, const std::vector< Go::Point > & add_corner_pts, double gap )`

Assuming input `corner_pts` lie inside gap of `outer_loop`, and not in the middle of a segment.

## 28.7 Go::CoonsPatchGen Namespace Reference

### Classes

- class [UnknownError](#)  
*Exception class.*

### Functions

- [SplineSurface](#) \* [createCoonsPatch](#) (const [CurveLoop](#) &boundary)
- [SplineSurface](#) \* [createCoonsPatch](#) (std::vector< shared\_ptr< [ParamCurve](#) > > &bd\_curves, std::vector< shared\_ptr< [ParamCurve](#) > > &cross\_curves, double [epsge](#), double [kink\\_tol](#))
- [SplineSurface](#) \* [createCoonsPatch](#) (std::vector< shared\_ptr< [SplineCurve](#) > > &bd\_curves, std::vector< shared\_ptr< [SplineCurve](#) > > &cross\_curves)
- [SplineSurface](#) \* [createGordonSurface](#) (std::vector< shared\_ptr< [SplineCurve](#) > > &mesh\_curves, std::vector< double > &params, int &nmb\_u\_crvs, bool [use\\_param\\_values](#))
- [SplineSurface](#) \* [createGordonSurface](#) (std::vector< shared\_ptr< [SplineCurve](#) > > &mesh\_curves, std::vector< double > &params, int &nmb\_u\_crvs, std::vector< shared\_ptr< [SplineCurve](#) > > &cross\_curves, std::vector< int > &cross\_index, bool [use\\_param\\_values=true](#))
- [SplineSurface](#) \* [doCreateSurface](#) (std::vector< shared\_ptr< [SplineCurve](#) > > &mesh\_curves, std::vector< double > &params, int &nmb\_u\_crvs, std::vector< shared\_ptr< [SplineCurve](#) > > &cross\_curves, std::vector< int > &cross\_index)
- [SplineSurface](#) \* [loftSurface](#) (std::vector< shared\_ptr< [SplineCurve](#) > >::iterator first\_curve, int nmb\_crvs)
- [SplineSurface](#) \* [loftSurface](#) (std::vector< shared\_ptr< [SplineCurve](#) > >::iterator first\_curve, std::vector< double >::iterator first\_param, int nmb\_crvs)
- [SplineSurface](#) \* [loftSurface](#) (std::vector< shared\_ptr< [SplineCurve](#) > >::iterator first\_curve, std::vector< double >::iterator first\_param, int nmb\_crvs, std::vector< shared\_ptr< [SplineCurve](#) > >::iterator first\_cross\_curve, std::vector< int > &cross\_index)

- `SplineSurface * tpSurface (const std::vector< shared_ptr< SplineCurve > > &mesh_curves, std::vector< double > params, int nmb_u_crvs, const std::vector< shared_ptr< SplineCurve > > &cross_curves, std::vector< int > &cross_index)`
- `void splitMeshCurves (std::vector< shared_ptr< SplineCurve > > &mesh_curves, std::vector< double > &params, int &nmb_u_crvs, std::vector< int > &cross_index, double epsgeo)`
- `void sortMeshCurves (std::vector< shared_ptr< SplineCurve > > &mesh_curves, std::vector< double > &params, int nmb_u_crvs, std::vector< int > &cross_index)`
- `void getCrossTangs (const std::vector< shared_ptr< SplineCurve > > &curves, std::vector< shared_ptr< SplineCurve > > &mod_cross_curves, double tol1, double tol2)`
- `void addMissingCrossCurves (const std::vector< shared_ptr< SplineCurve > > &bnd_curves, std::vector< shared_ptr< SplineCurve > > &cross_crvs)`
- `void getTangBlends (std::vector< shared_ptr< SplineCurve > > &curves, int iedge, std::vector< shared_ptr< SplineCurve > > &blend_functions)`
- `void blendcoef (double evecu[], double evecv[], double etang[], int idim, int isign, double *coef1, double *coef2)`
- `void hermit (double econd[], int icond, bool hasder1, double astart, double aend, int idim)`
- `void fixCrossEndPts (const std::vector< shared_ptr< SplineCurve > > &bd_curves, const std::vector< shared_ptr< SplineCurve > > &cross_curves)`
- `void makeLoftParams (std::vector< shared_ptr< SplineCurve > >::const_iterator first_curve, int nmb_crvs, double param_length, std::vector< double > &params)`
- `void reparamBoundaryCurve (std::vector< shared_ptr< SplineCurve > > &curves, double aconst)`

### 28.7.1 Detailed Description

This namespace contains functions used to create a Coons Patch or a Gordon Surface.

### 28.7.2 Function Documentation

**28.7.2.1** `void Go::CoonsPatchGen::addMissingCrossCurves ( const std::vector< shared_ptr< SplineCurve > > & bnd_curves, std::vector< shared_ptr< SplineCurve > > & cross_crvs )`

Generates missing cross boundary curves. In case of missing cross\_crv, we're expecting a NULL pointer. boundary\_crvs form a loop. cross\_crvs share spline space, and all cross curves point inwards = into the surface. A missing cross\_curve is indicated by a null pointer. The added cross curve will also point inwards, following parametrization given by boundary curves.

**28.7.2.2** `void Go::CoonsPatchGen::blendcoef ( double evecu[], double evecv[], double etang[], int idim, int isign, double * coef1, double * coef2 )`

Project the etang vector onto the plane defined by evecu and evecv. The projected vector will be represented as  $(*coef1)*evecu + (*coef2)*evecv$ .

#### Parameters

|              |                                           |
|--------------|-------------------------------------------|
| <i>evecu</i> | one of the vectors spanning the plane.    |
| <i>evecv</i> | one of the vectors spanning the plane.    |
| <i>etang</i> | vector to be projected.                   |
| <i>idim</i>  | the geometric dimension of the space.     |
| <i>isign</i> | sign with wich etang is to be multiplied. |
| <i>coef1</i> | scalar corresponding to evecu.            |
| <i>coef2</i> | scalar corresponding to evecv.            |

### 28.7.2.3 `SplineSurface* Go::CoonsPatchGen::createCoonsPatch ( const CurveLoop & boundary )`

Create a new [SplineSurface](#) representing the coons patch as defined by a loop of four [SplineCurves](#).

#### Parameters

|                 |                                                                                                                                                                                                                      |
|-----------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>boundary</i> | the <a href="#">CurveLoop</a> describing the boundary of the coons patch to be generated. There must be exactly four curves, they must all be of type ' <a href="#">SplineCurve</a> ', and they must be nonrational. |
|-----------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

#### Returns

a pointer to a newly created [SplineSurface](#) representing the coons patch. The user assumes ownership of the object.

### 28.7.2.4 `SplineSurface* Go::CoonsPatchGen::createCoonsPatch ( std::vector< shared_ptr< ParamCurve > > & bd_curves, std::vector< shared_ptr< ParamCurve > > & cross_curves, double epsge, double kink_tol )`

*bd\_curves* form a loop, all *cross\_curves* point into the surface. *cross\_curves* need not fulfill twist and tangent requirements. Create a new [SplineSurface](#) representing the coons patch defined by a loop of four boundary curves, and their respective cross-tangent curves.

#### Parameters

|                     |                                                                                                              |
|---------------------|--------------------------------------------------------------------------------------------------------------|
| <i>bd_curves</i>    | ccw loop defining the boundary of the patch, size = 4.                                                       |
| <i>cross_curves</i> | corresponding cross tangent curves, size = 4, an element may be a NULL pointer.                              |
| <i>epsge</i>        | allowed distance between corresponding end points of <i>bd_curves</i> .                                      |
| <i>kink_tol</i>     | allowed angle between end tangents of <i>bd_curves</i> and corresponding end points of <i>cross_curves</i> . |

#### Returns

pointer to the created Gordon Surface.

### 28.7.2.5 `SplineSurface* Go::CoonsPatchGen::createCoonsPatch ( std::vector< shared_ptr< SplineCurve > > & bd_curves, std::vector< shared_ptr< SplineCurve > > & cross_curves )`

Create a Gordon Surface from input curves. *bd\_curves* forms a loop, all *cross\_curves* point into the surface. Assumes curves have been preprocessed, i.e. *cross\_curves* fulfill twist and tangent requirements.

#### Parameters

|                     |                                                                                 |
|---------------------|---------------------------------------------------------------------------------|
| <i>bd_curves</i>    | ccw loop defining the boundary of the patch, size = 4.                          |
| <i>cross_curves</i> | corresponding cross tangent curves, size = 4, an element may be a NULL pointer. |

#### Returns

pointer to the created Gordon Surface.



**28.7.2.6 SplineSurface\* Go::CoonsPatchGen::createGordonSurface ( std::vector< shared\_ptr< SplineCurve > > & mesh\_curves, std::vector< double > & params, int & nmb\_u\_crvs, bool use\_param\_values )**

Create a Gordon Surface from input curves and parameters. We require that both boundary curves in one direction (at least) are given. If use\_param\_values == true, we require the curves to be ordered (with parameters as given by params), with nmb\_u\_crvs u-curves up front. If use\_param\_values == false, mesh\_curves is reordered as explained in [splitMeshCurves\(\)](#) and [sortMeshCurves\(\)](#). Intersection parameters of curves are found and set in params, and nmb\_u\_crvs is also set.

#### Parameters

|                         |                                                                                             |
|-------------------------|---------------------------------------------------------------------------------------------|
| <i>mesh_curves</i>      | the iso-curves defining the Gordon Surface.                                                 |
| <i>params</i>           | the corresponding iso parameters. Updated if use_param_values == false.                     |
| <i>nmb_u_crvs</i>       | the number of curves parametrized in the u-direction. Updated if use_param_values == false. |
| <i>use_param_values</i> | whether to use the input parameters.                                                        |

#### Returns

pointer to the created Gordon Surface.

**28.7.2.7 SplineSurface\* Go::CoonsPatchGen::createGordonSurface ( std::vector< shared\_ptr< SplineCurve > > & mesh\_curves, std::vector< double > & params, int & nmb\_u\_crvs, std::vector< shared\_ptr< SplineCurve > > & cross\_curves, std::vector< int > & cross\_index, bool use\_param\_values = true )**

Create a Gordon Surface from input curves and parameters. cross\_index refers to indexing of mesh\_curves, which means that the boundary curves to which it is a tangent must be included. All cross\_curves are assumed to fulfill twist and tangent requirements.

#### Parameters

|                         |                                                                                             |
|-------------------------|---------------------------------------------------------------------------------------------|
| <i>mesh_curves</i>      | the iso-curves defining the Gordon Surface.                                                 |
| <i>params</i>           | the corresponding iso parameters. Updated if use_param_values == false.                     |
| <i>nmb_u_crvs</i>       | the number of curves parametrized in the u-direction. Updated if use_param_values == false. |
| <i>cross_curves</i>     | cross tangent curves for the Gordon Surface, corresponding to element in mesh_curves.       |
| <i>cross_index</i>      | referring to index element in mesh_curves.                                                  |
| <i>use_param_values</i> | whether to use the input parameters.                                                        |

#### Returns

pointer to the created Gordon Surface.

**28.7.2.8 SplineSurface\* Go::CoonsPatchGen::doCreateSurface ( std::vector< shared\_ptr< SplineCurve > > & mesh\_curves, std::vector< double > & params, int & nmb\_u\_crvs, std::vector< shared\_ptr< SplineCurve > > & cross\_curves, std::vector< int > & cross\_index )**

Create a Gordon Surface interpolating the input curves in the input parameters. mesh\_curves assumed to be ordered (i.e. u-curves first, with ascending parameter values), and missing boundary curve(s) in at most one direction. Expecting end conditions are satisfied.

## Parameters

|                     |                                                                                                       |
|---------------------|-------------------------------------------------------------------------------------------------------|
| <i>mesh_curves</i>  | iso-curves for Gordon Surface.                                                                        |
| <i>params</i>       | iso-parameters for the mesh_curves.                                                                   |
| <i>nmb_u_crvs</i>   | the number of curves parametrized in the u-direction. May alter as the parameter directions may swap. |
| <i>cross_curves</i> | the cross tangent curves along iso-curves for the Gordon Surface.                                     |
| <i>cross_index</i>  | index in mesh_curves of corresponding boundary curve.                                                 |

## Returns

pointer to the created Gordon Surface.

**28.7.2.9** void Go::CoonsPatchGen::fixCrossEndPts ( const std::vector< shared\_ptr< SplineCurve > > & *bd\_curves*, const std::vector< shared\_ptr< SplineCurve > > & *cross\_curves* )

Make sure that the end points of the existing cross\_tangent\_curves match the end derivatives of the corresponding bd\_curves. Curves are oriented CCW.

## Parameters

|                     |                                                                                                                          |
|---------------------|--------------------------------------------------------------------------------------------------------------------------|
| <i>bd_curves</i>    | curves along the boundary, ccw orientated. Will not be altered.                                                          |
| <i>cross_curves</i> | the corresponding cross tangent curves (same orientation). The curve objects referred to by the pointers may be altered. |

**28.7.2.10** void Go::CoonsPatchGen::getCrossTangs ( const std::vector< shared\_ptr< SplineCurve > > & *curves*, std::vector< shared\_ptr< SplineCurve > > & *mod\_cross\_curves*, double *tol1*, double *tol2* )

Prepare cross tangents for surface creation. curves contains boundary\_curve, cross\_curve; one edge at a time (iedge\*2 curves). boundary curves form a loop. All curves share spline space, and all cross curves point inwards = into the surface. A missing cross\_curve is indicated by a null pointer. Output mod\_cross\_curves fulfills tangent and twist conditions.

**28.7.2.11** void Go::CoonsPatchGen::getTangBlends ( std::vector< shared\_ptr< SplineCurve > > & *curves*, int *iedge*, std::vector< shared\_ptr< SplineCurve > > & *blend\_functions* )

Find blending functions used to blend two derivative along some boundary curves corresponding to a surface, into a cross derivative curve pr edge. Get curves for blending of two tangent curves into one cross tangent curve. Size of curves = iedge \* 3 (i.e. bd\_curve, cross\_curve, tangent curve). Boundary curves form a loop, and all curves share orientation and line space. cross-curves point inwards.

**28.7.2.12** void Go::CoonsPatchGen::hermit ( double *econd*[], int *icond*, bool *hasder1*, double *astart*, double *aend*, int *idim* )

Hermite interpolate the input points.

## Parameters

|                |                                                                                                                                                                                                                         |
|----------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>econd</i>   | array containing the points and derivatives to be interpolated. The ordering is <start_pt, (start_tan,) end_der, end_pt> The array will contain the coefficients of the spline curve fulfilling the input requirements. |
| <i>icond</i>   | number of interpolation conditions.                                                                                                                                                                                     |
| <i>hasder1</i> | Whether derivative is given in the start of the interval                                                                                                                                                                |
| <i>astart</i>  | start parameter of the produced curve.                                                                                                                                                                                  |
| <i>aend</i>    | end parameter of the produced curve.                                                                                                                                                                                    |
| <i>idim</i>    | dimension of the geometric space.                                                                                                                                                                                       |

28.7.2.13 **SplineSurface\*** `Go::CoonsPatchGen::loftSurface ( std::vector< shared_ptr< SplineCurve > >::iterator first_curve, int nmb_crvs )`

Create a lofting surface based on the input curves.

## Parameters

|                    |                                                    |
|--------------------|----------------------------------------------------|
| <i>first_curve</i> | iterator to first iso-curve in the lofted surface. |
| <i>nmb_crvs</i>    | the number of curves referred to by first_curve.   |

## Returns

pointer to the created lofting surface.

28.7.2.14 **SplineSurface\*** `Go::CoonsPatchGen::loftSurface ( std::vector< shared_ptr< SplineCurve > >::iterator first_curve, std::vector< double >::iterator first_param, int nmb_crvs )`

Create a lofting surface interpolating the input curves in the input parameters.

## Parameters

|                    |                                                                  |
|--------------------|------------------------------------------------------------------|
| <i>first_curve</i> | iterator to first iso-curve in the lofted surface.               |
| <i>first_param</i> | iso parameter to corresponding curve referred to by first_curve. |
| <i>nmb_crvs</i>    | the number of curves referred to by first_curve.                 |

## Returns

pointer to the created lofting surface.

28.7.2.15 **SplineSurface\*** `Go::CoonsPatchGen::loftSurface ( std::vector< shared_ptr< SplineCurve > >::iterator first_curve, std::vector< double >::iterator first_param, int nmb_crvs, std::vector< shared_ptr< SplineCurve > >::iterator first_cross_curve, std::vector< int > & cross_index )`

Create a lofting surface interpolating the input curves and cross tangent curves in the input parameters. All curves assumed to share spline space.

## Parameters

|                          |                                                                          |
|--------------------------|--------------------------------------------------------------------------|
| <i>first_curve</i>       | iterator to first iso-curve in the lofted surface.                       |
| <i>first_param</i>       | iso parameter to corresponding curve referred to by <i>first_curve</i> . |
| <i>nmb_crvs</i>          | the number of curves referred to by <i>first_curve</i> .                 |
| <i>first_cross_curve</i> | iterator to first cross tangent curve in the lofted surface.             |
| <i>cross_index</i>       | referring to index of corresponding boundary curve.                      |

## Returns

pointer to the created lofting surface.

```
28.7.2.16 void Go::CoonsPatchGen::makeLoftParams (std::vector< shared_ptr< SplineCurve > >::const_iterator
 first_curve, int nmb_crvs, double param_length, std::vector< double > & params)
```

Calculate iso parameters for the input curves. The curves are expected to be ordered, i.e. corresponding to increasing iso parameters. Curves are given iso-parameters in the range 0.0 to *param\_length*.

## Parameters

|                     |                                                   |
|---------------------|---------------------------------------------------|
| <i>first_curve</i>  | iterator to first iso curve.                      |
| <i>nmb_crvs</i>     | the number of input curves.                       |
| <i>param_length</i> | length of parameter domain.                       |
| <i>params</i>       | the computed iso parameters for the input curves. |

```
28.7.2.17 void Go::CoonsPatchGen::reparamBoundaryCurve (std::vector< shared_ptr< SplineCurve > > & curves,
 double aconst)
```

Check length of tangent vectors at the endpoints of a curve compared to the size of the curve. If the vectors are too long, reparametrize the curve and all the other curves in vector. All curves expected to share parametrization. First curve is a bnd curve. Optional additional curves may be cross tangent curve. Function reparametrizes bnd curve so that end tangents have length equal to *aconst*. Typical value is 1/3 of edge-length.

## Parameters

|               |                                     |
|---------------|-------------------------------------|
| <i>curves</i> | input and output curves.            |
| <i>aconst</i> | scalar defining end tangent length. |

```
28.7.2.18 void Go::CoonsPatchGen::sortMeshCurves (std::vector< shared_ptr< SplineCurve > > & mesh_curves,
 std::vector< double > & params, int nmb_u_crvs, std::vector< int > & cross_index)
```

Given that *mesh\_curves* has been regrouped with u-curves up front, we sort the vector according to values in *params*.

28.7.2.19 void Go::CoonsPatchGen::splitMeshCurves ( std::vector< shared\_ptr< SplineCurve > > & mesh\_curves, std::vector< double > & params, int & nmb\_u\_crvs, std::vector< int > & cross\_index, double epsgeo )

Given input iso-curves, the curves are analyzed and the ordering altered such that the nmb\_u\_crvs first elements are u-curves, and the rest are v-curves. Calculated iso parameter for the curves is returned in params.

#### Parameters

|             |                                                                   |
|-------------|-------------------------------------------------------------------|
| mesh_curves | iso-curves for a surface, sorted inside function.                 |
| params      | calculated parameters for the iso-curves.                         |
| nmb_u_crvs  | the number of curves parametrized in the u-direction.             |
| cross_index | elements referring to mesh_curves, updated inside function.       |
| epsgeo      | geometrical tolerance defining intersections between mesh_curves. |

28.7.2.20 SplineSurface\* Go::CoonsPatchGen::tpSurface ( const std::vector< shared\_ptr< SplineCurve > > & mesh\_curves, std::vector< double > params, int nmb\_u\_crvs, const std::vector< shared\_ptr< SplineCurve > > & cross\_curves, std::vector< int > & cross\_index )

Make tensor product surface which interpolates given grid points. All curves assumed to share spline space.

#### Parameters

|              |                                                                        |
|--------------|------------------------------------------------------------------------|
| mesh_curves  | curves to be interpolated in the Gordon Surface, both u- and v-curves. |
| params       | the iso parameters for the curves.                                     |
| nmb_u_crvs   | the number of curves parametrized in the u-direction.                  |
| cross_curves | cross tangent curves for the Gordon Surface.                           |
| cross_index  | referring to index of corresponding boundary curve.                    |

#### Returns

pointer the created lofting surface.

## 28.8 Go::CoonsPatchVolumeGen Namespace Reference

This namespace contains functions used to create a Coons Patch volume.

### Functions

- void [get\\_corners](#) (shared\_ptr< Go::SplineSurface > surf, std::vector< Go::Point > &pts)  
*Used internally in the geometry construction.*
- void [push\\_corners](#) (std::vector< Go::Point > &pts\_to, const std::vector< Go::Point > pts\_from, int pos0, int pos1, int pos2, int pos3)  
*Used internally in the geometry construction.*
- bool [edge\\_curves\\_equal](#) (shared\_ptr< Go::SplineSurface > sf1, double par1, bool is\_u\_dir1, shared\_ptr< Go::SplineSurface > sf2, double par2, bool is\_u\_dir2, double tol)  
*Used internally in the geometry construction.*

- [SplineVolume](#) \* `createCoonsPatchDirectly` (const [Go::SplineSurface](#) \*surf\_u\_min, const [Go::SplineSurface](#) \*surf\_u\_max, const [Go::SplineSurface](#) \*surf\_v\_min, const [Go::SplineSurface](#) \*surf\_v\_max, const [Go::SplineSurface](#) \*surf\_w\_min, const [Go::SplineSurface](#) \*surf\_w\_max)
- [SplineVolume](#) \* `createCoonsPatch` (const [Go::SplineSurface](#) \*surf\_u\_min, const [Go::SplineSurface](#) \*surf\_u\_max, const [Go::SplineSurface](#) \*surf\_v\_min, const [Go::SplineSurface](#) \*surf\_v\_max, const [Go::SplineSurface](#) \*surf\_w\_min, const [Go::SplineSurface](#) \*surf\_w\_max, double tol=1.0e-05)

### 28.8.1 Detailed Description

This namespace contains functions used to create a Coons Patch volume.

### 28.8.2 Function Documentation

**28.8.2.1** `SplineVolume* Go::CoonsPatchVolumeGen::createCoonsPatch ( const Go::SplineSurface * surf_u_min, const Go::SplineSurface * surf_u_max, const Go::SplineSurface * surf_v_min, const Go::SplineSurface * surf_v_max, const Go::SplineSurface * surf_w_min, const Go::SplineSurface * surf_w_max, double tol = 1.0e-05 )`

Create a new [SplineVolume](#) representing the coons patch of six [SplineSurfaces](#), the six faces of the volume. We test if the faces are non-rational, lie in the same space, and have coinciding edge curves where needed. The B-spline bases of the surfaces might be reversed, swapped, have order raised, knots inserted and parameter interval rescaled to [0,1], in order to be able to use `createCoonsPatchDirectly()`

#### Parameters

|                         |                                                     |
|-------------------------|-----------------------------------------------------|
| <code>surf_u_min</code> | One of faces/isosurfaces for the u-direction        |
| <code>surf_u_max</code> | The other faces/isosurfaces for the u-direction     |
| <code>surf_v_min</code> | One of faces/isosurfaces for the v-direction        |
| <code>surf_v_max</code> | The other faces/isosurfaces for the v-direction     |
| <code>surf_w_min</code> | One of faces/isosurfaces for the w-direction        |
| <code>surf_w_max</code> | The other faces/isosurfaces for the w-direction     |
| <code>tol</code>        | Tolerance when identifying equal curves and corners |

#### Returns

a pointer to a newly created [SplineVolume](#) representing the coons patch. The user assumes ownership of the object

**28.8.2.2** `SplineVolume* Go::CoonsPatchVolumeGen::createCoonsPatchDirectly ( const Go::SplineSurface * surf_u_min, const Go::SplineSurface * surf_u_max, const Go::SplineSurface * surf_v_min, const Go::SplineSurface * surf_v_max, const Go::SplineSurface * surf_w_min, const Go::SplineSurface * surf_w_max )`

Create a new [SplineVolume](#) representing the coons patch of six [SplineSurfaces](#), the six faces of the volume. Here we assume that the faces are non-rational and lie in the same space, and all have the same B-spline basis (knot vectors and degrees are the same) along coinciding sides, i.e. for two surfaces with a common side, the two coinciding edge curves have the same B-spline basis. Also, all B-spline bases are assumed to have knot interval [0.0, 1.0].

## Parameters

|                   |                                 |
|-------------------|---------------------------------|
| <i>surf_u_min</i> | The face/isosurface for u = 0.0 |
| <i>surf_u_max</i> | The face/isosurface for u = 1.0 |
| <i>surf_v_min</i> | The face/isosurface for v = 0.0 |
| <i>surf_v_max</i> | The face/isosurface for v = 1.0 |
| <i>surf_w_min</i> | The face/isosurface for w = 0.0 |
| <i>surf_w_max</i> | The face/isosurface for u = 1.0 |

## Returns

a pointer to a newly created [SplineVolume](#) representing the coons patch. The user assumes ownership of the object.

**28.8.2.3** `bool Go::CoonsPatchVolumeGen::edge_curves_equal ( shared_ptr< Go::SplineSurface > sf1, double par1, bool is_u_dir1, shared_ptr< Go::SplineSurface > sf2, double par2, bool is_u_dir2, double tol )`

Used internally in the geometry construction.

**28.8.2.4** `void Go::CoonsPatchVolumeGen::get_corners ( shared_ptr< Go::SplineSurface > surf, std::vector< Go::Point > & pts )`

Used internally in the geometry construction.

**28.8.2.5** `void Go::CoonsPatchVolumeGen::push_corners ( std::vector< Go::Point > & pts_to, const std::vector< Go::Point > pts_from, int pos0, int pos1, int pos2, int pos3 )`

Used internally in the geometry construction.

## 28.9 Go::CreatorsUtils Namespace Reference

Related to the generation of cross tangent curves.

### Functions

- [SplineCurve](#) `GO_API * getParametricCurve (const std::vector< shared_ptr< const CurveOnSurface > > &cv)`
- `shared_ptr< Go::SplineCurve > GO_API createCrossTangent (const Go::CurveOnSurface &cv)`
- `shared_ptr< Go::SplineCurve > GO_API createCrossTangent (const Go::CurveOnSurface &cv, shared_ptr< Go::SplineCurve > basis_space_cv, const Go::SplineCurve *cross_cv_ref, bool appr_offset_cv=true)`
- `std::vector< Go::Point > GO_API projectPoint (const Go::ParamSurface *sf, bool closed_dir_u, bool closed_dir_v, const Go::Point &space_pt, double epsgeo=1e-04)`
- `shared_ptr< Go::Point > projectCurvePoint (const ParamSurface *sf, bool closed_dir_u, bool closed_dir_v, const Go::ParamCurve *space_cv, double cv_par, double epsgeo=1e-04)`
- `shared_ptr< Go::Point > projectCurvePoint (const SplineSurface &sf, bool closed_dir_u, bool closed_dir_v, const Go::ParamCurve *space_cv, double cv_par, double epsgeo=1e-04)`
- `void GO_API fixSeemCurves (shared_ptr< BoundedSurface > bd_sf, std::vector< shared_ptr< CurveOnSurface > > &loop_cvs, bool closed_dir_u, bool closed_dir_v, double tol)`
- `void GO_API fixTrimCurves (shared_ptr< Go::BoundedSurface > bd_sf, double epsgeo_frac=1.0, double tol=1.0e-3, double tol2=1.0e-2, double ang_tol=1.0e-2)`

## 28.9.1 Detailed Description

Related to the generation of cross tangent curves.

## 28.9.2 Function Documentation

28.9.2.1 `shared_ptr<Go::SplineCurve> GO_API Go::CreatorsUtils::createCrossTangent ( const Go::CurveOnSurface & cv )`

Generate the inwards pointing cross-tangent curve along the input trim curve.

### Parameters

|                 |                 |
|-----------------|-----------------|
| <code>cv</code> | the trim curve. |
|-----------------|-----------------|

### Returns

the inwards cross tangent curve.

28.9.2.2 `shared_ptr<Go::SplineCurve> GO_API Go::CreatorsUtils::createCrossTangent ( const Go::CurveOnSurface & cv, shared_ptr< Go::SplineCurve > basis_space_cv, const Go::SplineCurve * cross_cv_ref, bool appr_offset_cv = true )`

The cross tangent cv along input cv is created. Length given by length of derivs in sf along cv or by input\_cross\_cv if it exists. Direction of created cross tangent cv is to the left of cv (i.e. it should point into the surface). If input\_cross\_cv exists so does basis\_space\_cv and they must share parametrization. If input\_cross\_cv exists, but not basis\_space\_cv, it shares parametrization with cv.

### Parameters

|                             |                                                                                                                                                                       |
|-----------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>cv</code>             | the trim curve along which we are to compute the cross tangent cv.                                                                                                    |
| <code>basis_space_cv</code> | if != NULL we use the basis and parametrization of basis_space_cv.                                                                                                    |
| <code>cross_cv_ref</code>   | if != NULL the curve defines the angle between the boundary curve and the new cross tangent curve. Additionally it defines the length of the new cross tangent curve. |
| <code>appr_offset_cv</code> | whether the method should approximate the offset curve (as opposed to the cross tangent curve).                                                                       |

### Returns

the cross tangent (or offset) curve.

28.9.2.3 `void GO_API Go::CreatorsUtils::fixSeemCurves ( shared_ptr< BoundedSurface > bd_sf, std::vector< shared_ptr< CurveOnSurface > > & loop_cvs, bool closed_dir_u, bool closed_dir_v, double tol )`

Repair erranous seem curves in bounded surfaces with closed underlying surfaces



28.9.2.4 `void GO_API Go::CreatorsUtils::fixTrimCurves ( shared_ptr< Go::BoundedSurface > bd_sf, double epsgeo_frac = 1.0, double tol = 1.0e-3, double tol2 = 1.0e-2, double ang_tol = 1.0e-2 )`

Repair errantous trimming curves in bounded surfaces and compute missing parameter curves in the corresponding [CurveOnSurface](#) curves

28.9.2.5 `SplineCurve GO_API* Go::CreatorsUtils::getParametricCurve ( const std::vector< shared_ptr< const CurveOnSurface > > & cv )`

Helper function. Returns the parametric representation of the input curve. The returned curve is created (i.e. it should be handled by a smart ptr)! This function takes as argument a vector of (shared pointers to) [CurveOnSurfaces](#), which are assumed to lie on the same surface and to represent consecutive parts of a larger curve, and return the parametric representation of this larger curve.

#### Parameters

|           |                                                                                                                                                                                                 |
|-----------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>cv</i> | a vector of (shared pointers to) <a href="#">CurveOnSurfaces</a> . We suppose that these curves are all lying on the same surface and that they constitute consecutive parts of a global curve. |
|-----------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

#### Returns

if successful, a pointer to the newly generated [SplineCurve](#), which represents the global curve that is the union of the input curves. This [SplineCurve](#) is represented in the parameter plane of the surface, and is therefore a 2D curve. If it could not be constructed (due a failure of any of the provided [CurveOnSurfaces](#) to provide their internal parameter curve), a null pointer is returned instead. The user assumes ownership of the [SplineCurve](#).

28.9.2.6 `shared_ptr<Go::Point> Go::CreatorsUtils::projectCurvePoint ( const ParamSurface * sf, bool closed_dir_u, bool closed_dir_v, const Go::ParamCurve * space_cv, double cv_par, double epsgeo = 1e-04 )`

28.9.2.7 `shared_ptr<Go::Point> Go::CreatorsUtils::projectCurvePoint ( const SplineSurface & sf, bool closed_dir_u, bool closed_dir_v, const Go::ParamCurve * space_cv, double cv_par, double epsgeo = 1e-04 )`

Project a point on a space curve onto a surface. If the surface is closed, and the curve follows the seam at the point in question, then the point corresponding to a counterclockwise curve is chosen. If the programmer knows that this is not the case, then this must be handled.

#### Parameters

|                     |                                                                                       |
|---------------------|---------------------------------------------------------------------------------------|
| <i>sf</i>           | the surface the point is projected onto                                               |
| <i>closed_dir_u</i> | boolean that is <code>true</code> if the surface is closed in the <i>u</i> -direction |
| <i>closed_dir_v</i> | boolean that is <code>true</code> if the surface is closed in the <i>v</i> -direction |
| <i>space_cv</i>     | the curve to be projected                                                             |
| <i>cv_par</i>       | the parameter of the curve point to be projected                                      |

#### Returns

the projected point in the form of a parameter pair

28.9.2.8 `std::vector<Go::Point> GO_API Go::CreatorsUtils::projectPoint ( const Go::ParamSurface * sf, bool closed_dir_u, bool closed_dir_v, const Go::Point & space_pt, double epsgeo = 1e-04 )`

Project a point in space onto a surface. If the surface is closed, and the point is on the seam, then the set of possible candidates is returned.

#### Parameters

|                                        |                                                                                       |
|----------------------------------------|---------------------------------------------------------------------------------------|
| <i>sf</i>                              | the surface the point is projected onto                                               |
| <i>closed_dir</i> <sub>↔<i>u</i></sub> | boolean that is <code>true</code> if the surface is closed in the <i>u</i> -direction |
| <i>closed_dir</i> <sub>↔<i>v</i></sub> | boolean that is <code>true</code> if the surface is closed in the <i>v</i> -direction |
| <i>space_pt</i>                        | the point to be projected                                                             |

#### Returns

vector of candidates of projected points in the form of parameter pairs

## 28.10 Go::Curvature Namespace Reference

[Curvature](#) analysis related to curves.

### Functions

- void [curvatureRadiusPoints](#) (const [SplineCurve](#) &*curve*, double *curveRad*, std::vector< double > &*pos*)
- void [minimalCurvatureRadius](#) (const [SplineCurve](#) &*curve*, double &*mincurv*, double &*pos*)

#### 28.10.1 Detailed Description

[Curvature](#) analysis related to curves.

#### 28.10.2 Function Documentation

28.10.2.1 void [Go::Curvature::curvatureRadiusPoints](#) ( const [SplineCurve](#) & *curve*, double *curveRad*, std::vector< double > & *pos* )

Get the points on a [SplineCurve](#) with a given curvature radius Result is stored in a vector as parameter values

28.10.2.2 void [Go::Curvature::minimalCurvatureRadius](#) ( const [SplineCurve](#) & *curve*, double & *mincurv*, double & *pos* )

Get the minimal curvature radius, and the parameter value of the point with the minimal curvature radius

## 28.11 Go::CurvatureAnalysis Namespace Reference

### Functions

- void `computeFirstFundamentalForm` (const ParamSurface &sf, double u, double v, int derivs, std::vector< double > &form)
- void `computeSecondFundamentalForm` (const ParamSurface &sf, double u, double v, double form1[3], double form2[3])
- void `curvatures` (const ParamSurface &sf, double u, double v, double &K, double &H)
- void `principalCurvatures` (const ParamSurface &sf, double u, double v, double &k1, Point &d1, double &k2, Point &d2)
- void `minimalCurvatureRadius` (const ParamSurface &sf, double tolerance, double &mincurv, double &pos\_u, double &pos\_v, double degenerate\_eps, double curv\_tol=1.0e-3)
- void `evaluateMinCurvatureRadius` (const ParamSurface &sf, double start\_u, double end\_u, double start\_v, double end\_v, double tolerance, std::vector< double > &param\_u, std::vector< double > &param\_v, std::vector< std::vector< double > > &curvs, double &mincurv, double &minpos\_u, double &minpos\_v, bool initialize)

### 28.11.1 Detailed Description

[Curvature](#) analysis related to surfaces. Functions computing fundamental forms and curvature.

### 28.11.2 Function Documentation

28.11.2.1 void `Go::CurvatureAnalysis::computeFirstFundamentalForm` ( const ParamSurface &sf, double u, double v, int derivs, std::vector< double > &form )

Computes the coefficients of the first fundamental form. The returned values are stored { E F G [Eu Fu Gu Ev Fv Gv [Euu Fuu Guu Euv Fuv Guv Evv Fvv Gvv [...]] ]. Only one derivative is implemented so far.

#### Parameters

|               |                                                                                                                                                                                    |
|---------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>sf</i>     | reference to the concerned surface                                                                                                                                                 |
| <i>u</i>      | first parameter of point where we want to compute the first fundamental form                                                                                                       |
| <i>v</i>      | second parameter of point where we want to compute the first fundamental form                                                                                                      |
| <i>derivs</i> | number of (partial) derivatives of the first fundamental form that we want to include in the result. So far, only the computation of the first derivative is actually implemented. |
| <i>form</i>   | the computed values, on the format described above                                                                                                                                 |

28.11.2.2 void `Go::CurvatureAnalysis::computeSecondFundamentalForm` ( const ParamSurface &sf, double u, double v, double form1[3], double form2[3] )

Computes the coefficients of the first and second fundamental forms. The returned values are stored { E F G } in form1 and { e f g } in form2. This function cannot compute derivatives.

#### Parameters

|           |                                    |
|-----------|------------------------------------|
| <i>sf</i> | reference to the concerned surface |
|-----------|------------------------------------|

## Parameters

|              |                                                                               |
|--------------|-------------------------------------------------------------------------------|
| <i>u</i>     | first parameter of point where we want to compute the fundamental forms.      |
| <i>v</i>     | second parameter of point where we want to compute the fundamental forms      |
| <i>form1</i> | the values associated with the first fundamental form will be returned here.  |
| <i>form2</i> | the values associated with the second fundamental form will be returned here. |

28.11.2.3 void Go::CurvatureAnalysis::curvatures ( const ParamSurface & *sf*, double *u*, double *v*, double & *K*, double & *H* )

Computes the Gaussian (K) and mean (H) curvatures.

## Parameters

|           |                                                                   |
|-----------|-------------------------------------------------------------------|
| <i>sf</i> | reference to the concerned surface                                |
| <i>u</i>  | first parameter of point where we want to carry out computation   |
| <i>v</i>  | second parameter of point where weh want to carry out computation |
| <i>K</i>  | value of Gaussian curvature returned here                         |
| <i>H</i>  | value of mean curvature returned here                             |

28.11.2.4 void Go::CurvatureAnalysis::evaluateMinCurvatureRadius ( const ParamSurface & *sf*, double *start\_u*, double *end\_u*, double *start\_v*, double *end\_v*, double *tolerance*, std::vector< double > & *param\_u*, std::vector< double > & *param\_v*, std::vector< std::vector< double > > & *curvs*, double & *mincurv*, double & *minpos\_u*, double & *minpos\_v*, bool *initialize* )

Estimate the minium curvature radius of the surface *sf* in the parameter domain [*start\_u*,*end\_u*]x[*start\_v*,*end\_v*] by evaluating the curvature radius in a grid and fetch the smallest one. The grid density is computed from the given tolerance

28.11.2.5 void Go::CurvatureAnalysis::minimalCurvatureRadius ( const ParamSurface & *sf*, double *tolerance*, double & *mincurv*, double & *pos\_u*, double & *pos\_v*, double *degenerate\_eps*, double *curv\_tol* = 1.0e-3 )

Estimate the minimum curvature radius of the surface *sf*.

## Parameters

|                       |                                                                                                |
|-----------------------|------------------------------------------------------------------------------------------------|
| <i>sf</i>             | the given surface                                                                              |
| <i>tolerance</i>      | influences the density of the search                                                           |
| <i>mincurve</i>       | the computed minimum curvature radius                                                          |
| <i>pos_u</i>          | parameter in the first parameter direction corresponding to the minimum curvature radius       |
| <i>pos_v</i>          | parameter in the second parameter direction corresponding to the minimum curvature radius      |
| <i>degenerate_eps</i> | maximum length of a degenenerate boundary, used to check whether special treatment is required |
| <i>curv_tol</i>       | default parameter used to limit the search                                                     |

28.11.2.6 void Go::CurvatureAnalysis::principalCurvatures ( const ParamSurface & sf, double u, double v, double & k1, Point & d1, double & k2, Point & d2 )

Computes the principal curvatures of a surface, sf, in the parameter value (u,v) along with the corresponding parameter directions.

## 28.12 Go::CurveCreators Namespace Reference

### Functions

- SplineCurve GO\_API \* multCurveWithFunction (const SplineCurve &alpha, const SplineCurve &f)
- SplineCurve GO\_API \* blend (const SplineCurve &alpha\_1, const SplineCurve &f\_1, const SplineCurve &alpha\_2, const SplineCurve &f\_2)
- SplineCurve GO\_API \* approxCurves (shared\_ptr< ParamCurve > \*first\_crv, shared\_ptr< ParamCurve > \*last\_crv, const std::vector< Point > &start\_pt, const std::vector< Point > &end\_pt, double approx\_tol, double &maxdist, int max\_iter=5, int degree=3)
- void projectCurve (shared\_ptr< ParamCurve > &space\_cv, shared\_ptr< ParamSurface > &surf, double epsge, shared\_ptr< SplineCurve > &proj\_cv, shared\_ptr< SplineCurve > &par\_cv)
- std::vector< shared\_ptr< SplineCurve > > curveApprox (shared\_ptr< ParamCurve > cvs[], int nmb\_cvs, const BsplineBasis &init\_basis, double tol)
- std::vector< shared\_ptr< SplineCurve > > curveApprox (shared\_ptr< ParamCurve > cvs[], int nmb\_cvs, double tol, double degree=3)
- SplineCurve GO\_API \* projectSpaceCurve (shared\_ptr< ParamCurve > &space\_cv, shared\_ptr< ParamSurface > &surf, shared\_ptr< Point > &start\_par\_pt, shared\_ptr< Point > &end\_par\_pt, double epsge, const RectDomain \*domain\_of\_interest=NULL)
- SplineCurve GO\_API \* liftParameterCurve (shared\_ptr< ParamCurve > &parameter\_cv, shared\_ptr< ParamSurface > &surf, double epsge)
 

*Lift the parameter\_cv onto surf.*
- SplineCurve GO\_API \* createCircle (Point center, Point axis, Point normal, double radius)
 

*Create a circle.*
- shared\_ptr< Go::SplineCurve > GO\_API insertParamDomain (const Go::SplineCurve &cv\_1d, double knot\_tol=1e-08)
 

*Given input of 1-dimensional curve, return the 2-dimensional visualization (x, f(x)).*
- SplineCurve GO\_API \* offsetCurve (const SplineCurve &cv, Point offset\_val)
 

*Return the spline curve given by cv(t) + offset\_val.*

### 28.12.1 Detailed Description

Various functions for generating SplineCurve s by approximation, blending, etc.

### 28.12.2 Function Documentation

28.12.2.1 SplineCurve GO\_API \* Go::CurveCreators::approxCurves ( shared\_ptr< ParamCurve > \* first\_crv, shared\_ptr< ParamCurve > \* last\_crv, const std::vector< Point > & start\_pt, const std::vector< Point > & end\_pt, double approx\_tol, double & maxdist, int max\_iter = 5, int degree = 3 )

Given input curves, we approximate with one cubic spline curve, fulfilling end requirements. This function is used generate one SplineCurve which approximates a sequence of (supposedly end-to-end continuous) input SplineCurves. In addition, the user can define explicitly the start and end point and tangent of the generated curve.

## Parameters

|                  |                                                                                                                                                                                                                 |
|------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>first_crv</i> | pointer to the start of a range of (shared pointers to) SplineCurves, supposedly stored so that the end point of each curve coincides with the start point of the succeeding one.                               |
| <i>last_crv</i>  | pointer to one-past-last of the range of SplineCurves.                                                                                                                                                          |
| <i>start_pt</i>  | The user can choose to explicitly specify the start point and tangent of the curve to be generated. If so, then this vector must contain one or two elements: the start point and optionally the start tangent. |
| <i>end_pt</i>    | The user can choose to explicitly specify the end point and tangent of the curve to be generated. If so, then this vector must contain one or two elements: the end point and optionally the end tangent.       |
| <i>approxtol</i> | The routine will try to generate a curve that fits the given curves within a tolerance of 'approxtol', but is not guaranteed to succeed. If unsuccessful, a message will be written to standard output.         |
| <i>maxdist</i>   | Reports the maximum found distance between the generated curve and the input curves.                                                                                                                            |

## Return values

|                 |                                                                                              |
|-----------------|----------------------------------------------------------------------------------------------|
| <i>max_iter</i> | Specify the maximum number of iterations that is allowed in order to converge to a solution. |
|-----------------|----------------------------------------------------------------------------------------------|

## Returns

a raw pointer to the generated [SplineCurve](#). User assumes ownership.

### 28.12.2.2 SplineCurve GO\_API\* Go::CurveCreators::blend ( const SplineCurve & *alpha\_1*, const SplineCurve & *f\_1*, const SplineCurve & *alpha\_2*, const SplineCurve & *f\_2* )

Given blend functions *alpha\_1* & *alpha\_2* (represented as 1-dimensional SplineCurves), and the spline curves *f\_1* and *f\_2*, return the blended expression ' $\alpha_1 * f_1 + \alpha_2 * f_2$ '. **NB:** All the curves given as input must be defined on the same parameter interval. Their knotvectors and orders do not have to be equal, though.

## Parameters

|                |                                                                                                                                   |
|----------------|-----------------------------------------------------------------------------------------------------------------------------------|
| <i>alpha_1</i> | the multiplier function for the first curve, ' <i>f_1</i> '. It is represented as an 1-dimensional <a href="#">SplineCurve</a> .  |
| <i>f_1</i>     | The first <a href="#">SplineCurve</a> to be blended.                                                                              |
| <i>alpha_2</i> | the multiplier function for the second curve, ' <i>f_2</i> '. It is represented as an 1-dimensional <a href="#">SplineCurve</a> . |
| <i>f_2</i>     | The second <a href="#">SplineCurve</a> to be blended.                                                                             |

## Returns

A raw pointer to the newly generated [SplineCurve](#), a blend between '*f\_1*' and '*f\_2*'. The user assumes ownership.

### 28.12.2.3 SplineCurve GO\_API\* Go::CurveCreators::createCircle ( Point *center*, Point *axis*, Point *normal*, double *radius* )

Create a circle.

Generate a 3D [SplineCurve](#) representing a circle. Assuming dimension is 3.

## Parameters

|               |                                                                                                      |
|---------------|------------------------------------------------------------------------------------------------------|
| <i>center</i> | the center point of the circle,                                                                      |
| <i>axis</i>   | a vector pointing from the center point of the circle and towards start and end point of the circle. |
| <i>normal</i> | the normal to the plane in which the circle lie.                                                     |
| <i>radius</i> | the radius of the circle to generate.                                                                |

## Returns

a raw pointer to the generated [SplineCurve](#), representing the specified circle. User assumes ownership.

**28.12.2.4** `std::vector<shared_ptr<SplineCurve>> Go::CurveCreators::curveApprox ( shared_ptr< ParamCurve > cvs[], int nmb_cvs, const BsplineBasis & init_basis, double tol )`

Approximate a number of curves in the same spline space given an initial spline space that may be refined. The approximation tolerance is tol

**28.12.2.5** `std::vector<shared_ptr<SplineCurve>> Go::CurveCreators::curveApprox ( shared_ptr< ParamCurve > cvs[], int nmb_cvs, double tol, double degree = 3 )`

Approximate a number of curves in the same spline space. The approximation tolerance is tol

**28.12.2.6** `shared_ptr<Go::SplineCurve> GO_API Go::CurveCreators::insertParamDomain ( const Go::SplineCurve & cv_1d, double knot_tol = 1e-08 )`

Given input of 1-dimensional curve, return the 2-dimensional visualization (x, f(x)).

**28.12.2.7** `SplineCurve GO_API* Go::CurveCreators::liftParameterCurve ( shared_ptr< ParamCurve > & parameter_cv, shared_ptr< ParamSurface > & surf, double epsge )`

Lift the parameter\_cv onto surf.

'Lift' a 2D parameter curve onto a surface. (This means generate a space curve lying on the surface, defined as the curved obtained when evaluating the surface at the parameter points given by the 2D curve).

## Parameters

|                     |                                                                                          |
|---------------------|------------------------------------------------------------------------------------------|
| <i>parameter_cv</i> | the 2D parameter curve. Its values should be kept within the parameter domain of 'surf'. |
| <i>surf</i>         | the surface on which the obtained 3D curve will lie.                                     |
| <i>epsge</i>        | geometrical tolerance used when generating the curve                                     |

## Returns

a raw pointer to the generated spatial [SplineCurve](#). User assumes ownership.

### 28.12.2.8 SplineCurve GO\_API\* Go::CurveCreators::multCurveWithFunction ( const SplineCurve & *alpha*, const SplineCurve & *f* )

Given a spline curve *f\_i* on a space of dimension *n*, and a (1-dim) spline function *alpha\_i* (not necessarily a Bezier curve) defined on the same space, return the product. Construct a [SplineCurve](#) which is the 'product' of another [SplineCurve](#) and a function (which is given as a 1-dimensional [SplineCurve](#)). **NB:** The two [SplineCurves](#) given as argument must be defined on exactly the same parameter interval. Their knotvectors and orders do not have to be equal, though.

#### Parameters

|              |                                                                                    |
|--------------|------------------------------------------------------------------------------------|
| <i>alpha</i> | the multiplier function, given here as a 1-dimensional <a href="#">SplineCurve</a> |
| <i>f</i>     | the <a href="#">SplineCurve</a> to be multiplied with                              |

#### Returns

A raw pointer to the newly generated [SplineCurve](#), a product between '*f*' and the '*alpha*' function. The user assumes ownership.

### 28.12.2.9 SplineCurve GO\_API\* Go::CurveCreators::offsetCurve ( const SplineCurve & *cv*, Point *offset\_val* )

Return the spline curve given by  $cv(t) + offset\_val$ .

### 28.12.2.10 void Go::CurveCreators::projectCurve ( shared\_ptr< ParamCurve > & *space\_cv*, shared\_ptr< ParamSurface > & *surf*, double *epsge*, shared\_ptr< SplineCurve > & *proj\_cv*, shared\_ptr< SplineCurve > & *par\_cv* )

### 28.12.2.11 SplineCurve GO\_API\* Go::CurveCreators::projectSpaceCurve ( shared\_ptr< ParamCurve > & *space\_cv*, shared\_ptr< ParamSurface > & *surf*, shared\_ptr< Point > & *start\_par\_pt*, shared\_ptr< Point > & *end\_par\_pt*, double *epsge*, const RectDomain \* *domain\_of\_interest* = NULL )

Project the *space\_cv* into parameter domain given by *surf*. *start\_par\_pt* (& *end\*\_* not needed, but useful to ensure correct evaluation). Generate a [SplineCurve](#) which lies on a given (part of a) [SplineSurface](#) and is the projection of a given [SplineCurve](#) onto that surface.

#### Parameters

|                           |                                                                                                                                                                                                                                                         |
|---------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>space_cv</i>           | the <a href="#">SplineCurve</a> (in 3D space) that we want to project onto the surface.                                                                                                                                                                 |
| <i>surf</i>               | The surface we will project the curve onto. The resulting curve will therefore lie in this surface.                                                                                                                                                     |
| <i>start_par_pt</i>       | if the user wants to explicitly define where the start point of the curve should lie, ' <i>start_par_pt</i> ' could be set to point to this point. It can also be a zero pointer, in which case the start point is determined just like any other point |
| <i>end_par_pt</i>         | if the user wants to explicitly define where the end point of the curve should lie, ' <i>end_par_pt</i> ' could be set to point to this point. It can also be a zero pointer, in which case the end point is determined just like any other point.      |
| <i>epsge</i>              | The geometrical tolerance to use when generating the projected curve. Max dist in surface from exact projection.                                                                                                                                        |
| <i>domain_of_interest</i> | If the user do not want a projection onto the whole ' <i>surf</i> ', he/she can specify the part of the surface to use by limiting the parametric domain to use. If this pointer is left as a null pointer, the whole surface is used.                  |



**Returns**

a raw pointer to the generated [SplineCurve](#). User assumes ownership.

**28.13 Go::CurveInterpolator Namespace Reference**

Curve interpolation functionality not adapted to the [Interpolator](#) class.

**Functions**

- [SplineCurve](#) \* [regularInterpolation](#) ([const BsplineBasis](#) &basis, [std::vector< double >](#) &par, [std::vector< double >](#) &points, [int](#) dimension, [bool](#) rational, [std::vector< double >](#) &weights)

**28.13.1 Detailed Description**

Curve interpolation functionality not adapted to the [Interpolator](#) class.

**28.13.2 Function Documentation**

**28.13.2.1** [SplineCurve\\*](#) [Go::CurveInterpolator::regularInterpolation](#) ( [const BsplineBasis](#) & *basis*, [std::vector< double >](#) & *par*, [std::vector< double >](#) & *points*, [int](#) *dimension*, [bool](#) *rational*, [std::vector< double >](#) & *weights* )

Interpolate a set of regular, parameterized interpolation points. The parameterization is assumed to correspond to the given B-spline basis (select interpolation points in the Greville points). The function throws if the input parameters are inconsistent

**28.14 Go::FaceUtilities Namespace Reference****Functions**

- void [getBoundaryData](#) ([ftSurface](#) \*face, [int](#) nmb\_sample, [std::vector< SamplePointData >](#) &sample\_points)
- void [getInnerData](#) ([ftSurface](#) \*face, [int](#) nmb\_sample\_u, [int](#) nmb\_sample\_v, [std::vector< SamplePointData >](#) &sample\_points)
- [bool](#) [enforceCoLinearity](#) ([ftSurface](#) \*face1, [ftEdge](#) \*edge1, [ftSurface](#) \*face2, [double](#) tol, [double](#) ang\_tol)
- [bool](#) [enforceVxCoLinearity](#) ([shared\\_ptr< Vertex >](#) vx, [double](#) tol, [double](#) ang\_tol)

**28.14.1 Detailed Description**

Utility functionality for faces in order to release the need for private functions in [ftSurface](#)

## 28.14.2 Function Documentation

28.14.2.1 `bool Go::FaceUtilities::enforceCoLinearity ( ftSurface * face1, ftEdge * edge1, ftSurface * face2, double tol, double ang_tol )`

Enforce colinearity of coefficients of spline surfaces related to the given faces Return value = false : No modification

28.14.2.2 `bool Go::FaceUtilities::enforceVxCoLinearity ( shared_ptr< Vertex > vx, double tol, double ang_tol )`

28.14.2.3 `void Go::FaceUtilities::getBoundaryData ( ftSurface * face, int nmb_sample, std::vector< SamplePointData > & sample_points )`

28.14.2.4 `void Go::FaceUtilities::getInnerData ( ftSurface * face, int nmb_sample_u, int nmb_sample_v, std::vector< SamplePointData > & sample_points )`

## 28.15 Go::ftVolumeTools Namespace Reference

This namespace contains service functions related to [ftVolume](#).

### Functions

- `std::vector< shared_ptr< ftVolume > > splitVolumes (shared_ptr< ftVolume > &vol1, shared_ptr< ftVolume > &vol2, double eps, std::vector< int > &config)`
- `std::vector< shared_ptr< ftVolume > > splitVolumes (shared_ptr< ftVolume > &vol, shared_ptr< ftSurface > &face, double eps)`  
*Split one volume according to intersections with a given face.*
- `void updateWithSplitFaces (shared_ptr< SurfaceModel > shell, shared_ptr< ftSurface > &face1, shared_ptr< ftSurface > &face2, std::vector< std::pair< ftEdge *, ftEdge * > > &replaced_wires)`  
*Specific functionality. Used from ftVolume::generateMissingBdSurf.*
- `int boundaryStatus (ftVolume *vol, shared_ptr< ftSurface > &bd_face, double tol)`

### 28.15.1 Detailed Description

This namespace contains service functions related to [ftVolume](#).

### 28.15.2 Function Documentation

28.15.2.1 `int Go::ftVolumeTools::boundaryStatus ( ftVolume * vol, shared_ptr< ftSurface > & bd_face, double tol )`

Given a boundary or trimming surface related to an [ftVolume](#), check the status and update stored information if any new boundary status information is computed

#### Parameters

|                      |                                                                              |
|----------------------|------------------------------------------------------------------------------|
| <code>vol</code>     | the volume from which the boundary status is to be checked                   |
| <code>bd_face</code> | the boundary face where the requested information should be fetched/computed |
| <code>tol</code>     | tolerance to check for surface coincidence                                   |

**Returns**

-1 = not a boundary surface, 0 = boundary surface corresponding to umin, 1 = boundary surface corresponding to umax, 2 = boundary surface corresponding to vmin, 3 = boundary surface corresponding to vmax, 4 = boundary surface corresponding to wmin, 5 = boundary surface corresponding to wmax

28.15.2.2 `std::vector<shared_ptr<ftVolume>> Go::ftVolumeTools::splitVolumes ( shared_ptr< ftVolume > & vol1, shared_ptr< ftVolume > & vol2, double eps, std::vector< int > & config )`

Split two volumes with regard to the intersections between the boundary surfaces corresponding to these two volumes

28.15.2.3 `std::vector<shared_ptr<ftVolume>> Go::ftVolumeTools::splitVolumes ( shared_ptr< ftVolume > & vol, shared_ptr< ftSurface > & face, double eps )`

Split one volume according to intersections with a given face.

28.15.2.4 `void Go::ftVolumeTools::updateWithSplitFaces ( shared_ptr< SurfaceModel > shell, shared_ptr< ftSurface > & face1, shared_ptr< ftSurface > & face2, std::vector< std::pair< ftEdge *, ftEdge * > > & replaced_wires )`

Specific functionality. Used from `ftVolume::generateMissingBdSurf`.

## 28.16 Go::GapRemoval Namespace Reference

Functionality for removal of gaps between two adjacent surfaces of various types.

**Functions**

- void `removeGapSpline` (`shared_ptr< SplineSurface > &srf1`, `shared_ptr< CurveOnSurface > &bd_cv1`, `double start1`, `double end1`, `shared_ptr< SplineSurface > &srf2`, `shared_ptr< CurveOnSurface > &bd_cv2`, `double start2`, `double end2`, `Point vertex1`, `Point vertex2`, `double epsge`, `bool *same_orientation=NULL`)
- `double removeGapTrim` (`shared_ptr< CurveOnSurface > &bd_cv1`, `double start1`, `double end1`, `shared_ptr< CurveOnSurface > &bd_cv2`, `double start2`, `double end2`, `Point vertex1`, `Point vertex2`, `double epsge`)
- `bool removeGapSplineTrim` (`shared_ptr< SplineSurface > &srf1`, `std::vector< shared_ptr< CurveOnSurface > > &bd_cv1`, `std::vector< double > start1`, `std::vector< double > end1`, `std::vector< shared_ptr< CurveOnSurface > > &bd_cv2`, `std::vector< double > start2`, `std::vector< double > end2`, `Point vertex1`, `Point vertex2`, `double epsge`)
- void `modifySplineSf` (`shared_ptr< ParamSurface > &srf1`, `std::vector< shared_ptr< CurveOnSurface > > &bd_cv1`, `std::vector< double > start1`, `std::vector< double > end1`, `shared_ptr< ParamSurface > &srf2`, `std::vector< shared_ptr< CurveOnSurface > > &bd_cv2`, `std::vector< double > start2`, `std::vector< double > end2`, `double epsge`)
- void `modifySplines` (`shared_ptr< ParamSurface > &srf1`, `std::vector< shared_ptr< CurveOnSurface > > &bd_cv1`, `std::vector< double > &start1`, `std::vector< double > &end1`, `shared_ptr< ParamSurface > &srf2`, `std::vector< shared_ptr< CurveOnSurface > > &bd_cv2`, `std::vector< double > &start2`, `std::vector< double > &end2`, `std::vector< Point > &vertex`, `double epsge`)
- void `removeGapSpline2` (`std::vector< shared_ptr< CurveOnSurface > > &bd_cv1`, `std::vector< double > &start1`, `std::vector< double > &end1`, `std::vector< shared_ptr< CurveOnSurface > > &bd_cv2`, `std::vector< double > &start2`, `std::vector< double > &end2`, `std::vector< Point > &vertex`, `double epsge`)

- `std::vector< shared_ptr< sideConstraintSet > >` `getCoefConstraints` (`shared_ptr< SplineCurve >` &crv1, `int` idx1, `shared_ptr< SplineCurve >` &crv2, `int` idx2, `double` tol)
- `shared_ptr< SplineCurve >` `replaceCurvePiece` (`shared_ptr< SplineCurve >` crv, `shared_ptr< SplineCurve >` sub\_crv, `double` par1, `int` cont1, `double` par2, `int` cont2)
- `shared_ptr< SplineSurface >` `getSplineAndBd` (`shared_ptr< ParamSurface >` psurf, `std::vector< shared_ptr< CurveOnSurface > >` &bd\_crvs)
- `void` `getBoundarySamples` (`std::vector< shared_ptr< CurveOnSurface > >` &all\_bd, `std::vector< shared_ptr< CurveOnSurface > >` &bd\_cvs, `std::vector< double >` &pts, `std::vector< double >` &pars, `std::vector< int >` &bd\_idx, `double` epsge)
- `bool` `modifyAtVertex` (`shared_ptr< SplineSurface >` srf, `Point` face\_param, `Point` vertex, `double` epsge)
- `void` `checkBoundaryDist` (`shared_ptr< CurveOnSurface >` bd1, `shared_ptr< CurveOnSurface >` bd2, `double` start1, `double` end1, `double` start2, `double` end2, `int` nmb\_sample, `double` &mdist1, `double` &mdist2)
- `void` `removeGapSpline` (`shared_ptr< SplineVolume >` &vol1, `shared_ptr< SurfaceOnVolume >` &bd\_sf1, `double` sf1\_start1, `double` sf1\_end1, `double` sf1\_start2, `double` sf1\_end2, `shared_ptr< SplineVolume >` &vol2, `shared_ptr< SurfaceOnVolume >` &bd\_sf2, `double` sf2\_start1, `double` sf2\_end1, `double` sf2\_start2, `double` sf2\_end2, `Point` vertex\_ll, `Point` vertex\_ur, `double` epsge, `int` orientation)

### 28.16.1 Detailed Description

Functionality for removal of gaps between two adjacent surfaces of various types.

Removal of gaps between two adjacent volumes No implementation

### 28.16.2 Function Documentation

**28.16.2.1** `void` `Go::GapRemoval::checkBoundaryDist` ( `shared_ptr< CurveOnSurface >` *bd1*, `shared_ptr< CurveOnSurface >` *bd2*, `double` *start1*, `double` *end1*, `double` *start2*, `double` *end2*, `int` *nmb\_sample*, `double` & *mdist1*, `double` & *mdist2* )

Compute the distance between trimming curves on adjacent surfaces along a common boundary.

#### Parameters

|               |                                                                                    |
|---------------|------------------------------------------------------------------------------------|
| <i>bd1</i>    | trimming curve on first surface                                                    |
| <i>bd2</i>    | trimming curve on second surface                                                   |
| <i>start1</i> | start parameter to the relevant piece of bd1                                       |
| <i>end1</i>   | end parameter to the relevant piece of bd1                                         |
| <i>start2</i> | start parameter to the relevant piece of bd2                                       |
| <i>end2</i>   | end parameter to the relevant piece of bd2                                         |
| <i>mdist1</i> | maximum distance between given trimming curves                                     |
| <i>mdist2</i> | maximum distance between the trimming curves projected onto the associated surface |

**28.16.2.2** `void` `Go::GapRemoval::getBoundarySamples` ( `std::vector< shared_ptr< CurveOnSurface > >` & *all\_bd*, `std::vector< shared_ptr< CurveOnSurface > >` & *bd\_cvs*, `std::vector< double >` & *pts*, `std::vector< double >` & *pars*, `std::vector< int >` & *bd\_idx*, `double` *epsge* )

Given the trimming curves of a bounded surface. Sort the curves followin boundaries of the underlying surface into the vector `bd_cvs`. The remaining curves are sampled and the corresponding points and parameter values are returned in the vectors `pts` and `pars`.

28.16.2.3 `std::vector<shared_ptr<sideConstraintSet> > Go::GapRemoval::getCoefConstraints ( shared_ptr<SplineCurve > & crv1, int idx1, shared_ptr< SplineCurve > & crv2, int idx2, double tol )`

Define linear constraints between coefficients on two curves entities representing the same trace to make the trace of the two curves identical.

28.16.2.4 `shared_ptr<SplineSurface> Go::GapRemoval::getSplineAndBd ( shared_ptr< ParamSurface > psurf, std::vector< shared_ptr< CurveOnSurface > > & bd_crvs )`

Fetch the underlying spline surface and trimming curves from a bounded surface

28.16.2.5 `bool Go::GapRemoval::modifyAtVertex ( shared_ptr< SplineSurface > srf, Point face_param, Point vertex, double epsge )`

Modify a surface to adapt to a given point, vertex, in the parameter value face\_param.

28.16.2.6 `void Go::GapRemoval::modifySplines ( shared_ptr< ParamSurface > & srf1, std::vector< shared_ptr< CurveOnSurface > > & bd_cv1, std::vector< double > & start1, std::vector< double > & end1, shared_ptr< ParamSurface > & srf2, std::vector< shared_ptr< CurveOnSurface > > & bd_cv2, std::vector< double > & start2, std::vector< double > & end2, std::vector< Point > & vertex, double epsge )`

Modify the underlying spline surface of srf1 to improve the adaption to given trimming curves.

28.16.2.7 `void Go::GapRemoval::modifySplineSf ( shared_ptr< ParamSurface > & srf1, std::vector< shared_ptr< CurveOnSurface > > & bd_cv1, std::vector< double > start1, std::vector< double > end1, shared_ptr< ParamSurface > & srf2, std::vector< shared_ptr< CurveOnSurface > > & bd_cv2, std::vector< double > start2, std::vector< double > end2, double epsge )`

Modify the underlying spline surface of srf1 to improve the adaption to given trimming curves.

28.16.2.8 `void Go::GapRemoval::removeGapSpline ( shared_ptr< SplineVolume > & vol1, shared_ptr< SurfaceOnVolume > & bd_sf1, double sf1_start1, double sf1_end1, double sf1_start2, double sf1_end2, shared_ptr< SplineVolume > & vol2, shared_ptr< SurfaceOnVolume > & bd_sf2, double sf2_start1, double sf2_end1, double sf2_start2, double sf2_end2, Point vertex_ll, Point vertex_ur, double epsge, int orientation )`

We average the volumes along the matching faces on the rectangular domain given by vertex lower left and upper right (assuming that such a domain is well defined).

28.16.2.9 `void Go::GapRemoval::removeGapSpline ( shared_ptr< SplineSurface > & srf1, shared_ptr< CurveOnSurface > & bd_cv1, double start1, double end1, shared_ptr< SplineSurface > & srf2, shared_ptr< CurveOnSurface > & bd_cv2, double start2, double end2, Point vertex1, Point vertex2, double epsge, bool * same_orientation = NULL )`

Both surfaces are non-trimmed spline surfaces and the common boundary are described by [CurveOnSurface](#) entities Remove gap by averaging coefficients at the common boundary. May increase the associated spline spaces.

```
28.16.2.10 void Go::GapRemoval::removeGapSpline2 (std::vector< shared_ptr< CurveOnSurface > > & bd_cv1,
std::vector< double > & start1, std::vector< double > & end1, std::vector< shared_ptr< CurveOnSurface
> > & bd_cv2, std::vector< double > & start2, std::vector< double > & end2, std::vector< Point > &
vertex, double epsge)
```

Both surfaces are non-trimmed spline surfaces and the common boundary are described by [CurveOnSurface](#) entities. Remove gap by defining conditions between coefficients at the common boundary between the two surface and modify both surfaces accordingly.

```
28.16.2.11 bool Go::GapRemoval::removeGapSplineTrim (shared_ptr< SplineSurface > & srf1, std::vector<
shared_ptr< CurveOnSurface > > & bd_cv1, std::vector< double > start1, std::vector< double > end1,
std::vector< shared_ptr< CurveOnSurface > > & bd_cv2, std::vector< double > start2, std::vector<
double > end2, Point vertex1, Point vertex2, double epsge)
```

Both surfaces are trimmed and have an underlying spline surface. The common boundary are described by [CurveOnSurface](#) entities limited by bounding parameters. Sub curves of the same curve on surface entity may occur several times, but are split to have correspondance between the two surfaces along the common boundary.

```
28.16.2.12 double Go::GapRemoval::removeGapTrim (shared_ptr< CurveOnSurface > & bd_cv1, double start1,
double end1, shared_ptr< CurveOnSurface > & bd_cv2, double start2, double end2, Point vertex1,
Point vertex2, double epsge)
```

Both surfaces are trimmed of some type and the common boundary are described by [CurveOnSurface](#) entities.

```
28.16.2.13 shared_ptr< SplineCurve > Go::GapRemoval::replaceCurvePiece (shared_ptr< SplineCurve > crv,
shared_ptr< SplineCurve > sub_crv, double par1, int cont1, double par2, int cont2)
```

Replace a sub curve (between par1 and par2) in a given spline curve by a modified curve piece. The continuity at the joints are given by cont1 and cont2.

## 28.17 Go::GeometryTools Namespace Reference

### Functions

- int [GO\\_API analyzePeriodicity](#) (const [SplineCurve](#) &cv, double knot\_tol=1e-12)
- int [GO\\_API analyzePeriodicity](#) (const [SplineSurface](#) &sf, int direction, double knot\_tol=1e-12)
- int [GO\\_API analyzePeriodicity](#) (const [BsplineBasis](#) &basis, double knot\_tol=1e-12)
- int [GO\\_API analyzePeriodicityDerivs](#) (const [ParamCurve](#) &cv, int max\_derivs, double tol=1e-14)
- int [GO\\_API analyzePeriodicityDerivs](#) (const [SplineSurface](#) &sf, int direction, int max\_derivs, double tol=1e-14)
- shared\_ptr< [SplineCurve](#) > [GO\\_API curveSum](#) (const [SplineCurve](#) &crv1, double fac1, const [SplineCurve](#) &crv2, double fac2, double num\_tol=1e-05)
- shared\_ptr< [SplineSurface](#) > [GO\\_API surfaceSum](#) (const [SplineSurface](#) &sf1, double fac1, const [SplineSurface](#) &sf2, double fac2, double num\_tol=1e-05)
- void [GO\\_API estimateSurfaceSize](#) (const [ParamSurface](#) &srf, double &length\_u, double &length\_v, double \*area=NULL)
- void [GO\\_API estimateIsoCurveLength](#) (const [SplineSurface](#) &srf, bool dir\_u, double par, double &length)
- bool [GO\\_API degenerateToCurve](#) (const [SplineSurface](#) &srf, bool dir\_u, double tol)
- void [GO\\_API makeBdDegenerate](#) ([SplineSurface](#) &srf, int bd\_idx)

*Make a specified surface boundary exactly degenerate.*

- void `GO_API makeUnionKnots` (std::vector< [BsplineBasis](#) > &bbasis, double tol, std::vector< double > &union\_knots)  
*Compute the union of a set of knot vectors.*
- void `GO_API makeUnionKnots` (std::vector< std::vector< double > > &knots, double tol, std::vector< double > &union\_knots)
- bool `GO_API checkConstantCoef` ([SplineCurve](#) &cv, int idx, double val, double max\_dist, double tol)
- void `GO_API setSfBdCoefToConst` ([SplineSurface](#) &srf, int bd\_idx, int idx\_d, double val, double deg\_tol)
- void `GO_API findDominant` (const [SplineSurface](#) &surface, [Vector3D](#) &dominant\_u, [Vector3D](#) &dominant\_v)
- void `GO_API getGnJoints` (const [ParamCurve](#) &curve, const std::vector< double > &cont, std::vector< double > &gn\_joints)
- void `GO_API getGnJoints` (const [CurveLoop](#) &loop, const std::vector< double > &cont, std::vector< std::vector< double > > &gn\_joints)
- bool `GO_API isCoincident` (const [ParamCurve](#) &cv1, const [ParamCurve](#) &cv2, double epsge)
- bool `GO_API negativeProj` (const [SplineSurface](#) &surface, const [Array](#)< [Vector3D](#), 2 > &refvector, const double eps=0.0)
- shared\_ptr< [ParamCurve](#) > `GO_API projectCurve` (shared\_ptr< [ParamCurve](#) > incurve, const [Point](#) &normal, bool planar)
- shared\_ptr< [SplineCurve](#) > `GO_API projectCurve` (const [SplineCurve](#) &incurve, const [Point](#) &normal, bool planar)
- shared\_ptr< [SplineCurve](#) > `GO_API representSurfaceAsCurve` (const [SplineSurface](#) &surface, int cv\_dir)
- shared\_ptr< [SplineSurface](#) > `GO_API representCurveAsSurface` (const [SplineCurve](#) &curve, int cv\_dir, const [BsplineBasis](#) &other\_bas, bool rational)
- std::vector< double > `GO_API getRotationMatrix` (const [Point](#) &unit\_axis\_dir, double alpha)
- void `GO_API rotateSplineSurf` ([Point](#) rot\_axis, double alpha, [SplineSurface](#) &sf)
- void `GO_API rotateSplineCurve` ([Point](#) rot\_axis, double alpha, [SplineCurve](#) &cv)
- void `GO_API rotateLineCloud` ([Point](#) rot\_axis, double alpha, [LineCloud](#) &lc)
- void `GO_API rotatePoint` ([Point](#) rot\_axis, double alpha, double \*space\_pt)
- void `GO_API rotatePoint` ([Point](#) rot\_axis, double alpha, [Point](#) &space\_pt)
- void `GO_API splitCurveIntoSegments` (const [SplineCurve](#) &cv, std::vector< [SplineCurve](#) > &seg)  
*Split a spline curve into Bezier segments.*
- std::vector< shared\_ptr< [SplineSurface](#) > > `GO_API splitInKinks` (const [SplineSurface](#) &sf, const std::vector< double > &u\_kinks, const std::vector< double > &v\_kinks)  
*Extract sub patches from the surface given by input parameters.*
- void `GO_API splitSurfaceIntoPatches` (const [SplineSurface](#) &sf, std::vector< [SplineSurface](#) > &pat)  
*Splits a spline surface into Bezier patches.*
- void `GO_API surfaceKinks` (const [SplineSurface](#) &sf, double max\_normal\_angle, std::vector< double > &g1\_disc\_u, std::vector< double > &g1\_disc\_v, bool compute\_g1\_disc=true)
- void `GO_API curveKinks` (const [SplineCurve](#) &cv, double tol, double ang\_tol, std::vector< double > &c1\_disconts, std::vector< double > &g1\_disconts)  
*Find parameter values where a curve is G1- or C1-discontinuous.*
- void `GO_API translateSplineSurf` (const [Point](#) &trans\_vec, [SplineSurface](#) &sf)  
*Translate the given [SplineSurface](#) by trans\_vec.*
- void `GO_API translateSplineCurve` (const [Point](#) &trans\_vec, [SplineCurve](#) &cv)  
*Translate the given [SplineCurve](#) by trans\_vec.*
- void `GO_API translateLineCloud` (const [Point](#) &trans\_vec, [LineCloud](#) &lc)  
*Translate the given [LineCloud](#) by trans\_vec.*
- void `GO_API averageBoundaryCoefs` (shared\_ptr< [SplineSurface](#) > &srf1, int bd1, bool keep\_first, shared\_ptr< [SplineSurface](#) > &srf2, int bd2, bool keep\_second, bool found\_corner1, [Point](#) corner1, bool found\_corner2, [Point](#) corner2, bool opposite)
- void `GO_API unifyCurveSplineSpace` (std::vector< shared\_ptr< [SplineCurve](#) > > &curves, double tol)
- void `GO_API unifySurfaceSplineSpace` (std::vector< shared\_ptr< [SplineSurface](#) > > &surfaces, double tol, int dir=0)
- void `GO_API unifySurfaceSplineSpaceOneDir` (std::vector< shared\_ptr< [SplineSurface](#) > > &surfaces, double tol, bool unify\_u\_dir)



- `shared_ptr< SplineSurface > GO_API joinPatches (const std::vector< shared_ptr< SplineSurface > > &patches, const SplineSurface &spline_space)`
- `void GO_API insertKnotsEvenly (BsplineBasis &basis, int num_knots)`
- `void GO_API insertKnotsEvenly (BsplineBasis &basis, double tmin, double tmax, int num_knots, double knot_diff_tol=1e-05)`
- `double GO_API getKnotAtLargestInterval (const BsplineBasis &basis)`
- `std::pair< double, double > GO_API getLargestParameterInterval (const BsplineBasis &basis)`
- `void setParameterDomain (std::vector< shared_ptr< BoundedSurface > > &sfs, double u1, double u2, double v1, double v2)`

*Reparameterize a set of bounded surfaces to the same domain.*

## 28.17.1 Function Documentation

28.17.1.1 `int GO_API Go::GeometryTools::analyzePeriodicity ( const SplineCurve & cv, double knot_tol = 1e-12 )`

Analyze periodicity of curve based on number of repeating knots and control points. The return value is -1 if the curve ends are disjoint, otherwise k if cv is  $C^k$  continuous. These are sufficient but not necessary conditions for periodicity, so it is possible that a call to [analyzePeriodicityDerivs\(\)](#) will yield a higher degree of periodicity.

### Parameters

|                 |                                                   |
|-----------------|---------------------------------------------------|
| <i>cv</i>       | the curve to analyse                              |
| <i>knot_tol</i> | the tolerance used when comparing knot intervals. |

### Returns

-1 if the curve ends are disjoint, or k if the curve is proven to be  $C^k$  continuous.

28.17.1.2 `int GO_API Go::GeometryTools::analyzePeriodicity ( const SplineSurface & sf, int direction, double knot_tol = 1e-12 )`

Analyze periodicity of surface based on number of repeating knots and control points. The return value is -1 if the surface edges are disjoint, otherwise k if sf is  $C^k$  continuous across the seam. These are sufficient but not necessary conditions for periodicity, so it is possible that a call to [analyzePeriodicityDerivs\(\)](#) will yield a higher degree of periodicity. The current implementation is quite slow, and not optimized for speed.

### Parameters

|                  |                                                                                                                                          |
|------------------|------------------------------------------------------------------------------------------------------------------------------------------|
| <i>sf</i>        | reference to the <a href="#">SplineSurface</a> to be analyzed                                                                            |
| <i>direction</i> | specify 'direction' to be '0' to check for periodicity in the first parameter direction, or '1' to check the second parameter direction. |
| <i>knot_tol</i>  | the tolerance used when comparing knot intervals                                                                                         |

### Returns

-1 if the surface edges are disjoint, otherwise k if the



28.17.1.3 `int GO_API Go::GeometryTools::analyzePeriodicity ( const BsplineBasis & basis, double knot_tol = 1e-12 )`

Analyze periodicity of basis based on number of repeating knots. The return value is -1 if there is no repeating (or order-tuple end knots), up to k if there are (order-1-k)-tuple end knots and sufficient repeats for a  $C^k$  curve. This is mostly a helper function for the curve and surface analysis functions.

#### Parameters

|                 |                                                            |
|-----------------|------------------------------------------------------------|
| <i>basis</i>    | reference to the <a href="#">BsplineBasis</a> to analyze   |
| <i>knot_tol</i> | the tolerance used when comparing knots and knot intervals |

#### Returns

-1 if there is no repeating (or order-tuple end knots), up to k if there are (order-1-k)-tuple end knots and sufficient repeats for a  $C^k$  curve.

28.17.1.4 `int GO_API Go::GeometryTools::analyzePeriodicityDerivs ( const ParamCurve & cv, int max_derivs, double tol = 1e-14 )`

Analyze periodicity of curve based on evaluating derivatives at both endpoints. The return value is -1 if the curve ends are disjoint, otherwise k if cv is  $C^k$  continuous. A maximum of max\_derivs derivatives are computed, so the analysis cannot yield a higher return value than max\_derivs. The tolerance is the amount the point and derivatives are allowed to be different at the two ends. The default is quite tight!

#### Parameters

|                   |                                                                                                   |
|-------------------|---------------------------------------------------------------------------------------------------|
| <i>cv</i>         | the curve to be analyzed for periodicity                                                          |
| <i>max_derivs</i> | the maximum of computed derivatives (the analysis can not detect continuities beyond this value). |
| <i>tol</i>        | the tolerance used when comparing points and derivatives for approximate equality                 |

#### Returns

-1 if the curve ends are disjoint, k if the curve is proven to be  $C^k$  continuous.

28.17.1.5 `int GO_API Go::GeometryTools::analyzePeriodicityDerivs ( const SplineSurface & sf, int direction, int max_derivs, double tol = 1e-14 )`

Analyze periodicity of surface based on evaluating derivatives at opposing ends of the surface. The return value is -1 if the surface edges are disjoint, otherwise k if sf is  $C^k$  continuous across the seam. These are sufficient but not necessary conditions for periodicity, so it is possible that a call to [analyzePeriodicityDerivs\(\)](#) will yield a higher degree of periodicity. The current implementation is quite slow, and not optimized for speed.

#### Parameters

|                   |                                                                                                                           |
|-------------------|---------------------------------------------------------------------------------------------------------------------------|
| <i>sf</i>         | the <a href="#">SplineSurface</a> to analyze for periodicity                                                              |
| <i>direction</i>  | the parameter direction to check for periodicity (0 to check in the first parameter direction, 1 to check in the second). |
| <i>max_derivs</i> | the maximum of computed derivatives (the analysis cannot detect continuities beyond this value)                           |
| <i>tol</i>        | the tolerance used when comparing points and derivatives for approximate equality.                                        |

## Returns

-1 if the surface edges are disjoint, or k if the surface is proven to be  $C^k$  continuous across the seam.

28.17.1.6 `void GO_API Go::GeometryTools::averageBoundaryCoefs ( shared_ptr< SplineSurface > & srf1, int bd1, bool keep_first, shared_ptr< SplineSurface > & srf2, int bd2, bool keep_second, bool found_corner1, Point corner1, bool found_corner2, Point corner2, bool opposite )`

Average specified boundary coefficients between two spline surfaces to ensure a C0 transition

28.17.1.7 `bool GO_API Go::GeometryTools::checkConstantCoef ( SplineCurve & cv, int idx, double val, double max_dist, double tol )`

Check if a curve coefficient is equal to a constant in a specified dimension provided it already lies close

28.17.1.8 `void GO_API Go::GeometryTools::curveKinks ( const SplineCurve & cv, double tol, double ang_tol, std::vector< double > & c1_disconts, std::vector< double > & g1_disconts )`

Find parameter values where a curve is G1- or C1-discontinuous.

28.17.1.9 `shared_ptr<SplineCurve> GO_API Go::GeometryTools::curveSum ( const SplineCurve & crv1, double fac1, const SplineCurve & crv2, double fac2, double num_tol = 1e-05 )`

Addition of two signed SplineCurves, i.e. this function can also be used for subtraction. The curves is assumed to live on the same parameter domain, but may have different knot vectors. resulting curve = fac1 \* crv1 + fac2 \* crv2

## Parameters

|                |                                                                                |
|----------------|--------------------------------------------------------------------------------|
| <i>crv1</i>    | the first of the curves to be added                                            |
| <i>fac1</i>    | multiplicative factor of the first curve                                       |
| <i>crv2</i>    | the second of the curves to be added                                           |
| <i>fac2</i>    | multiplicative factor of the second curve                                      |
| <i>num_tol</i> | tolerance used when unifying the spline spaces in which 'crv1' and 'crv2' lie. |

## Returns

shared pointed to a newly created [SplineCurve](#) which is the sum of 'crv1' and 'crv2'.

28.17.1.10 `bool GO_API Go::GeometryTools::degenerateToCurve ( const SplineSurface & srf, bool dir_u, double tol )`

Check if a given spline surface degenerates to a curve within a given tolerance

28.17.1.11 `void GO_API Go::GeometryTools::estimateIsoCurveLength ( const SplineSurface & srf, bool dir_u, double par, double & length )`

estimate the length of an iso-curve on the surface

## Parameters

|               |                                                                                                                                    |
|---------------|------------------------------------------------------------------------------------------------------------------------------------|
| <i>srf</i>    | the surface containing the iso-curve                                                                                               |
| <i>dir_u</i>  | 'true' if the iso-curve is along the first parameter (second parameter fixed), 'false' if it is the first parameter that is fixed. |
| <i>par</i>    | parameter value for the fixed parameter                                                                                            |
| <i>length</i> | returns the estimated length of the iso-curve                                                                                      |

28.17.1.12 void **GO\_API** Go::GeometryTools::estimateSurfaceSize ( const ParamSurface & *srf*, double & *length\_u*, double & *length\_v*, double \* *area* = NULL )

Rough estimate of the size of a parametric surface

## Parameters

|                 |                                                                                |
|-----------------|--------------------------------------------------------------------------------|
| <i>srf</i>      | the surface we want to estimate the size of                                    |
| <i>length_u</i> | the estimated average length of the surface, moving along the first parameter  |
| <i>length_v</i> | the estimated average length of the surface, moving along the second parameter |

28.17.1.13 void **GO\_API** Go::GeometryTools::findDominant ( const SplineSurface & *surface*, Vector3D & *dominant\_u*, Vector3D & *dominant\_v* )

Finds the "dominant u- and v-vectors" for a surface, defined to be the sum of all the vectors pointing from one control point to the next in the u- and v-directions

## Parameters

|                   |                                               |
|-------------------|-----------------------------------------------|
| <i>surface</i>    | the surface we want to analyze                |
| <i>dominant_u</i> | returns the dominant u-vector for the surface |
| <i>dominant_v</i> | returns the dominant v-vector for the surface |

28.17.1.14 void **GO\_API** Go::GeometryTools::getGnJoints ( const ParamCurve & *curve*, const std::vector< double > & *cont*, std::vector< double > & *gn\_joints* )

Partition the given curve into segments where each segment is at least  $G^n$  continuous (currently supporting up to  $G^2$  continuity).

## Parameters

|              |                                                                                                                                                                                                                                                                           |
|--------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>curve</i> | the curve to analyze                                                                                                                                                                                                                                                      |
| <i>cont</i>  | the tolerance used for defining continuity. <i>cont</i> [0] is the tolerance used for defining the $G^0$ continuity, <i>cont</i> [1] is used for checking $G^1$ continuity, etc. The length of this vector also determines 'n' ('n' is equal to <i>cont</i> .size() - 1). |

## Return values

|                  |                                                                                                                                                                   |
|------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>gn_joints</i> | returns the parameter values where one segment stops and the next begins. The vector will at least contain two values (the start and end parameter of the curve). |
|------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------|

**28.17.1.15** `void GO_API Go::GeometryTools::getGnJoints ( const CurveLoop & loop, const std::vector< double > & cont, std::vector< std::vector< double > > & gn_joints )`

Partition a given [CurveLoop](#) into segments where each segment is at least  $G^n$  continuous (currently supporting up to  $G^2$  continuity).

## Parameters

|                  |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>loop</i>      | the <a href="#">CurveLoop</a> to analyze                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| <i>cont</i>      | the tolerance used for defining continuity. <code>cont[0]</code> is the tolerance used for defining the $G^0$ continuity, <code>cont[1]</code> is used for checking $G^1$ continuity, etc. The length of this vector also determines 'n' ('n' is equal to <code>cont.size() - 1</code> ).                                                                                                                                                                                                                                                                                                                                                                                              |
| <i>gn_joints</i> | a vector of a vector, reporting the result of the analysis. The outer vector contains one entry per curve in the <a href="#">CurveLoop</a> . This entry contains the parameters for which that curve must be split in order to obtain $G^n$ segments. NB: the start- and end parameters are NOT included here, as opposed to the result obtained from the other <a href="#">getGnJoints()</a> function. The exception to this is when there is an $G^n$ discontinuity at the transition between the curve 'i-1' and curve 'i' in the <a href="#">CurveLoop</a> . To indicate this, the start parameter value for curve 'i' will be included in the corresponding entry in 'gn_joints'. |

**28.17.1.16** `double GO_API Go::GeometryTools::getKnotAtLargestInterval ( const BsplineBasis & basis )`

Return the mid parameter of the largest parameter interval in a given B-spline basis. If several knot intervals have the same size, the first is returned.

**28.17.1.17** `std::pair<double, double> GO_API Go::GeometryTools::getLargestParameterInterval ( const BsplineBasis & basis )`

Return the start and end parameter corresponding to the largest parameter interval in a given B-spline basis. If several knot intervals have the same size, the first is returned.

**28.17.1.18** `std::vector<double> GO_API Go::GeometryTools::getRotationMatrix ( const Point & unit_axis_dir, double alpha )`

Compute the elements of the matrix describing a rotation or a given number of radians around a given axis going through the origin.

## Parameters

|                      |                                                                                                                    |
|----------------------|--------------------------------------------------------------------------------------------------------------------|
| <i>unit_axis_dir</i> | this vector defines the axis of rotation. It must be of unit length, although this is not checked by the function. |
| <i>alpha</i>         | the angle of rotation, in radians                                                                                  |

**Returns**

a vector containing the matrix elements of the corresponding rotation matrix, stored row-wise.

**28.17.1.19** void **GO\_API** Go::GeometryTools::insertKnotsEvenly ( **BsplineBasis** & *basis*, int *num\_knots* )

Insert *num\_knots* new knots in the *num\_knots* largest knot intervals in the B-spline basis basis. The knots are inserted one by one. Thus, more than one knot may be inserted in one knot interval of the initial basis. And we do not end up with the most even distribution of knots possible.

**28.17.1.20** void **GO\_API** Go::GeometryTools::insertKnotsEvenly ( **BsplineBasis** & *basis*, double *tmin*, double *tmax*, int *num\_knots*, double *knot\_diff\_tol* = 1e-05 )

Insert *num\_knots* new knots in the *num\_knots* largest knot intervals in the B-spline basis basis in the parameter interval [*tmin*,*tmax*]. The knots are inserted one by one. Thus, more than one knot may be inserted in one knot interval of the initial basis. And we do not end up with the most even distribution of knots possible.

**28.17.1.21** bool **GO\_API** Go::GeometryTools::isCoincident ( const **ParamCurve** & *cv1*, const **ParamCurve** & *cv2*, double *epsge* )

Returns true if the curves are approximately coincident (that is, they have the same parameter space and overlap in space). The function uses a (possibly high) number of closest point calculations to check for spatial coincidence.

**Parameters**

|              |                                                                              |
|--------------|------------------------------------------------------------------------------|
| <i>cv1</i>   | the first curve to check for coincidence                                     |
| <i>cv2</i>   | the second curve to check for coincidence                                    |
| <i>epsge</i> | the tolerance for specifying what we mean by <i>approximately</i> coincident |

**Returns**

'true' if the curves are found to be approximately coincident, 'false' otherwise.

**28.17.1.22** shared\_ptr<**SplineSurface**> **GO\_API** Go::GeometryTools::joinPatches ( const std::vector< shared\_ptr<**SplineSurface**>> & *patches*, const **SplineSurface** & *spline\_space* )

Join patches with continuity according to input *basis\_u* & *basis\_v*. The patches are assumed to be in Bezier form, and form a continuous set. There should be *num\_u* x *num\_v* patches, where *num\_u* = *numcoefs\_u*/*order\_u* and *num\_v* = *numcoefs\_v*/*order\_v* (as given by *basis\_u* and *basis\_v*).

**28.17.1.23** void **GO\_API** Go::GeometryTools::makeBdDegenerate ( **SplineSurface** & *srf*, int *bd\_idx* )

Make a specified surface boundary exactly degenerate.

28.17.1.24 void **GO\_API** Go::GeometryTools::makeUnionKnots ( std::vector< **BsplineBasis** > & *bbasis*, double *tol*, std::vector< double > & *union\_knots* )

Compute the union of a set of knot vectors.

28.17.1.25 void **GO\_API** Go::GeometryTools::makeUnionKnots ( std::vector< std::vector< double > > & *knots*, double *tol*, std::vector< double > & *union\_knots* )

28.17.1.26 bool **GO\_API** Go::GeometryTools::negativeProj ( const **SplineSurface** & *surface*, const Array< **Vector3D**, 2 > & *refvector*, const double *eps* = 0.0 )

Returns true if any vector difference between neighboring control points in the u- or v-directions has negative projection on the given reference vector.

#### Parameters

|                  |                                                                                                                                       |
|------------------|---------------------------------------------------------------------------------------------------------------------------------------|
| <i>surface</i>   | the <a href="#">SplineSurface</a> containing the control points                                                                       |
| <i>refvector</i> | the vector onto which we carry out the projection                                                                                     |
| <i>eps</i>       | tolerance used when defining whether a vector has a negative projection onto 'refvector' (the scalar product must be less than -eps). |

#### Returns

'true' if we have found a difference between neighboring control points that has a negative projection on the reference vector.

28.17.1.27 shared\_ptr<**ParamCurve**> **GO\_API** Go::GeometryTools::projectCurve ( shared\_ptr< **ParamCurve** > *incurve*, const **Point** & *normal*, bool *planar* )

Project a 3D parametric curve into a given plane. The curve can be returned either as a 3D curve lying in the plane or as a 2D curve. In the latter case, the coordinate system is rotated such that the given plane normal coincides with the z-axis.

#### Parameters

|                |                                                        |
|----------------|--------------------------------------------------------|
| <i>incurve</i> | the curve to project                                   |
| <i>normal</i>  | the normal to the plane of projection                  |
| <i>planar</i>  | 'true' if we want the returned curve to be a 2D curve. |

#### Returns

a shared pointer to a newly constructed, planar [SplineCurve](#) represented as a parametric curve expressing the projection of 'incurve' onto the given plane.

28.17.1.28 shared\_ptr<**SplineCurve**> **GO\_API** Go::GeometryTools::projectCurve ( const **SplineCurve** & *incurve*, const **Point** & *normal*, bool *planar* )

Project a 3D [SplineCurve](#) into a given plane. The curve can be returned either as a 3D curve lying in the plane or as a 2D curve. In the latter case, the coordinate system is rotated such that the given plane normal coincides with

the z-axis.

#### Parameters

|                |                                                        |
|----------------|--------------------------------------------------------|
| <i>incurve</i> | the curve to project                                   |
| <i>normal</i>  | the normal to the plane of projection                  |
| <i>planar</i>  | 'true' if we want the returned curve to be a 2D curve. |

#### Returns

a shared pointer to a newly constructed, planar [SplineCurve](#) expressing the projection of 'incurve' onto the given plane.

**28.17.1.29** `shared_ptr<SplineSurface> GO_API Go::GeometryTools::representCurveAsSurface ( const SplineCurve & curve, int cv_dir, const BsplineBasis & other_bas, bool rational )`

Describe a curve as surface in a given direction. The surface output is rational if the argument *rational* is set to 'true'. In that case the curve is supposed to live in homogenous space. Describe a curve as a lower-dimensional surface in a given direction.

#### Parameters

|                  |                                                                                                                                                                                                                                                                    |
|------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>curve</i>     | the curve that we want to express as a surface                                                                                                                                                                                                                     |
| <i>cv_dir</i>    | If this variable is set to '1', then the curve's parameter will become the <i>first</i> parameter in the generated surface. If it is set to '2', the curve's parameter will become the <i>second</i> parameter in the generated surface. Other values are illegal. |
| <i>other_bas</i> | the <a href="#">BsplineBasis</a> for the additional parameter direction.                                                                                                                                                                                           |
| <i>rational</i>  | define whether the generated surface shall be specified as <i>rational</i> or not.                                                                                                                                                                                 |

#### Returns

a shared pointer to a new [SplineSurface](#), expressing the curve in a space of lower dimensionality.

**28.17.1.30** `shared_ptr<SplineCurve> GO_API Go::GeometryTools::representSurfaceAsCurve ( const SplineSurface & surface, int cv_dir )`

Describe a surface as a high-dimensional curve in a given direction. If the surface is rational, the curve will be non-rational and living in the homogenous space.

#### Parameters

|                |                                                                                                                                                                                                                                                                                                                                                               |
|----------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>surface</i> | the surface to express as a curve                                                                                                                                                                                                                                                                                                                             |
| <i>cv_dir</i>  | the parameter direction that will be kept when defining the curve (the other one will disappear, as the control points in this direction will be lumped together and expressed as single control points in a higher-dimensional space. 'cv_dir' takes either the value '1' (keep the first parameter direction) or '2' (keep the second parameter direction). |

**Returns**

shared pointer to a new [SplineCurve](#), expressing the surface as a curve in a high-dimensional space.

28.17.1.31 void `GO_API Go::GeometryTools::rotateLineCloud ( Point rot_axis, double alpha, LineCloud & lc )`

Rotate the given [LineCloud](#) a certain angle around a given axis.

**Parameters**

|                 |                                                                                         |
|-----------------|-----------------------------------------------------------------------------------------|
| <i>rot_axis</i> | the axis of rotation. It does not have to be normalized, but must of course be nonzero. |
| <i>alpha</i>    | the angle of rotation, given in radians                                                 |
| <i>lc</i>       | reference to the <a href="#">LineCloud</a> that is to be rotated                        |

28.17.1.32 void `GO_API Go::GeometryTools::rotatePoint ( Point rot_axis, double alpha, double * space_pt )`

Rotate the given 3D point a certain angle around a certain axis.

**Parameters**

|                              |                                                                                                                                          |
|------------------------------|------------------------------------------------------------------------------------------------------------------------------------------|
| <i>rot_axis</i>              | the axis of rotation. It does not have to be normalized, but must of course be nonzero.                                                  |
| <i>alpha</i>                 | the angle of rotation, given in radians                                                                                                  |
| <i>space</i> ↔<br><i>_pt</i> | pointer to the memory location where the 3D coordinates of the point are stored. These will be overwritten with the rotated coordinates. |

28.17.1.33 void `GO_API Go::GeometryTools::rotatePoint ( Point rot_axis, double alpha, Point & space_pt )`

Rotate the given 3D point a certain angle around a certain axis.

**Parameters**

|                              |                                                                                              |
|------------------------------|----------------------------------------------------------------------------------------------|
| <i>rot_axis</i>              | the axis of rotation. It does not have to be normalized, but must of course be nonzero.      |
| <i>alpha</i>                 | the angle of rotation, given in radians                                                      |
| <i>space</i> ↔<br><i>_pt</i> | reference to teh point to be rotated. This will be overwritten with the rotated coordinates. |

28.17.1.34 void `GO_API Go::GeometryTools::rotateSplineCurve ( Point rot_axis, double alpha, SplineCurve & cv )`

Rotate the given [SplineCurve](#) a certain angle around a given axis.

**Parameters**

|                 |                                                                                         |
|-----------------|-----------------------------------------------------------------------------------------|
| <i>rot_axis</i> | the axis of rotation. It does not have to be normalized, but must of course be nonzero. |
| <i>alpha</i>    | the angle of rotation, given in radians                                                 |
| <i>cv</i>       | reference to the curve that is to be rotated                                            |



28.17.1.35 void **GO\_API** Go::GeometryTools::rotateSplineSurf ( Point *rot\_axis*, double *alpha*, SplineSurface & *sf* )

Rotate the given [SplineSurface](#) a certain angle around a given axis.

#### Parameters

|                 |                                                                                         |
|-----------------|-----------------------------------------------------------------------------------------|
| <i>rot_axis</i> | the axis of rotation. It does not have to be normalized, but must of course be nonzero. |
| <i>alpha</i>    | the angle of rotation, given in radians                                                 |
| <i>sf</i>       | reference to the surface that is to be rotated.                                         |

28.17.1.36 void Go::GeometryTools::setParameterDomain ( std::vector< shared\_ptr< BoundedSurface > > & *sfs*, double *u1*, double *u2*, double *v1*, double *v2* )

Reparameterize a set of bounded surfaces to the same domain.

28.17.1.37 void **GO\_API** Go::GeometryTools::setSfBdCoefToConst ( SplineSurface & *sf*, int *bd\_idx*, int *idx\_d*, double *val*, double *deg\_tol* )

Modify surface along specified boundary to match a specific constant in one direction. Currently only non-rational surfaces

28.17.1.38 void **GO\_API** Go::GeometryTools::splitCurveIntoSegments ( const SplineCurve & *cv*, std::vector< SplineCurve > & *seg* )

Split a spline curve into Bezier segments.

28.17.1.39 std::vector<shared\_ptr<SplineSurface>> **GO\_API** Go::GeometryTools::splitInKinks ( const SplineSurface & *sf*, const std::vector< double > & *u\_kinks*, const std::vector< double > & *v\_kinks* )

Extract sub patches from the surface given by input parameters.

28.17.1.40 void **GO\_API** Go::GeometryTools::splitSurfaceIntoPatches ( const SplineSurface & *sf*, std::vector< SplineSurface > & *pat* )

Splits a spline surface into Bezier patches.

28.17.1.41 void **GO\_API** Go::GeometryTools::surfaceKinks ( const SplineSurface & *sf*, double *max\_normal\_angle*, std::vector< double > & *g1\_disc\_u*, std::vector< double > & *g1\_disc\_v*, bool *compute\_g1\_disc* = true )

Surface assumed to be continuous. Return parameter values failing to achieve G1-continuity.

28.17.1.42 `shared_ptr<SplineSurface> GO_API Go::GeometryTools::surfaceSum ( const SplineSurface & sf1, double fac1, const SplineSurface & sf2, double fac2, double num_tol = 1e-05 )`

28.17.1.43 `void GO_API Go::GeometryTools::translateLineCloud ( const Point & trans_vec, LineCloud & lc )`

Translate the given [LineCloud](#) by `trans_vec`.

28.17.1.44 `void GO_API Go::GeometryTools::translateSplineCurve ( const Point & trans_vec, SplineCurve & cv )`

Translate the given [SplineCurve](#) by `trans_vec`.

28.17.1.45 `void GO_API Go::GeometryTools::translateSplineSurf ( const Point & trans_vec, SplineSurface & sf )`

Translate the given [SplineSurface](#) by `trans_vec`.

28.17.1.46 `void GO_API Go::GeometryTools::unifyCurveSplineSpace ( std::vector< shared_ptr< SplineCurve > > & curves, double tol )`

Make sure that a set of curves live on the same knot vector tol-equal knots are set equal (i.e. if they differ within tol).

28.17.1.47 `void GO_API Go::GeometryTools::unifySurfaceSplineSpace ( std::vector< shared_ptr< SplineSurface > > & surfaces, double tol, int dir = 0 )`

Make sure that a set of surfaces live on the same knot vectors tol-equal knots are set equal (i.e. if they differ within tol). `dir` 0 means both, 1 is u, 2 is v

28.17.1.48 `void GO_API Go::GeometryTools::unifySurfaceSplineSpaceOneDir ( std::vector< shared_ptr< SplineSurface > > & surfaces, double tol, bool unify_u_dir )`

Make sure that a set of surfaces live on the same knot vectors in one parameter direction. tol-equal knots are set equal (i.e. if they differ within tol). Nothing is changed in the other parameter direction (as opposed to [unifySurfaceSplineSpace\(\)](#) where orders are raised to same value for all surfaces in both directions) Warning! Objects being pointed to may be recreated inside function. Remember to update other shared pointers if needed.

#### Parameters

|                          |                                                                                                           |
|--------------------------|-----------------------------------------------------------------------------------------------------------|
| <code>surfaces</code>    | Surfaces to end up with common knot vectors                                                               |
| <code>tol</code>         | tolerance for identifying equal knot values                                                               |
| <code>unify_u_dir</code> | if 'true', unify bases in first parameter direction if 'false', unify bases in second parameter direction |

## 28.18 Go::HahnsSurfaceGen Namespace Reference

## Functions

- `std::vector< shared_ptr< Go::ParamSurface > > constructPolygonalSurface` (`std::vector< shared_ptr< Go::ParamCurve > > &bnd_curves`, `std::vector< shared_ptr< Go::ParamCurve > > &cross_curves`, `double neighbour_tol`, `double kink_tol`, `double knot_diff_tol`)
- `std::vector< shared_ptr< Go::ParamSurface > > constructHahnsSurface` (`std::vector< shared_ptr< Go::SplineCurve > > &bnd_curves`, `std::vector< shared_ptr< Go::SplineCurve > > &mod_cross_curves`, `double neighbour_tol`, `double kink_tol`, `double knot_diff_tol`)

### 28.18.1 Detailed Description

This namespace contains functions used to create a number of surface covering a whole defined by of a number of boundary curves ( $3 \leq \text{nmb\_cvs} \leq 6$ ). Hahns method is used in the construction. Typically used for patching a model in which one or more surfaces are missing.

### 28.18.2 Function Documentation

**28.18.2.1** `std::vector<shared_ptr<Go::ParamSurface> > Go::HahnsSurfaceGen::constructHahnsSurface ( std::vector< shared_ptr< Go::SplineCurve > > & bnd_curves, std::vector< shared_ptr< Go::SplineCurve > > & mod_cross_curves, double neighbour_tol, double kink_tol, double knot_diff_tol )`

Given input of `bnd_curves` and `cross_tangent` curves, create the corresponding Hahns Surfaces. Assuming input curves fulfill corner conditions. All curves expected to live in 3-dimensional space.

#### Parameters

|                         |                                                    |
|-------------------------|----------------------------------------------------|
| <i>bnd_curves</i>       | edge curves for the Hahns Surface.                 |
| <i>mod_cross_curves</i> | corresponding cross tangent curves.                |
| <i>neighbour_tol</i>    | allowed distance between corresponding end points. |
| <i>kink_tol</i>         | allowed angle between corresponding tangents.      |
| <i>knot_diff_tol</i>    | parametric tolerance for equality of knots.        |

#### Returns

vector containing the Hahns Surface.

**28.18.2.2** `std::vector<shared_ptr<Go::ParamSurface> > Go::HahnsSurfaceGen::constructPolygonalSurface ( std::vector< shared_ptr< Go::ParamCurve > > & bnd_curves, std::vector< shared_ptr< Go::ParamCurve > > & cross_curves, double neighbour_tol, double kink_tol, double knot_diff_tol )`

The routine which actually creates vector of blending surfaces, with given curves as their total bnd curve. # of return surfaces equals # of curves. All bnd curves share orientation (clockwise), and all cross curves point inwards. A missing cross curve is indicated by a 0 pointer. `bnd_curves` must form a loop. As we approximate modified cross curves, we include parameters to denote the exactness of the approximation. All curves expected to live in 3-dimensional space.

#### Parameters

|                   |                                    |
|-------------------|------------------------------------|
| <i>bnd_curves</i> | edge curves for the Hahns Surface. |
|-------------------|------------------------------------|

## Parameters

|                      |                                                    |
|----------------------|----------------------------------------------------|
| <i>cross_curves</i>  | corresponding cross tangent curves.                |
| <i>neighbour_tol</i> | allowed distance between corresponding end points. |
| <i>kink_tol</i>      | allowed angle between corresponding tangents.      |
| <i>knot_diff_tol</i> | parametric tolerance for equality of knots.        |

## Returns

vector containing the Hahns Surface.

## 28.19 Go::IntersectionUtils Namespace Reference

Various functions related to the intersection algorithms.

### Functions

- `shared_ptr< SplineCurve > create1DSplineCurve (const SplineCurve &cv, int dim_id)`
- `shared_ptr< SplineSurface > create1DSplineSurface (const SplineSurface &sf, int dim_id)`
- `shared_ptr< SplineCurve > splineCurveProduct (std::vector< shared_ptr< SplineCurve > > &cv, Alg2D↔Elem term)`
- `shared_ptr< SplineSurface > splineSurfaceProduct (std::vector< shared_ptr< SplineSurface > > &sf, Alg3DElem term)`
- `shared_ptr< SplineCurve > insertCvInAlgcv (const SplineCurve &cv, AlgObj2DInt *alg_obj2d_int)`
- `shared_ptr< SplineSurface > insertSfnAlgsf (const SplineSurface &sf, AlgObj3DInt *alg_obj3d_int)`
- `shared_ptr< SplineSurface > insertSfnAlgsf2 (const SplineSurface &sf, AlgObj3DInt *alg_obj3d_int)`
- `shared_ptr< SplineSurface > insertSfnImplObj (const SplineSurface &spline_sf, const Bernstein↔TetrahedralPoly &impl, const BaryCoordSystem3D &bc)`
- `double distImplRepresentationCompFunction (const SplineSurface &spline_sf, const BernsteinTetrahedral↔Poly &impl, const BaryCoordSystem3D &bc, const SplineSurface &comp_1d_sf, double upar, double vpar)`

### 28.19.1 Detailed Description

Various functions related to the intersection algorithms.

### 28.19.2 Function Documentation

28.19.2.1 `shared_ptr<SplineCurve> Go::IntersectionUtils::create1DSplineCurve ( const SplineCurve & cv, int dim_id )`

28.19.2.2 `shared_ptr<SplineSurface> Go::IntersectionUtils::create1DSplineSurface ( const SplineSurface & sf, int dim_id )`

28.19.2.3 `double Go::IntersectionUtils::distImplRepresentationCompFunction ( const SplineSurface & spline_sf, const BernsteinTetrahedralPoly & impl, const BaryCoordSystem3D & bc, const SplineSurface & comp_1d_sf, double upar, double vpar )`

- 28.19.2.4 `shared_ptr<SplineCurve> Go::IntersectionUtils::insertCvInAlgcv ( const SplineCurve & cv, AlgObj2DInt * alg_obj2d_int )`
- 28.19.2.5 `shared_ptr<SplineSurface> Go::IntersectionUtils::insertSflnAlgsf ( const SplineSurface & sf, AlgObj3DInt * alg_obj3d_int )`
- 28.19.2.6 `shared_ptr<SplineSurface> Go::IntersectionUtils::insertSflnAlgsf2 ( const SplineSurface & sf, AlgObj3DInt * alg_obj3d_int )`
- 28.19.2.7 `shared_ptr<SplineSurface> Go::IntersectionUtils::insertSflnImplObj ( const SplineSurface & spline_sf, const BernsteinTetrahedralPoly & impl, const BaryCoordSystem3D & bc )`
- 28.19.2.8 `shared_ptr<SplineCurve> Go::IntersectionUtils::splineCurveProduct ( std::vector< shared_ptr< SplineCurve > > & cv, Alg2DElem term )`
- 28.19.2.9 `shared_ptr<SplineSurface> Go::IntersectionUtils::splineSurfaceProduct ( std::vector< shared_ptr< SplineSurface > > & sf, Alg3DElem term )`

## 28.20 Go::LinDepUtils Namespace Reference

### Functions

- [bool isPeelable \(const LRSplineSurface &\)](#)
- `std::vector< LRBSpline2D * > unpeelableBasisFunctions (const LRSplineSurface &)`

### 28.20.1 Function Documentation

- 28.20.1.1 `bool Go::LinDepUtils::isPeelable ( const LRSplineSurface & )`
- 28.20.1.2 `std::vector<LRBSpline2D*> Go::LinDepUtils::unpeelableBasisFunctions ( const LRSplineSurface & )`

## 28.21 Go::LoftSurfaceCreator Namespace Reference

This namespace contains functions used to create lofted surfaces.

### Functions

- [SplineSurface \\* loftSurface \(std::vector< shared\\_ptr< SplineCurve > >::iterator first\\_curve, int nmb\\_crvs\)](#)
- [SplineSurface \\* loftSurface \(std::vector< shared\\_ptr< SplineCurve > >::iterator first\\_curve, std::vector< double >::iterator first\\_param, int nmb\\_crvs\)](#)
- [SplineSurface \\* loftSurface \(std::vector< shared\\_ptr< SplineCurve > > &loft\\_curves, int nmb\\_crvs\)](#)
- `std::vector< shared_ptr< SplineCurve > > unifiedCurvesCopy (std::vector< shared_ptr< SplineCurve > >::iterator first_curve, int nmb_crvs)`
- `void makeLoftParams (std::vector< shared_ptr< SplineCurve > >::const_iterator first_curve, int nmb_crvs, double param_length, std::vector< double > &params)`
- [SplineSurface \\* loftSurfaceFromUnifiedCurves \(std::vector< shared\\_ptr< SplineCurve > >::iterator first\\_↵ curve, std::vector< double >::iterator first\\_param, int nmb\\_crvs\)](#)
- [SplineSurface \\* loftNonrationalSurface \(std::vector< shared\\_ptr< SplineCurve > >::iterator first\\_curve, std::vector< double >::iterator first\\_param, int nmb\\_crvs\)](#)
- [SplineSurface \\* loftRationalSurface \(std::vector< shared\\_ptr< SplineCurve > >::iterator first\\_curve, std↵::vector< double >::iterator first\\_param, int nmb\\_crvs\)](#)

### 28.21.1 Detailed Description

This namespace contains functions used to create lofted surfaces.

### 28.21.2 Function Documentation

#### 28.21.2.1 `SplineSurface* Go::LoftSurfaceCreator::loftNonrationalSurface ( std::vector< shared_ptr< SplineCurve > >::iterator first_curve, std::vector< double >::iterator first_param, int nmb_crvs )`

Create a lofting surface interpolating the input curves in the input parameters. The input curves are expected to have identical B-spline space. Either all or none of the curves are expected to be rational. In case the curves are rational, we also loft the denominator function. In this way, we might risk that the resulting surface has negative control points, this must be handled by the calling function. The returned lofted surface will be non-rational if the curves are non-rational.

#### Parameters

|                    |                                                                          |
|--------------------|--------------------------------------------------------------------------|
| <i>first_curve</i> | iterator to first iso-curve in the lofted surface.                       |
| <i>first_param</i> | iso parameter to corresponding curve referred to by <i>first_curve</i> . |
| <i>nmb_crvs</i>    | the number of curves referred to by <i>first_curve</i> .                 |

#### Returns

pointer to the created lofting surface.

#### 28.21.2.2 `SplineSurface* Go::LoftSurfaceCreator::loftRationalSurface ( std::vector< shared_ptr< SplineCurve > >::iterator first_curve, std::vector< double >::iterator first_param, int nmb_crvs )`

Create a lofting surface interpolating the input rational curves in the input parameters by Hermite interpolation, to ensure positive denominator function. The input curves are expected to have identical B-spline space. All of the curves are expected to be rational. The returned lofted surface will also be rational.

#### Parameters

|                    |                                                                          |
|--------------------|--------------------------------------------------------------------------|
| <i>first_curve</i> | iterator to first iso-curve in the lofted surface.                       |
| <i>first_param</i> | iso parameter to corresponding curve referred to by <i>first_curve</i> . |
| <i>nmb_crvs</i>    | the number of curves referred to by <i>first_curve</i> .                 |

#### Returns

pointer to the created lofting surface.

#### 28.21.2.3 `SplineSurface* Go::LoftSurfaceCreator::loftSurface ( std::vector< shared_ptr< SplineCurve > >::iterator first_curve, int nmb_crvs )`

Create a lofting surface based on the input curves. The curves are not changed during the lofting process. The curves must all lie in the same space

## Parameters

|                    |                                                          |
|--------------------|----------------------------------------------------------|
| <i>first_curve</i> | iterator to first iso-curve in the lofted surface.       |
| <i>nmb_crvs</i>    | the number of curves referred to by <i>first_curve</i> . |

## Returns

pointer to the created lofting surface.

**28.21.2.4** `SplineSurface*` `Go::LoftSurfaceCreator::loftSurface ( std::vector< shared_ptr< SplineCurve > >::iterator first_curve, std::vector< double >::iterator first_param, int nmb_crvs )`

Create a lofting surface interpolating the input curves in the input parameters. The curves are not changed during the lofting process. The curves must all lie in the same space

## Parameters

|                    |                                                                          |
|--------------------|--------------------------------------------------------------------------|
| <i>first_curve</i> | iterator to first iso-curve in the lofted surface.                       |
| <i>first_param</i> | iso parameter to corresponding curve referred to by <i>first_curve</i> . |
| <i>nmb_crvs</i>    | the number of curves referred to by <i>first_curve</i> .                 |

## Returns

pointer to the created lofting surface.

**28.21.2.5** `SplineSurface*` `Go::LoftSurfaceCreator::loftSurface ( std::vector< shared_ptr< SplineCurve > > & loft_curves, int nmb_crvs )`

Create a lofting surface based on the input curves. The curves are changed during the lofting process. The curves must all lie in the same space.

## Parameters

|                    |                             |
|--------------------|-----------------------------|
| <i>loft_curves</i> | the input curves.           |
| <i>nmb_crvs</i>    | the number of input curves. |

## Returns

pointer to the created lofting surface.

**28.21.2.6** `SplineSurface*` `Go::LoftSurfaceCreator::loftSurfaceFromUnifiedCurves ( std::vector< shared_ptr< SplineCurve > >::iterator first_curve, std::vector< double >::iterator first_param, int nmb_crvs )`

Create a lofting surface interpolating the input curves in the input parameters. The input curves are expected to have identical B-spline space. Either all or none of the curves are expected to be rational. ??? The returned lofted surface will be rational if and only if the curves are rational.

## Parameters

|                    |                                                                          |
|--------------------|--------------------------------------------------------------------------|
| <i>first_curve</i> | iterator to first iso-curve in the lofted surface.                       |
| <i>first_param</i> | iso parameter to corresponding curve referred to by <i>first_curve</i> . |
| <i>nmb_crvs</i>    | the number of curves referred to by <i>first_curve</i> .                 |

## Returns

pointer to the created lofting surface.

**28.21.2.7** void `Go::LoftSurfaceCreator::makeLoftParams ( std::vector< shared_ptr< SplineCurve > >::const_iterator first_curve, int nmb_crvs, double param_length, std::vector< double > &params )`

Calculate iso parameters for the input curves. The curves are expected to be ordered, i.e. corresponding to increasing iso parameters. Curves are given iso-parameters in the range 0.0 to *param\_length*. All the parameter differences between two neighbouring curves will be proportional to the quadratic mean of the distance between corresponding control points, i.e. if, for two curves  $S=first\_curve[i]$  and  $T=first\_curve[i+1]$ ,  $Q$  is the quadratic mean of the distance between corresponding control points of  $S$  and  $T$ , and  $P$  is the difference between the calculated parameter values for  $S$  and  $T$ , then the ratio  $Q/P$  is the same for all  $i$ .

## Parameters

|                     |                                                   |
|---------------------|---------------------------------------------------|
| <i>first_curve</i>  | iterator to first iso curve.                      |
| <i>nmb_crvs</i>     | the number of input curves.                       |
| <i>param_length</i> | length of parameter domain.                       |
| <i>params</i>       | the computed iso parameters for the input curves. |

**28.21.2.8** `std::vector<shared_ptr<SplineCurve> > Go::LoftSurfaceCreator::unifiedCurvesCopy ( std::vector< shared_ptr< SplineCurve > >::iterator first_curve, int nmb_crvs )`

Create a vector of curves holding a copy of the input curves, but where the spline bases have been changed (reparametrized, knot inserted and order raised) so that the B-spline spaces for the new curves all are identical, and where all curves are now rational if at least one of the input curves were rational. The input curves are not changed during the process. The curves must all lie in the same space.

## Parameters

|                    |                                                          |
|--------------------|----------------------------------------------------------|
| <i>first_curve</i> | iterator to first input curve.                           |
| <i>nmb_crvs</i>    | the number of curves referred to by <i>first_curve</i> . |

## Returns

vector holding the unified curves.

## 28.22 Go::LoftVolumeCreator Namespace Reference

This namespace contains functions used to create lofted volumes.



## Functions

- [SplineVolume](#) \* [loftVolume](#) (std::vector< shared\_ptr< [SplineSurface](#) > >::iterator first\_surface, int nmb\_srfs)
- [SplineVolume](#) \* [loftVolume](#) (std::vector< shared\_ptr< [SplineSurface](#) > >::iterator first\_surface, std::vector< double >::iterator first\_param, int nmb\_srfs)
- std::vector< shared\_ptr< [SplineSurface](#) > > [unifiedSurfacesCopy](#) (std::vector< shared\_ptr< [SplineSurface](#) > >::iterator first\_surface, int nmb\_srfs)
- void [makeLoftParams](#) (std::vector< shared\_ptr< [SplineSurface](#) > >::const\_iterator first\_surface, int nmb\_crvs, double param\_length, std::vector< double > &params)
- [SplineVolume](#) \* [loftVolumeFromUnifiedSurfaces](#) (std::vector< shared\_ptr< [SplineSurface](#) > >::iterator first\_surface, std::vector< double >::iterator first\_param, int nmb\_srfs)
- [SplineVolume](#) \* [loftNonrationalVolume](#) (std::vector< shared\_ptr< [SplineSurface](#) > >::iterator first\_surface, std::vector< double >::iterator first\_param, int nmb\_srfs)
- [SplineVolume](#) \* [loftRationalVolume](#) (std::vector< shared\_ptr< [SplineSurface](#) > >::iterator first\_surface, std::vector< double >::iterator first\_param, int nmb\_srfs)

### 28.22.1 Detailed Description

This namespace contains functions used to create lofted volumes.

### 28.22.2 Function Documentation

#### 28.22.2.1 [SplineVolume](#)\* [Go::LoftVolumeCreator::loftNonrationalVolume](#) ( std::vector< shared\_ptr< [SplineSurface](#) > >::iterator *first\_surface*, std::vector< double >::iterator *first\_param*, int *nmb\_srfs* )

Create a lofting volume interpolating the input surfaces in the input parameters. For each parameter direction (u or v), the input surfaces are expected to have identical B-spline space. Either all or none of the surfaces are expected to be rational. In case the surfaces are rational, we also loft the denominator function. In this way, we might risk that the resulting volume has negative control points, this must be handled by the calling function. The returned lofted volume will non-rational if the surfaces are non-rational.

#### Parameters

|                      |                                                                              |
|----------------------|------------------------------------------------------------------------------|
| <i>first_surface</i> | iterator to first iso-surface in the lofted volume.                          |
| <i>first_param</i>   | iso parameter to corresponding surface referred to by <i>first_surface</i> . |
| <i>nmb_srfs</i>      | the number of surfaces referred to by <i>first_surface</i> .                 |

#### Returns

pointer to the created lofting volume.

#### 28.22.2.2 [SplineVolume](#)\* [Go::LoftVolumeCreator::loftRationalVolume](#) ( std::vector< shared\_ptr< [SplineSurface](#) > >::iterator *first\_surface*, std::vector< double >::iterator *first\_param*, int *nmb\_srfs* )

Create a lofting volume interpolating the input rational surfaces in the input parameters by Hermite interpolation, to ensure positive denominator function. For each parameter direction (u or v), the input surfaces are expected to have identical B-spline space. All of the surfaces are expected to be rational. The returned lofted volume will also be rational.

## Parameters

|                      |                                                                              |
|----------------------|------------------------------------------------------------------------------|
| <i>first_surface</i> | iterator to first iso-surface in the lofted volume.                          |
| <i>first_param</i>   | iso parameter to corresponding surface referred to by <i>first_surface</i> . |
| <i>nmb_srfs</i>      | the number of surfaces referred to by <i>first_surface</i> .                 |

## Returns

pointer to the created lofting volume.

**28.22.2.3** `SplineVolume* Go::LoftVolumeCreator::loftVolume ( std::vector< shared_ptr< SplineSurface > >::iterator first_surface, int nmb_srfs )`

Create a lofting volume based on the input surfaces. The surfaces are not changed during the lofting process. The surfaces must all lie in the same space

## Parameters

|                      |                                                              |
|----------------------|--------------------------------------------------------------|
| <i>first_surface</i> | iterator to first iso-surface in the lofted volume.          |
| <i>nmb_srfs</i>      | the number of surfaces referred to by <i>first_surface</i> . |

## Returns

pointer to the created lofting volume.

**28.22.2.4** `SplineVolume* Go::LoftVolumeCreator::loftVolume ( std::vector< shared_ptr< SplineSurface > >::iterator first_surface, std::vector< double >::iterator first_param, int nmb_srfs )`

Create a lofting volume interpolating the input surfaces in the input parameters. The surfaces are not changed during the lofting process. The surfaces must all lie in the same space

## Parameters

|                      |                                                                              |
|----------------------|------------------------------------------------------------------------------|
| <i>first_surface</i> | iterator to first iso-surface in the lofted volume.                          |
| <i>first_param</i>   | iso parameter to corresponding surface referred to by <i>first_surface</i> . |
| <i>nmb_srfs</i>      | the number of surfaces referred to by <i>first_surface</i> .                 |

## Returns

pointer to the created lofting volume.

**28.22.2.5** `SplineVolume* Go::LoftVolumeCreator::loftVolumeFromUnifiedSurfaces ( std::vector< shared_ptr< SplineSurface > >::iterator first_surface, std::vector< double >::iterator first_param, int nmb_srfs )`

Create a lofting volume interpolating the input surfaces in the input parameters. For each parameter direction (u or v), the input surfaces are expected to have identical B-spline space. Either all or none of the surfaces are expected to be rational. The returned lofted volume will be rational if and only if the surfaces are rational.

## Parameters

|                      |                                                                              |
|----------------------|------------------------------------------------------------------------------|
| <i>first_surface</i> | iterator to first iso-surface in the lofted volume.                          |
| <i>first_param</i>   | iso parameter to corresponding surface referred to by <i>first_surface</i> . |
| <i>nmb_srfs</i>      | the number of surfaces referred to by <i>first_surface</i> .                 |

## Returns

pointer to the created lofting volume.

**28.22.2.6** void Go::LoftVolumeCreator::makeLoftParams ( std::vector< shared\_ptr< SplineSurface > >::const\_iterator *first\_surface*, int *nmb\_crvs*, double *param\_length*, std::vector< double > & *params* )

Calculate iso parameters for the input surfaces. The surfaces are expected to be ordered, i.e. corresponding to increasing iso parameters. Surfaces are given iso-parameters in the range 0.0 to *param\_length*. All the parameter differences between two neighbouring surfaces will be proportional to the quadratic mean of the distance between corresponding control points, i.e. if, for two surfaces  $S=first\_surface[i]$  and  $T=first\_surface[i+1]$ ,  $Q$  is the quadratic mean of the distance between corresponding control points of  $S$  and  $T$ , and  $P$  is the difference between the calculated parameter values for  $S$  and  $T$ , then the ratio  $Q/P$  is the same for all  $i$ .

## Parameters

|                      |                                                     |
|----------------------|-----------------------------------------------------|
| <i>first_surface</i> | iterator to first iso surface.                      |
| <i>nmb_crvs</i>      | the number of input surfaces.                       |
| <i>param_length</i>  | length of parameter domain.                         |
| <i>params</i>        | the computed iso parameters for the input surfaces. |

**28.22.2.7** std::vector<shared\_ptr<SplineSurface> > Go::LoftVolumeCreator::unifiedSurfacesCopy ( std::vector< shared\_ptr< SplineSurface > >::iterator *first\_surface*, int *nmb\_srfs* )

Create a vector of surfaces holding a copy of the input surfaces, but where the spline bases have been changed (reparametrized, knot inserted and order raised) so that the B-spline spaces in the  $u$ -direction for the new surfaces all are identical, same with  $v$ -direction, and where all surfaces are now rational if at least one of the input surfaces were rational. The input surfaces are not changed during the process. The surfaces must all lie in the same space.

## Parameters

|                      |                                                              |
|----------------------|--------------------------------------------------------------|
| <i>first_surface</i> | iterator to first input surface.                             |
| <i>nmb_srfs</i>      | the number of surfaces referred to by <i>first_surface</i> . |

## Returns

vector holding the unified surfaces.

## 28.23 Go::LoopUtils Namespace Reference

## Functions

- void `representAsSurfaceCurves` (std::vector< shared\_ptr< [ParamCurve](#) > > &curves, shared\_ptr< [BoundedSurface](#) > surf, std::vector< shared\_ptr< [CurveOnSurface](#) > > &cvs\_on\_sf)
- bool `loopsCCW` (const std::vector< shared\_ptr< [Go::SplineCurve](#) > > &simple\_par\_loop, double space\_epsilon, double int\_tol)
- bool `paramsCCW` (const std::vector< shared\_ptr< [Go::CurveOnSurface](#) > > &loop, double space\_epsilon, double int\_tol)
- bool `loopsCCW` (const [CurveLoop](#) &loop, double int\_tol)
- bool `firstLoopInsideSecond` (const std::vector< shared\_ptr< [Go::CurveOnSurface](#) > > &first\_loop, const std::vector< shared\_ptr< [Go::CurveOnSurface](#) > > &second\_loop, double loop\_tol, double int\_tol)  
*Loops expected to be disjoint, except possibly share part of boundary.*
- bool `makeLoopCCW` (std::vector< shared\_ptr< [ParamCurve](#) > > &loop\_cvs, double tol)

### 28.23.1 Detailed Description

Functions for checking the orientation of loops (closed curves), and whether one loop on a surface encloses another.

### 28.23.2 Function Documentation

**28.23.2.1** bool `Go::LoopUtils::firstLoopInsideSecond` ( const std::vector< shared\_ptr< [Go::CurveOnSurface](#) > > & *first\_loop*, const std::vector< shared\_ptr< [Go::CurveOnSurface](#) > > & *second\_loop*, double *loop\_tol*, double *int\_tol* )

Loops expected to be disjoint, except possibly share part of boundary.

Test whether one loop lies entirely within another. This function does not work in the general case; it makes the assumption that the loops do NOT intersect each other transversally (their boundaries are allowed to tangentially touch though). The algorithm works by testing a single point on the first loop for being inside the second loop, so if the first loop lay partially inside, partially outside the second, the answer would be arbitrary.

#### Parameters

|                    |                                                                                                                       |
|--------------------|-----------------------------------------------------------------------------------------------------------------------|
| <i>first_loop</i>  | the first loop                                                                                                        |
| <i>second_loop</i> | the second loop                                                                                                       |
| <i>loop_tol</i>    | the tolerance for defining coincidence between start/endpoints on the consecutive curve segments constituting a loop. |
| <i>int_tol</i>     | tolerance used for intersection calculations                                                                          |

#### Returns

'true' if 'first\_loop' was found to be located inside 'second\_loop' (given the assumptions above). 'false' otherwise.

**28.23.2.2** bool `Go::LoopUtils::loopsCCW` ( const std::vector< shared\_ptr< [Go::SplineCurve](#) > > & *simple\_par\_loop*, double *space\_epsilon*, double *int\_tol* )

Check if a closed 2D-loop is oriented counterclockwise or not.

## Parameters

|                        |                                                                                                |
|------------------------|------------------------------------------------------------------------------------------------|
| <i>simple_par_loop</i> | a sequence of 2D curves that are joined start-to-end and that form a closed loop in the plane. |
| <i>int_tol</i>         | (geometric) tolerance used for internal computations (intersection detections)                 |

## Returns

'true' if the loop was found to be oriented CCW, otherwise 'false'.

**28.23.2.3** `bool Go::LoopUtils::loopsCCW ( const CurveLoop & loop, double int_tol )`

Check if a closed 2D-loop is oriented counterclockwise or not. The loop is given as a [CurveLoop](#).

**28.23.2.4** `bool Go::LoopUtils::makeLoopCCW ( std::vector< shared_ptr< ParamCurve > > & loop_cvs, double tol )`

Reorganize curves to create a ccw loop The curves are assumed to have correct sequence, but possible wrong parameter direction Return value: false = reorganization not possible

**28.23.2.5** `bool Go::LoopUtils::paramsCCW ( const std::vector< shared_ptr< Go::CurveOnSurface > > & loop, double space_epsilon, double int_tol )`

Check if a loop defined by [CurveOnSurface](#) s is oriented counterclockwise in the surface's parametric domain.

## Parameters

|                |                                                                                                                          |
|----------------|--------------------------------------------------------------------------------------------------------------------------|
| <i>loop</i>    | a sequence of <a href="#">CurveOnSurface</a> s that are jointed start-to-end and that form a closed loop on the surface. |
| <i>int_tol</i> | (geometric) tolerance used for internal computations (intersection detections)                                           |

## Returns

'true' if the loop was found to be oriented CCW, otherwise 'false'.

**28.23.2.6** `void Go::LoopUtils::representAsSurfaceCurves ( std::vector< shared_ptr< ParamCurve > > & curves, shared_ptr< BoundedSurface > surf, std::vector< shared_ptr< CurveOnSurface > > & cvs_on_sf )`

Represent a vector of [ParamCurve](#) curves as a vector of [CurveOnSurface](#) curves The surface is given as additional input Note that the function throws if the surface information is inconsistent

## 28.24 Go::LRApproxApp Namespace Reference

### Functions

- void [pointCloud2Spline](#) (std::vector< double > &points, int dim, double domain[], double reduced\_domain[], double eps, int max\_iter, shared\_ptr< [LRSplineSurface](#) > &surf, double &maxdist, double &avdist, double &avdist\_out, int &nmb\_out)

- void `computeDistPointSpline` (std::vector< double > &points, shared\_ptr< LRSplineSurface > &surf, double &max\_above, double &max\_below, double &avdist, int &nmb\_points, std::vector< double > &pointsdist)
- void `computeDistPointSpline_omp` (std::vector< double > &points, shared\_ptr< LRSplineSurface > &surf, double &max\_above, double &max\_below, double &avdist, int &nmb\_points, std::vector< double > &pointsdist)
- void `classifyCloudFromDist` (std::vector< double > &points, shared\_ptr< LRSplineSurface > &surf, std::vector< double > &limits, double &max\_above, double &max\_below, double &avdist, int &nmb\_points, std::vector< std::vector< double > > &level\_points, std::vector< int > &nmb\_group)
- void `classifyCloudFromDist_omp` (std::vector< double > &points, shared\_ptr< LRSplineSurface > &surf, std::vector< double > &limits, double &max\_above, double &max\_below, double &avdist, int &nmb\_points, std::vector< std::vector< double > > &level\_points, std::vector< int > &nmb\_group)
- void `categorizeCloudFromDist` (std::vector< double > &points, shared\_ptr< LRSplineSurface > &surf, std::vector< double > &limits, double &max\_above, double &max\_below, double &avdist, int &nmb\_points, std::vector< int > &classification, std::vector< int > &nmb\_group)
- void `categorizeCloudFromDist_omp` (std::vector< double > &points, shared\_ptr< LRSplineSurface > &surf, std::vector< double > &limits, double &max\_above, double &max\_below, double &avdist, int &nmb\_points, std::vector< int > &classification, std::vector< int > &nmb\_group)

### 28.24.1 Function Documentation

- 28.24.1.1 void `Go::LRApproxApp::categorizeCloudFromDist` ( std::vector< double > & *points*, shared\_ptr< LRSplineSurface > & *surf*, std::vector< double > & *limits*, double & *max\_above*, double & *max\_below*, double & *avdist*, int & *nmb\_points*, std::vector< int > & *classification*, std::vector< int > & *nmb\_group* )
- 28.24.1.2 void `Go::LRApproxApp::categorizeCloudFromDist_omp` ( std::vector< double > & *points*, shared\_ptr< LRSplineSurface > & *surf*, std::vector< double > & *limits*, double & *max\_above*, double & *max\_below*, double & *avdist*, int & *nmb\_points*, std::vector< int > & *classification*, std::vector< int > & *nmb\_group* )
- 28.24.1.3 void `Go::LRApproxApp::classifyCloudFromDist` ( std::vector< double > & *points*, shared\_ptr< LRSplineSurface > & *surf*, std::vector< double > & *limits*, double & *max\_above*, double & *max\_below*, double & *avdist*, int & *nmb\_points*, std::vector< std::vector< double > > & *level\_points*, std::vector< int > & *nmb\_group* )
- 28.24.1.4 void `Go::LRApproxApp::classifyCloudFromDist_omp` ( std::vector< double > & *points*, shared\_ptr< LRSplineSurface > & *surf*, std::vector< double > & *limits*, double & *max\_above*, double & *max\_below*, double & *avdist*, int & *nmb\_points*, std::vector< std::vector< double > > & *level\_points*, std::vector< int > & *nmb\_group* )
- 28.24.1.5 void `Go::LRApproxApp::computeDistPointSpline` ( std::vector< double > & *points*, shared\_ptr< LRSplineSurface > & *surf*, double & *max\_above*, double & *max\_below*, double & *avdist*, int & *nmb\_points*, std::vector< double > & *pointsdist* )
- 28.24.1.6 void `Go::LRApproxApp::computeDistPointSpline_omp` ( std::vector< double > & *points*, shared\_ptr< LRSplineSurface > & *surf*, double & *max\_above*, double & *max\_below*, double & *avdist*, int & *nmb\_points*, std::vector< double > & *pointsdist* )
- 28.24.1.7 void `Go::LRApproxApp::pointCloud2Spline` ( std::vector< double > & *points*, int *dim*, double *domain*[], double *reduced\_domain*[], double *eps*, int *max\_iter*, shared\_ptr< LRSplineSurface > & *surf*, double & *maxdist*, double & *avdist*, double & *avdist\_out*, int & *nmb\_out* )

## 28.25 Go::LRBSpline2DUtils Namespace Reference

### Functions

- std::vector< int > `derive_knots` (const Mesh2D &m, Direction2D d, int beg, int end, int orto\_min, int orto\_max)

- void `split_several` (const double \*knotvals, const std::vector< int > &k\_vec\_in, const std::vector< int > &new\_knots, std::vector< int > &k\_vec\_out, std::vector< double > &b\_spline\_weights)
- void `split_function` (const LRBSpline2D &orig, const Mesh2D &mesh, Direction2D d, const double \*const knotvalues, int new\_knot\_ix, LRBSpline2D \*&new\_1, LRBSpline2D \*&new\_2)
- bool `try_split_once` (const LRBSpline2D &b, const Mesh2D &mesh, LRBSpline2D \*&b1, LRBSpline2D \*&b2)

### 28.25.1 Function Documentation

- 28.25.1.1 `std::vector<int> Go::LRBSpline2DUtils::derive_knots ( const Mesh2D & m, Direction2D d, int beg, int end, int orto_min, int orto_max )`
- 28.25.1.2 `void Go::LRBSpline2DUtils::split_function ( const LRBSpline2D & orig, const Mesh2D & mesh, Direction2D d, const double *const knotvalues, int new_knot_ix, LRBSpline2D *& new_1, LRBSpline2D *& new_2 )`
- 28.25.1.3 `void Go::LRBSpline2DUtils::split_several ( const double * knotvals, const std::vector< int > & k_vec_in, const std::vector< int > & new_knots, std::vector< int > & k_vec_out, std::vector< double > & b_spline_weights )`
- 28.25.1.4 `bool Go::LRBSpline2DUtils::try_split_once ( const LRBSpline2D & b, const Mesh2D & mesh, LRBSpline2D *& b1, LRBSpline2D *& b2 )`

## 28.26 Go::LRBSplineMBA Namespace Reference

### Functions

- void `MBADistAndUpdate` (LRBSplineSurface \*srf)
- void `MBADistAndUpdate_omp` (LRBSplineSurface \*srf)
- void `MBAUpdate` (LRBSplineSurface \*srf)
- void `MBAUpdate_omp` (LRBSplineSurface \*srf)
- void `MBAUpdate` (LRBSplineSurface \*srf, std::vector< Element2D \* > &elems, std::vector< Element2D \* > &elems2)
- void `add_contribution` (int dim, std::map< const LRBSpline2D \*, Array< double, 2 > > &target, const LRBSpline2D \*bspline, double nom[], double denom)
- void `add_contribution2` (int dim, std::map< const LRBSpline2D \*, Array< double, 4 > > &target, const LRBSpline2D \*bspline, double nom[], double denom)

### 28.26.1 Function Documentation

- 28.26.1.1 `void Go::LRBSplineMBA::add_contribution ( int dim, std::map< const LRBSpline2D *, Array< double, 2 > > & target, const LRBSpline2D * bspline, double nom[], double denom )`
- 28.26.1.2 `void Go::LRBSplineMBA::add_contribution2 ( int dim, std::map< const LRBSpline2D *, Array< double, 4 > > & target, const LRBSpline2D * bspline, double nom[], double denom )`
- 28.26.1.3 `void Go::LRBSplineMBA::MBADistAndUpdate ( LRBSplineSurface * srf )`
- 28.26.1.4 `void Go::LRBSplineMBA::MBADistAndUpdate_omp ( LRBSplineSurface * srf )`

28.26.1.5 void Go::LRSplineMBA::MBAUpdate ( LRSplineSurface \* srf )

28.26.1.6 void Go::LRSplineMBA::MBAUpdate ( LRSplineSurface \* srf, std::vector< Element2D \* > & elems, std::vector< Element2D \* > & elems2 )

28.26.1.7 void Go::LRSplineMBA::MBAUpdate\_omp ( LRSplineSurface \* srf )

## 28.27 Go::LRSplineUtils Namespace Reference

### Classes

- struct [support\\_compare](#)

### Functions

- [LRSplineSurface::ElementMap identify\\_elements\\_from\\_mesh](#) (const Mesh2D &m)
- void [update\\_elements\\_with\\_single\\_bspline](#) (LRBSpline2D \*b, LRSplineSurface::ElementMap &emap, const Mesh2D &mesh, bool remove)
- int [locate\\_interval](#) (const Mesh2D &m, Direction2D d, double value, double other\_value, bool at\_end)
- void [increment\\_knotvec\\_indices](#) (LRSplineSurface::BSplineMap &bmap, Direction2D d, int from\_ix)
- LRBSpline2D \* [insert\\_basis\\_function](#) (std::unique\_ptr< LRBSpline2D > &b, const Mesh2D &mesh, LRSplineSurface::BSplineMap &bmap)
- std::vector< int > [set\\_uniform\\_meshlines](#) (Direction2D d, Mesh2D &mesh)
- bool [all\\_meshlines\\_uniform](#) (Direction2D d, const Mesh2D &m)
- double [compute\\_greville](#) (const std::vector< int > &v\_ixs, const double \*const vals)
- std::vector< double > [compute\\_greville](#) (int deg, const std::vector< int > &k\_vec\_in, const double \*const knotvals)
- std::vector< int > [knots\\_to\\_insert](#) (const std::vector< int > &ref, const std::vector< int > &mults)
- double [compute\\_alpha](#) (int degree, const int \*const oldvec\_ix, const int \*const newvec\_ix, const double \*const kval)
- std::tuple< std::vector< double >, std::vector< int > > [insert\\_knots](#) (const std::vector< int > &new\_knots, std::unique\_ptr< LRBSpline2D > &bfun, const Direction2D d, const double \*const kval)
- void [tensor\\_split](#) (std::unique\_ptr< LRBSpline2D > &bfun, const std::vector< int > &x\_mults, const std::vector< int > &y\_mults, const Mesh2D &tensor\_mesh, LRSplineSurface::BSplineMap &bmap)
- void [iteratively\\_split](#) (std::vector< std::unique\_ptr< LRBSpline2D > > &bfuns, const Mesh2D &mesh)
- void [iteratively\\_split2](#) (std::vector< LRBSpline2D \* > &bsplines, const Mesh2D &mesh, LRSplineSurface::BSplineMap &bmap, double domain[])
- std::tuple< int, int, int, int > [refine\\_mesh](#) (Direction2D d, double fixed\_val, double start, double end, int mult, bool absolute, int spline\_degree, double knot\_tol, Mesh2D &mesh, LRSplineSurface::BSplineMap &bmap)
- bool [support\\_equal](#) (const LRBSpline2D \*b1, const LRBSpline2D \*b2)
- bool [elementOK](#) (const Element2D \*elem, const Mesh2D &m)
- void [insertParameterFunctions](#) (LRSplineSurface \*lr\_spline\_sf)
- [SplineSurface \\* fullTensorProductSurface](#) (const LRSplineSurface &lr\_spline\_sf)
- LRBSpline2D \* [mostComparableBspline](#) (LRSplineSurface \*lr\_spline\_sf, Point pos)
- std::vector< std::vector< double > > [elementLineClouds](#) (const LRSplineSurface &lr\_spline\_sf)
- void [distributeDataPoints](#) (LRSplineSurface \*srf, std::vector< double > &points, bool add\_distance\_field=false, bool primary\_points=true)



## 28.27.1 Function Documentation

- 28.27.1.1 `bool Go::LRSplineUtils::all_meshlines_uniform ( Direction2D d, const Mesh2D & m )`
- 28.27.1.2 `double Go::LRSplineUtils::compute_alpha ( int degree, const int *const oldvec_ix, const int *const newvec_ix, const double *const kvals )`
- 28.27.1.3 `double Go::LRSplineUtils::compute_greville ( const std::vector< int > & v_ixs, const double *const vals )`
- 28.27.1.4 `std::vector<double> Go::LRSplineUtils::compute_greville ( int deg, const std::vector< int > & k_vec_in, const double *const knotvals )`
- 28.27.1.5 `void Go::LRSplineUtils::distributeDataPoints ( LRSplineSurface * srf, std::vector< double > & points, bool add_distance_field = false, bool primary_points = true )`
- 28.27.1.6 `std::vector<std::vector<double>> Go::LRSplineUtils::elementLineClouds ( const LRSplineSurface & lr_spline_sf )`
- 28.27.1.7 `bool Go::LRSplineUtils::elementOK ( const Element2D * elem, const Mesh2D & m )`
- 28.27.1.8 `SplineSurface* Go::LRSplineUtils::fullTensorProductSurface ( const LRSplineSurface & lr_spline_sf )`
- 28.27.1.9 `LRSplineSurface::ElementMap Go::LRSplineUtils::identify_elements_from_mesh ( const Mesh2D & m )`
- 28.27.1.10 `void Go::LRSplineUtils::increment_knotvec_indices ( LRSplineSurface::BSplineMap & bmap, Direction2D d, int from_ix )`
- 28.27.1.11 `LRBSpline2D* Go::LRSplineUtils::insert_basis_function ( std::unique_ptr< LRBSpline2D > & b, const Mesh2D & mesh, LRSplineSurface::BSplineMap & bmap )`
- 28.27.1.12 `std::tuple<std::vector<double>, std::vector<int>> Go::LRSplineUtils::insert_knots ( const std::vector< int > & new_knots, std::unique_ptr< LRBSpline2D > & bfun, const Direction2D d, const double *const kvals )`
- 28.27.1.13 `void Go::LRSplineUtils::insertParameterFunctions ( LRSplineSurface * lr_spline_sf )`
- 28.27.1.14 `void Go::LRSplineUtils::iteratively_split ( std::vector< std::unique_ptr< LRBSpline2D >> & bfuns, const Mesh2D & mesh )`
- 28.27.1.15 `void Go::LRSplineUtils::iteratively_split2 ( std::vector< LRBSpline2D * > & bsplines, const Mesh2D & mesh, LRSplineSurface::BSplineMap & bmap, double domain[ ] )`
- 28.27.1.16 `std::vector<int> Go::LRSplineUtils::knots_to_insert ( const std::vector< int > & ref, const std::vector< int > & mults )`
- 28.27.1.17 `int Go::LRSplineUtils::locate_interval ( const Mesh2D & m, Direction2D d, double value, double other_value, bool at_end )`

- 28.27.1.18 **LRBSpline2D\*** Go::LRsplineUtils::mostComparableBspline ( *LRsplineSurface \* lr\_spline\_sf, Point pos* )
- 28.27.1.19 **std::tuple<int, int, int, int>** Go::LRsplineUtils::refine\_mesh ( *Direction2D d, double fixed\_val, double start, double end, int mult, bool absolute, int spline\_degree, double knot\_tol, Mesh2D & mesh, LRSplineSurface::BSplineMap & bmap* )
- 28.27.1.20 **std::vector<int>** Go::LRsplineUtils::set\_uniform\_meshlines ( *Direction2D d, Mesh2D & mesh* )
- 28.27.1.21 **bool** Go::LRsplineUtils::support\_equal ( *const LRBSpline2D \* b1, const LRBSpline2D \* b2* )
- 28.27.1.22 **void** Go::LRsplineUtils::tensor\_split ( *std::unique\_ptr< LRBSpline2D > & bfun, const std::vector< int > & x\_mults, const std::vector< int > & y\_mults, const Mesh2D & tensor\_mesh, LRSplineSurface::BSplineMap & bmap* )
- 28.27.1.23 **void** Go::LRsplineUtils::update\_elements\_with\_single\_bspline ( *LRBSpline2D \* b, LRSplineSurface::ElementMap & emap, const Mesh2D & mesh, bool remove* )

## 28.28 Go::LRSurfStitch Namespace Reference

### Functions

- **int** [averageCorner](#) ( *std::vector< std::pair< shared\_ptr< ParamSurface >, int > > &sfs, double tol* )
- **bool** [averageEdge](#) ( *shared\_ptr< ParamSurface > surf1, int edge1, shared\_ptr< ParamSurface > surf2, int edge2, double tol* )
- **bool** [averageEdge](#) ( *shared\_ptr< LRSplineSurface > surf1, int edge1, shared\_ptr< LRSplineSurface > surf2, int edge2, double tol* )
- **void** [fetchEdgeCorners](#) ( *shared\_ptr< LRSplineSurface > surf, int edge, double &u1, double &v1, double &u2, double &v2* )
- **void** [extractMissingKnots](#) ( *std::vector< double > &union\_vec, std::vector< double > &vec, double tol, int order, std::vector< double > &resvec* )
- **void** [defineRefinements](#) ( *const Mesh2D &mesh, Direction2D dir, int edge, int ix, std::vector< double > &knot\_vals, int element\_width, std::vector< LRSplineSurface::Refinement2D > &refs* )
- **void** [extractBoundaryBsplines](#) ( *shared\_ptr< LRSplineSurface > surf, int edge, std::vector< LRBSpline2D \* > &bsplines* )

### 28.28.1 Function Documentation

- 28.28.1.1 **int** Go::LRSurfStitch::averageCorner ( *std::vector< std::pair< shared\_ptr< ParamSurface >, int > > & sfs, double tol* )
- 28.28.1.2 **bool** Go::LRSurfStitch::averageEdge ( *shared\_ptr< ParamSurface > surf1, int edge1, shared\_ptr< ParamSurface > surf2, int edge2, double tol* )
- 28.28.1.3 **bool** Go::LRSurfStitch::averageEdge ( *shared\_ptr< LRSplineSurface > surf1, int edge1, shared\_ptr< LRSplineSurface > surf2, int edge2, double tol* )
- 28.28.1.4 **void** Go::LRSurfStitch::defineRefinements ( *const Mesh2D & mesh, Direction2D dir, int edge, int ix, std::vector< double > & knot\_vals, int element\_width, std::vector< LRSplineSurface::Refinement2D > & refs* )

- 28.28.1.5 void Go::LRSurfStitch::extractBoundaryBsplines ( shared\_ptr< LRSplineSurface > surf, int edge, std::vector< LRBSpline2D \* > & bsplines )
- 28.28.1.6 void Go::LRSurfStitch::extractMissingKnots ( std::vector< double > & union\_vec, std::vector< double > & vec, double tol, int order, std::vector< double > & resvec )
- 28.28.1.7 void Go::LRSurfStitch::fetchEdgeCorners ( shared\_ptr< LRSplineSurface > surf, int edge, double & u1, double & v1, double & u2, double & v2 )

## 28.29 Go::Mesh2DUtils Namespace Reference

### Functions

- int last\_nonlarger\_knotvalue\_ix (const Mesh2D &m, Direction2D d, double par)
- int first\_larger\_knotvalue\_ix (const Mesh2D &m, Direction2D d, double par)
- bool identify\_patch\_lower\_left (const Mesh2D &m, double u, double v, int &x\_ix, int &y\_ix)
- bool identify\_patch\_upper\_right (const Mesh2D &m, double u, double v, int &x\_ix, int &y\_ix)
- int search\_downwards\_for\_nonzero\_multiplicity (const Mesh2D &m, Direction2D d, int start\_ix, int other\_ix)
- int search\_upwards\_for\_nonzero\_multiplicity (const Mesh2D &m, Direction2D d, int start\_ix, int other\_ix)
- int search\_downwards\_for\_nonzero\_multiplicity (const Mesh2D &m, Direction2D d, int start\_ix, int ix1, int ix2)
- int search\_upwards\_for\_nonzero\_multiplicity (const Mesh2D &m, Direction2D d, int start\_ix, int ix1, int ix2)

### 28.29.1 Function Documentation

- 28.29.1.1 int Go::Mesh2DUtils::first\_larger\_knotvalue\_ix ( const Mesh2D & m, Direction2D d, double par )
- 28.29.1.2 bool Go::Mesh2DUtils::identify\_patch\_lower\_left ( const Mesh2D & m, double u, double v, int & x\_ix, int & y\_ix )
- 28.29.1.3 bool Go::Mesh2DUtils::identify\_patch\_upper\_right ( const Mesh2D & m, double u, double v, int & x\_ix, int & y\_ix )
- 28.29.1.4 int Go::Mesh2DUtils::last\_nonlarger\_knotvalue\_ix ( const Mesh2D & m, Direction2D d, double par )
- 28.29.1.5 int Go::Mesh2DUtils::search\_downwards\_for\_nonzero\_multiplicity ( const Mesh2D & m, Direction2D d, int start\_ix, int other\_ix )
- 28.29.1.6 int Go::Mesh2DUtils::search\_downwards\_for\_nonzero\_multiplicity ( const Mesh2D & m, Direction2D d, int start\_ix, int ix1, int ix2 )
- 28.29.1.7 int Go::Mesh2DUtils::search\_upwards\_for\_nonzero\_multiplicity ( const Mesh2D & m, Direction2D d, int start\_ix, int other\_ix )
- 28.29.1.8 int Go::Mesh2DUtils::search\_upwards\_for\_nonzero\_multiplicity ( const Mesh2D & m, Direction2D d, int start\_ix, int ix1, int ix2 )

## 28.30 Go::ModifySurf Namespace Reference

### Functions

- void replaceBoundary (shared\_ptr< SplineSurface > surf, shared\_ptr< SplineCurve > curve, int bd\_idx, double tol)
- bool enforceCoefCoLinearity (shared\_ptr< SplineSurface > sf1, int bd1, shared\_ptr< SplineSurface > sf2, int bd2, double tol, std::vector< int > & enumeration)
- bool enforceVxCoefCoLinearity (std::vector< shared\_ptr< SplineSurface > > &sfs, std::vector< int > &vx←\_enum, std::vector< std::pair< std::vector< int >, std::pair< int, int > > &coef\_cond, double tol)

### 28.30.1 Detailed Description

Functionality used to modify one or more spline surfaces with respect to given conditions

### 28.30.2 Function Documentation

**28.30.2.1** `bool Go::ModifySurf::enforceCoefCoLinearity ( shared_ptr< SplineSurface > sf1, int bd1, shared_ptr< SplineSurface > sf2, int bd2, double tol, std::vector< std::vector< int > > & enumeration )`

Enforce colinearity between coefficients at the common boundary between two spline surfaces. The two outer rows of coefficients are involved in the linearity constraints, but a few additional rows are modified for reasons of smoothness. All surface boundaries, except the affected one, are kept fixed.

**28.30.2.2** `bool Go::ModifySurf::enforceVxCoefCoLinearity ( std::vector< shared_ptr< SplineSurface > > & sfs, std::vector< int > & vx_enum, std::vector< std::pair< std::vector< int >, std::pair< int, int > > > & coef_cond, double tol )`

Enforce colinearity at vertices where 4 edge pairs meet. The two outer rows of coefficients around the vertex are involved in the linearity constraints, but a few additional rows are modified for reasons of smoothness.

**28.30.2.3** `void Go::ModifySurf::replaceBoundary ( shared_ptr< SplineSurface > surf, shared_ptr< SplineCurve > curve, int bd_idx, double tol )`

Replace one boundary curve of a [SplineSurface](#) Approximate the initial surface bd\_idx = 0: umin 1: umax 2: vmin

## 28.31 Go::OffsetSurfaceUtils Namespace Reference

### Functions

- [OffsetSurfaceStatus](#) `offsetSurfaceSet (const std::vector< shared_ptr< ParamSurface > > &param_sfs, double offset_dist, double epsgeo, shared_ptr< SplineSurface > &offset_sf)`

### 28.31.1 Function Documentation

**28.31.1.1** `OffsetSurfaceStatus Go::OffsetSurfaceUtils::offsetSurfaceSet ( const std::vector< shared_ptr< ParamSurface > > & param_sfs, double offset_dist, double epsgeo, shared_ptr< SplineSurface > & offset_sf )`

Compute the offset surface for a set of parametric surfaces (possibly 1 surface only). The surface set must have 4 corners.

#### Parameters

|                          |                                             |
|--------------------------|---------------------------------------------|
| <code>param_sfs</code>   | input parametric surfaces                   |
| <code>offset_dist</code> | distance for the offset surface.            |
| <code>epsgeo</code>      | geometric tolerance for the offset surface. |

**Returns**

status of the offset function. 0 => success, everything else a failure.

## 28.32 Go::OffsetUtils Namespace Reference

Related to the generation of cross tangent curves.

**Functions**

- void `blend_s1421` (`const SplineSurface *ps`, `double aoffset`, `int ider`, `const Point &epar`, `int &ilfs`, `int &ilft`, `std::vector< Point > &effpnt`, `std::vector< Point > &epnt`, `int *jstat`)

### 28.32.1 Detailed Description

Related to the generation of cross tangent curves.

### 28.32.2 Function Documentation

28.32.2.1 void Go::OffsetUtils::blend\_s1421 ( `const SplineSurface * ps`, `double aoffset`, `int ider`, `const Point & epar`, `int & ilfs`, `int & ilft`, `std::vector< Point > & effpnt`, `std::vector< Point > & epnt`, `int * jstat` )

## 28.33 Go::orientCurves Namespace Reference

Sorts and orients a set of curves.

**Functions**

- `template<class PtrToCurveType >`  
void `orientCurves` (`const std::vector< PtrToCurveType > &curves`, `std::vector< int > &permutation`, `std::vector< bool > &reversed`, `double neighbour_tol`, `bool assume_manifold=true`)

### 28.33.1 Detailed Description

Sorts and orients a set of curves.

### 28.33.2 Function Documentation

28.33.2.1 `template<class PtrToCurveType > void Go::orientCurves::orientCurves ( const std::vector< PtrToCurveType > & curves`, `std::vector< int > & permutation`, `std::vector< bool > & reversed`, `double neighbour_tol`, `bool assume_manifold = true` ) `[inline]`

This function sorts and orients a set of curves so that curves whose endpoints coincide will be ordered consecutively, and eventually 'reversed' so that startpoints meet endpoints. It is assumed that the set of curves given constitute one or several manifolds, ie., that there will not be cases where three or more endpoints meet at a common point (this would make the above mentioned ordering impossible).

## Parameters

|                        |                                                                                                                                                                                                                                                                                                                                                                |
|------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>curves</i>          | a vector containing the set of (pointers to) curves to be analysed. It is expected that they constitute one or more 1-manifolds.                                                                                                                                                                                                                               |
| <i>permutation</i>     | upon return, this vector will contain a permutation of the indices to the input curves such that after this permutation, curves whose endpoints are connected will become 'consecutive'.                                                                                                                                                                       |
| <i>reversed</i>        | upon return, this vector, whose length will be equal to that of 'curves' and 'permutation', will contain bool values. If curve at position 'i' in the 'curves' vector needs to be reversed in order to connect startpoint-to-endpoint with its neighbours, then the corresponding value in 'reversed' will be 'true', and 'false' if no reversal is necessary. |
| <i>neighbour_tol</i>   | the tolerance used when checking for coincident points.                                                                                                                                                                                                                                                                                                        |
| <i>assume_manifold</i> | if the user specifies 'true', then the manifold property will be assumed, but will not be explicitly checked. If 'false' is specified, then the input <i>will</i> be checked for consistency with respect to this, and an exception will be cast if the manifold condition is violated.                                                                        |

Definition at line 74 of file orientCurves.h.

## 28.34 Go::Path Namespace Reference

Functions related to sequence of edges.

### Functions

- `bool estimateHoleInfo (const std::vector< ftEdge * > &edges, Point &centre, Point &axis, double &radius, double &angle)`  
*sequence*
- `void classifyCorners (const std::vector< ftEdge * > &edges, double tol, std::vector< shared_ptr< Vertex > > &corner, std::vector< shared_ptr< Vertex > > &non_corner)`
- `std::vector< ftEdge * > identifyLoop (std::vector< ftEdge * > edges, shared_ptr< Vertex > vx)`
- `void closestPoint (std::vector< ftEdge * > edges, const Point &pt, int &clo_ind, double &clo_par, Point &clo_pt, double &clo_dist)`
- `void getEdgeCurves (std::vector< ftEdge * > &loop, std::vector< shared_ptr< ParamCurve > > &space_cvs, std::vector< Point > &joint_points, double eps, double tol, bool corner_in_Tjoint=true)`
- `std::vector< ftEdge * > edgeChain (ftEdge *edg, double angtol, shared_ptr< Vertex > &v1, shared_ptr< Vertex > &v2)`

### 28.34.1 Detailed Description

Functions related to sequence of edges.

### 28.34.2 Function Documentation

28.34.2.1 `void Go::Path::classifyCorners ( const std::vector< ftEdge * > &edges, double tol, std::vector< shared_ptr< Vertex > > &corner, std::vector< shared_ptr< Vertex > > &non_corner )`

Classify vertices in a path of edges as corner or non-corner depending on a given tolerance

28.34.2.2 void Go::Path::closestPoint ( std::vector< ftEdge \* > edges, const Point & pt, int & clo\_ind, double & clo\_par, Point & clo\_pt, double & clo\_dist )

28.34.2.3 std::vector<ftEdge\*> Go::Path::edgeChain ( ftEdge \* edg, double angtol, shared\_ptr< Vertex > & v1, shared\_ptr< Vertex > & v2 )

Extract edge chain with no joints between more than two edges or corners

28.34.2.4 bool Go::Path::estimateHoleInfo ( const std::vector< ftEdge \* > & edges, Point & centre, Point & axis, double & radius, double & angle )

sequence

Estimate mid point, normal and radius defined by an edge

28.34.2.5 void Go::Path::getEdgeCurves ( std::vector< ftEdge \* > & loop, std::vector< shared\_ptr< ParamCurve > > & space\_cvs, std::vector< Point > & joint\_points, double eps, double tol, bool corner\_in\_Tjoint = true )

Combine edges into 4 curves, preferably with splits in real corners

28.34.2.6 std::vector<ftEdge\*> Go::Path::identifyLoop ( std::vector< ftEdge \* > edges, shared\_ptr< Vertex > vx )

Identify a loops starting and ending in a given vertex in an ordered sequence of edges

## 28.35 Go::PointSetApp Namespace Reference

[Point](#) set and triangulation applications.

### Functions

- void [parameterizeTriang](#) (const double \*xyz\_points, int nmbp, const int \*triangles, int nmbt, std::vector< double > &uv\_pars)  
*Parameterize triangulated point set.*
- bool [recognizeBoundary](#) (shared\_ptr< ftPointSet > &triang, std::vector< int > &corner\_ix)  
*Service functionality for parameterizeTriang.*
- bool [recognizeCornerNodes](#) (const shared\_ptr< ftPointSet > &triang, const std::vector< int > &bd\_nodes, std::vector< int > &corner\_ix)

### 28.35.1 Detailed Description

[Point](#) set and triangulation applications.

### 28.35.2 Function Documentation

28.35.2.1 `void Go::PointSetApp::parameterizeTriang ( const double * xyz_points, int nmbp, const int * triangles, int nmbt, std::vector< double > & uv_pars )`

Parameterize triangulated point set.

28.35.2.2 `bool Go::PointSetApp::recognizeBoundary ( shared_ptr< ftPointSet > & triang, std::vector< int > & corner_ix )`

Service functionality for parameterizeTriang.

28.35.2.3 `bool Go::PointSetApp::recognizeCornerNodes ( const shared_ptr< ftPointSet > & triang, const std::vector< int > & bd_nodes, std::vector< int > & corner_ix )`

## 28.36 Go::qualityUtils Namespace Reference

### Functions

- [bool isSliverFace](#) (shared\_ptr< ParamSurface >, double thickness, double factor=2.0)
- [bool isSliverFace](#) (const SplineSurface &sf, double thickness, double factor=2.0)
- [bool isSliverFace](#) (const BoundedSurface &sf, double thickness, double factor=2.0)
- [bool isSliverFace2](#) (const BoundedSurface &sf, double thickness, double factor=2.0)
- [bool hasIndistinctKnots](#) (shared\_ptr< ParamSurface > surf, double tol, std::vector< shared\_ptr< ParamCurve > > &trim\_cv\_knots)
- [double estimateArea](#) (shared\_ptr< ParamSurface > surf)
- [double estimateLoopArea](#) (shared\_ptr< CurveLoop > loop)

### 28.36.1 Function Documentation

28.36.1.1 `double Go::qualityUtils::estimateArea ( shared_ptr< ParamSurface > surf )`

28.36.1.2 `double Go::qualityUtils::estimateLoopArea ( shared_ptr< CurveLoop > loop )`

28.36.1.3 `bool Go::qualityUtils::hasIndistinctKnots ( shared_ptr< ParamSurface > surf, double tol, std::vector< shared_ptr< ParamCurve > > & trim_cv_knots )`

28.36.1.4 `bool Go::qualityUtils::isSliverFace ( shared_ptr< ParamSurface >, double thickness, double factor = 2.0 )`

28.36.1.5 `bool Go::qualityUtils::isSliverFace ( const SplineSurface & sf, double thickness, double factor = 2.0 )`

28.36.1.6 `bool Go::qualityUtils::isSliverFace ( const BoundedSurface & sf, double thickness, double factor = 2.0 )`

28.36.1.7 `bool Go::qualityUtils::isSliverFace2 ( const BoundedSurface & sf, double thickness, double factor = 2.0 )`

## 28.37 Go::RegularizeUtils Namespace Reference

Utility functionality for [RegularizeFace](#) and [RegularizeFaceSet](#).



## Functions

- `std::vector< shared_ptr< ftSurface > > divideVertex` (`shared_ptr< ftSurface > face`, `shared_ptr< Vertex > vx`, `std::vector< shared_ptr< Vertex > > &cand_vx`, `ftEdge *cand_edge`, `std::vector< shared_ptr< Vertex > > &prio_vx`, `double epsge`, `double tol2`, `double angtol`, `double bend`, `std::vector< shared_ptr< Vertex > > &non_corner`, `const Point &centre`, `const Point &axis`, `bool strong=false`)
- `std::vector< shared_ptr< CurveOnSurface > > findVertexSplit` (`shared_ptr< ftSurface > face`, `shared_ptr< Vertex > vx`, `std::vector< shared_ptr< Vertex > > &cand_vx`, `ftEdge *cand_edge`, `std::vector< shared_ptr< Vertex > > &prio_vx`, `double epsge`, `double tol2`, `double angtol`, `double bend`, `std::vector< shared_ptr< Vertex > > &non_corner`, `const Point &centre`, `const Point &axis`, `shared_ptr< BoundedSurface > &bd_sf`, `bool strong=false`)
- `std::vector< shared_ptr< ftSurface > > createFaces` (`std::vector< shared_ptr< BoundedSurface > > &sub_sfs`, `shared_ptr< ftSurface > face`, `double epsge`, `double tol2`, `double angtol`, `std::vector< shared_ptr< Vertex > > non_corner`)
- `void getDivisionPlane` (`shared_ptr< ftSurface > face`, `shared_ptr< Vertex > vx`, `double epsge`, `Point &pnt`, `Point &normal`)
- `void getClosestBoundaryPar` (`shared_ptr< ftSurface > face`, `shared_ptr< Vertex > vx`, `std::vector< shared_ptr< ParamCurve > > &vx_cvs`, `const Point &pnt`, `double epsge`, `int &close_idx`, `double &close_dist`, `Point &close_par`, `int loop_idx=-1`)
- `bool cornerInShortestPath` (`shared_ptr< Vertex > vx1`, `shared_ptr< Vertex > vx2`, `shared_ptr< ftSurface > face`, `double angtol`)
- `bool getPath` (`ftEdge *edg`, `shared_ptr< Vertex > vx`, `shared_ptr< Vertex > last`, `shared_ptr< ftSurface > face`, `std::vector< ftEdge * > &path`)
- `int noExtension` (`shared_ptr< Vertex > vx`, `ftSurface *face`, `shared_ptr< Vertex > &vx2`, `std::pair< Point, Point > &co_par1`, `std::pair< Point, Point > &co_par2`, `int &dir1`, `int &dir2`, `double &val1`, `double &val2`, `double angtol`, `bool check_constant_curve`)
- `bool mergeSituationContinuation` (`ftSurface *init_face`, `shared_ptr< Vertex > vx`, `ftEdge *edge`, `double angtol`)
- `double getMaxParFrac` (`shared_ptr< ftSurface > face`)
- `int selectCandVx` (`shared_ptr< ftSurface > face`, `shared_ptr< Vertex > vx`, `const Point &in_vec`, `vector< shared_ptr< Vertex > > cand_vx`, `RectDomain &dom`, `double epsge`, `double angtol`, `const Point &centre`, `const Point &normal`, `std::vector< shared_ptr< ParamCurve > > &vx_cvs`, `double close_dist`, `const Point &close_pt`, `double &cyl_rad`, `bool strong=false`)
- `void adjustTrimSeg` (`std::vector< shared_ptr< CurveOnSurface > > &trim_segments`, `Point *parval1`, `Point *parval2`, `shared_ptr< ftSurface > face`, `shared_ptr< BoundedSurface > &bd_sf`, `std::vector< shared_ptr< Vertex > > &non_corner`, `double tol`, `double epsge`)
- `void checkTrimSeg` (`std::vector< shared_ptr< CurveOnSurface > > &trim_segments`, `std::vector< shared_ptr< Vertex > > &next_vxs`, `const Point &vx_point`, `const Point &other_pt`, `double epsge`)
- `void checkTrimSeg2` (`std::vector< shared_ptr< CurveOnSurface > > &trim_segments`, `const Point &vx_←par1`, `const Point &vx_←par2`, `double epsge`)
- `void checkTrimSeg3` (`std::vector< shared_ptr< CurveOnSurface > > &trim_segments`, `const Point &vx_←par1`, `const Point &vx_←par2`, `double epsge`)
- `void checkTrimConfig` (`shared_ptr< ftSurface > face`, `std::vector< shared_ptr< CurveOnSurface > > &trim_segments`, `shared_ptr< Vertex > vx`, `std::vector< shared_ptr< Vertex > > &corners`, `double epsge`)
- `ftEdge * getOppositeBoundaryPar` (`shared_ptr< ftSurface > face`, `shared_ptr< Vertex > vx`, `std::vector< shared_ptr< Vertex > > &corners`, `double epsge`, `Point &point`, `double &par`, `double &dist`)
- `Point getInVec` (`shared_ptr< Vertex > vx`, `shared_ptr< ftSurface > face`)
- `shared_ptr< ParamCurve > checkStrightParCv` (`shared_ptr< ftSurface > face`, `const Point &pos1`, `const Point &pos2`, `double epsge`)
- `shared_ptr< ParamCurve > checkStrightParCv` (`shared_ptr< ftSurface > face`, `shared_ptr< Vertex > vx1`, `shared_ptr< Vertex > vx2`, `double epsge`)
- `shared_ptr< ParamCurve > checkStrightParCv` (`shared_ptr< ftSurface > face`, `shared_ptr< Vertex > vx1`, `const Point &mid`, `double epsge`)
- `bool checkPath` (`shared_ptr< Vertex > vx1`, `shared_ptr< Vertex > vx2`, `shared_ptr< Vertex > vx`, `shared_ptr< ftSurface > face`, `double angtol`)
- `bool checkRegularity` (`std::vector< shared_ptr< Vertex > > &cand_vx`, `shared_ptr< ftSurface > face`, `bool checkConvex=true`)

- `std::vector< shared_ptr< Vertex > > endVxInChain` (`shared_ptr< ftSurface > face`, `ftSurface *face1`, `ftSurface *face2`, `shared_ptr< Vertex > vx`, `shared_ptr< Vertex > prev`, `shared_ptr< Vertex > vx0`, `std::vector< shared_ptr< Vertex > > &met_already`)
- `int traverseUntilTJoint` (`std::vector< ftSurface * > vx_faces`, `shared_ptr< Vertex > vx`, `shared_ptr< Vertex > &vx2`, `std::vector< ftSurface * > &vx_faces2`)
- `void angleInEndpoints` (`shared_ptr< CurveOnSurface > seg`, `shared_ptr< Vertex > vx1`, `shared_ptr< Vertex > vx2`, `shared_ptr< ftSurface > face`, `double &min_ang1`, `double &min_ang2`)
- `void getSourceCvs` (`std::vector< shared_ptr< ftEdge > > &all_edg`, `std::vector< shared_ptr< ParamCurve > > &all_cvs`)

### 28.37.1 Detailed Description

Utility functionality for [RegularizeFace](#) and [RegularizeFaceSet](#).

### 28.37.2 Function Documentation

- 28.37.2.1 `void Go::RegularizeUtils::adjustTrimSeg` (`std::vector< shared_ptr< CurveOnSurface > > &trim_segments`, `Point * parval1`, `Point * parval2`, `shared_ptr< ftSurface > face`, `shared_ptr< BoundedSurface > &bd_sf`, `std::vector< shared_ptr< Vertex > > &non_corner`, `double tol`, `double epsge` )
- 28.37.2.2 `void Go::RegularizeUtils::angleInEndpoints` (`shared_ptr< CurveOnSurface > seg`, `shared_ptr< Vertex > vx1`, `shared_ptr< Vertex > vx2`, `shared_ptr< ftSurface > face`, `double &min_ang1`, `double &min_ang2` )
- 28.37.2.3 `bool Go::RegularizeUtils::checkPath` (`shared_ptr< Vertex > vx1`, `shared_ptr< Vertex > vx2`, `shared_ptr< Vertex > vx`, `shared_ptr< ftSurface > face`, `double angtol` )
- 28.37.2.4 `bool Go::RegularizeUtils::checkRegularity` (`std::vector< shared_ptr< Vertex > > &cand_vx`, `shared_ptr< ftSurface > face`, `bool checkConvex = true` )
- 28.37.2.5 `shared_ptr< ParamCurve > Go::RegularizeUtils::checkStrightParCv` (`shared_ptr< ftSurface > face`, `const Point &pos1`, `const Point &pos2`, `double epsge` )
- 28.37.2.6 `shared_ptr< ParamCurve > Go::RegularizeUtils::checkStrightParCv` (`shared_ptr< ftSurface > face`, `shared_ptr< Vertex > vx1`, `shared_ptr< Vertex > vx2`, `double epsge` )
- 28.37.2.7 `shared_ptr< ParamCurve > Go::RegularizeUtils::checkStrightParCv` (`shared_ptr< ftSurface > face`, `shared_ptr< Vertex > vx1`, `const Point &mid`, `double epsge` )
- 28.37.2.8 `void Go::RegularizeUtils::checkTrimConfig` (`shared_ptr< ftSurface > face`, `std::vector< shared_ptr< CurveOnSurface > > &trim_segments`, `shared_ptr< Vertex > vx`, `std::vector< shared_ptr< Vertex > > &corners`, `double epsge` )
- 28.37.2.9 `void Go::RegularizeUtils::checkTrimSeg` (`std::vector< shared_ptr< CurveOnSurface > > &trim_segments`, `std::vector< shared_ptr< Vertex > > &next_vxs`, `const Point &vx_point`, `const Point &other_pt`, `double epsge` )
- 28.37.2.10 `void Go::RegularizeUtils::checkTrimSeg2` (`std::vector< shared_ptr< CurveOnSurface > > &trim_segments`, `const Point &vx_par1`, `const Point &vx_par2`, `double epsge` )

- 28.37.2.11 `void Go::RegularizeUtils::checkTrimSeg3 ( std::vector< shared_ptr< CurveOnSurface > > & trim_segments, const Point & vx_par1, const Point & vx_par2, double epsge )`
- 28.37.2.12 `bool Go::RegularizeUtils::cornerInShortestPath ( shared_ptr< Vertex > vx1, shared_ptr< Vertex > vx2, shared_ptr< ftSurface > face, double angtol )`
- 28.37.2.13 `std::vector<shared_ptr<ftSurface> > Go::RegularizeUtils::createFaces ( std::vector< shared_ptr< BoundedSurface > > & sub_sfs, shared_ptr< ftSurface > face, double epsge, double tol2, double angtol, std::vector< shared_ptr< Vertex > > non_corner )`
- 28.37.2.14 `std::vector<shared_ptr<ftSurface> > Go::RegularizeUtils::divideVertex ( shared_ptr< ftSurface > face, shared_ptr< Vertex > vx, std::vector< shared_ptr< Vertex > > & cand_vx, ftEdge * cand_edge, std::vector< shared_ptr< Vertex > > & prio_vx, double epsge, double tol2, double angtol, double bend, std::vector< shared_ptr< Vertex > > & non_corner, const Point & centre, const Point & axis, bool strong = false )`
- 28.37.2.15 `std::vector<shared_ptr<Vertex> > Go::RegularizeUtils::endVxInChain ( shared_ptr< ftSurface > face, ftSurface * face1, ftSurface * face2, shared_ptr< Vertex > vx, shared_ptr< Vertex > prev, shared_ptr< Vertex > vx0, std::vector< shared_ptr< Vertex > > & met_already )`
- 28.37.2.16 `std::vector<shared_ptr<CurveOnSurface> > Go::RegularizeUtils::findVertexSplit ( shared_ptr< ftSurface > face, shared_ptr< Vertex > vx, std::vector< shared_ptr< Vertex > > & cand_vx, ftEdge * cand_edge, std::vector< shared_ptr< Vertex > > & prio_vx, double epsge, double tol2, double angtol, double bend, std::vector< shared_ptr< Vertex > > & non_corner, const Point & centre, const Point & axis, shared_ptr< BoundedSurface > & bd_sf, bool strong = false )`
- 28.37.2.17 `void Go::RegularizeUtils::getClosestBoundaryPar ( shared_ptr< ftSurface > face, shared_ptr< Vertex > vx, std::vector< shared_ptr< ParamCurve > > & vx_cvs, const Point & pnt, double epsge, int & close_idx, double & close_dist, Point & close_par, int loop_idx = -1 )`
- 28.37.2.18 `void Go::RegularizeUtils::getDivisionPlane ( shared_ptr< ftSurface > face, shared_ptr< Vertex > vx, double epsge, Point & pnt, Point & normal )`
- 28.37.2.19 `Point Go::RegularizeUtils::getInVec ( shared_ptr< Vertex > vx, shared_ptr< ftSurface > face )`
- 28.37.2.20 `double Go::RegularizeUtils::getMaxParFrac ( shared_ptr< ftSurface > face )`
- 28.37.2.21 `ftEdge* Go::RegularizeUtils::getOppositeBoundaryPar ( shared_ptr< ftSurface > face, shared_ptr< Vertex > vx, std::vector< shared_ptr< Vertex > > & corners, double epsge, Point & point, double & par, double & dist )`
- 28.37.2.22 `bool Go::RegularizeUtils::getPath ( ftEdge * edg, shared_ptr< Vertex > vx, shared_ptr< Vertex > last, shared_ptr< ftSurface > face, std::vector< ftEdge * > & path )`
- 28.37.2.23 `void Go::RegularizeUtils::getSourceCvs ( std::vector< shared_ptr< ftEdge > > & all_edg, std::vector< shared_ptr< ParamCurve > > & all_cvs )`
- 28.37.2.24 `bool Go::RegularizeUtils::mergeSituationContinuation ( ftSurface * init_face, shared_ptr< Vertex > vx, ftEdge * edg, double angtol )`

- 28.37.2.25 `int Go::RegularizeUtils::noExtension ( shared_ptr< Vertex > vx, ftSurface * face, shared_ptr< Vertex > & vx2, std::pair< Point, Point > & co_par1, std::pair< Point, Point > & co_par2, int & dir1, int & dir2, double & val1, double & val2, double angtol, bool check_constant_curve )`
- 28.37.2.26 `int Go::RegularizeUtils::selectCandVx ( shared_ptr< ftSurface > face, shared_ptr< Vertex > vx, const Point & in_vec, vector< shared_ptr< Vertex > > cand_vx, RectDomain & dom, double epsge, double angtol, const Point & centre, const Point & normal, std::vector< shared_ptr< ParamCurve > > & vx_cvs, double close_dist, const Point & close_pt, double & cyl_rad, bool strong = false )`
- 28.37.2.27 `int Go::RegularizeUtils::traverseUntilTJoint ( std::vector< ftSurface * > vx_faces, shared_ptr< Vertex > vx, shared_ptr< Vertex > & vx2, std::vector< ftSurface * > & vx_faces2 )`

## 28.38 Go::Singular Namespace Reference

### Functions

- void `vanishingNormal` (shared\_ptr< ParamSurface > srf, double tol, std::vector< Point > &singular\_pts, std::vector< std::vector< Point > > &singular\_sequences)
- void `vanishingTangent` (shared\_ptr< ParamCurve > crv, double start, double end, double tol, std::vector< double > &singular\_pts, std::vector< std::vector< double > > &singular\_sequences)

### 28.38.1 Function Documentation

- 28.38.1.1 `void Go::Singular::vanishingNormal ( shared_ptr< ParamSurface > srf, double tol, std::vector< Point > & singular_pts, std::vector< std::vector< Point > > & singular_sequences )`
- 28.38.1.2 `void Go::Singular::vanishingTangent ( shared_ptr< ParamCurve > crv, double start, double end, double tol, std::vector< double > & singular_pts, std::vector< std::vector< double > > & singular_sequences )`

## 28.39 Go::SplineDebugUtils Namespace Reference

### Functions

- void `GO_API writeSpaceParamCurve` (const SplineCurve &pcurve, std::ostream &os, double z=0.0)
- void `GO_API writeSpaceParamCurve` (const Line &pline, std::ostream &os, double z=0.0)
- void `GO_API writeTrimmedInfo` (BoundedSurface &bd\_sf, std::ostream &os, double z=0.0)
- void `GO_API writeOuterBoundaryLoop` (ParamSurface &sf, std::ostream &os)
- void `GO_API objToFile` (GeomObject \*geom\_obj, char \*to\_file)
- void `GO_API objsToFile` (std::vector< shared\_ptr< GeomObject > > &geom\_objs, char \*to\_file)
- void `GO_API writeSISLFormat` (const SplineCurve &spline\_cv, std::ostream &os)
- void `GO_API writeCvsOnSf` (const std::vector< shared\_ptr< Go::ParamCurve > > &loop\_cvs, double epsgeo, std::ofstream &fileout)
- void `GO_API writeCvsOnSf` (const std::vector< shared\_ptr< Go::CurveOnSurface > > &loop\_cvs, double epsgeo, std::ofstream &fileout)

### 28.39.1 Detailed Description

For debugging. Writes a parameter curve in the xy-plane. Remove when GoViewer handles 2D curves.

### 28.39.2 Function Documentation

- 28.39.2.1 `void GO_API Go::SplineDebugUtils::objsToFile ( std::vector< shared_ptr< GeomObject > > & geom_objs, char * to_file )`

writes the geometric objects (with header) to the specified file name.

## Parameters

|                  |                                                     |
|------------------|-----------------------------------------------------|
| <i>geom_objs</i> | the objects to write to file.                       |
| <i>to_file</i>   | the file name to which the objects will be written. |

28.39.2.2 void GO\_API Go::SplineDebugUtils::objToFile ( *GeomObject* \* *geom\_obj*, char \* *to\_file* )

writes the geometric object (with header) to the specified file name.

## Parameters

|                 |                                                    |
|-----------------|----------------------------------------------------|
| <i>geom_obj</i> | the object to write to file.                       |
| <i>to_file</i>  | the file name to which the object will be written. |

28.39.2.3 void GO\_API Go::SplineDebugUtils::writeCvsOnSf ( const std::vector< shared\_ptr< Go::ParamCurve > > & *loop\_cvs*, double *epsgeo*, std::ofstream & *fileout* )

28.39.2.4 void GO\_API Go::SplineDebugUtils::writeCvsOnSf ( const std::vector< shared\_ptr< Go::CurveOnSurface > > & *loop\_cvs*, double *epsgeo*, std::ofstream & *fileout* )

28.39.2.5 void GO\_API Go::SplineDebugUtils::writeOuterBoundaryLoop ( *ParamSurface* & *sf*, std::ostream & *os* )

28.39.2.6 void GO\_API Go::SplineDebugUtils::writeSISLFormat ( const *SplineCurve* & *spline\_cv*, std::ostream & *os* )

Write a [SplineCurve](#) to a stream using the SISL file format (not the [Go](#) format).

## Parameters

|                  |                                                                 |
|------------------|-----------------------------------------------------------------|
| <i>spline_cv</i> | the curve to write to a stream                                  |
| <i>os</i>        | the stream to which the curve will be written (in SISL format)x |

28.39.2.7 void GO\_API Go::SplineDebugUtils::writeSpaceParamCurve ( const *SplineCurve* & *pcurve*, std::ostream & *os*, double *z = 0.0* )

For debugging. Writes a parameter curve (2D) in the xy-plane, for a given z-value. It will be written (with header) to the specified stream as a 3D curve.

## Parameters

|               |                                                    |
|---------------|----------------------------------------------------|
| <i>pcurve</i> | the parameter curve we want to write as a 3D curve |
| <i>os</i>     | the stream to which we want to write the 3D curve  |
| <i>z</i>      | the constant z-value for the generated curve       |

28.39.2.8 void **GO\_API** Go::SplineDebugUtils::writeSpaceParamCurve ( const Line & *pline*, std::ostream & *os*, double *z* = 0.0 )

28.39.2.9 void **GO\_API** Go::SplineDebugUtils::writeTrimmedInfo ( BoundedSurface & *bd\_sf*, std::ostream & *os*, double *z* = 0.0 )

Write the parameter curve (if existing) and the space curve (if existing) to the output stream. Both curves are written as 3D curves extending 2D curves with the given z-value.

## 28.40 Go::SplineUtils Namespace Reference

### Functions

- void **GO\_API** transpose\_array (int *dim*, int *m*, int *n*, double \*array\_start)
- int **GO\_API** closest\_in\_array (const double \*pt, const double \*array, int *n*, int *dim*)
- Vector3D **GO\_API** closest\_on\_triangle (const Vector3D &pt, const Vector3D tri[3], double &clo\_dist2)
- double **GO\_API** closest\_on\_line\_segment (const Vector3D &pt, const Vector3D &beg, const Vector3D &end)
- void **GO\_API** closest\_on\_rectgrid (const double \*pt, const double \*array, int *m*, int *n*, double &clo\_u, double &clo\_v)
- void **GO\_API** closest\_on\_rectgrid (const double \*pt, const double \*array, int *u\_min*, int *u\_max*, int *v\_min*, int *v\_max*, int *nmb\_coefs\_u*, double &clo\_u, double &clo\_v)
- void **GO\_API** make\_coef\_array\_from\_rational\_coefs (const double \*rationals, double \*coefs, int *num\_coefs*, int *dim*)
- void **GO\_API** curve\_ratder (double const eder[], int *idim*, int *ider*, double gder[])
- void **GO\_API** surface\_ratder (double const eder[], int *idim*, int *ider*, double gder[])
- void **GO\_API** osloalg (int *ij*, int *imy*, int *ik*, int *in*, int \*jpl, int \*jfi, int \*jla, const double \*et, const double \*etau, double \*galfa)
- void **GO\_API** refmatrix (const double \*et, int *im*, int *ik*, const double \*etau, int *in*, double \*ea, int \*nfirst, int \*nlast)
- void **GO\_API** splineToBezierTransfMat (const double \*knots, std::vector< double > &transf\_mat)
- void **GO\_API** extractBezierCoefs (const double \*coefs, const int *num\_coefs\_u*, const int *num\_coefs\_v*, const int *ind\_u\_min*, const int *ind\_v\_min*, const std::vector< double > &transf\_mat\_u, const std::vector< double > &transf\_mat\_v, std::vector< double > &bezier\_coefs)
- void **GO\_API** refinedBezierCoefsCubic (Go::SplineSurface &spline\_sf, int *ind\_u\_min*, int *ind\_v\_min*, std::vector< double > &bez\_coefs)  
*Method expecting bi-cubic input.*
- shared\_ptr< SplineSurface > **GO\_API** refineToBezier (const Go::SplineSurface &spline\_sf)
- shared\_ptr< SplineSurface > **GO\_API** insertKnots (const Go::SplineSurface &spline\_sf, const std::vector< double > new\_knots\_u, const std::vector< double > new\_knots\_v)

### 28.40.1 Detailed Description

Utility functionality related to spline curves and surfaces or functions operating on spline curves and surfaces

### 28.40.2 Function Documentation

28.40.2.1 int **GO\_API** Go::SplineUtils::closest\_in\_array ( const double \* *pt*, const double \* *array*, int *n*, int *dim* )

Find the point in an array closest to the given base point (measured in the usual, Euclidean distance)

## Parameters

|              |                                              |
|--------------|----------------------------------------------|
| <i>pt</i>    | pointer to the base point (of dimension 'n') |
| <i>array</i> | pointer to the array in which we will search |
| <i>n</i>     | number of elements in the array              |
| <i>dim</i>   | dimension of the elements/base point         |

28.40.2.2 **double GO\_API Go::SplineUtils::closest\_on\_line\_segment ( const Vector3D & *pt*, const Vector3D & *beg*, const Vector3D & *end* )**

Find the closest point on a line segment to a given point.

## Parameters

|            |                                                                           |
|------------|---------------------------------------------------------------------------|
| <i>pt</i>  | the point for which we want to find the closest point on the line segment |
| <i>beg</i> | the start point of the line segment                                       |
| <i>end</i> | the end point of the line segment                                         |

## Returns

the barycentric coordinate of the closest point on the line segment (number between 0 and 1, where 0 is the start point and 1 is the end point)

28.40.2.3 **void GO\_API Go::SplineUtils::closest\_on\_rectgrid ( const double \* *pt*, const double \* *array*, int *m*, int *n*, double & *clo\_u*, double & *clo\_v* )**

Find an approximate closest point on a rectangular grid of 3-dimensional points. The method used is to 'triangulate' the grid, and locate the closest point on this triangulation. We imagine that the point grid is uniformly parametrized from 0 to *m*-1 along each row and from 0 to *n*-1 along each column, and we refer to these parameters as 'u' and 'v'.

## Parameters

|                           |                                                                                                        |
|---------------------------|--------------------------------------------------------------------------------------------------------|
| <i>pt</i>                 | the point we want to search the closest point for                                                      |
| <i>array</i>              | pointer to the array of 3D-points; coordinates should be stored consecutively in a row-wise fashion.   |
| <i>m</i>                  | number of columns in the point array                                                                   |
| <i>n</i>                  | number of rows in the point array                                                                      |
| <i>clo</i> ↔<br><i>_u</i> | upon function return; the u-parameter of the closest point, using the parametrization explained above. |
| <i>clo</i> ↔<br><i>_v</i> | upon function return; the v-parameter of the closest point, using the parametrization explained above. |

28.40.2.4 **void GO\_API Go::SplineUtils::closest\_on\_rectgrid ( const double \* *pt*, const double \* *array*, int *u\_min*, int *u\_max*, int *v\_min*, int *v\_max*, int *nmb\_coefs\_u*, double & *clo\_u*, double & *clo\_v* )**

Find an approximate closest point on a sub-grid of a rectangular grid of 3-dimensional points. The method used is to 'triangulate' the grid, and locate the closest point on this triangulation. We imagine that the parametrization of the sub-grid is from 'u\_min' to 'u\_max' along each row and from 'v\_min' to 'v\_max' along each column.

## Parameters

|                               |                                                                                                                                               |
|-------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------|
| <i>pt</i>                     | the point we want to search the closest point for                                                                                             |
| <i>array</i>                  | pointer to the array of 3D-points; coordinates should be stored consecutively in a row-wise fashion.                                          |
| <i>u_min</i>                  | lowest column index of the subgrid we want to search                                                                                          |
| <i>u_max</i>                  | highest column index of the subgrid we want to search                                                                                         |
| <i>v_min</i>                  | lowest row index of the subgrid we want to search                                                                                             |
| <i>v_max</i>                  | highest row index of the subgrid we want to search                                                                                            |
| <i>nmb_coefs</i><br><i>_u</i> | total number of columns in the complete grid (necessary to know in order to determine how far to jump to get to the next row in the subgrid). |
| <i>clo_u</i>                  | upon function return; the u-parameter of the closest point, using the parametrization explained above.                                        |
| <i>clo_v</i>                  | upon function return; the v-parameter of the closest point, using the parametrization explained above.                                        |

28.40.2.5 **Vector3D GO\_API Go::SplineUtils::closest\_on\_triangle ( const Vector3D & *pt*, const Vector3D *tri*[3], double & *clo\_dist2* )**

Find the closest point on a triangle, in barycentric coordinates.

## Parameters

|                  |                                                                                |
|------------------|--------------------------------------------------------------------------------|
| <i>pt</i>        | the point for which we want to find the closest point on the triangle          |
| <i>tri</i>       | the triangle, specified by its three corners                                   |
| <i>clo_dist2</i> | the squared Euclidean distance from 'pt' to the closest point in the triangle. |

## Returns

the barycentric coordinates of the closest point on the triangle

28.40.2.6 **void GO\_API Go::SplineUtils::curve\_ratder ( double const *eder*[], int *idim*, int *ider*, double *gder*[] )**

This function takes as input the position and a certain number of derivatives in homogenous space of a point on a rational curve. It outputs the position and derivatives of this point in non-homogenous ("ordinary") space. (Corresponds to *s6ratder* in SISL)

## Parameters

|             |                                                                                                                            |
|-------------|----------------------------------------------------------------------------------------------------------------------------|
| <i>eder</i> | pointer to the array where the point's position and derivatives in homogenous coordinates (input) are consecutively stored |
| <i>idim</i> | the dimension of the non-homogenous space                                                                                  |
| <i>ider</i> | the number of derivatives sought (0 means that we only want to convert the <i>position</i> to non-homogenous coordinates). |
| <i>gder</i> | pointer to the array where the result will be written (in same order as ' <i>eder</i> ').                                  |



28.40.2.7 void GO\_API Go::SplineUtils::extractBezierCoefs ( const double \* *coefs*, const int *num\_coefs\_u*, const int *num\_coefs\_v*, const int *ind\_u\_min*, const int *ind\_v\_min*, const std::vector< double > & *transf\_mat\_u*, const std::vector< double > & *transf\_mat\_v*, std::vector< double > & *bezier\_coefs* )

Assuming surface is bi-cubic (i.e. order 4). Extract the bezier patch corr to the domain (knots\_u[*ind\_u\_min*], knots\_u[*ind\_u\_min*+1])x(knots\_v[*ind\_v\_min*], knots\_v[*ind\_v\_min*+1]).

#### Parameters

|                  |                                              |
|------------------|----------------------------------------------|
| <i>ind_u_min</i> | Basis pointer corresponding to <i>umin</i> . |
| <i>ind_v_min</i> | Basis pointer corresponding to <i>vmin</i> . |

28.40.2.8 shared\_ptr<SplineSurface> GO\_API Go::SplineUtils::insertKnots ( const Go::SplineSurface & *spline\_sf*, const std::vector< double > *new\_knots\_u*, const std::vector< double > *new\_knots\_v* )

28.40.2.9 void GO\_API Go::SplineUtils::make\_coef\_array\_from\_rational\_coefs ( const double \* *rationals*, double \* *coefs*, int *num\_coefs*, int *dim* )

convert an array of rational coefficients to an array of nonrational coefficients

#### Parameters

|                  |                                                                           |
|------------------|---------------------------------------------------------------------------|
| <i>rationals</i> | pointer to the array of rational coefficients                             |
| <i>coefs</i>     | pointer to the array where the nonrational coefficients are to be written |
| <i>num_coefs</i> | total number of coefficients                                              |
| <i>dim</i>       | dimension of coefficients (not counting the rational component).          |

28.40.2.10 void GO\_API Go::SplineUtils::osloalg ( int *ij*, int *imy*, int *ik*, int *in*, int \* *jpl*, int \* *jfi*, int \* *jla*, const double \* *et*, const double \* *etau*, double \* *galfa* )

Corresponds to s1701 in SISL This function computes in a compact format a line in the discrete B-spline matrix converting between an original basis "etau" and a new basis "et".

#### Parameters

|              |                                                                                                                                                                                                                                                                       |
|--------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>ij</i>    | The index of the new vertice                                                                                                                                                                                                                                          |
| <i>imy</i>   | An index on etau, where the input value are to be etau( <i>imy</i> ) <= et( <i>ij</i> ) < etau( <i>imy</i> + 1).                                                                                                                                                      |
| <i>ik</i>    | The order of the B-spline.                                                                                                                                                                                                                                            |
| <i>in</i>    | The number of the original vertices.                                                                                                                                                                                                                                  |
| <i>et</i>    | The new knot vector.                                                                                                                                                                                                                                                  |
| <i>etau</i>  | The old knot vector.                                                                                                                                                                                                                                                  |
| <i>et</i>    | An array ep( <i>ik</i> ) to local use. Such that we do not need to allocate the array locally after each call.                                                                                                                                                        |
| <i>jpl</i>   | The negativ difference between the index in galfa and the real knot inserten matrix.                                                                                                                                                                                  |
| <i>jfi</i>   | The index of the first element in the line j in the the real knot inserten matrix whice is not zero. The element with the index ( <i>jfi</i> + <i>jpl</i> ) in galfa is the same as the element with index <i>jfi</i> in the real line j in the knot inserten matrix. |
| <i>jla</i>   | The index of the last element in the line j in the real knot inserten matrix whice is not zero. The element with the index ( <i>jla</i> + <i>jpl</i> ) in galfa is the same as the element with index <i>jla</i> in the real line j in the knot inserten matrix.      |
| <i>galfa</i> | A compressed line in the knot inserten matrix.                                                                                                                                                                                                                        |

28.40.2.11 void **GO\_API** Go::SplineUtils::refinedBezierCoefsCubic ( Go::SplineSurface & *spline\_sf*, int *ind\_u\_min*, int *ind\_v\_min*, std::vector< double > & *bez\_coefs* )

Method expecting bi-cubic input.

28.40.2.12 shared\_ptr<SplineSurface> **GO\_API** Go::SplineUtils::refineToBezier ( const Go::SplineSurface & *spline\_sf* )

28.40.2.13 void **GO\_API** Go::SplineUtils::refmatrix ( const double \* *et*, int *im*, int *ik*, const double \* *etau*, int *in*, double \* *ea*, int \* *nfirst*, int \* *nlast* )

Corresponds to sh1922 in SISL. Computes the B-spline refinement transformation matrix from the spline space generated by the knot vector etau to the refined spline space generated by the refined knot vector et.

#### Parameters

|               |                                                                                                                                                                                                                                                                                                                                                                 |
|---------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>et</i>     | Real array of length (im+ik) containing the refined knot vector.                                                                                                                                                                                                                                                                                                |
| <i>im</i>     | The dimension of the spline space corresponding to et.                                                                                                                                                                                                                                                                                                          |
| <i>ik</i>     | The order of the spline space.                                                                                                                                                                                                                                                                                                                                  |
| <i>etau</i>   | Real array of length (in+ik) containing the original knot vector.                                                                                                                                                                                                                                                                                               |
| <i>in</i>     | The dimension of the spline space corresponding to etau.                                                                                                                                                                                                                                                                                                        |
| <i>ea</i>     | Real array of dimension (im*ik) containing the B-spline refinement matrix from the knot vector etau to the knot vector et. This matrix has dimension im*in but since at most ik entries are nonzero in each row, it can be stored in a im*ik array together with two integer arrays indicating the position of the first and last nonzero elements in each row. |
| <i>nfirst</i> | Integer array of dimension (im) containing pointers to the first nonzero element of each row of the B-spline refinement matrix from etau to et.                                                                                                                                                                                                                 |
| <i>nlast</i>  | Integer array of dimension (im) containing pointers to the last nonzero element of each row of the B-spline refinement matrix from etau to et.                                                                                                                                                                                                                  |

28.40.2.14 void **GO\_API** Go::SplineUtils::splineToBezierTransfMat ( const double \* *knots*, std::vector< double > & *transf\_mat* )

Assuming basis is cubic (i.e. order 4). Create the transformation matrix which extract the bezier coefs for the interval (knots[3], knots[4]).

28.40.2.15 void **GO\_API** Go::SplineUtils::surface\_ratder ( double const *eder*[], int *idim*, int *ider*, double *gder*[]) )

This function takes as input the position and a certain number of derivatives in homogenous space of a point on a rational surface. It outputs the position and derivatives of this point in non-homogenous ("ordinary") space. (Corresponds to s6strider in SISL)

## Parameters

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|-------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>eder</i> | pointer to double array of dimension $[(ider+1)*(ider+2)*(idim+1)/2]$ containing the position and the derivative vectors of the homogeneous surface at the point with parameter value (epar[0],epar[1]). (idim+1 is the number of components of each B-spline coefficient, i.e. the dimension of the homogeneous space in which the surface lies.) These vectors are stored in the following order: First the idim+1 components of the position vector, then the idim+1 components of the D(1,0) vector, then the idim+1 components of the D(0,1) vector, then the idim+1 components of the D(2,0) vector, followed by D(1,1), D(0,2) and so on up to the idim+1 components of the D(0,ider). |
| <i>idim</i> | The dimension of the non homogenous space                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| <i>ider</i> | The number of input derivatives with respect to both parameter directions.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| <i>gder</i> | pointer to the array where the result will be written (in same order as 'eder').                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |

28.40.2.16 void **GO\_API** Go::SplineUtils::transpose\_array ( int *dim*, int *m*, int *n*, double \* *array\_start* )

Utility functionality related to spline curves and surfaces or functions operating on spline curves and surfaces Transpose an (m x n) matrix of dim-dimensional points stored as an array in row-major order (i.e. all elements of a single row are stored together).

## Parameters

|                    |                                                      |
|--------------------|------------------------------------------------------|
| <i>dim</i>         | the dimension of each of the points in the array     |
| <i>m</i>           | number of lines in the matrix                        |
| <i>n</i>           | number of columns in the matrix                      |
| <i>array_start</i> | pointer to the start of array where matrix is stored |

## 28.41 Go::SplitModelUtils Namespace Reference

Utility functionality for splitting of surface models.

### Functions

- void [splitInFreeCorners](#) (shared\_ptr< [SurfaceModel](#) > *sfmodel*, const [Point](#) &*pnt*, const [Point](#) &*axis*)
- void [splitInNonCorners](#) (shared\_ptr< [SurfaceModel](#) > *sfmodel*, const [Point](#) &*pnt*, const [Point](#) &*axis*)
- void [splitInOuterVertices](#) (shared\_ptr< [SurfaceModel](#) > *sfmodel*, shared\_ptr< [ftSurface](#) > *face*, const [Point](#) &*pnt*, const [Point](#) &*axis*)

#### 28.41.1 Detailed Description

Utility functionality for splitting of surface models.

#### 28.41.2 Function Documentation

28.41.2.1 void Go::SplitModelUtils::splitInFreeCorners ( shared\_ptr< [SurfaceModel](#) > *sfmodel*, const [Point](#) & *pnt*, const [Point](#) & *axis* )

The model is split from corners vertices belonging only to one face towards the axis defined by pnt and axis

28.41.2.2 void Go::SplitModelUtils::splitInNonCorners ( shared\_ptr< SurfaceModel > *sfmodel*, const Point & *pnt*, const Point & *axis* )

The model is split from vertices that do not constitute a corner for the current face towards the axis defined by *pnt* and *axis*

28.41.2.3 void Go::SplitModelUtils::splitInOuterVertices ( shared\_ptr< SurfaceModel > *sfmodel*, shared\_ptr< ftSurface > *face*, const Point & *pnt*, const Point & *axis* )

The model is split from vertices in outer loops towards the axis defined by *pnt* and *axis* or towards vertices in inner loops

## 28.42 Go::SurfaceCreators Namespace Reference

### Functions

- shared\_ptr< SplineSurface > createSmoothTransition (const std::vector< shared\_ptr< const ParamSurface > > &surfs, const std::vector< shared\_ptr< const CurveOnSurface > > &int\_cvs, double dist\_0, double dist\_1, double epsge, std::vector< shared\_ptr< SplineCurve > > &trim\_crvs)
- shared\_ptr< SplineSurface > mult1DSurfaces (const SplineSurface &sf1, const SplineSurface &sf2)
- shared\_ptr< SplineSurface > mult1DBezierPatches (const SplineSurface &patch1, const SplineSurface &patch2)
- shared\_ptr< Go::SplineSurface > mergeRationalParts (const Go::SplineSurface &nom\_sf, const Go::SplineSurface &den\_sf, bool weights\_in\_first=false)
- shared\_ptr< Go::SplineSurface > insertParamDomain (const Go::SplineSurface &sf\_1d)
- std::vector< shared\_ptr< Go::SplineSurface > > separateRationalParts (const Go::SplineSurface &sf)

### 28.42.1 Detailed Description

Various functions for generating [SplineSurface](#) s by approximation, blending, etc.

### 28.42.2 Function Documentation

28.42.2.1 shared\_ptr<SplineSurface> Go::SurfaceCreators::createSmoothTransition ( const std::vector< shared\_ptr< const ParamSurface > > & *surfs*, const std::vector< shared\_ptr< const CurveOnSurface > > & *int\_cvs*, double *dist\_0*, double *dist\_1*, double *epsge*, std::vector< shared\_ptr< SplineCurve > > & *trim\_crvs* )

Given input of two parametric surfaces, which intersect along an edge, create a smooth surface by moving boundary of *surfs* to inner part of current *surfs*. Normals of input *surfs* are assumed to be consistent.

#### Parameters

|                  |                                                                                                                                          |
|------------------|------------------------------------------------------------------------------------------------------------------------------------------|
| <i>surfs</i>     | the input surfaces, size of vector is 2.                                                                                                 |
| <i>int_cvs</i>   | intersection curves between the two surfaces. Vector has size 2. Both parameter curves should exist, the space curve should be the same. |
| <i>dist_0</i>    | the offset space distance in <i>surfs</i> [0].                                                                                           |
| <i>dist_1</i>    | the offset space distance in <i>surfs</i> [1].                                                                                           |
| <i>epsge</i>     | the geometrical tolerance when offsetting.                                                                                               |
| <i>trim_crvs</i> | the offset trim curves: <i>par_cv0</i> , <i>space_cv0</i> , <i>par_cv1</i> , <i>space_cv1</i> .                                          |

28.42.2.2 `shared_ptr<Go::SplineSurface> Go::SurfaceCreators::insertParamDomain ( const Go::SplineSurface & sf_1d )`

Given input of 1d-sf, return the 3d visualization (u, v, f(u, v)).

#### Parameters

|                    |                        |
|--------------------|------------------------|
| <code>sf_1d</code> | 1-dimensional surface. |
|--------------------|------------------------|

#### Returns

the 3-dimensional surface.

28.42.2.3 `shared_ptr<Go::SplineSurface> Go::SurfaceCreators::mergeRationalParts ( const Go::SplineSurface & nom_sf, const Go::SplineSurface & den_sf, bool weights_in_first = false )`

Return the rational surface 'nom\_sf/den\_sf'.

#### Parameters

|                               |                                                                                                                   |
|-------------------------------|-------------------------------------------------------------------------------------------------------------------|
| <code>nom_sf</code>           | the nominator surface.                                                                                            |
| <code>den_sf</code>           | the denominator surface.                                                                                          |
| <code>weights_in_first</code> | true if the coefficients of the nom_sf have been multiplied by the corresponding rational coefficients in den_sf. |

#### Returns

the rational surface.

28.42.2.4 `shared_ptr<SplineSurface> Go::SurfaceCreators::mult1DBezierPatches ( const SplineSurface & patch1, const SplineSurface & patch2 )`

Return the product of the two Bezier patches. Expecting them to be non-rational.

#### Parameters

|                     |                                                                     |
|---------------------|---------------------------------------------------------------------|
| <code>patch1</code> | <a href="#">SplineSurface</a> of Bezier type (i.e. no inner knots). |
| <code>patch2</code> | <a href="#">SplineSurface</a> of Bezier type (i.e. no inner knots). |

#### Returns

the surface product.

28.42.2.5 `shared_ptr<SplineSurface> Go::SurfaceCreators::mult1DSurfaces ( const SplineSurface & sf1, const SplineSurface & sf2 )`

Return the product of the two spline surfaces. Expecting them to be non-rational.

## Parameters

|            |                     |
|------------|---------------------|
| <i>sf1</i> | the first surface.  |
| <i>sf2</i> | the second surface. |

## Returns

the surface product.

28.42.2.6 `std::vector<shared_ptr<Go::SplineSurface> > Go::SurfaceCreators::separateRationalParts ( const Go::SplineSurface & sf )`

Assuming the *sf* is rational, separate the geometric space from the homogenous.

## Parameters

|           |                           |
|-----------|---------------------------|
| <i>sf</i> | the input spline surface. |
|-----------|---------------------------|

## Returns

the non-rational parts of *sf*.

## 28.43 Go::SurfaceInterpolator Namespace Reference

Functionality for creating interpolating surfaces.

### Functions

- `SplineSurface * regularInterpolation (const BsplineBasis &basis_u, const BsplineBasis &basis_v, std::vector< double > &par_u, std::vector< double > &par_v, std::vector< double > &points, int dimension, bool rational, std::vector< double > &weights)`

#### 28.43.1 Detailed Description

Functionality for creating interpolating surfaces.

#### 28.43.2 Function Documentation

28.43.2.1 `SplineSurface* Go::SurfaceInterpolator::regularInterpolation ( const BsplineBasis & basis_u, const BsplineBasis & basis_v, std::vector< double > & par_u, std::vector< double > & par_v, std::vector< double > & points, int dimension, bool rational, std::vector< double > & weights )`

Interpolate a set of regular, parameterized interpolation points. The parameterization is assumed to correspond to the given B-spline bases (select interpolation points in the Greville points). The function throws if the input parameters are inconsistent

## 28.44 Go::SurfaceModelUtils Namespace Reference

Utility functionality for splitting of surface models.

### Functions

- `std::vector< shared_ptr< ParamSurface > >` `checkClosedFaces` (`shared_ptr< ParamSurface >` `surface`, `double tol`)

#### 28.44.1 Detailed Description

Utility functionality for splitting of surface models.

#### 28.44.2 Function Documentation

28.44.2.1 `std::vector<shared_ptr<ParamSurface>>` `Go::SurfaceModelUtils::checkClosedFaces` ( `shared_ptr< ParamSurface >` `surface`, `double tol` )

## 28.45 Go::SurfaceOnVolumeTools Namespace Reference

### Functions

- `CurveLoop` `getOuterBoundaryLoop` (`shared_ptr< SurfaceOnVolume >` `sf`, `double eps`)

#### 28.45.1 Detailed Description

This namespace contains functions used to fetch boundary curves from surface on volumes

#### 28.45.2 Function Documentation

28.45.2.1 `CurveLoop` `Go::SurfaceOnVolumeTools::getOuterBoundaryLoop` ( `shared_ptr< SurfaceOnVolume >` `sf`, `double eps` )

Fetch the outer loop of the surface. This function is implemented outside of `SurfaceOnVolume` to allow the resulting curves to be of type `CurveOnSurface` and possess all the information related to this curve type.

## 28.46 Go::SurfaceTools Namespace Reference

Free functions operating on parametric surfaces.

## Functions

- [CurveLoop](#) [outerBoundarySfLoop](#) (shared\_ptr< [ParamSurface](#) > surf, double degenerate\_epsilon)
 

*Fetch the outer boundary loop of a parametric surface.*
- std::vector< [CurveLoop](#) > [allBoundarySfLoops](#) (shared\_ptr< [ParamSurface](#) > surf, double degenerate\_epsilon)
 

*Fetch all boundary loops of a parametric surface.*
- std::vector< [CurveLoop](#) > [absolutelyAllBoundarySfLoops](#) (shared\_ptr< [ParamSurface](#) > surf, double degenerate\_epsilon)
- void [iterateCornerPos](#) ([Point](#) &vertex, std::vector< std::pair< shared\_ptr< [ParamSurface](#) >, [Point](#) > > sfs, double tol)
- bool [cornerToCornerSfs](#) (shared\_ptr< [ParamSurface](#) > sf1, shared\_ptr< [CurveOnSurface](#) > sf\_cv1, shared\_ptr< [ParamSurface](#) > sf2, shared\_ptr< [CurveOnSurface](#) > sf\_cv2, double tol)
- bool [getSfAdjacencyInfo](#) (shared\_ptr< [ParamSurface](#) > sf1, shared\_ptr< [CurveOnSurface](#) > sf\_cv1, shared\_ptr< [ParamSurface](#) > sf2, shared\_ptr< [CurveOnSurface](#) > sf\_cv2, double tol, int &bd1, int &bd2, bool &same\_orient)
- bool [getCorrCoefEnum](#) (shared\_ptr< [SplineSurface](#) > sf1, shared\_ptr< [SplineSurface](#) > sf2, int bd1, int bd2, bool same\_orient, std::vector< std::pair< int, int > > &enumeration)
- bool [getCoefEnumeration](#) (shared\_ptr< [SplineSurface](#) > sf, int bd, std::vector< int > &enumeration)
- bool [getCoefEnumeration](#) (shared\_ptr< [SplineSurface](#) > sf, int bd, std::vector< int > &enumeration\_bd, std::vector< int > &enumeration\_bd2)
- bool [getCornerCoefEnum](#) (shared\_ptr< [SplineSurface](#) > sf, int bd1, int bd2, int &enumeration)
- int [checkCoefCoLinearity](#) (shared\_ptr< [SplineSurface](#) > sf1, shared\_ptr< [SplineSurface](#) > sf2, int bd1, int bd2, bool same\_orient, double tol, double ang\_tol, std::vector< std::vector< int > > &enumeration)
- void [surface\\_seedfind](#) (const [Point](#) &pt, const [ParamSurface](#) &sf, const [RectDomain](#) \*rd, double &u, double &v)
- void [parameterizeByBaseSurf](#) (const [ParamSurface](#) &sf, const std::vector< double > &points, std::vector< double > &parvals)
- double [estimateTangentLength](#) ([SplineSurface](#) \*surf, int pardir, bool at\_start)
- void [GO\\_API checkSurfaceClosed](#) (const [Go::ParamSurface](#) &sf, bool &closed\_dir\_u, bool &closed\_dir\_v, double closed\_tol=1e-06)
- void [GO\\_API surfaceClosed](#) (const [Go::SplineSurface](#) &sf, bool &closed\_dir\_u, bool &closed\_dir\_v, double closed\_tol=1e-06)
- [Point](#) [getParEpsilon](#) (const [ParamSurface](#) &sf, double epsgeo)

### 28.46.1 Detailed Description

Free functions operating on parametric surfaces.

### 28.46.2 Function Documentation

28.46.2.1 `std::vector<CurveLoop> Go::SurfaceTools::absolutelyAllBoundarySfLoops ( shared_ptr< ParamSurface > surf, double degenerate_epsilon )`

Fetch all boundary loops of a parametric surface including degenerate curves

28.46.2.2 `std::vector<CurveLoop> Go::SurfaceTools::allBoundarySfLoops ( shared_ptr< ParamSurface > surf, double degenerate_epsilon )`

Fetch all boundary loops of a parametric surface.



28.46.2.3 `int Go::SurfaceTools::checkCoefCoLinearity ( shared_ptr< SplineSurface > sf1, shared_ptr< SplineSurface > sf2, int bd1, int bd2, bool same_orient, double tol, double ang_tol, std::vector< std::vector< int > > & enumeration )`

Find coefficient enumeration of possible colinear coefficients at a common edge and check whether the coefficients are indeed colinear (return value: 0 = no correspondance, 1 = almost colinear, 2 = conditions defined)

28.46.2.4 `void GO_API Go::SurfaceTools::checkSurfaceClosed ( const Go::ParamSurface & sf, bool & closed_dir_u, bool & closed_dir_v, double closed_tol = 1e-06 )`

Check if the surface is closed in one or both paramater directions

28.46.2.5 `bool Go::SurfaceTools::cornerToCornerSfs ( shared_ptr< ParamSurface > sf1, shared_ptr< CurveOnSurface > sf_cv1, shared_ptr< ParamSurface > sf2, shared_ptr< CurveOnSurface > sf_cv2, double tol )`

Check if two neighbouring surfaces in a surface set have a corner-to-corner configuration, i.e. no T-joints

#### Parameters

|               |                                                                                        |
|---------------|----------------------------------------------------------------------------------------|
| <i>sf1</i>    | one surface                                                                            |
| <i>sf_cv1</i> | the boundary curve/trimming curve corresponding to sf1 at the boundary common with sf2 |
| <i>sf2</i>    | the other surface                                                                      |
| <i>sf_cv2</i> | the boundary curve/trimming curve corresponding to sf2 at the boundary common with sf1 |
| <i>return</i> | parameter true of the configuration is corner-to-corner                                |

28.46.2.6 `double Go::SurfaceTools::estimateTangentLength ( SplineSurface * surf, int paddir, bool at_start )`

Estimate the average length of the cross tangent of a spline surface at a given boundary

#### Parameters

|                 |                                                                                                                         |
|-----------------|-------------------------------------------------------------------------------------------------------------------------|
| <i>surf</i>     | the surface                                                                                                             |
| <i>paddir</i>   | = 1 : the cross tangent curve has constant u-parameter<br>paddir = 0 : the cross tangent curve has constant v-parameter |
| <i>at_start</i> | the cross tangent curve is evaluated at the start of the parameter domain in the given parameter direction              |

28.46.2.7 `bool Go::SurfaceTools::getCoefEnumeration ( shared_ptr< SplineSurface > sf, int bd, std::vector< int > & enumeration )`

Fetch the local enumeration of the surface coefficients along a specified boundary (bd=0 - umin, bd=1 - umax, bd=2 - vmin, bd=3 - vmax)

28.46.2.8 **bool** Go::SurfaceTools::getCoefEnumeration ( *shared\_ptr*< **SplineSurface** > *sf*, *int* *bd*, *std::vector*< *int* > & *enumeration\_bd*, *std::vector*< *int* > & *enumeration\_bd2* )

Return both the boundary coefficients as well as coefficient row number 2 when counting from the edge.

28.46.2.9 **bool** Go::SurfaceTools::getCornerCoefEnum ( *shared\_ptr*< **SplineSurface** > *sf*, *int* *bd1*, *int* *bd2*, *int* & *enumeration* )

Fetch the coefficient enumeration of the corner of a spline surface specified by the two boundaries meeting in this corner

28.46.2.10 **bool** Go::SurfaceTools::getCorrCoefEnum ( *shared\_ptr*< **SplineSurface** > *sf1*, *shared\_ptr*< **SplineSurface** > *sf2*, *int* *bd1*, *int* *bd2*, **bool** *same\_orient*, *std::vector*< *std::pair*< *int*, *int* > > & *enumeration* )

Given two spline surfaces and information about a common boundary between these surfaces, compute the enumeration of corresponding coefficients along this boundary provided that the surfaces have got the same spline space along the boundary

#### Parameters

|                    |                                                                                                                              |
|--------------------|------------------------------------------------------------------------------------------------------------------------------|
| <i>sf1</i>         | first surface                                                                                                                |
| <i>sf2</i>         | second surface                                                                                                               |
| <i>bd1</i>         | boundary curve of first surface which follows the common boundary, -1=no such boundary curve, 0=umin, 1=umax, 2=vmin, 3=vmax |
| <i>bd2</i>         | boundary curve of second surface which follows the common boundary                                                           |
| <i>same_orient</i> | whether or not the surface boundaries have got the same orientation along the common boundary                                |
| <i>return</i>      | value whether or not the surfaces have got the same spline space                                                             |
| <i>enumeration</i> | pairwise enumeration of the corresponding coefficients in the two surfaces                                                   |

28.46.2.11 **Point** Go::SurfaceTools::getParEpsilon ( **const** **ParamSurface** & *sf*, **double** *epsgeo* )

28.46.2.12 **bool** Go::SurfaceTools::getSfAdjacencyInfo ( *shared\_ptr*< **ParamSurface** > *sf1*, *shared\_ptr*< **CurveOnSurface** > *sf\_cv1*, *shared\_ptr*< **ParamSurface** > *sf2*, *shared\_ptr*< **CurveOnSurface** > *sf\_cv2*, **double** *tol*, *int* & *bd1*, *int* & *bd2*, **bool** & *same\_orient* )

Given two neighbouring surfaces, *sf1* and *sf2*, and information about the common boundary, fetch information about the adjacency configuration

#### Parameters

|               |                                                                                                                   |
|---------------|-------------------------------------------------------------------------------------------------------------------|
| <i>sf1</i>    | one surface                                                                                                       |
| <i>sf_cv1</i> | the boundary curve/trimming curve corresponding to <i>sf1</i> at the boundary common with <i>sf2</i>              |
| <i>sf2</i>    | the other surface                                                                                                 |
| <i>sf_cv2</i> | the boundary curve/trimming curve corresponding to <i>sf2</i> at the boundary common with <i>sf1</i>              |
| <i>tol</i>    | adjacency tolerance                                                                                               |
| <i>return</i> | parameter true if the two surfaces share the same boundary and that boundary is boundary trimmed in both surfaces |

## Parameters

|                    |                                                                                                                              |
|--------------------|------------------------------------------------------------------------------------------------------------------------------|
| <i>bd1</i>         | boundary curve of first surface which follows the common boundary, -1=no such boundary curve, 0=umin, 1=umax, 2=vmin, 3=vmax |
| <i>bd2</i>         | boundary curve of second surface which follows the common boundary                                                           |
| <i>same_orient</i> | whether or not the surface boundaries have got the same orientation along the common boundary                                |

**28.46.2.13** void Go::SurfaceTools::iterateCornerPos ( Point & *vertex*, std::vector< std::pair< shared\_ptr< ParamSurface >, Point > > *sfs*, double *tol* )

Compute the corner where a number of surfaces meet. This function is relevant if the surfaces almost interpolate the same corner. If the surface is elementary (cone, sphere, ...), then this surface has more influence on the corner position than spline surfaces.

## Parameters

|               |                                                                                                                                                                                                                                                                                                   |
|---------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>vertex</i> | the position of the corner. This point is moved during the computation to get closer to a point shared by all surfaces                                                                                                                                                                            |
| <i>sfs</i>    | the surfaces meeting in this corner and the surface parameter associated with the corner. Note that the parameter does not necessarily lie at a corner of the underlying non-trimmed surface and it might also represent a vertex in a surface model and may lie internally at a surface boundary |

**28.46.2.14** CurveLoop Go::SurfaceTools::outerBoundarySfLoop ( shared\_ptr< ParamSurface > *surf*, double *degenerate\_epsilon* )

Fetch the outer boundary loop of a parametric surface.

**28.46.2.15** void Go::SurfaceTools::parameterizeByBaseSurf ( const ParamSurface & *sf*, const std::vector< double > & *points*, std::vector< double > & *parvals* )

Parameterize a point set by projecting the points onto a given base surface

**28.46.2.16** void Go::SurfaceTools::surface\_seedfind ( const Point & *pt*, const ParamSurface & *sf*, const RectDomain \* *rd*, double & *u*, double & *v* )

Estimate a start point to a closest point iteration between a given point, *pt*, and a given surface, *sf*, possibly restricted to the parameter domain *rd*.

**28.46.2.17** void GO\_API Go::SurfaceTools::surfaceClosed ( const Go::SplineSurface & *sf*, bool & *closed\_dir\_u*, bool & *closed\_dir\_v*, double *closed\_tol* = 1e-06 )

Check if the surface is closed in one or both parameter directions

## 28.47 Go::TesselatorUtils Namespace Reference

Related to the relative resolution of tessellation.

### Functions

- void [getResolution](#) (const [ParamSurface](#) \*surf, int &u\_nmb, int &v\_nmb, int uv\_nmb=400)
- [shared\\_ptr](#)<[LineCloud](#)> [getCtrPol](#) ([GeomObject](#) \*obj)  
*Fetch the control polygon of some geometric entity.*

### 28.47.1 Detailed Description

Related to the relative resolution of tessellation.

### 28.47.2 Function Documentation

28.47.2.1 [shared\\_ptr](#)<[LineCloud](#)> [Go::TesselatorUtils::getCtrPol](#) ( [GeomObject](#) \* obj )

Fetch the control polygon of some geometric entity.

28.47.2.2 void [Go::TesselatorUtils::getResolution](#) ( const [ParamSurface](#) \* surf, int & u\_nmb, int & v\_nmb, int uv\_nmb = 400 )

Get the mesh size of a surface in the two parameter directions given the total number of nodes in the mesh. NB! u\_nmb\*v\_nmb are probably not exactly equal to uv\_nmb

## 28.48 Go::tpUtils Namespace Reference

### Functions

- [template](#)<class [edgeType](#)>  
void [adjacentEdges](#) ([edgeType](#) \*edge, [bool](#) at\_start\_of\_edge, [std::vector](#)< [edgeType](#) \* > &adjacent, [std::vector](#)< [bool](#) > &at\_start)
- [template](#)<class [edgeType](#)>  
[tpJointType](#) [checkContinuity](#) (const [edgeType](#) \*edge1, const [edgeType](#) \*edge2, [double](#) neighbour, [double](#) gap, [double](#) bend, [double](#) kink)  
*Return continuity between end of edge1 and start of edge2.*

### 28.48.1 Function Documentation

28.48.1.1 [template](#)<class [edgeType](#)> void [Go::tpUtils::adjacentEdges](#) ( [edgeType](#) \* edge, [bool](#) at\_start\_of\_edge, [std::vector](#)< [edgeType](#) \* > & adjacent, [std::vector](#)< [bool](#) > & at\_start )

Consider the 'node' or 'corner' implied by the startpoint (if at\_start\_of\_edge is true) or endpoint (if it is false). The vector adjacent is filled with all edges with that 'node' as a startpoint (indicated by a true at the corresponding place in the at\_start vector) or endpoint (false in at\_start). The edge originally asked for is not included.

Definition at line 60 of file tpUtils.h.

28.48.1.2 `template<class edgeType > tpJointType Go::tpUtils::checkContinuity ( const edgeType * edge1, const edgeType * edge2, double neighbour, double gap, double bend, double kink )`

Return continuity between end of edge1 and start of edge2.

Definition at line 138 of file tpUtils.h.

## 28.49 Go::Utils Namespace Reference

Namespace for some utility functions.

### Functions

- `template<typename ForwardIterator > go_iterator_traits< ForwardIterator >::value_type sum (ForwardIterator first, ForwardIterator last)`  
*sum finds the sum of the elements*
- `template<typename ForwardIterator > go_iterator_traits< ForwardIterator >::value_type sum_squared (ForwardIterator first, ForwardIterator last)`  
*sum\_squared finds the squared sum of the elements*
- `template<typename ForwardIterator > go_iterator_traits< ForwardIterator >::value_type distance_squared (ForwardIterator first1, ForwardIterator last1, ForwardIterator first2)`  
*distance\_squared*
- `template<typename ForwardIterator > void normalize (ForwardIterator first, ForwardIterator last)`  
*normalize makes the length of a vector 1.0*
- `template<typename ForwardIterator > go_iterator_traits< ForwardIterator >::value_type inner (ForwardIterator first, ForwardIterator last, ForwardIterator second)`  
*inner product*
- `template<typename InputStream > InputStream & eatwhite (InputStream &is)`  
*eat white space*

### 28.49.1 Detailed Description

Namespace for some utility functions.

### 28.49.2 Function Documentation

28.49.2.1 `template<typename ForwardIterator > go_iterator_traits<ForwardIterator>::value_type Go::Utils::distance_squared ( ForwardIterator first1, ForwardIterator last1, ForwardIterator first2 ) [inline]`

distance\_squared

Definition at line 113 of file Utils.h.

28.49.2.2 `template<typename InputStream > InputStream& Go::Utils::eatwhite ( InputStream & is ) [inline]`

eat white space

Definition at line 151 of file Utils.h.

28.49.2.3 `template<typename ForwardIterator > go_iterator_traits<ForwardIterator>::value_type Go::Utils::inner ( ForwardIterator first, ForwardIterator last, ForwardIterator second ) [inline]`

inner product

Definition at line 139 of file Utils.h.

28.49.2.4 `template<typename ForwardIterator > void Go::Utils::normalize ( ForwardIterator first, ForwardIterator last ) [inline]`

normalize makes the length of a vector 1.0

Definition at line 126 of file Utils.h.

28.49.2.5 `template<typename ForwardIterator > go_iterator_traits<ForwardIterator>::value_type Go::Utils::sum ( ForwardIterator first, ForwardIterator last ) [inline]`

sum finds the sum of the elements

Definition at line 89 of file Utils.h.

28.49.2.6 `template<typename ForwardIterator > go_iterator_traits<ForwardIterator>::value_type Go::Utils::sum_squared ( ForwardIterator first, ForwardIterator last ) [inline]`

sum\_squared finds the squared sum of the elements

Definition at line 101 of file Utils.h.

## 28.50 Go::VolumeInterpolator Namespace Reference

This namespace contains functions used to interpolate a set of points.

### Functions

- [SplineVolume](#) \* [regularInterpolation](#) ([const BsplineBasis](#) &basis\_u, [const BsplineBasis](#) &basis\_v, [const BsplineBasis](#) &basis\_w, [std::vector< double >](#) &par\_u, [std::vector< double >](#) &par\_v, [std::vector< double >](#) &par\_w, [std::vector< double >](#) &points, [int](#) dimension, [bool](#) rational, [std::vector< double >](#) &weights)

### 28.50.1 Detailed Description

This namespace contains functions used to interpolate a set of points.

### 28.50.2 Function Documentation

28.50.2.1 **SplineVolume\*** `Go::VolumeInterpolator::regularInterpolation ( const BsplineBasis & basis_u, const BsplineBasis & basis_v, const BsplineBasis & basis_w, std::vector< double > & par_u, std::vector< double > & par_v, std::vector< double > & par_w, std::vector< double > & points, int dimension, bool rational, std::vector< double > & weights )`

Interpolate a set of regular, parameterized interpolation points. The parameterization is assumed to correspond to the given B-spline bases (select interpolation points in the Greville points). The function throws if the input parameters are inconsistent

#### Parameters

|                  |                                                                              |
|------------------|------------------------------------------------------------------------------|
| <i>basis_u</i>   | spline basis in the first parameter direction                                |
| <i>basis_v</i>   | spline basis in the second parameter direction                               |
| <i>basis_w</i>   | spline basis in the third parameter direction                                |
| <i>par_u</i>     | parameter values in 1. parameter direction corresponding to point            |
| <i>par_v</i>     | parameter values in 2. parameter direction corresponding to point            |
| <i>par_w</i>     | parameter values in 2. parameter direction corresponding to point            |
| <i>points</i>    | the regular point set                                                        |
| <i>dimension</i> | dimension of geometry space                                                  |
| <i>rational</i>  | whether or not a rational surface is expected                                |
| <i>weights</i>   | the weights of the rational volume, used only if <code>rational==true</code> |

## 28.51 Go::VolumeModelCreator Namespace Reference

### Functions

- `bool createRotationalModel ( shared_ptr< SurfaceModel > &sfmodel, shared_ptr< VolumeModel > &volmodel )`
- `bool linearSweptModel ( shared_ptr< SurfaceModel > &sfmodel, shared_ptr< VolumeModel > &volmodel )`

### 28.51.1 Function Documentation

28.51.1.1 `bool Go::VolumeModelCreator::createRotationalModel ( shared_ptr< SurfaceModel > & sfmodel, shared_ptr< VolumeModel > & volmodel )`

28.51.1.2 `bool Go::VolumeModelCreator::linearSweptModel ( shared_ptr< SurfaceModel > & sfmodel, shared_ptr< VolumeModel > & volmodel )`

## 28.52 Go::VolumeTools Namespace Reference

This namespace contains free functions operating on parametric volumes.

## Functions

- `int GO_API analyzePeriodicity (const SplineVolume &sf, int direction, double knot_tol=1e-12)`
- `shared_ptr< SplineCurve > representVolumeAsCurve (const SplineVolume &volume, int cv_dir)`
- `shared_ptr< SplineVolume > representCurveAsVolume (const SplineCurve &curve, int cv_dir, const BsplineBasis &other_bas1, const BsplineBasis &other_bas2, bool rational)`
- `shared_ptr< SplineSurface > representVolumeAsSurface (const SplineVolume &volume, int sf_dir1, int sf_dir2)`
- `shared_ptr< SplineVolume > representSurfaceAsVolume (const SplineSurface &surface, int sf_dir1, int sf_dir2, const BsplineBasis &other_bas, bool rational)`
- `bool cornerToCornerVols (shared_ptr< ParamVolume > vol1, shared_ptr< SurfaceOnVolume > vol_sf1, shared_ptr< ParamVolume > vol2, shared_ptr< SurfaceOnVolume > vol_sf2, double tol)`
- `bool getVolAdjacencyInfo (shared_ptr< ParamVolume > vol1, shared_ptr< SurfaceOnVolume > vol_sf1, shared_ptr< ParamVolume > vol2, shared_ptr< SurfaceOnVolume > vol_sf2, double tol, int &bd1, int &bd2, int &orientation, bool &same_seq)`
- `bool getCorrCoefVolEnum (shared_ptr< SplineVolume > vol1, shared_ptr< SplineVolume > vol2, int bd1, int bd2, int orientation, bool same_seq, std::vector< std::pair< int, int > > &enumeration)`
- `bool getVolCoefEnumeration (shared_ptr< SplineVolume > vol, int bd, std::vector< int > &enumeration)`
- `bool getVolCoefEnumeration (shared_ptr< SplineVolume > vol, int bd, std::vector< int > &enumeration_bd, std::vector< int > &enumeration_bd2)`
- `bool getVolBdCoefEnumeration (shared_ptr< SplineVolume > vol, int bd, int bd_cv, std::vector< int > &enumeration)`
- `std::vector< shared_ptr< ParamSurface > > getBoundarySurfaces (shared_ptr< ParamVolume > vol)`  
*Given a parametric volume, fetch all boundary surfaces.*
- `std::vector< shared_ptr< ParamSurface > > getOrientedBoundarySurfaces (shared_ptr< ParamVolume > vol)`
- `shared_ptr< SurfaceOnVolume > getBoundarySurface (shared_ptr< SplineVolume > vol, int idx)`  
*Given a spline volume, fetch all boundary surfaces.*
- `shared_ptr< SurfaceOnVolume > getOrientedBoundarySurface (shared_ptr< SplineVolume > vol, int idx)`
- `void volCommonSplineSpace (shared_ptr< SplineVolume > vol1, int bd1, shared_ptr< SplineVolume > vol2, int bd2, int orientation, bool same_seq)`
- `shared_ptr< SplineCurve > liftVolParamCurve (shared_ptr< ParamCurve > pcurve, shared_ptr< ParamVolume > vol, double tol)`
- `shared_ptr< SplineCurve > projectVolParamCurve (shared_ptr< ParamCurve > spacecurve, shared_ptr< ParamVolume > vol, double tol)`
- `shared_ptr< SplineCurve > approxVolParamCurve (shared_ptr< ParamCurve > spacecurve, shared_ptr< ParamVolume > vol, double tol, int max_iter, double &maxdist)`

### 28.52.1 Detailed Description

This namespace contains free functions operating on parametric volumes.

### 28.52.2 Function Documentation

**28.52.2.1** `int GO_API Go::VolumeTools::analyzePeriodicity ( const SplineVolume & sf, int direction, double knot_tol = 1e-12 )`

Analyze periodicity of volume based on number of repeating knots and control points. The return value is -1 if the volume edges are disjoint, otherwise  $k$  if  $sf$  is  $C^k$  continuous across the seam. These are sufficient but not necessary conditions for periodicity, so it is possible that a call to `analyzePeriodicityDerivs()` will yield a higher degree of periodicity. The current implementation is quite slow, and not optimized for speed.



## Parameters

|                  |                                                                                                                                                              |
|------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>sf</i>        | reference to the <a href="#">SplineVolume</a> to be analyzed                                                                                                 |
| <i>direction</i> | specify 'direction' to be '0' to check for periodicity in the first parameter direction, or '1' to check the second parameter direction, or 2 for the third. |
| <i>knot_tol</i>  | the tolerance used when comparing knot intervals                                                                                                             |

## Returns

-1 if the volume edges are disjoint, otherwise k if the

**28.52.2.2** `shared_ptr<SplineCurve> Go::VolumeTools::approxVolParamCurve ( shared_ptr< ParamCurve > spacecurve, shared_ptr< ParamVolume > vol, double tol, int max_iter, double & maxdist )`

**28.52.2.3** `bool Go::VolumeTools::cornerToCornerVols ( shared_ptr< ParamVolume > vol1, shared_ptr< SurfaceOnVolume > vol_sf1, shared_ptr< ParamVolume > vol2, shared_ptr< SurfaceOnVolume > vol_sf2, double tol )`

Check if two neighbouring volumes in a volume set have a corner-to-corner configuration, i.e. no T-joints

## Parameters

|                |                                                                          |
|----------------|--------------------------------------------------------------------------|
| <i>vol1</i>    | one volume                                                               |
| <i>vol_sf1</i> | the boundary face corresponding to vol1 at the boundary common with vol2 |
| <i>vol2</i>    | the other volume                                                         |
| <i>vol_sf1</i> | the boundary face corresponding to vol2 at the boundary common with vol1 |
| <i>return</i>  | parameter true of the configuration is corner-to-corner                  |

**28.52.2.4** `shared_ptr<SurfaceOnVolume> Go::VolumeTools::getBoundarySurface ( shared_ptr< SplineVolume > vol, int idx )`

Given a spline volume, fetch all boundary surfaces.

**28.52.2.5** `std::vector<shared_ptr<ParamSurface>> Go::VolumeTools::getBoundarySurfaces ( shared_ptr< ParamVolume > vol )`

Given a parametric volume, fetch all boundary surfaces.

**28.52.2.6** `bool Go::VolumeTools::getCorrCoefVolEnum ( shared_ptr< SplineVolume > vol1, shared_ptr< SplineVolume > vol2, int bd1, int bd2, int orientation, bool same_seq, std::vector< std::pair< int, int > > & enumeration )`

Given information about the adjacency relationship between two spline volumes (the enumeration of the common boundary and the correspondance in parameterization, see the function `getVolAdjacencyInfo`), fetch the enumeration of pairwise corresponding coefficients. The function returns false if the spline spaces of the two surfaces at the common boundary are not the same.

28.52.2.7 `shared_ptr<SurfaceOnVolume> Go::VolumeTools::getOrientedBoundarySurface ( shared_ptr<SplineVolume > vol, int idx )`

Given a spline volume, fetch all boundary surfaces and ensure a consistent normal vector direction of these surfaces

28.52.2.8 `std::vector<shared_ptr<ParamSurface> > Go::VolumeTools::getOrientedBoundarySurfaces ( shared_ptr<ParamVolume > vol )`

Given a parametric volume, fetch all boundary surfaces and ensure a consistent normal vector direction of these surfaces

28.52.2.9 `bool Go::VolumeTools::getVolAdjacencyInfo ( shared_ptr< ParamVolume > vol1, shared_ptr< SurfaceOnVolume > vol_sf1, shared_ptr< ParamVolume > vol2, shared_ptr< SurfaceOnVolume > vol_sf2, double tol, int & bd1, int & bd2, int & orientation, bool & same_seq )`

Given two neighbouring volumes, vol1 and vol2, and information about the common boundary, fetch information about the adjacency configuration

#### Parameters

|                    |                                                                                                                                                                                                                                                                                                   |
|--------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>vol1</i>        | the first volume                                                                                                                                                                                                                                                                                  |
| <i>vol_sf1</i>     | the boundary surface/trimming surface corresponding to vol1 at the boundary common with vol2                                                                                                                                                                                                      |
| <i>vol2</i>        | the other volume                                                                                                                                                                                                                                                                                  |
| <i>vol_sf2</i>     | the boundary surface/trimming surface corresponding to vol2 at the boundary common with vol1                                                                                                                                                                                                      |
| <i>tol</i>         | adjacency tolerance                                                                                                                                                                                                                                                                               |
| <i>return</i>      | parameter true if the two volumes share the same boundary and that boundary is boundary trimmed in both volumes                                                                                                                                                                                   |
| <i>bd1</i>         | boundary surface of first volume which follows the common boundary, -1=no such boundary surface, 0=umin, 1=umax, 2=vmin, 3=vmax, 4=wmin, 5=wmax                                                                                                                                                   |
| <i>bd2</i>         | boundary surface of second volume which follows the common boundary                                                                                                                                                                                                                               |
| <i>orientation</i> | =0: The surfaces at the common boundary have the same orientation in both parameter directions, =1: the orientation is opposite in the first parameter direction, =2: the orientation is opposite in the second parameter direction, =3: the orientation is opposite in both parameter directions |
| <i>same_seq</i>    | tells if 1. parameter direction of the first boundary surface corresponds to the 1. parameter direction of the second boundary surface (true) or if the parameter directions are swapped (false)                                                                                                  |

28.52.2.10 `bool Go::VolumeTools::getVolBdCoefEnumeration ( shared_ptr< SplineVolume > vol, int bd, int bd_cv, std::vector< int > & enumeration )`

Given a spline volume, a boundary surface enumeration (bd=0:umin, bd=1:umax, bd=2:vmin, bd=3:vmax, bd=4:wmin, bd=5:wmax) and a boundary curve enumeration corresponding to the boundary surface (bd\_cv=0:umin, bd\_cv=1:umax, bd\_cv=2:vmin, bd\_cv=3:vmax), fetch the enumeration of the volume coefficients along that boundary curve

28.52.2.11 `bool Go::VolumeTools::getVolCoefEnumeration ( shared_ptr< SplineVolume > vol, int bd, std::vector< int > & enumeration )`

Given a spline volume and a boundary enumeration (bd=0:umin, bd=1:umax, bd=2:vmin, bd=3:vmax, bd=4:wmin, bd=5:wmax), fetch the enumeration of the volume coefficients at that boundary surface

28.52.2.12 `bool Go::VolumeTools::getVolCoefEnumeration ( shared_ptr< SplineVolume > vol, int bd, std::vector< int > & enumeration_bd, std::vector< int > & enumeration_bd2 )`

Given a spline volume and a boundary enumeration (bd=0:umin, bd=1:umax, bd=2:vmin, bd=3:vmax, bd=4:wmin, bd=5:wmax), fetch the enumeration of the volume coefficients at that boundary surface

28.52.2.13 `shared_ptr< SplineCurve > Go::VolumeTools::liftVolParamCurve ( shared_ptr< ParamCurve > pcurve, shared_ptr< ParamVolume > vol, double tol )`

Approximate a parameter curve in the parameter space of a given volume by a space curve

28.52.2.14 `shared_ptr< SplineCurve > Go::VolumeTools::projectVolParamCurve ( shared_ptr< ParamCurve > spacecurve, shared_ptr< ParamVolume > vol, double tol )`

Approximate a given space curve by a curve in the parameter space of a given volume

28.52.2.15 `shared_ptr< SplineVolume > Go::VolumeTools::representCurveAsVolume ( const SplineCurve & curve, int cv_dir, const BsplineBasis & other_bas1, const BsplineBasis & other_bas2, bool rational )`

Describe a curve as a lower-dimensional volume in a given direction.

#### Parameters

|                   |                                                                                                                                                                                                                                                                                                                                                                       |
|-------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>curve</i>      | the curve that we want to express as a volume                                                                                                                                                                                                                                                                                                                         |
| <i>cv_dir</i>     | If this variable is set to 0, then the curve's parameter will become the <i>first</i> parameter in the generated volume. If it is set to 1, the curve's parameter will become the <i>second</i> parameter in the generated volume. If it is set to 2, the curve's parameter will become the <i>third</i> parameter in the generated volume. Other values are illegal. |
| <i>other_bas1</i> | the first <a href="#">BsplineBasis</a> for the additional parameter directions. If <i>cv_dir</i> is 0, this will be the second parameter direction on the volume. Otherwise, it will be the first parameter direction                                                                                                                                                 |
| <i>other_bas2</i> | the second <a href="#">BsplineBasis</a> for the additional parameter directions. If <i>cv_dir</i> is 2, this will be the second parameter direction on the volume. Otherwise, it will be the third parameter direction                                                                                                                                                |
| <i>rational</i>   | define whether the generated volume shall be specified as <i>rational</i> or not.                                                                                                                                                                                                                                                                                     |

#### Returns

a shared pointer to a new [SplineVolume](#), expressing the curve in a space of lower dimensionality.

28.52.2.16 `shared_ptr< SplineVolume > Go::VolumeTools::representSurfaceAsVolume ( const SplineSurface & surface, int sf_dir1, int sf_dir2, const BsplineBasis & other_bas, bool rational )`

Describe a surface as a lower-dimensional volume in given directions.

## Parameters

|                  |                                                                                                                                                                                                 |
|------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>surface</i>   | the surface that we want to express as a volume                                                                                                                                                 |
| <i>sf_dir1</i>   | The volume parameter direction from the first surface parameter direction. The value of <i>sf_dir1</i> must be either 0, 1 or 2. Other values are illegal.                                      |
| <i>sf_dir2</i>   | The volume parameter direction from the second surface parameter direction. The value of <i>sf_dir2</i> must be either 0, 1 or 2, and different from <i>sf_dir1</i> . Other values are illegal. |
| <i>other_bas</i> | the <a href="#">BsplineBasis</a> for the additional volume parameter direction, different <i>sf_dir1</i> and <i>sf_dir2</i> .                                                                   |
| <i>rational</i>  | define whether the generated volume shall be specified as <i>rational</i> or not.                                                                                                               |

## Returns

a shared pointer to a new [SplineVolume](#), expressing the surface in a space of lower dimensionality.

**28.52.2.17** `shared_ptr<SplineCurve> Go::VolumeTools::representVolumeAsCurve ( const SplineVolume & volume, int cv_dir )`

Describe a volume as a high-dimensional curve in a given direction. If the volume is rational, the curve will be non-rational and living in the homogenous space.

## Parameters

|               |                                                                                                                                                                                                                                                                                                                                                             |
|---------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>volume</i> | the volume to express as a curve                                                                                                                                                                                                                                                                                                                            |
| <i>cv_dir</i> | the parameter direction that will be kept when defining the curve (the other two will disappear, as the control points in this direction will be lumped together and expressed as single control points in a higher-dimensional space. ' <i>cv_dir</i> ' takes the values 0, 1 or 2 for keeping the first, second or third parameter direction respectively |

## Returns

shared pointer to a new [SplineCurve](#), expressing the volume as a curve in a high-dimensional space.

**28.52.2.18** `shared_ptr<SplineSurface> Go::VolumeTools::representVolumeAsSurface ( const SplineVolume & volume, int sf_dir1, int sf_dir2 )`

Describe a volume as a high-dimensional surface in two given directions. If the volume is rational, the surface will be non-rational and living in the homogenous space.

## Parameters

|                |                                                                                                                                                                                                                                                                                                                                  |
|----------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>volume</i>  | the volume to express as a surface                                                                                                                                                                                                                                                                                               |
| <i>sf_dir1</i> | the volume parameter direction to become the first parameter direction of the surface, either 0, 1 or 2.                                                                                                                                                                                                                         |
| <i>sf_dir2</i> | the volume parameter direction to become the second parameter direction of the surface, either 0, 1 or 2, and different from <i>sf_dir1</i> . The control points in the direction different from <i>sf_dir1</i> and <i>sf_dir2</i> will be lumped together and expressed as single control points in a higher-dimensional space. |

## Returns

shared pointer to a new [SplineSurface](#), expressing the volume as a surface in a high-dimensional space.

```
28.52.2.19 void Go::VolumeTools::volCommonSplineSpace (shared_ptr< SplineVolume > vol1, int bd1, shared_ptr< SplineVolume > vol2, int bd2, int orientation, bool same_seq)
```

Given two spline volumes sharing a common boundary and information about the configuration of this boundary in the spline volumes (see the explanation of `getVolAdjacencyInfo`), ensure that the two volumes share the same spline space at the common boundary

## 28.53 hed Namespace Reference

### Classes

- class [Dart](#)  
*Dart* class for the half-edge data structure.
- class [Edge](#)  
*Edge* class in the in the half-edge data structure.
- class [Node](#)  
*Node* class for data structures (Inherits from [HandleId](#))
- class [Triangulation](#)  
*Triangulation* class for the half-edge data structure with adaption to TTL.
- struct [TTLtraits](#)  
*Traits* class (static struct) for the half-edge data structure.

## 28.54 hetriang Namespace Reference

### Classes

- class [Dart](#)  
*Dart* class for the half-edge data structure.
- class [Edge](#)  
*Edge* class in the half-edge data structure
- class [Node](#)  
*Node* class in the half-edge data structure
- class [Triangulation](#)  
*Triangulation* class for the half-edge data structure with adaption to TTL.
- struct [TTLtraits](#)  
*Traits* class (static struct) for the half-edge data structure.

## 28.55 NEWMAT Namespace Reference

### Classes

- class [AddedMatrix](#)
- class [ArrayLengthSpecifier](#)
- class [BandLUMatrix](#)
- class [BandMatrix](#)
- class [BaseMatrix](#)
- class [CannotBuildException](#)
- class [ColedMatrix](#)
- class [ColumnVector](#)
- class [ConcatenatedMatrix](#)
- class [ConvergenceException](#)
- class [CroutMatrix](#)
- class [DiagedMatrix](#)
- class [DiagonalMatrix](#)
- class [FFT\\_Controller](#)
- class [FindMaximum2](#)
- class [FloatingPointPrecision](#)
  - *Floating point precision (type double).*
- class [GeneralMatrix](#)
- class [GenericMatrix](#)
- class [GetSubMatrix](#)
- class [IdentityMatrix](#)
- class [IncompatibleDimensionsException](#)
- class [IndexException](#)
- class [InternalException](#)
- class [InvertedMatrix](#)
- class [KPMatrix](#)
- class [LinearEquationSolver](#)
- class [LL\\_D\\_FI](#)
- class [LogAndSign](#)
- class [LowerBandMatrix](#)
- class [LowerTriangularMatrix](#)
- class [MatedMatrix](#)
- class [Matrix](#)
- class [MatrixBandWidth](#)
- class [MatrixInput](#)
- class [MatrixType](#)
- class [MLE\\_D\\_FI](#)
- class [MultipliedMatrix](#)
- class [MultiRadixCounter](#)
- class [NegatedMatrix](#)
- class [NegShiftedMatrix](#)
- class [NonLinearLeastSquares](#)
- class [NotDefinedException](#)
- class [NotSquareException](#)
- class [NPDException](#)
- class [nricMatrix](#)
- class [OverflowException](#)
- class [ProgramException](#)
- class [R1\\_Col\\_I\\_D](#)
- class [ReturnMatrixX](#)

- class [ReversedMatrix](#)
- class [RowedMatrix](#)
- class [RowVector](#)
- class [ScaledMatrix](#)
- class [ShiftedMatrix](#)
- class [SimpleIntArray](#)
- class [SingularException](#)
- class [SolvedMatrix](#)
- class [SPMatrix](#)
- class [StackedMatrix](#)
- class [SubMatrixDimensionException](#)
- class [SubtractedMatrix](#)
- class [SymmetricBandMatrix](#)
- class [SymmetricEigenAnalysis](#)
- class [SymmetricMatrix](#)
- class [TransposedMatrix](#)
- class [UpperBandMatrix](#)
- class [UpperTriangularMatrix](#)
- class [VectorException](#)

## Functions

- void [MatrixErrorNoSpace](#) (void \*)
- bool [operator==](#) (const [GeneralMatrix](#) &A, const [GeneralMatrix](#) &B)
- bool [operator==](#) (const [BaseMatrix](#) &A, const [BaseMatrix](#) &B)
- bool [operator!=](#) (const [GeneralMatrix](#) &A, const [GeneralMatrix](#) &B)
- bool [operator!=](#) (const [BaseMatrix](#) &A, const [BaseMatrix](#) &B)
- bool [operator<=](#) (const [BaseMatrix](#) &A, const [BaseMatrix](#) &B)
- bool [operator>=](#) (const [BaseMatrix](#) &A, const [BaseMatrix](#) &B)
- bool [operator<](#) (const [BaseMatrix](#) &A, const [BaseMatrix](#) &B)
- bool [operator>](#) (const [BaseMatrix](#) &A, const [BaseMatrix](#) &B)
- bool [IsZero](#) (const [BaseMatrix](#) &A)
- bool [Rectangular](#) ([MatrixType](#) a, [MatrixType](#) b, [MatrixType](#) c)
- bool [Compare](#) (const [MatrixType](#) &A, [MatrixType](#) &B)
- [Real](#) [DotProduct](#) (const [Matrix](#) &A, const [Matrix](#) &B)
- [SPMatrix](#) [SP](#) (const [BaseMatrix](#) &A, const [BaseMatrix](#) &B)
- [KPMatrix](#) [KP](#) (const [BaseMatrix](#) &A, const [BaseMatrix](#) &B)
- [ShiftedMatrix](#) [operator+](#) ([Real](#) f, const [BaseMatrix](#) &BM)
- [NegShiftedMatrix](#) [operator-](#) ([Real](#) f, const [BaseMatrix](#) &BM)
- [ScaledMatrix](#) [operator\\*](#) ([Real](#) f, const [BaseMatrix](#) &BM)
- [LogAndSign](#) [LogDeterminant](#) (const [BaseMatrix](#) &B)
- [Real](#) [Determinant](#) (const [BaseMatrix](#) &B)
- [Real](#) [SumSquare](#) (const [BaseMatrix](#) &B)
- [Real](#) [NormFrobenius](#) (const [BaseMatrix](#) &B)
- [Real](#) [Trace](#) (const [BaseMatrix](#) &B)
- [Real](#) [SumAbsoluteValue](#) (const [BaseMatrix](#) &B)
- [Real](#) [Sum](#) (const [BaseMatrix](#) &B)
- [Real](#) [MaximumAbsoluteValue](#) (const [BaseMatrix](#) &B)
- [Real](#) [MinimumAbsoluteValue](#) (const [BaseMatrix](#) &B)
- [Real](#) [Maximum](#) (const [BaseMatrix](#) &B)
- [Real](#) [Minimum](#) (const [BaseMatrix](#) &B)
- [Real](#) [Norm1](#) (const [BaseMatrix](#) &B)
- [Real](#) [Norm1](#) ([RowVector](#) &RV)
- [Real](#) [NormInfinity](#) (const [BaseMatrix](#) &B)

- [Real NormInfinity](#) ([ColumnVector](#) &CV)
- [bool IsZero](#) ([const GeneralMatrix](#) &A)
- [void QRZT](#) ([Matrix](#) &, [LowerTriangularMatrix](#) &)
- [void QRZT](#) ([const Matrix](#) &, [Matrix](#) &, [Matrix](#) &)
- [void QRZ](#) ([Matrix](#) &, [UpperTriangularMatrix](#) &)
- [void QRZ](#) ([const Matrix](#) &, [Matrix](#) &, [Matrix](#) &)
- [void HHDecompose](#) ([Matrix](#) &X, [LowerTriangularMatrix](#) &L)
- [void HHDecompose](#) ([const Matrix](#) &X, [Matrix](#) &Y, [Matrix](#) &M)
- [ReturnMatrix Cholesky](#) ([const SymmetricMatrix](#) &)
- [ReturnMatrix Cholesky](#) ([const SymmetricBandMatrix](#) &)
- [void SVD](#) ([const Matrix](#) &, [DiagonalMatrix](#) &, [Matrix](#) &, [Matrix](#) &, [bool=true](#), [bool=true](#))
- [void SVD](#) ([const Matrix](#) &, [DiagonalMatrix](#) &)
- [void SVD](#) ([const Matrix](#) &A, [DiagonalMatrix](#) &D, [Matrix](#) &U, [bool withU=true](#))
- [void SortSV](#) ([DiagonalMatrix](#) &D, [Matrix](#) &U, [bool ascending=false](#))
- [void SortSV](#) ([DiagonalMatrix](#) &D, [Matrix](#) &U, [Matrix](#) &V, [bool ascending=false](#))
- [void Jacobi](#) ([const SymmetricMatrix](#) &, [DiagonalMatrix](#) &)
- [void Jacobi](#) ([const SymmetricMatrix](#) &, [DiagonalMatrix](#) &, [SymmetricMatrix](#) &)
- [void Jacobi](#) ([const SymmetricMatrix](#) &, [DiagonalMatrix](#) &, [Matrix](#) &)
- [void Jacobi](#) ([const SymmetricMatrix](#) &, [DiagonalMatrix](#) &, [SymmetricMatrix](#) &, [Matrix](#) &, [bool=true](#))
- [void EigenValues](#) ([const SymmetricMatrix](#) &, [DiagonalMatrix](#) &)
- [void EigenValues](#) ([const SymmetricMatrix](#) &, [DiagonalMatrix](#) &, [SymmetricMatrix](#) &)
- [void EigenValues](#) ([const SymmetricMatrix](#) &, [DiagonalMatrix](#) &, [Matrix](#) &)
- [void SortAscending](#) ([GeneralMatrix](#) &)
- [void SortDescending](#) ([GeneralMatrix](#) &)
- [void FFT](#) ([const ColumnVector](#) &, [const ColumnVector](#) &, [ColumnVector](#) &, [ColumnVector](#) &)
- [void FFTI](#) ([const ColumnVector](#) &, [const ColumnVector](#) &, [ColumnVector](#) &, [ColumnVector](#) &)
- [void RealFFT](#) ([const ColumnVector](#) &, [ColumnVector](#) &, [ColumnVector](#) &)
- [void RealFFTI](#) ([const ColumnVector](#) &, [const ColumnVector](#) &, [ColumnVector](#) &)
- [void DCT\\_II](#) ([const ColumnVector](#) &, [ColumnVector](#) &)
- [void DCT\\_II\\_inverse](#) ([const ColumnVector](#) &, [ColumnVector](#) &)
- [void DST\\_II](#) ([const ColumnVector](#) &, [ColumnVector](#) &)
- [void DST\\_II\\_inverse](#) ([const ColumnVector](#) &, [ColumnVector](#) &)
- [void DCT](#) ([const ColumnVector](#) &, [ColumnVector](#) &)
- [void DCT\\_inverse](#) ([const ColumnVector](#) &, [ColumnVector](#) &)
- [void DST](#) ([const ColumnVector](#) &, [ColumnVector](#) &)
- [void DST\\_inverse](#) ([const ColumnVector](#) &, [ColumnVector](#) &)
- [std::ostream & operator<<](#) ([std::ostream](#) &, [const BaseMatrix](#) &)
- [std::ostream & operator<<](#) ([std::ostream](#) &, [const GeneralMatrix](#) &)

## 28.55.1 Function Documentation

- 28.55.1.1 [ReturnMatrix NEWMAT::Cholesky](#) ( [const SymmetricMatrix](#) & )
- 28.55.1.2 [ReturnMatrix NEWMAT::Cholesky](#) ( [const SymmetricBandMatrix](#) & )
- 28.55.1.3 [bool NEWMAT::Compare](#) ( [const MatrixType](#) &, [MatrixType](#) & )
- 28.55.1.4 [void NEWMAT::DCT](#) ( [const ColumnVector](#) &, [ColumnVector](#) & )
- 28.55.1.5 [void NEWMAT::DCT\\_II](#) ( [const ColumnVector](#) &, [ColumnVector](#) & )
- 28.55.1.6 [void NEWMAT::DCT\\_II\\_inverse](#) ( [const ColumnVector](#) &, [ColumnVector](#) & )
- 28.55.1.7 [void NEWMAT::DCT\\_inverse](#) ( [const ColumnVector](#) &, [ColumnVector](#) & )
- 28.55.1.8 [Real NEWMAT::Determinant](#) ( [const BaseMatrix](#) & B ) `[inline]`

Definition at line 1744 of file newmat.h.



28.55.1.9 Real NEWMAT::DotProduct ( const Matrix & A, const Matrix & B )

28.55.1.10 void NEWMAT::DST ( const ColumnVector &, ColumnVector & )

28.55.1.11 void NEWMAT::DST\_II ( const ColumnVector &, ColumnVector & )

28.55.1.12 void NEWMAT::DST\_II\_inverse ( const ColumnVector &, ColumnVector & )

28.55.1.13 void NEWMAT::DST\_inverse ( const ColumnVector &, ColumnVector & )

28.55.1.14 void NEWMAT::EigenValues ( const SymmetricMatrix &, DiagonalMatrix & )

28.55.1.15 void NEWMAT::EigenValues ( const SymmetricMatrix &, DiagonalMatrix &, SymmetricMatrix & )

28.55.1.16 void NEWMAT::EigenValues ( const SymmetricMatrix &, DiagonalMatrix &, Matrix & )

28.55.1.17 void NEWMAT::FFT ( const ColumnVector &, const ColumnVector &, ColumnVector &, ColumnVector & )

28.55.1.18 void NEWMAT::FFTI ( const ColumnVector &, const ColumnVector &, ColumnVector &, ColumnVector & )

28.55.1.19 void NEWMAT::HHDecompose ( Matrix & X, LowerTriangularMatrix & L ) [inline]

Definition at line 26 of file newmatap.h.

28.55.1.20 void NEWMAT::HHDecompose ( const Matrix & X, Matrix & Y, Matrix & M ) [inline]

Definition at line 29 of file newmatap.h.

28.55.1.21 bool NEWMAT::IsZero ( const BaseMatrix & A )

28.55.1.22 bool NEWMAT::IsZero ( const GeneralMatrix & A ) [inline]

Definition at line 1764 of file newmat.h.

28.55.1.23 void NEWMAT::Jacobi ( const SymmetricMatrix &, DiagonalMatrix & )

28.55.1.24 void NEWMAT::Jacobi ( const SymmetricMatrix &, DiagonalMatrix &, SymmetricMatrix & )

28.55.1.25 void NEWMAT::Jacobi ( const SymmetricMatrix &, DiagonalMatrix &, Matrix & )

28.55.1.26 void NEWMAT::Jacobi ( const SymmetricMatrix &, DiagonalMatrix &, SymmetricMatrix &, Matrix &, bool = true )

28.55.1.27 KPMatrix NEWMAT::KP ( const BaseMatrix &, const BaseMatrix & )

28.55.1.28 LogAndSign NEWMAT::LogDeterminant ( const BaseMatrix & B ) [inline]

Definition at line 1742 of file newmat.h.

28.55.1.29 `void NEWMAT::MatrixErrorNoSpace ( void * )`

Definition at line 275 of file newmatex.cpp.

28.55.1.30 `Real NEWMAT::Maximum ( const BaseMatrix & B ) [inline]`

Definition at line 1757 of file newmat.h.

28.55.1.31 `Real NEWMAT::MaximumAbsoluteValue ( const BaseMatrix & B ) [inline]`

Definition at line 1753 of file newmat.h.

28.55.1.32 `Real NEWMAT::Minimum ( const BaseMatrix & B ) [inline]`

Definition at line 1758 of file newmat.h.

28.55.1.33 `Real NEWMAT::MinimumAbsoluteValue ( const BaseMatrix & B ) [inline]`

Definition at line 1755 of file newmat.h.

28.55.1.34 `Real NEWMAT::Norm1 ( const BaseMatrix & B ) [inline]`

Definition at line 1759 of file newmat.h.

28.55.1.35 `Real NEWMAT::Norm1 ( RowVector & RV ) [inline]`

Definition at line 1760 of file newmat.h.

28.55.1.36 `Real NEWMAT::NormFrobenius ( const BaseMatrix & B ) [inline]`

Definition at line 1747 of file newmat.h.

28.55.1.37 `Real NEWMAT::NormInfinity ( const BaseMatrix & B ) [inline]`

Definition at line 1761 of file newmat.h.

28.55.1.38 `Real NEWMAT::NormInfinity ( ColumnVector & CV ) [inline]`

Definition at line 1762 of file newmat.h.

28.55.1.39 **bool** NEWMAT::operator!=( const GeneralMatrix & A, const GeneralMatrix & B ) [inline]

Definition at line 1703 of file newmat.h.

28.55.1.40 **bool** NEWMAT::operator!=( const BaseMatrix & A, const BaseMatrix & B ) [inline]

Definition at line 1705 of file newmat.h.

28.55.1.41 **ScaledMatrix** NEWMAT::operator\*( Real f, const BaseMatrix & BM ) [inline]

Definition at line 1778 of file newmat.h.

28.55.1.42 **ShiftedMatrix** NEWMAT::operator+( Real f, const BaseMatrix & BM ) [inline]

Definition at line 1776 of file newmat.h.

28.55.1.43 **NegShiftedMatrix** NEWMAT::operator-( Real , const BaseMatrix & )

28.55.1.44 **bool** NEWMAT::operator<( const BaseMatrix & A, const BaseMatrix & ) [inline]

Definition at line 1714 of file newmat.h.

28.55.1.45 **std::ostream&** NEWMAT::operator<<( std::ostream & , const BaseMatrix & )

28.55.1.46 **std::ostream&** NEWMAT::operator<<( std::ostream & , const GeneralMatrix & )

28.55.1.47 **bool** NEWMAT::operator<=( const BaseMatrix & A, const BaseMatrix & ) [inline]

Definition at line 1710 of file newmat.h.

28.55.1.48 **bool** NEWMAT::operator==( const GeneralMatrix & A, const GeneralMatrix & B )

28.55.1.49 **bool** NEWMAT::operator==( const BaseMatrix & A, const BaseMatrix & B )

28.55.1.50 **bool** NEWMAT::operator>( const BaseMatrix & A, const BaseMatrix & ) [inline]

Definition at line 1716 of file newmat.h.

28.55.1.51 **bool** NEWMAT::operator>=( const BaseMatrix & A, const BaseMatrix & ) [inline]

Definition at line 1712 of file newmat.h.

28.55.1.52 void NEWMAT::QRZ ( Matrix & , UpperTriangularMatrix & )

28.55.1.53 void NEWMAT::QRZ ( const Matrix & , Matrix & , Matrix & )

28.55.1.54 void NEWMAT::QRZT ( Matrix & , LowerTriangularMatrix & )

28.55.1.55 void NEWMAT::QRZT ( const Matrix & , Matrix & , Matrix & )

28.55.1.56 void NEWMAT::RealFFT ( const ColumnVector & , ColumnVector & , ColumnVector & )

28.55.1.57 void NEWMAT::RealFFTI ( const ColumnVector & , const ColumnVector & , ColumnVector & )

28.55.1.58 bool NEWMAT::Rectangular ( MatrixType a , MatrixType b , MatrixType c )

28.55.1.59 void NEWMAT::SortAscending ( GeneralMatrix & )

28.55.1.60 void NEWMAT::SortDescending ( GeneralMatrix & )

28.55.1.61 void NEWMAT::SortSV ( DiagonalMatrix & D , Matrix & U , bool ascending = false )

28.55.1.62 void NEWMAT::SortSV ( DiagonalMatrix & D , Matrix & U , Matrix & V , bool ascending = false )

28.55.1.63 SPMatrix NEWMAT::SP ( const BaseMatrix & , const BaseMatrix & )

28.55.1.64 Real NEWMAT::Sum ( const BaseMatrix & B ) [inline]

Definition at line 1751 of file newmat.h.

28.55.1.65 Real NEWMAT::SumAbsoluteValue ( const BaseMatrix & B ) [inline]

Definition at line 1749 of file newmat.h.

28.55.1.66 Real NEWMAT::SumSquare ( const BaseMatrix & B ) [inline]

Definition at line 1746 of file newmat.h.

28.55.1.67 void NEWMAT::SVD ( const Matrix & , DiagonalMatrix & , Matrix & , Matrix & , bool = true , bool = true )

28.55.1.68 void NEWMAT::SVD ( const Matrix & , DiagonalMatrix & )

28.55.1.69 void NEWMAT::SVD ( const Matrix & A , DiagonalMatrix & D , Matrix & U , bool withU = true ) [inline]

Definition at line 41 of file newmatap.h.

28.55.1.70 `Real NEWMAT::Trace ( const BaseMatrix & B ) [inline]`

Definition at line 1748 of file newmat.h.

## 28.56 RBD\_COMMON Namespace Reference

### Classes

- class [Bad\\_alloc](#)
- class [BaseException](#)
- class [Domain\\_error](#)
- class [Invalid\\_argument](#)
- class [Janitor](#)
- class [Length\\_error](#)
- class [Logic\\_error](#)
- class [OneDimSolve](#)
- class [Out\\_of\\_range](#)
- class [Overflow\\_error](#)
- class [R1\\_R1](#)
- class [Range\\_error](#)
- class [Runtime\\_error](#)
- class [SolutionException](#)
- class [Tracer](#)

### Typedefs

- typedef [double](#) [Real](#)
- typedef long [double](#) [long\\_Real](#)
- typedef [BaseException](#) [Exception](#)

### Functions

- void [Terminate](#) ()

#### 28.56.1 Typedef Documentation

28.56.1.1 typedef [BaseException](#) [RBD\\_COMMON::Exception](#)

Definition at line 98 of file myexcept.h.

28.56.1.2 typedef long [double](#) [RBD\\_COMMON::long\\_Real](#)

Definition at line 313 of file include.h.

### 28.56.1.3 typedef double RBD\_COMMON::Real

Definition at line 312 of file include.h.

## 28.56.2 Function Documentation

### 28.56.2.1 void RBD\_COMMON::Terminate ( )

Definition at line 226 of file myexcept.cpp.

## 28.57 RBD\_LIBRARIES Namespace Reference

## 28.58 std Namespace Reference

### Functions

- `template<int Dim>`  
`std::istream & operator>> (std::istream &is, Go::BaryCoordSystem< Dim > &bc)`  
*Read BaryCoordSystem from input stream.*
- `template<int Dim>`  
`std::ostream & operator<< (std::ostream &os, const Go::BaryCoordSystem< Dim > &bc)`  
*Write BaryCoordSystem to output stream.*
- `std::istream & operator>> (std::istream &is, Go::BoundingBox &bbox)`  
*Read BoundingBox from input stream.*
- `std::ostream & operator<< (std::ostream &os, const Go::BoundingBox &bbox)`  
*Write BoundingBox to output stream.*
- `std::istream & operator>> (std::istream &is, Go::BernsteinMulti &m)`  
*Read BernsteinMulti from input stream.*
- `std::ostream & operator<< (std::ostream &os, const Go::BernsteinMulti &m)`  
*Write BernsteinMulti to output stream.*
- `std::istream & operator>> (std::istream &is, Go::BernsteinPoly &p)`  
*Read BernsteinPoly from input stream.*
- `std::ostream & operator<< (std::ostream &os, const Go::BernsteinPoly &p)`  
*Write BernsteinPoly to output stream.*
- `template<class _Tp >`  
`_Tp identity\_element (plus< _Tp >)`
- `template<class _Tp >`  
`_Tp identity\_element (multiplies< _Tp >)`
- `template<class _Tp, class _Integer, class _MonoidOperation >`  
`_Tp \_\_power (_Tp __x, _Integer __n, _MonoidOperation __oper)`
- `template<class _Tp, class _Integer >`  
`_Tp \_\_power (_Tp __x, _Integer __n)`
- `template<class _Tp, class _Integer, class _MonoidOperation >`  
`_Tp power (_Tp __x, _Integer __n, _MonoidOperation __oper)`
- `template<class _Tp, class _Integer >`  
`_Tp power (_Tp __x, _Integer __n)`

### 28.58.1 Detailed Description

We implement two extensions: **identity\_element** and **power**.

### 28.58.2 Function Documentation

28.58.2.1 `template<class _Tp, class _Integer, class _MonoidOperation > _Tp std::__power ( _Tp __x, _Integer __n, _MonoidOperation __oper )`

Definition at line 72 of file BezierTriangle.h.

28.58.2.2 `template<class _Tp, class _Integer > _Tp std::__power ( _Tp __x, _Integer __n ) [inline]`

Definition at line 96 of file BezierTriangle.h.

28.58.2.3 `template<class _Tp > _Tp std::identity_element ( plus<_Tp > ) [inline]`

Definition at line 59 of file BezierTriangle.h.

28.58.2.4 `template<class _Tp > _Tp std::identity_element ( multiplies<_Tp > ) [inline]`

Definition at line 62 of file BezierTriangle.h.

28.58.2.5 `template<int Dim> std::ostream& std::operator<< ( std::ostream & os, const Go::BaryCoordSystem< Dim > & bc ) [inline]`

Write BaryCoordSystem to output stream.

Definition at line 205 of file BaryCoordSystem.h.

28.58.2.6 `std::ostream& std::operator<< ( std::ostream & os, const Go::BoundingBox & bbox ) [inline]`

Write BoundingBox to output stream.

Definition at line 217 of file BoundingBox.h.

28.58.2.7 `std::ostream& std::operator<< ( std::ostream & os, const Go::BernsteinPoly & p ) [inline]`

Write BernsteinPoly to output stream.

Definition at line 315 of file BernsteinPoly.h.

**28.58.2.8** `std::ostream& std::operator<< ( std::ostream & os, const Go::BernsteinMulti & m )` `[inline]`

Write BernsteinMulti to output stream.

Definition at line 415 of file BernsteinMulti.h.

**28.58.2.9** `template<int Dim> std::istream& std::operator>> ( std::istream & is, Go::BaryCoordSystem< Dim > & bc )`  
`[inline]`

Read BaryCoordSystem from input stream.

Definition at line 195 of file BaryCoordSystem.h.

**28.58.2.10** `std::istream& std::operator>> ( std::istream & is, Go::BoundingBox & bbox )` `[inline]`

Read BoundingBox from input stream.

Definition at line 208 of file BoundingBox.h.

**28.58.2.11** `std::istream& std::operator>> ( std::istream & is, Go::BernsteinPoly & p )` `[inline]`

Read BernsteinPoly from input stream.

Definition at line 306 of file BernsteinPoly.h.

**28.58.2.12** `std::istream& std::operator>> ( std::istream & is, Go::BernsteinMulti & m )` `[inline]`

Read BernsteinMulti from input stream.

Definition at line 406 of file BernsteinMulti.h.

**28.58.2.13** `template<class _Tp, class _Integer, class _MonoidOperation > _Tp std::power ( _Tp __x, _Integer __n,`  
`_MonoidOperation __oper )` `[inline]`

Definition at line 105 of file BezierTriangle.h.

**28.58.2.14** `template<class _Tp, class _Integer > _Tp std::power ( _Tp __x, _Integer __n )` `[inline]`

Definition at line 111 of file BezierTriangle.h.

## 28.59 ttl Namespace Reference

Main interface to TTL.



## Functions

### Delaunay Triangulation

- `template<class TraitsType, class DartType, class PointType >`  
`bool insertNode (DartType &dart, PointType &point)`
- `template<class TraitsType, class DartType >`  
`void removeRectangularBoundary (DartType &dart)`
- `template<class TraitsType, class DartType >`  
`void removeNode (DartType &dart)`
- `template<class TraitsType, class DartType >`  
`void removeBoundaryNode (DartType &dart)`
- `template<class TraitsType, class DartType >`  
`void removeInteriorNode (DartType &dart)`
- `template<class TraitsType, class ForwardIterator, class DartType >`  
`void insertNodes (ForwardIterator first, ForwardIterator last, DartType &dart)`

### Topological and Geometric Queries

- `template<class TraitsType, class PointType, class DartType >`  
`bool locateFaceSimplest (const PointType &point, DartType &dart)`
- `template<class TraitsType, class PointType, class DartType >`  
`bool locateTriangle (const PointType &point, DartType &dart)`
- `template<class TraitsType, class PointType, class DartType >`  
`bool inTriangleSimplest (const PointType &point, const DartType &dart)`
- `template<class TraitsType, class PointType, class DartType >`  
`bool inTriangle (const PointType &point, const DartType &dart)`
- `template<class DartType, class DartListType >`  
`void getBoundary (const DartType &dart, DartListType &boundary)`
- `template<class DartType >`  
`bool isBoundaryEdge (const DartType &dart)`
- `template<class DartType >`  
`bool isBoundaryFace (const DartType &dart)`
- `template<class DartType >`  
`bool isBoundaryNode (const DartType &dart)`
- `template<class DartType >`  
`int getDegreeOfNode (const DartType &dart)`
- `template<class DartType, class DartListType >`  
`void get_0_orbit_interior (const DartType &dart, DartListType &orbit)`
- `template<class DartType, class DartListType >`  
`void get_0_orbit_boundary (const DartType &dart, DartListType &orbit)`
- `template<class DartType >`  
`bool same_0_orbit (const DartType &d1, const DartType &d2)`
- `template<class DartType >`  
`bool same_1_orbit (const DartType &d1, const DartType &d2)`
- `template<class DartType >`  
`bool same_2_orbit (const DartType &d1, const DartType &d2)`
- `template<class TraitsType, class DartType >`  
`bool swappableEdge (const DartType &dart, bool allowDegeneracy=false)`
- `template<class DartType >`  
`void positionAtNextBoundaryEdge (DartType &dart)`
- `template<class TraitsType, class DartType >`  
`bool convexBoundary (const DartType &dart)`
- `template<class TopologyElementType, class DartType >`  
`bool isMemberOfFace (const TopologyElementType &topologyElement, const DartType &dart)`
- `template<class TraitsType, class NodeType, class DartType >`  
`bool locateFaceWithNode (const NodeType &node, DartType &dart_iter)`
- `template<class DartType >`  
`void getAdjacentTriangles (const DartType &dart, DartType &t1, DartType &t2, DartType &t3)`
- `template<class DartType >`  
`void getNeighborNodes (const DartType &dart, std::list< DartType > &node_list, bool &boundary)`

- `template<class TraitsType , class DartType >`  
`bool degenerateTriangle (const DartType &dart)`

### Utilities for Delaunay Triangulation

- `template<class TraitsType , class DartType , class DartListType >`  
`void optimizeDelaunay (DartListType &elist)`
- `template<class TraitsType , class DartType , class DartListType >`  
`void optimizeDelaunay (DartListType &elist, const typename DartListType::iterator end)`
- `template<class TraitsType , class DartType >`  
`bool swapTestDelaunay (const DartType &dart, bool cycling_check=false)`
- `template<class TraitsType , class DartType >`  
`void recSwapDelaunay (DartType &diagonal)`
- `template<class TraitsType , class DartType , class ListType >`  
`void swapEdgesAwayFromInteriorNode (DartType &dart, ListType &swapped_edges)`
- `template<class TraitsType , class DartType , class ListType >`  
`void swapEdgesAwayFromBoundaryNode (DartType &dart, ListType &swapped_edges)`
- `template<class TraitsType , class DartType , class DartListType >`  
`void swapEdgeInList (const typename DartListType::iterator &it, DartListType &elist)`

### Constrained (Delaunay) Triangulation

- `template<class TraitsType , class DartType >`  
`DartType insertConstraint (DartType &dstart, DartType &dend, bool optimize_delaunay)`

## 28.59.1 Detailed Description

Main interface to TTL.

This namespace contains the basic generic algorithms for the TTL, the Triangulation Template Library. Examples of functionality are:

- Incremental Delaunay triangulation
- Constrained triangulation
- Insert/remove nodes and constrained edges
- Traversal operations
- Misc. queries for extracting information for visualisation systems etc.

#### General requirements and assumptions:

- *DartType* and *TraitsType* should be implemented in accordance with the description in [Application Programming Interface to TTL \(API\)](#).
- A "**Requires:**" section in the documentation of a function template shows which functionality is required in *TraitsType* to support that specific function. Functionality required in *DartType* is the same (almost) for all function templates; see [Application Programming Interface to TTL \(API\)](#) and the example referred to.
- When a reference to a *dart* object is passed to a function in TTL, it is assumed that it is oriented *counterclockwise* (CCW) in a triangle unless it is explicitly mentioned that it can also be *clockwise* (CW). The same applies for a dart that is passed from a function in TTL to the users *TraitsType* class (or struct).
- When an edge (represented with a dart) is swapped, it is assumed that darts outside the quadrilateral where the edge is a diagonal are not affected by the swap. Thus, [TraitsType::swapEdge](#) must be implemented in accordance with this rule.

## Glossary:

- General terms are explained in [Application Programming Interface to TTL \(API\)](#).
- *CCW* - counterclockwise
- *CW* - clockwise
- *0\_orbit*, *1\_orbit* and *2\_orbit*: A sequence of darts around a node, around an edge and in a triangle respectively; see [ttl::get\\_0\\_orbit\\_interior](#) and [ttl::get\\_0\\_orbit\\_boundary](#)
- *arc* - In a triangulation an arc is equivalent with an edge

## See also

[ttl\\_util](#) and [Application Programming Interface to TTL \(API\)](#)

## Author

Oyvind Hjelle, [oyvindhj@ifi.uio.no](mailto:oyvindhj@ifi.uio.no)

## 28.59.2 Function Documentation

28.59.2.1 `template<class TraitsType , class DartType > bool ttl::convexBoundary ( const DartType & dart )`

Checks if the boundary of a triangulation is convex.

## Parameters

|             |                                                 |
|-------------|-------------------------------------------------|
| <i>dart</i> | A CCW dart at the boundary of the triangulation |
|-------------|-------------------------------------------------|

- [TraitsType::crossProduct2d](#) (const Dart&, const Dart&)

Definition at line 1368 of file `ttl.h`.

28.59.2.2 `template<class TraitsType , class DartType > bool ttl::degenerateTriangle ( const DartType & dart )`

Definition at line 1262 of file `ttl.h`.

28.59.2.3 `template<class DartType , class DartListType > void ttl::get_0_orbit_boundary ( const DartType & dart, DartListType & orbit )`

Gets the 0-orbit around a node at the boundary

## Parameters

|             |                                                                                         |
|-------------|-----------------------------------------------------------------------------------------|
| <i>dart</i> | A dart (CCW or CW) positioned at a <i>boundary node</i> and at a <i>boundary edge</i> . |
|-------------|-----------------------------------------------------------------------------------------|

## Return values

|              |                                                                                                                                                                                                        |
|--------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>orbit</i> | Sequence of darts with one orbit for each arc. All the darts, <i>except the last</i> one, have the same orientation (CCW or CW) as <i>dart</i> , and <i>dart</i> is the first element in the sequence. |
|--------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

- `DartListType::push_back (DartType&)`

## Note

- The last dart in the sequence have opposite orientation compared to the others!

## See also

[ttl::get\\_0\\_orbit\\_interior](#)

Definition at line 1171 of file `ttl.h`.

```
28.59.2.4 template<class DartType , class DartListType > void ttl::get_0_orbit_interior (const DartType & dart, DartListType & orbit)
```

Gets the 0-orbit around an interior node.

## Parameters

|             |                                                           |
|-------------|-----------------------------------------------------------|
| <i>dart</i> | A dart (CCW or CW) positioned at an <i>interior</i> node. |
|-------------|-----------------------------------------------------------|

## Return values

|              |                                                                                                                                                                           |
|--------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>orbit</i> | Sequence of darts with one orbit for each arc. All the darts have the same orientation (CCW or CW) as <i>dart</i> , and <i>dart</i> is the first element in the sequence. |
|--------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

- `DartListType::push_back (DartType&)`

## See also

[ttl::get\\_0\\_orbit\\_boundary](#)

Definition at line 1137 of file `ttl.h`.

```
28.59.2.5 template<class DartType > void ttl::getAdjacentTriangles (const DartType & dart, DartType & t1, DartType & t2, DartType & t3)
```

Definition at line 844 of file `ttl.h`.

28.59.2.6 `template<class DartType , class DartListType > void tti::getBoundary ( const DartType & dart, DartListType & boundary )`

Gets the boundary as sequence of darts, where the edges associated with the darts are boundary edges, given a dart with an associating edge at the boundary of a topology structure. The first dart in the sequence will be the given one, and the others will have the same orientation (CCW or CW) as the first. Assumes that the given dart is at the boundary.

#### Parameters

|                 |                                                                        |
|-----------------|------------------------------------------------------------------------|
| <i>dart</i>     | A dart at the boundary (CCW or CW)                                     |
| <i>boundary</i> | A sequence of darts, where the associated edges are the boundary edges |

- `DartListType::push_back (DartType&)`

Definition at line 889 of file tti.h.

28.59.2.7 `template<class DartType > int tti::getDegreeOfNode ( const DartType & dart )`

Returns the degree of the node associated with *dart*.

#### Definition:

The *degree* (or valency) of a node *V* in a triangulation, is defined as the number of edges incident with *V*, i.e., the number of edges joining *V* with another node in the triangulation.

Definition at line 1012 of file tti.h.

28.59.2.8 `template<class DartType > void tti::getNeighborNodes ( const DartType & dart, std::list< DartType > & node_list, bool & boundary )`

Definition at line 1072 of file tti.h.

28.59.2.9 `template<class TraitsType , class DartType > DartType tti::insertConstraint ( DartType & dstart, DartType & dend, bool optimize_delaunay )`

Inserts a constrained edge between two existing nodes in a triangulation. If the constraint falls on one or more existing nodes and this is detected by the predicate `TraitsType::orient2d`, which should return zero in this case, the constraint is split. Otherwise a degenerate triangle will be made along the constraint.

#### Parameters

|                          |                                                                                                                                                                                                                                  |
|--------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>dstart</i>            | A CCW dart with the start node of the constraint as the source node                                                                                                                                                              |
| <i>dend</i>              | A CCW dart with the end node of the constraint as the source node                                                                                                                                                                |
| <i>optimize_delaunay</i> | If set to <code>true</code> , the resulting triangulation will be a <i>constrained Delaunay triangulation</i> . If set to <code>false</code> , the resulting triangulation will not necessarily be of constrained Delaunay type. |

## Return values

|                 |                                           |
|-----------------|-------------------------------------------|
| <i>DartType</i> | A dart representing the constrained edge. |
|-----------------|-------------------------------------------|

- [TraitsType::orient2d](#) (DartType&, DartType&, PointType&)
- [TraitsType::swapEdge](#) (DartType&)
- [ttl::optimizeDelaunay](#) if *optimize\_delaunay* is set to `true`

## Assumes:

- The constrained edge must be inside the existing triangulation (and it cannot cross the boundary of the triangulation).

Definition at line 556 of file `ttl_constr.h`.

```
28.59.2.10 template<class TraitsType , class DartType , class PointType > bool ttl::insertNode (DartType & dart, PointType & point)
```

Inserts a new node in an existing Delaunay triangulation and swaps edges to obtain a new Delaunay triangulation. This is the basic function for incremental Delaunay triangulation. When starting from a set of points, an initial Delaunay triangulation can be created as two triangles forming a rectangle that contains all the points. After `insertNode` has been called repeatedly with all the points, [ttl::removeRectangularBoundary](#) can be called to remove triangles at the boundary of the triangulation so that the boundary form the convex hull of the points.

Note that this incremental scheme will run much faster if the points have been sorted lexicographically on *x* and *y*.

## Parameters

|              |                                                                                            |
|--------------|--------------------------------------------------------------------------------------------|
| <i>dart</i>  | An arbitrary CCW dart in the tringulation.<br>Output: A CCW dart incident to the new node. |
| <i>point</i> | A point (node) to be inserted in the triangulation.                                        |

## Return values

|             |                                                                                                                                                                                             |
|-------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>bool</i> | <code>true</code> if <i>point</i> was inserted; <code>false</code> if not.<br>If <i>point</i> is outside the triangulation, or the input dart is not valid, <code>false</code> is returned. |
|-------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

- [TraitsType::splitTriangle](#) (DartType&, const PointType&)
- [ttl::locateTriangle](#)
- [ttl::recSwapDelaunay](#)

## Note

- For efficiency reasons *dart* should be close to the insertion *point*.

See also

[ttl::removeRectangularBoundary](#)

Definition at line 281 of file ttl.h.

28.59.2.11 `template<class TraitsType , class ForwardIterator , class DartType > void ttl::insertNodes ( ForwardIterator first, ForwardIterator last, DartType & dart )`

Definition at line 335 of file ttl.h.

28.59.2.12 `template<class TraitsType , class PointType , class DartType > bool ttl::inTriangle ( const PointType & point, const DartType & dart )`

Checks if *point* is inside the triangle associated with *dart*. This function deals with degeneracy to some extent, but round-off errors may still lead to wrong result if the triangle is degenerate.

Parameters

|             |                            |
|-------------|----------------------------|
| <i>dart</i> | A CCW dart in the triangle |
|-------------|----------------------------|

- [TraitsType::crossProduct2d](#) (DartType&, PointType&)
- [TraitsType::scalarProduct2d](#) (DartType&, PointType&)

See also

[ttl::inTriangleSimplest](#)

Definition at line 763 of file ttl.h.

28.59.2.13 `template<class TraitsType , class PointType , class DartType > bool ttl::inTriangleSimplest ( const PointType & point, const DartType & dart )`

Checks if *point* is inside the triangle associated with *dart*. A fast and simple function that does not deal with degeneracy.

Parameters

|             |                            |
|-------------|----------------------------|
| <i>dart</i> | A CCW dart in the triangle |
|-------------|----------------------------|

- [TraitsType::orient2d](#) (DartType&, DartType&, PointType&)

See also

[ttl::inTriangle](#) for a more robust function

Definition at line 719 of file ttl.h.

**28.59.2.14** `template<class DartType > bool ttl::isBoundaryEdge ( const DartType & dart )`

Checks if the edge associated with *dart* is at the boundary of the triangulation.

Implements:

```
DartType dart_iter = dart;
if (dart_iter.alpha2() == dart)
 return true;
else
 return false;
```

Definition at line 935 of file ttl.h.

**28.59.2.15** `template<class DartType > bool ttl::isBoundaryFace ( const DartType & dart )`

Checks if the face associated with *dart* is at the boundary of the triangulation.

Definition at line 950 of file ttl.h.

**28.59.2.16** `template<class DartType > bool ttl::isBoundaryNode ( const DartType & dart )`

Checks if the node associated with *dart* is at the boundary of the triangulation.

Definition at line 979 of file ttl.h.

**28.59.2.17** `template<class TopologyElementType , class DartType > bool ttl::isMemberOfFace ( const TopologyElementType & topologyElement, const DartType & dart )`

Definition at line 530 of file ttl.h.

**28.59.2.18** `template<class TraitsType , class PointType , class DartType > bool ttl::locateFaceSimplest ( const PointType & point, DartType & dart )`

Locates the face containing a given point. It is assumed that the tessellation (e.g. a triangulation) is *regular* in the sense that there are no holes, the boundary is convex and there are no degenerate faces.

Parameters

|              |                                                                                      |
|--------------|--------------------------------------------------------------------------------------|
| <i>point</i> | A point to be located                                                                |
| <i>dart</i>  | An arbitrary CCW dart in the triangulation<br>Output: A CCW dart in the located face |

Return values

|             |                                        |
|-------------|----------------------------------------|
| <i>bool</i> | true if a face is found; false if not. |
|-------------|----------------------------------------|



- [TraitsType::orient2d](#) (DartType&, DartType&, PointType&)

#### Note

- If `false` is returned, *point* may still be inside a face if the tessellation is not *regular* as explained above.

#### See also

[ttl::locateTriangle](#)

Definition at line 600 of file `ttl.h`.

**28.59.2.19** `template<class TraitsType , class NodeType , class DartType > bool ttl::locateFaceWithNode ( const NodeType & node, DartType & dart_iter )`

Definition at line 550 of file `ttl.h`.

**28.59.2.20** `template<class TraitsType , class PointType , class DartType > bool ttl::locateTriangle ( const PointType & point, DartType & dart )`

Locates the triangle containing a given point. It is assumed that the triangulation is *regular* in the sense that there are no holes and the boundary is convex. This function deals with degeneracy to some extent, but round-off errors may still lead to a wrong result if triangles are degenerate.

#### Parameters

|              |                                                                                          |
|--------------|------------------------------------------------------------------------------------------|
| <i>point</i> | A point to be located                                                                    |
| <i>dart</i>  | An arbitrary CCW dart in the triangulation<br>Output: A CCW dart in the located triangle |

#### Return values

|             |                                                                                                                                                                                                                                                             |
|-------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>bool</i> | <code>true</code> if a triangle is found; <code>false</code> if not.<br>If <i>point</i> is outside the triangulation, in which case <code>false</code> is returned, then the edge associated with <i>dart</i> will be at the boundary of the triangulation. |
|-------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

- [ttl::locateFaceSimplest](#)
- [ttl::inTriangle](#)

Definition at line 667 of file `ttl.h`.

**28.59.2.21** `template<class TraitsType , class DartType , class DartListType > void ttl::optimizeDelaunay ( DartListType & elist )`

Optimizes the edges in the given sequence according to the *Delaunay* criterion, i.e., such that the edge will fulfill the *circumcircle* criterion (or equivalently the *MaxMin* angle criterion) with respect to the quadrilaterals where they are diagonals.

## Parameters

|              |                       |
|--------------|-----------------------|
| <i>elist</i> | The sequence of edges |
|--------------|-----------------------|

- [TraitsType::swapEdge](#) (DartType& *dart*)  
**Note:** Must be implemented such that *dart* is delivered back in a position as seen if it was glued to the edge when swapping (rotating) the edge CCW
- [ttl::swapTestDelaunay](#)

Definition at line 1434 of file ttl.h.

```
28.59.2.22 template<class TraitsType , class DartType , class DartListType > void ttl::optimizeDelaunay (DartListType & elist,
 const typename DartListType::iterator end)
```

Definition at line 1441 of file ttl.h.

```
28.59.2.23 template<class DartType > void ttl::positionAtNextBoundaryEdge (DartType & dart)
```

Given a *dart*, CCW or CW, positioned in a 0-orbit at the boundary of a tessellation. Position *dart* at a boundary edge in the same 0-orbit.

If the given *dart* is CCW, *dart* is positioned at the left boundary edge and will be CW.

If the given *dart* is CW, *dart* is positioned at the right boundary edge and will be CCW.

## Note

- The given *dart* must have a source node at the boundary, otherwise an infinit loop occurs.

Definition at line 1343 of file ttl.h.

```
28.59.2.24 template<class TraitsType , class DartType > void ttl::recSwapDelaunay (DartType & diagonal)
```

Recursively swaps edges in the triangulation according to the *Delaunay* criterion.

## Parameters

|                 |                                                                   |
|-----------------|-------------------------------------------------------------------|
| <i>diagonal</i> | A CCW dart representing the edge where the recursion starts from. |
|-----------------|-------------------------------------------------------------------|

- [TraitsType::swapEdge](#) (DartType&)  
**Note:** Must be implemented such that the darts outside the quadrilateral are not affected by the swap.
- Calls itself recursively

Definition at line 1630 of file ttl.h.

28.59.2.25 `template<class TraitsType , class DartType > void ttl::removeBoundaryNode ( DartType & dart )`

Removes the boundary node associated with *dart* and updates the triangulation to be Delaunay.

- [ttl::swapEdgesAwayFromBoundaryNode](#)
- [ttl::optimizeDelaunay](#)
- [TraitsType::removeBoundaryTriangle](#) (Dart&)

Definition at line 415 of file `ttl.h`.

28.59.2.26 `template<class TraitsType , class DartType > void ttl::removeInteriorNode ( DartType & dart )`

Removes the interior node associated with *dart* and updates the triangulation to be Delaunay.

- [ttl::swapEdgesAwayFromInteriorNode](#)
- [ttl::optimizeDelaunay](#)
- [TraitsType::reverse\\_splitTriangle](#) (Dart&)

#### Note

- The node cannot belong to a fixed (constrained) edge that is not swappable. (An endless loop is likely to occur in this case).

Definition at line 479 of file `ttl.h`.

28.59.2.27 `template<class TraitsType , class DartType > void ttl::removeNode ( DartType & dart )`

Removes the node associated with *dart* and updates the triangulation to be Delaunay.

- [ttl::removeBoundaryNode](#) if *dart* represents a node at the boundary
- [ttl::removeInteriorNode](#) if *dart* represents an interior node

#### Note

- The node cannot belong to a fixed (constrained) edge that is not swappable. (An endless loop is likely to occur in this case).

Definition at line 394 of file `ttl.h`.

28.59.2.28 `template<class TraitsType , class DartType > void ttl::removeRectangularBoundary ( DartType & dart )`

Removes the rectangular boundary of a triangulation as a final step of an incremental Delaunay triangulation. The four nodes at the corners will be removed and the resulting triangulation will have a convex boundary and be Delaunay.

## Parameters

|             |                                                                                           |
|-------------|-------------------------------------------------------------------------------------------|
| <i>dart</i> | A CCW dart at the boundary of the triangulation<br>Output: A CCW dart at the new boundary |
|-------------|-------------------------------------------------------------------------------------------|

- [ttl::removeBoundaryNode](#)

## Note

- This function requires that the boundary of the triangulation is a rectangle with four nodes (one in each corner).

Definition at line 365 of file ttl.h.

**28.59.2.29** `template<class DartType > bool ttl::same_0_orbit ( const DartType & d1, const DartType & d2 )`

Checks if the two darts belong to the same 0-orbit, i.e., if they share a node. *d1* and/or *d2* can be CCW or CW.

(This function also examines if the the node associated with *d1* is at the boundary, which slows down the function (slightly). If it is known that the node associated with *d1* is an interior node and a faster version is needed, the user should implement his/her own version.)

Definition at line 1198 of file ttl.h.

**28.59.2.30** `template<class DartType > bool ttl::same_1_orbit ( const DartType & d1, const DartType & d2 )`

Checks if the two darts belong to the same 1-orbit, i.e., if they share an edge. *d1* and/or *d2* can be CCW or CW.

Definition at line 1232 of file ttl.h.

**28.59.2.31** `template<class DartType > bool ttl::same_2_orbit ( const DartType & d1, const DartType & d2 )`

Checks if the two darts belong to the same 2-orbit, i.e., if they lie in the same triangle. *d1* and/or *d2* can be CCW or CW

Definition at line 1248 of file ttl.h.

**28.59.2.32** `template<class TraitsType , class DartType , class DartListType > void ttl::swapEdgeInList ( const typename DartListType::iterator & it, DartListType & elist )`

Swap the the edge associated with iterator *it* and update affected darts in *elist* accordingly. The darts affected by the swap are those in the same quadrilateral. Thus, if one want to preserve one or more of these darts on should keep them in *elist*.

Definition at line 1842 of file ttl.h.

**28.59.2.33** `template<class TraitsType , class DartType , class ListType > void ttl::swapEdgesAwayFromBoundaryNode ( DartType & dart, ListType & swapped_edges )`

Swaps edges away from the (boundary) node associated with *dart* in such a way that when removing the edges that remain incident with the node, the boundary of the triangulation will be convex. This function is used as a first step in [ttl::removeBoundaryNode](#)

## Return values

|             |                                   |
|-------------|-----------------------------------|
| <i>dart</i> | A CCW dart incident with the node |
|-------------|-----------------------------------|

- [TraitsType::swapEdge](#) (DartType& *dart*)

**Note:** Must be implemented such that *dart* is delivered back in a position as seen if it was glued to the edge when swapping (rotating) the edge CCW

## Assumes:

- The node associated with *dart* is at the boundary of the triangulation.

## See also

[ttl::swapEdgesAwayFromInteriorNode](#)

Definition at line 1753 of file ttl.h.

```
28.59.2.34 template<class TraitsType , class DartType , class ListType > void ttl::swapEdgesAwayFromInteriorNode (DartType
 & dart, ListType & swapped_edges)
```

Swaps edges away from the (interior) node associated with *dart* such that that exactly three edges remain incident with the node. This function is used as a first step in [ttl::removeInteriorNode](#)

## Return values

|             |                                   |
|-------------|-----------------------------------|
| <i>dart</i> | A CCW dart incident with the node |
|-------------|-----------------------------------|

## Assumes:

- The node associated with *dart* is interior to the triangulation.
- [TraitsType::swapEdge](#) (DartType& *dart*)
 

**Note:** Must be implemented such that *dart* is delivered back in a position as seen if it was glued to the edge when swapping (rotating) the edge CCW

## Note

- A degenerate triangle may be left at the node.
- The function is not unique as it depends on which dart at the node that is given as input.

## See also

[ttl::swapEdgesAwayFromBoundaryNode](#)

Definition at line 1695 of file ttl.h.

```
28.59.2.35 template<class TraitsType , class DartType > bool ttl::swappableEdge (const DartType & dart, bool
 allowDegeneracy)
```

Checks if the edge associated with *dart* is swappable, i.e., if the edge is a diagonal in a *strictly* convex (or convex) quadrilateral.

## Parameters

|                        |                                                                                                                                                                       |
|------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>allowDegeneracy</i> | If set to true, the function will also return true if the numerical calculations indicate that the quadrilateral is convex only, and not necessarily strictly convex. |
|------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------|

- [TraitsType::crossProduct2d](#) (Dart&, Dart&)

Definition at line 1290 of file ttl.h.

```
28.59.2.36 template<class TraitsType , class DartType > bool ttl::swapTestDelaunay (const DartType & dart, bool
 cycling_check)
```

Checks if the edge associated with *dart* should be swapped according to the *Delaunay* criterion, i.e., the *circumcircle* criterion (or equivalently the *MaxMin* angle criterion).

## Parameters

|                      |                                                                                                                                                                       |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>cycling_check</i> | Must be set to true when used in connection with optimization algorithms, e.g., optimizeDelaunay. This will avoid cycling and infinite loops in nearly neutral cases. |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------|

- [TraitsType::scalarProduct2d](#) (DartType&, DartType&)
- [TraitsType::crossProduct2d](#) (DartType&, DartType&)

Definition at line 1518 of file ttl.h.

## 28.60 ttl\_constr Namespace Reference

Constrained Delaunay triangulation.

### Functions

- template<class DartType >  
[bool isTheConstraint](#) (const DartType &dart, const DartType &dstart, const DartType &dend)
- template<class TraitsType , class DartType >  
[bool crossesConstraint](#) (DartType &dstart, DartType &dend, DartType &d1, DartType &d2)
- template<class TraitsType , class DartType >  
DartType [getAtSmallestAngle](#) (const DartType &dstart, const DartType &dend)
- template<class TraitsType , class DartType , class ListType >  
DartType [findCrossingEdges](#) (const DartType &dstart, const DartType &dend, ListType &elist)
- template<class TraitsType , class DartType >  
void [transformToConstraint](#) (DartType &dstart, DartType &dend, std::list< DartType > &elist)

### 28.60.1 Detailed Description

Constrained Delaunay triangulation.

Basic generic algorithms in TTL for inserting a constrained edge between two existing nodes. See documentation for the namespace `tfl` for general requirements and assumptions.

Author

Oyvind Hjelle, [oyvindhj@ifi.uio.no](mailto:oyvindhj@ifi.uio.no)

### 28.60.2 Function Documentation

**28.60.2.1** `template<class TraitsType , class DartType > bool tfl_constr::crossesConstraint ( DartType & dstart, DartType & dend, DartType & d1, DartType & d2 )`

Definition at line 126 of file `tfl_constr.h`.

**28.60.2.2** `template<class TraitsType , class DartType , class ListType > DartType tfl_constr::findCrossingEdges ( const DartType & dstart, const DartType & dend, ListType & elist )`

Definition at line 276 of file `tfl_constr.h`.

**28.60.2.3** `template<class TraitsType , class DartType > DartType tfl_constr::getAtSmallestAngle ( const DartType & dstart, const DartType & dend )`

Definition at line 164 of file `tfl_constr.h`.

**28.60.2.4** `template<class DartType > bool tfl_constr::isTheConstraint ( const DartType & dart, const DartType & dstart, const DartType & dend )`

Definition at line 95 of file `tfl_constr.h`.

**28.60.2.5** `template<class TraitsType , class DartType > void tfl_constr::transformToConstraint ( DartType & dstart, DartType & dend, std::list< DartType > & elist )`

Definition at line 396 of file `tfl_constr.h`.

## 28.61 `tfl_util` Namespace Reference

Utilities.

## Functions

### Computational geometry

- `template<class real_type >`  
`real_type scalarProduct2d` (`real_type dx1`, `real_type dy1`, `real_type dx2`, `real_type dy2`)
- `template<class real_type >`  
`real_type crossProduct2d` (`real_type dx1`, `real_type dy1`, `real_type dx2`, `real_type dy2`)
- `template<class real_type >`  
`real_type orient2dfast` (`real_type pa[2]`, `real_type pb[2]`, `real_type pc[2]`)

### Utilities involving points

- `template<class PointType >`  
`std::vector< PointType * > * createRandomData` (`int noPoints`, `int seed=1`)

## 28.61.1 Detailed Description

Utilities.

This name space contains utility functions for TTL.

Point and vector algebra such as scalar product and cross product between vectors are implemented here. These functions are required by functions in the `tll` namespace, where they are assumed to be present in the `TTLtraits` class. Thus, the user can call these functions from the traits class. For efficiency reasons, the user may consider implementing these functions in the the API directly on the actual data structure; see [Application Programming Interface to TTL \(API\)](#).

#### Note

- Cross product between vectors in the xy-plane delivers a scalar, which is the z-component of the actual cross product (the x and y components are both zero).

#### See also

[tll](#) and [Application Programming Interface to TTL \(API\)](#)

#### Author

Oyvind Hjelle, [oyvindhj@ifi.uio.no](mailto:oyvindhj@ifi.uio.no)

## 28.61.2 Function Documentation

**28.61.2.1** `template<class PointType > std::vector<PointType*>* tll_util::createRandomData ( int noPoints, int seed = 1 )`

Creates random data on the unit square.

#### Parameters

|                 |                                                 |
|-----------------|-------------------------------------------------|
| <i>noPoints</i> | Number of random points to be generated         |
| <i>seed</i>     | Initial value for pseudorandom number generator |



- Constructor `PointType::PointType(double x, double y)`.  
For example, one can use `pair<double, double>`.

**Note**

- To deduce template argument for `PointType` the function must be called with the syntax: `create↵ RandomData<MyPoint>(…)` where `MyPoint` is the actual point type.

Definition at line 279 of file `ttl_util.h`.

**28.61.2.2** `template<class real_type > real_type ttl_util::crossProduct2d ( real_type dx1, real_type dy1, real_type dx2, real_type dy2 )`

Cross product between two 2D vectors. (The z-component of the actual cross product.)

**Returns:**

`dx1*dy2 - dy1*dx2`

Definition at line 117 of file `ttl_util.h`.

**28.61.2.3** `template<class real_type > real_type ttl_util::orient2dfast ( real_type pa[2], real_type pb[2], real_type pc[2] )`

Returns a positive value if the 2D nodes/points *pa*, *pb*, and *pc* occur in counterclockwise order; a negative value if they occur in clockwise order; and zero if they are collinear.

**Note**

- This is a finite arithmetic fast version. It can be made more robust using exact arithmetic schemes by Jonathan Richard Shewchuk. See <http://www-2.cs.cmu.edu/~quake/robust.html>

Definition at line 133 of file `ttl_util.h`.

**28.61.2.4** `template<class real_type > real_type ttl_util::scalarProduct2d ( real_type dx1, real_type dy1, real_type dx2, real_type dy2 )`

Scalar product between two 2D vectors.

**Returns:**

`dx1*dx2 + dy1*dy2`

Definition at line 103 of file `ttl_util.h`.



# Chapter 29

## Class Documentation

### 29.1 Go::AdaptCurve Class Reference

```
#include <AdaptCurve.h>
```

#### Public Member Functions

- [AdaptCurve](#) ([const EvalCurve](#) \*evalcrv, [double](#) aepsge)
- [AdaptCurve](#) ([const EvalCurve](#) \*evalcrv, [double](#) aepsge, int in, int ik)
- [AdaptCurve](#) ([const EvalCurve](#) \*evalcrv, [double](#) aepsge, int in, int ik, [std::vector](#)< [double](#) > &knots)
- [AdaptCurve](#) ([const EvalCurve](#) \*evalcrv, [double](#) aepsge, [shared\\_ptr](#)< [SplineCurve](#) > curve)
- [~AdaptCurve](#) ()  
*Destructor.*
- void [setSmooth](#) ([double](#) w)
- void [unsetSmooth](#) ()  
*Unset smoothing weight (set it to zero)*
- void [setEndpoints](#) ([const](#) [std::vector](#)< [Point](#) > &start\_point, [const](#) [std::vector](#)< [Point](#) > &end\_point)
- int [approximate](#) (int max\_iter=5)
- [shared\\_ptr](#)< [SplineCurve](#) > [getAdaptCurve](#) ([double](#) &maxdist, [double](#) &avdist, int max\_iter=5)

#### Protected Member Functions

- [AdaptCurve](#) ()  
*Default constructor.*

#### 29.1.1 Detailed Description

This class can generate a B-spline curve that approximates an evaluator based curve within a given accuracy  
Definition at line 68 of file AdaptCurve.h.

#### 29.1.2 Constructor & Destructor Documentation

##### 29.1.2.1 Go::AdaptCurve::AdaptCurve ( [const EvalCurve](#) \* evalcrv, [double](#) aepsge )

Constructor where the evaluator based curve and the approximation tolerance are specified. The generated curve will have a spline basis of order 4 (cubic).

## Parameters

|                |                     |
|----------------|---------------------|
| <i>evalcrv</i> | curve to adapt to   |
| <i>aepsge</i>  | geometric tolerance |

29.1.2.2 `Go::AdaptCurve::AdaptCurve ( const EvalCurve * evalcrv, double aepsge, int in, int ik )`

Constructor where the evaluator based curve and the approximation tolerance are specified as well as the size of the initial spline space

## Parameters

|                |                                                          |
|----------------|----------------------------------------------------------|
| <i>evalcrv</i> | curve to adapt to                                        |
| <i>aepsge</i>  | geometric tolerance                                      |
| <i>in</i>      | the number of control points of the initial spline curve |
| <i>ik</i>      | the order of the initial spline curve (pol. degree + 1)  |

29.1.2.3 `Go::AdaptCurve::AdaptCurve ( const EvalCurve * evalcrv, double aepsge, int in, int ik, std::vector< double > & knots )`

Constructor where the evaluator based curve and the approximation tolerance are specified as well as the initial spline space

## Parameters

|                |                                                          |
|----------------|----------------------------------------------------------|
| <i>evalcrv</i> | curve to adapt to                                        |
| <i>aepsge</i>  | geometric tolerance                                      |
| <i>in</i>      | the number of control points of the initial spline curve |
| <i>ik</i>      | the order of the initial spline curve (pol. degree + 1)  |
| <i>knots</i>   | specifies the knotvector of the resulting spline curve.  |

29.1.2.4 `Go::AdaptCurve::AdaptCurve ( const EvalCurve * evalcrv, double aepsge, shared_ptr< SplineCurve > curve )`

Constructor where the evaluator based curve and the approximation tolerance are specified as well as an initial spline curve

## Parameters

|                |                      |
|----------------|----------------------|
| <i>evalcrv</i> | curve to adapt to    |
| <i>aepsge</i>  | geometric tolerance  |
| <i>crv</i>     | initial spline curve |

## 29.1.2.5 Go::AdaptCurve::~~AdaptCurve ( )

Destructor.

## 29.1.2.6 Go::AdaptCurve::AdaptCurve ( ) [protected]

Default constructor.

## 29.1.3 Member Function Documentation

29.1.3.1 int Go::AdaptCurve::approximate ( int *max\_iter* = 5 )

Perform the approximation

Parameters

|                 |                                                 |
|-----------------|-------------------------------------------------|
| <i>max_iter</i> | specify the maximum number of iterations to use |
|-----------------|-------------------------------------------------|

29.1.3.2 shared\_ptr<SplineCurve> Go::AdaptCurve::getAdaptCurve ( double & *maxdist*, double & *avdist*, int *max\_iter* = 5 )

Fetch the approximating curve

Return values

|                |                                                                                                            |
|----------------|------------------------------------------------------------------------------------------------------------|
| <i>maxdist</i> | report the maximum distance between the generated curve and the data points generated from the input curve |
| <i>avdist</i>  | report the average distance between the generated curve and the datapoints                                 |

Returns

a shared pointer to the generated [SplineCurve](#), approximating the input evaluatorbased curve

29.1.3.3 void Go::AdaptCurve::setEndPoints ( const std::vector< Point > & *start\_point*, const std::vector< Point > & *end\_point* )

The user may decide upon end tangents of approximation curve. If these are not set, the final end tangents result from smoothing equation. The user may specifically decide the start and end points and tangents of the approximation curve (if these are not set, this information will result from the smoothing equation). This function lets the user specify the start and end points and tangents.

Parameters

|                    |                                                                                                                |
|--------------------|----------------------------------------------------------------------------------------------------------------|
| <i>start_point</i> | this vector should contain one or two elements: the start point and optionally the start tangent of the curve. |
| <i>end_point</i>   | this vector should contain one or two elements: the end point and optionally the end tangent of the curve.     |

#### 29.1.3.4 void Go::AdaptCurve::setSmooth ( double $w$ )

Set smoothing weight  $0 < w < 1$

##### Parameters

|     |                      |
|-----|----------------------|
| $w$ | the smoothing weight |
|-----|----------------------|

#### 29.1.3.5 void Go::AdaptCurve::unsetSmooth ( )

Unset smoothing weight (set it to zero)

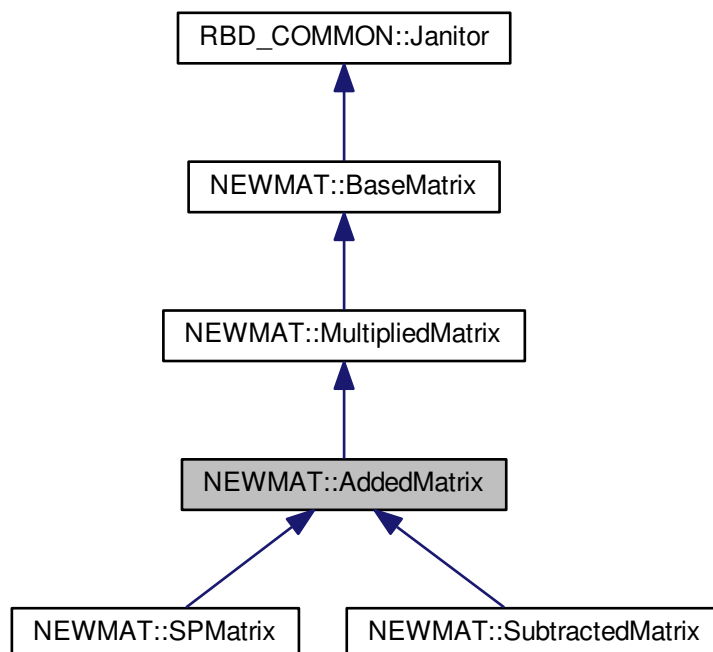
The documentation for this class was generated from the following file:

- [gotools-core/include/GoTools/creators/AdaptCurve.h](#)

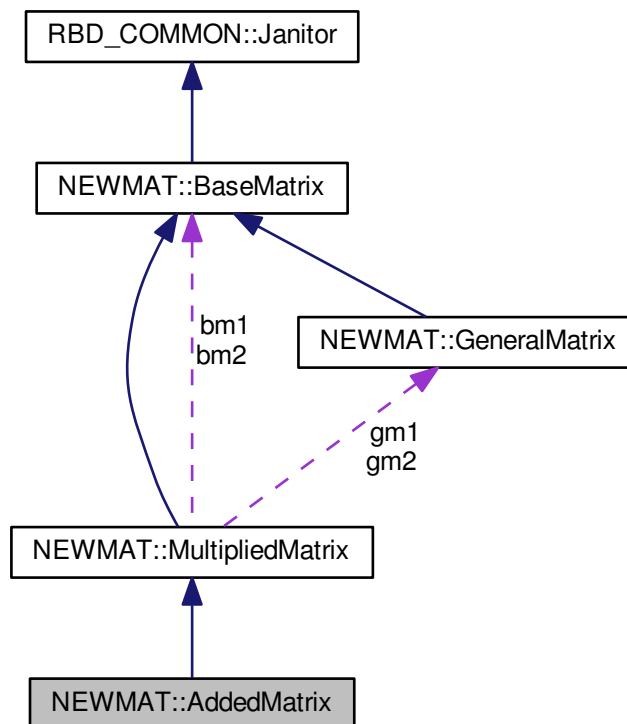
## 29.2 NEWMAT::AddedMatrix Class Reference

```
#include <newmat.h>
```

Inheritance diagram for NEWMAT::AddedMatrix:



Collaboration diagram for NEWMAT::AddedMatrix:



### Public Member Functions

- `~AddedMatrix ()`
- `GeneralMatrix * Evaluate (MatrixType mt=MatrixTypeUnSp)`
- `MatrixBandWidth BandWidth () const`

### Protected Member Functions

- `AddedMatrix (const BaseMatrix *bm1x, const BaseMatrix *bm2x)`

### Friends

- class `BaseMatrix`
- class `GeneralMatrix`
- class `GenericMatrix`

### Additional Inherited Members

#### 29.2.1 Detailed Description

Definition at line 1194 of file `newmat.h`.

## 29.2.2 Constructor & Destructor Documentation

**29.2.2.1** `NEWMAT::AddedMatrix::AddedMatrix ( const BaseMatrix * bm1x, const BaseMatrix * bm2x )` [inline], [protected]

Definition at line 1197 of file newmat.h.

**29.2.2.2** `NEWMAT::AddedMatrix::~~AddedMatrix ( )` [inline]

Definition at line 1204 of file newmat.h.

## 29.2.3 Member Function Documentation

**29.2.3.1** `MatrixBandWidth AddedMatrix::BandWidth ( ) const` [virtual]

Reimplemented from [NEWMAT::MultipliedMatrix](#).

Reimplemented in [NEWMAT::SPMatrix](#).

Definition at line 418 of file newmat4.cpp.

**29.2.3.2** `GeneralMatrix * AddedMatrix::Evaluate ( MatrixType mt = MatrixTypeUnSp )` [virtual]

Reimplemented from [NEWMAT::MultipliedMatrix](#).

Reimplemented in [NEWMAT::SubtractedMatrix](#), and [NEWMAT::SPMatrix](#).

Definition at line 509 of file newmat7.cpp.

## 29.2.4 Friends And Related Function Documentation

**29.2.4.1** `friend class BaseMatrix` [friend]

Definition at line 1200 of file newmat.h.

**29.2.4.2** `friend class GeneralMatrix` [friend]

Definition at line 1201 of file newmat.h.

**29.2.4.3** `friend class GenericMatrix` [friend]

Definition at line 1202 of file newmat.h.

The documentation for this class was generated from the following files:

- [newmat/include/newmat.h](#)
- [newmat/src/newmat4.cpp](#)
- [newmat/src/newmat7.cpp](#)

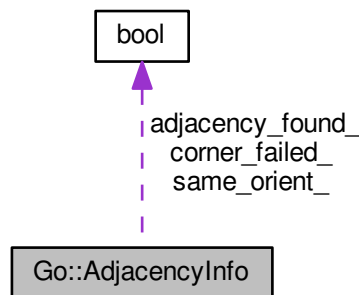


## 29.3 Go::AdjacencyInfo Struct Reference

Struct to store information about adjacency relations between two faces.

```
#include <ftSurface.h>
```

Collaboration diagram for Go::AdjacencyInfo:



### Public Member Functions

- [AdjacencyInfo \(\)](#)

### Public Attributes

- [bool adjacency\\_found\\_](#)  
*True if adjacency was found, false if not.*
- [int bd\\_idx\\_1\\_](#)  
*Boundary index of first surface. 0 = umin, 1 = umax, 2 = vmin, 3 = vmax.*
- [int bd\\_idx\\_2\\_](#)  
*Boundary index of second surface. 0 = umin, 1 = umax, 2 = vmin, 3 = vmax.*
- [bool same\\_orient\\_](#)  
*True if the surfaces are equally oriented along the common boundary.*
- [bool corner\\_failed\\_](#)

#### 29.3.1 Detailed Description

Struct to store information about adjacency relations between two faces.

Definition at line 61 of file `ftSurface.h`.

## 29.3.2 Constructor & Destructor Documentation

### 29.3.2.1 `Go::AdjacencyInfo::AdjacencyInfo ( )` [inline]

Definition at line 76 of file `ftSurface.h`.

## 29.3.3 Member Data Documentation

### 29.3.3.1 `bool Go::AdjacencyInfo::adjacency_found_`

True if adjacency was found, false if not.

Definition at line 65 of file `ftSurface.h`.

### 29.3.3.2 `int Go::AdjacencyInfo::bd_idx_1_`

Boundary index of first surface. 0 = `umin`, 1 = `umax`, 2 = `vmin`, 3 = `vmax`.

Definition at line 67 of file `ftSurface.h`.

### 29.3.3.3 `int Go::AdjacencyInfo::bd_idx_2_`

Boundary index of second surface. 0 = `umin`, 1 = `umax`, 2 = `vmin`, 3 = `vmax`.

Definition at line 69 of file `ftSurface.h`.

### 29.3.3.4 `bool Go::AdjacencyInfo::corner_failed_`

True if adjacency is found, user wanted to test if the surfaces meet in corner-to-corner configuration, and the test failed. Otherwise false.

Definition at line 74 of file `ftSurface.h`.

### 29.3.3.5 `bool Go::AdjacencyInfo::same_orient_`

True if the surfaces are equally oriented along the common boundary.

Definition at line 71 of file `ftSurface.h`.

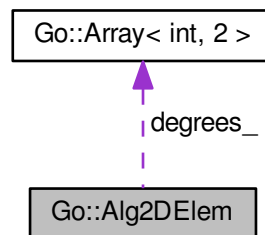
The documentation for this struct was generated from the following file:

- [compositemodel/include/GoTools/compositemodel/ftSurface.h](#)

## 29.4 Go::Alg2DElem Struct Reference

```
#include <AlgObj2DInt.h>
```

Collaboration diagram for Go::Alg2DElem:



### Public Member Functions

- [Alg2DElem](#) (double factor, int degree\_x, int degree\_y)  
*Constructor.*

### Public Attributes

- double factor\_  
*The factor a in  $ax^i y^j$ .*
- [Array](#)< int, 2 > degrees\_  
*The degrees i and j in  $ax^i y^j$ .*

#### 29.4.1 Detailed Description

Struct that represents a monomial in two variables. By this we mean a term in a polynomial of the form  $ax^i y^j$ . This struct is used by [AlgObj2DInt](#).

Definition at line 56 of file AlgObj2DInt.h.

#### 29.4.2 Constructor & Destructor Documentation

29.4.2.1 `Go::Alg2DElem::Alg2DElem ( double factor, int degree_x, int degree_y ) [inline]`

Constructor.

Definition at line 65 of file AlgObj2DInt.h.

### 29.4.3 Member Data Documentation

#### 29.4.3.1 `Array<int, 2> Go::Alg2DElem::degrees_`

The degrees  $i$  and  $j$  in  $ax^i y^j$ .

Definition at line 62 of file AlgObj2DInt.h.

#### 29.4.3.2 `double Go::Alg2DElem::factor_`

The factor  $a$  in  $ax^i y^j$ .

Definition at line 59 of file AlgObj2DInt.h.

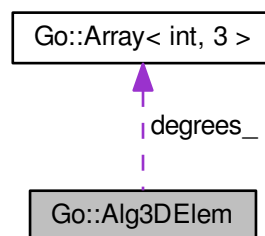
The documentation for this struct was generated from the following file:

- [intersections/include/GoTools/intersections/AlgObj2DInt.h](#)

## 29.5 `Go::Alg3DElem` Struct Reference

```
#include <AlgObj3DInt.h>
```

Collaboration diagram for `Go::Alg3DElem`:



### Public Member Functions

- [Alg3DElem](#) (`double` factor, `int` degree\_x, `int` degree\_y, `int` degree\_z)  
*Constructor.*

### Public Attributes

- `double` [factor\\_](#)  
*The factor  $a$  in  $ax^i y^j x^k$ .*
- `Array< int, 3 >` [degrees\\_](#)  
*The degrees  $i, j$  and  $k$  in  $ax^i y^j x^k$ .*

### 29.5.1 Detailed Description

Struct that represents a monomial in three variables. By this we mean a term in a polynomial of the form  $ax^i y^j x^k$ . This struct is used by [AlgObj3DInt](#).

Definition at line 58 of file AlgObj3DInt.h.

### 29.5.2 Constructor & Destructor Documentation

29.5.2.1 `Go::Alg3DElem::Alg3DElem ( double factor, int degree_x, int degree_y, int degree_z )` `[inline]`

Constructor.

Definition at line 67 of file AlgObj3DInt.h.

### 29.5.3 Member Data Documentation

29.5.3.1 `Array<int, 3> Go::Alg3DElem::degrees_`

The degrees  $i$ ,  $j$  and  $k$  in  $ax^i y^j x^k$ .

Definition at line 64 of file AlgObj3DInt.h.

29.5.3.2 `double Go::Alg3DElem::factor_`

The factor  $a$  in  $ax^i y^j x^k$ .

Definition at line 61 of file AlgObj3DInt.h.

The documentation for this struct was generated from the following file:

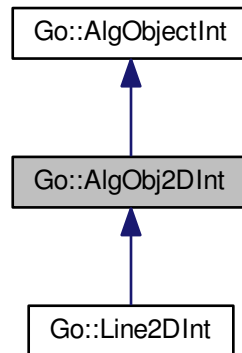
- [intersections/include/GoTools/intersections/AlgObj3DInt.h](#)

## 29.6 Go::AlgObj2DInt Class Reference

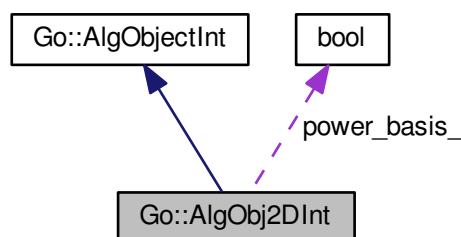
Class for 2-dimensional algebraic intersection objects.

```
#include <AlgObj2DInt.h>
```

Inheritance diagram for Go::AlgObj2DInt:



Collaboration diagram for Go::AlgObj2DInt:



### Public Member Functions

- [AlgObj2DInt](#) (int degree)
- [AlgObj2DInt](#) (const std::vector< [Alg2DElem](#) > &terms)
- virtual [~AlgObj2DInt](#) ()  
*Destructor.*
- int numTerms ()
- [Alg2DElem term](#) (int index)

## Protected Attributes

- int [degree\\_](#)
- std::vector< [Alg2DElem](#) > [terms\\_](#)
- bool [power\\_basis\\_](#)

### 29.6.1 Detailed Description

Class for 2-dimensional algebraic intersection objects.

Definition at line 73 of file AlgObj2DInt.h.

### 29.6.2 Constructor & Destructor Documentation

#### 29.6.2.1 Go::AlgObj2DInt::AlgObj2DInt ( int *degree* )

Constructor.

Parameters

|               |                                                                                              |
|---------------|----------------------------------------------------------------------------------------------|
| <i>degree</i> | the total degree of the algebraic expression, i.e. the maximum sum of exponents of a factor. |
|---------------|----------------------------------------------------------------------------------------------|

#### 29.6.2.2 Go::AlgObj2DInt::AlgObj2DInt ( const std::vector< Alg2DElem > & *terms* )

Constructor.

Parameters

|              |                                                                                    |
|--------------|------------------------------------------------------------------------------------|
| <i>terms</i> | the terms in the algebraic expression, i.e. elements on the form $a_{ij}x^i y^j$ . |
|--------------|------------------------------------------------------------------------------------|

#### 29.6.2.3 virtual Go::AlgObj2DInt::~~AlgObj2DInt ( ) [virtual]

Destructor.

### 29.6.3 Member Function Documentation

#### 29.6.3.1 int Go::AlgObj2DInt::numTerms ( ) [inline]

Get the number of terms in the algebraic object.

Returns

The number of terms in the object.

Definition at line 90 of file AlgObj2DInt.h.

### 29.6.3.2 Alg2DElem Go::AlgObj2DInt::term ( int *index* )

Get a term from the algebraic object.

#### Returns

the term with index *index*

## 29.6.4 Member Data Documentation

### 29.6.4.1 int Go::AlgObj2DInt::degree\_ [protected]

Definition at line 99 of file AlgObj2DInt.h.

### 29.6.4.2 bool Go::AlgObj2DInt::power\_basis\_ [protected]

Definition at line 101 of file AlgObj2DInt.h.

### 29.6.4.3 std::vector<Alg2DElem> Go::AlgObj2DInt::terms\_ [protected]

Definition at line 100 of file AlgObj2DInt.h.

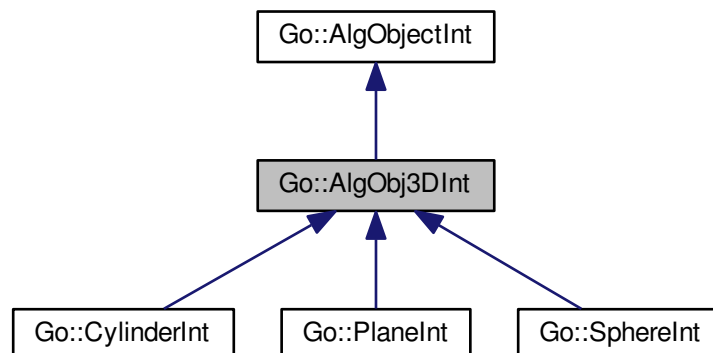
The documentation for this class was generated from the following file:

- [intersections/include/GoTools/intersections/AlgObj2DInt.h](#)

## 29.7 Go::AlgObj3DInt Class Reference

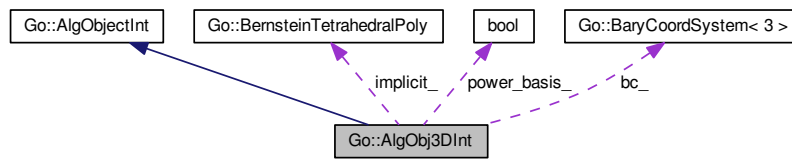
```
#include <AlgObj3DInt.h>
```

Inheritance diagram for Go::AlgObj3DInt:





Collaboration diagram for Go::AlgObj3DInt:



## Public Member Functions

- [AlgObj3DInt](#) (int *degree*)
- [AlgObj3DInt](#) (const std::vector< [Alg3DElem](#) > &terms)
- [AlgObj3DInt](#) (const [BernsteinTetrahedralPoly](#) &implicit, const [BaryCoordSystem3D](#) &bc)
- virtual [~AlgObj3DInt](#) ()
  - Destructor.*
- int [numTerms](#) ()
- int [degree](#) ()
- [Alg3DElem](#) [term](#) (int index)
- void [getImplicit](#) ([BernsteinTetrahedralPoly](#) &impl, [BaryCoordSystem3D](#) &bc)
- bool [usingPowerBasis](#) ()

## Protected Attributes

- int [degree\\_](#)
- std::vector< [Alg3DElem](#) > [terms\\_](#)
- bool [power\\_basis\\_](#)
- [BernsteinTetrahedralPoly](#) [implicit\\_](#)
- [BaryCoordSystem3D](#) [bc\\_](#)

### 29.7.1 Detailed Description

Class for 3-dimensional algebraic intersection objects. Supports two different representations: Polynomials on power basis, or Bernstein polynomials on a tetrahedron.

Definition at line 78 of file [AlgObj3DInt.h](#).

### 29.7.2 Constructor & Destructor Documentation

#### 29.7.2.1 Go::AlgObj3DInt::AlgObj3DInt ( int *degree* )

Constructor.

#### Parameters

|               |                                                                                            |
|---------------|--------------------------------------------------------------------------------------------|
| <i>degree</i> | the total degree of the algebraic expression, i.e. the maximum sum of exponents of a term. |
|---------------|--------------------------------------------------------------------------------------------|

29.7.2.2 `Go::AlgObj3DInt::AlgObj3DInt ( const std::vector< Alg3DElem > & terms )`

Constructor.

Parameters

|              |                                                                                         |
|--------------|-----------------------------------------------------------------------------------------|
| <i>terms</i> | the terms in the algebraic expression, i.e. elements on the form $a_{ijk}x^i y^j z^k$ . |
|--------------|-----------------------------------------------------------------------------------------|

29.7.2.3 `Go::AlgObj3DInt::AlgObj3DInt ( const BernsteinTetrahedralPoly & implicit, const BaryCoordSystem3D & bc )`

Constructor. We define the algebraic expression using another representation than the standard power basis formulation. This will (typically) be the result from an approximative implicitization of a spline surface.

Parameters

|                 |                                                           |
|-----------------|-----------------------------------------------------------|
| <i>implicit</i> | the implicit object.                                      |
| <i>bc</i>       | the barycentric coordinate system for the representation. |

29.7.2.4 `virtual Go::AlgObj3DInt::~~AlgObj3DInt ( ) [virtual]`

Destructor.

### 29.7.3 Member Function Documentation

29.7.3.1 `int Go::AlgObj3DInt::degree ( ) [inline]`

Get the degree of the algebraic object

Returns

the degree

Definition at line 111 of file AlgObj3DInt.h.

29.7.3.2 `void Go::AlgObj3DInt::getImplicit ( BernsteinTetrahedralPoly & impl, BaryCoordSystem3D & bc ) [inline]`

Get the implicit representation of the object.

Parameters

|             |                                      |
|-------------|--------------------------------------|
| <i>impl</i> | the implicit representation.         |
| <i>bc</i>   | the corresponding coordinate system. |

Definition at line 122 of file AlgObj3DInt.h.

#### 29.7.3.3 `int Go::AlgObj3DInt::numTerms ( ) [inline]`

Get the number of terms in the algebraic object.

##### Returns

The number of terms in the object.

Definition at line 106 of file AlgObj3DInt.h.

#### 29.7.3.4 `Alg3DElem Go::AlgObj3DInt::term ( int index )`

Get the corresponding term from the algebraic object.

##### Parameters

|              |                                                          |
|--------------|----------------------------------------------------------|
| <i>index</i> | the index of the term in question. Indexing starts at 0. |
|--------------|----------------------------------------------------------|

#### 29.7.3.5 `bool Go::AlgObj3DInt::usingPowerBasis ( ) [inline]`

Verify whether we are using a standard power basis representation for the object.

##### Returns

True if we are using a standard power basis representation. For typical algebraic objects like spheres this will be the case, but not for approximative implicitizations of spline surfaces.

Definition at line 134 of file AlgObj3DInt.h.

### 29.7.4 Member Data Documentation

#### 29.7.4.1 `BaryCoordSystem3D Go::AlgObj3DInt::bc_ [protected]`

Definition at line 156 of file AlgObj3DInt.h.

#### 29.7.4.2 `int Go::AlgObj3DInt::degree_ [protected]`

Definition at line 139 of file AlgObj3DInt.h.

#### 29.7.4.3 `BernsteinTetrahedralPoly Go::AlgObj3DInt::implicit_ [protected]`

Definition at line 153 of file AlgObj3DInt.h.

#### 29.7.4.4 `bool Go::AlgObj3DInt::power_basis_` [protected]

Definition at line 143 of file AlgObj3DInt.h.

#### 29.7.4.5 `std::vector<Alg3DElem> Go::AlgObj3DInt::terms_` [protected]

Definition at line 142 of file AlgObj3DInt.h.

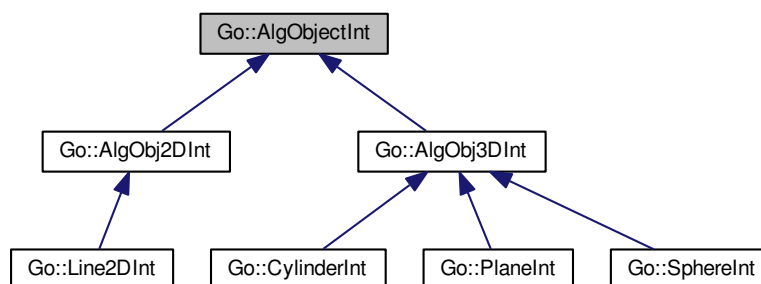
The documentation for this class was generated from the following file:

- [intersections/include/GoTools/intersections/AlgObj3DInt.h](#)

## 29.8 Go::AlgObjectInt Class Reference

```
#include <AlgObjectInt.h>
```

Inheritance diagram for Go::AlgObjectInt:



### Public Member Functions

- [AlgObjectInt](#) ()  
*Constructor.*
- virtual [~AlgObjectInt](#) ()  
*Destructor.*

#### 29.8.1 Detailed Description

This class is a purely abstract base class, providing an interface to the algebraic intersection objects.

Definition at line 50 of file AlgObjectInt.h.

## 29.8.2 Constructor & Destructor Documentation

### 29.8.2.1 Go::AlgObjectInt::AlgObjectInt ( ) [inline]

Constructor.

Definition at line 54 of file AlgObjectInt.h.

### 29.8.2.2 virtual Go::AlgObjectInt::~~AlgObjectInt ( ) [inline],[virtual]

Destructor.

Definition at line 57 of file AlgObjectInt.h.

The documentation for this class was generated from the following file:

- intersections/include/GoTools/intersections/[AlgObjectInt.h](#)

## 29.9 Go::ApproxCrvToSeqs Class Reference

```
#include <ApproxCrvToSeqs.h>
```

### Public Member Functions

- [ApproxCrvToSeqs](#) (const std::vector< std::vector< double > > &points, const std::vector< std::vector< double > > &parvals, int dim, double aepsge)
- [ApproxCrvToSeqs](#) (const std::vector< std::vector< double > > &points, const std::vector< std::vector< double > > &parvals, int dim, double aepsge, int in, int ik)
- [ApproxCrvToSeqs](#) (const std::vector< std::vector< double > > &points, const std::vector< std::vector< double > > &parvals, int dim, double aepsge, int in, int ik, std::vector< double > &knots)
- [~ApproxCrvToSeqs](#) ()  
*Destructor.*
- void [setSmooth](#) (double w)
- void [unsetSmooth](#) ()  
*Unset smoothing weight (set it to zero)*
- void [setEndPoints](#) (const std::vector< std::vector< Point > > &start\_point, const std::vector< std::vector< Point > > &end\_point)
- void [setC1Approx](#) (double fac)  
*Approximate C1 continuity with a given impartance  $0 \leq \text{fac}$ .*
- std::vector< shared\_ptr< [SplineCurve](#) > > [getApproxCurves](#) (double &maxdist, double &avdist, int max\_iter=5)

### Protected Member Functions

- [ApproxCrvToSeqs](#) ()  
*Default constructor.*

### 29.9.1 Detailed Description

This class can generate B-spline curves that approximates sequences of points for a given accuracy.

Definition at line 67 of file `ApproxCrvToSeqs.h`.

### 29.9.2 Constructor & Destructor Documentation

**29.9.2.1** `Go::ApproxCrvToSeqs::ApproxCrvToSeqs ( const std::vector< std::vector< double > > & points, const std::vector< std::vector< double > > & parvals, int dim, double aepsge )`

Constructor where the user specifies a set of parameterized points and a tolerance to be used. The generated curve will have a spline basis of order 4 (cubic), and a number of control points equal to one-sixth of the number of input points. The knotvector of the curve's basis will be set to uniform.

#### Parameters

|                |                                                                                                                                                                                                                            |
|----------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>points</i>  | vector containing the sequences points to approximate. These are stored consecutively in "xyzxyzxyz-fashion". The dimension of all points is expected to be the same, but the number of points in the sequences may differ |
| <i>parvals</i> | the parameter values associated with the points to approximate. The parameter values must correspond for all sequences                                                                                                     |
| <i>dim</i>     | the spatial dimension of the points (usually 3)                                                                                                                                                                            |
| <i>aepsge</i>  | the geometric tolerance to work with                                                                                                                                                                                       |

**29.9.2.2** `Go::ApproxCrvToSeqs::ApproxCrvToSeqs ( const std::vector< std::vector< double > > & points, const std::vector< std::vector< double > > & parvals, int dim, double aepsge, int in, int ik )`

Constructor where the user specifies a set of parameterized points and a tolerance to be used, as well as the spline order and number of control points. The knotvector of the curve's basis will be set to uniform.

#### Parameters

|                |                                                                                                                                                                                                                            |
|----------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>points</i>  | vector containing the sequences points to approximate. These are stored consecutively in "xyzxyzxyz-fashion". The dimension of all points is expected to be the same, but the number of points in the sequences may differ |
| <i>parvals</i> | the parameter values associated with the points to approximate. The parameter values must correspond for all sequences                                                                                                     |
| <i>dim</i>     | the spatial dimension of the points (usually 3)                                                                                                                                                                            |
| <i>aepsge</i>  | the geometric tolerance to work with                                                                                                                                                                                       |
| <i>in</i>      | the number of control points of the resulting spline curve                                                                                                                                                                 |
| <i>ik</i>      | the order of the resulting spline curve (pol. degree + 1)                                                                                                                                                                  |

29.9.2.3 `Go::ApproxCrvToSeqs::ApproxCrvToSeqs ( const std::vector< std::vector< double > > & points, const std::vector< std::vector< double > > & parvals, int dim, double aepsge, int in, int ik, std::vector< double > & knots )`

Constructor where the user specifies a set of parameterized points, a tolerance and the spline space to use for constructing the curve. Constructor where the user specifies a set of parameterized points and a tolerance to be used, as well as the spline order and number of control points. The knotvector of the curve's basis will be set to uniform.

#### Parameters

|                |                                                                                                                                                                                                                            |
|----------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>points</i>  | vector containing the sequences points to approximate. These are stored consecutively in "xyzxyzxyz-fashion". The dimension of all points is expected to be the same, but the number of points in the sequences may differ |
| <i>parvals</i> | the parameter values associated with the points to approximate. The parameter values must correspond for all sequences                                                                                                     |
| <i>dim</i>     | the spatial dimension of the points (usually 3)                                                                                                                                                                            |
| <i>aepsge</i>  | the geometric tolerance to work with                                                                                                                                                                                       |
| <i>in</i>      | the number of control points of the resulting spline curve                                                                                                                                                                 |
| <i>ik</i>      | the order of the resulting spline curve (pol. degree + 1)                                                                                                                                                                  |
| <i>knots</i>   | specifies the knotvector of the resulting spline curve.                                                                                                                                                                    |

29.9.2.4 `Go::ApproxCrvToSeqs::~~ApproxCrvToSeqs ( )`

Destructor.

29.9.2.5 `Go::ApproxCrvToSeqs::ApproxCrvToSeqs ( )` [protected]

Default constructor.

### 29.9.3 Member Function Documentation

29.9.3.1 `std::vector<shared_ptr<SplineCurve> > Go::ApproxCrvToSeqs::getApproxCurves ( double & maxdist, double & avdist, int max_iter = 5 )`

When everything else is set, this function can be used to fetch the approximating curves

#### Return values

|                |                                                                              |
|----------------|------------------------------------------------------------------------------|
| <i>maxdist</i> | report the maximum distance between the generated curves and the data points |
| <i>avdist</i>  | report the average distance between the generated curves and the datapoints  |

#### Parameters

|                 |                                                 |
|-----------------|-------------------------------------------------|
| <i>max_iter</i> | specify the maximum number of iterations to use |
|-----------------|-------------------------------------------------|

**Returns**

a shared pointer to the generated [SplineCurve](#), approximating the points as specified.

**29.9.3.2 void Go::ApproxCrvToSeqs::setC1Approx ( double fac )**

Approximate C1 continuity with a given impartance  $0 \leq \text{fac}$ .

**29.9.3.3 void Go::ApproxCrvToSeqs::setEndPoints ( const std::vector< std::vector< Point > > & start\_point, const std::vector< std::vector< Point > > & end\_point )**

The user may decide upon end tangents of approximation curve. If these are not set, the final end tangents result from smoothing equation. The user may specifically decide the start and end points and tangents of the approximation curve (if these are not set, this information will result from the smoothing equation). This function lets the user specify the start and end points and tangents.

**Parameters**

|                    |                                                                                                                               |
|--------------------|-------------------------------------------------------------------------------------------------------------------------------|
| <i>start_point</i> | for each curve, this vector should contain up to two elements: the start point and optionally the start tangent of the curve. |
| <i>end_point</i>   | for each curve, this vector should contain up to two elements: the end point and optionally the end tangent of the curve.     |

**29.9.3.4 void Go::ApproxCrvToSeqs::setSmooth ( double w )**

Set smoothing weight  $0 < w < 1$

**Parameters**

|          |                      |
|----------|----------------------|
| <i>w</i> | the smoothing weight |
|----------|----------------------|

**29.9.3.5 void Go::ApproxCrvToSeqs::unsetSmooth ( )**

Unset smoothing weight (set it to zero)

The documentation for this class was generated from the following file:

- [gotools-core/include/GoTools/creators/ApproxCrvToSeqs.h](#)

**29.10 Go::ApproxCurve Class Reference**

```
#include <ApproxCurve.h>
```



## Public Member Functions

- [ApproxCurve](#) (`const std::vector< double > &points`, `const std::vector< double > &parvals`, `int dim`, `double aepsge`)
- [ApproxCurve](#) (`const std::vector< double > &points`, `const std::vector< double > &parvals`, `int dim`, `double aepsge`, `int in`, `int ik`)
- [ApproxCurve](#) (`const std::vector< double > &points`, `const std::vector< double > &parvals`, `int dim`, `double aepsge`, `int in`, `int ik`, `const std::vector< double > &knots`)
- [~ApproxCurve](#) ()  
*Destructor.*
- void [setSmooth](#) (`double w`)
- void [unsetSmooth](#) ()  
*Unset smoothing weight (set it to zero)*
- void [setEndPoints](#) (`const std::vector< Point > &start_point`, `const std::vector< Point > &end_point`)
- void [setC1Approx](#) (`double fac`)  
*Approximate C1 continuity with a given impartance  $0 \leq fac < 1$ .*
- `shared_ptr< SplineCurve >` [getApproxCurve](#) (`double &maxdist`, `double &avdist`, `int max_iter=5`)

## Protected Member Functions

- [ApproxCurve](#) ()  
*Default constructor.*

### 29.10.1 Detailed Description

This class can generate a B-spline curve that approximates a set of points for a given accuracy.

Definition at line 67 of file `ApproxCurve.h`.

### 29.10.2 Constructor & Destructor Documentation

- 29.10.2.1 `Go::ApproxCurve::ApproxCurve ( const std::vector< double > & points, const std::vector< double > & parvals, int dim, double aepsge )`

Constructor where the user specifies a set of parameterized points and a tolerance to be used. The generated curve will have a spline basis of order 4 (cubic), and a number of control points equal to one-sixth of the number of input points. The knotvector of the curve's basis will be set to uniform.

#### Parameters

|                |                                                                                                     |
|----------------|-----------------------------------------------------------------------------------------------------|
| <i>points</i>  | vector containing the points to approximate. These are stored consecutively in "xyzxyzxyz-fashion". |
| <i>parvals</i> | the parameter values associated with the points to approximate                                      |
| <i>dim</i>     | the spatial dimension of the points (usually 3)                                                     |
| <i>aepsge</i>  | the geometric tolerance to work with                                                                |

**29.10.2.2** `Go::ApproxCurve::ApproxCurve ( const std::vector< double > & points, const std::vector< double > & parvals, int dim, double aepsge, int in, int ik )`

Constructor where the user specifies a set of parameterized points and a tolerance to be used, as well as the spline order and number of control points. The knotvector of the curve's basis will be set to uniform.

#### Parameters

|                |                                                                                                     |
|----------------|-----------------------------------------------------------------------------------------------------|
| <i>points</i>  | vector containing the points to approximate. These are stored consecutively in "xyzxyzxyz-fashion". |
| <i>parvals</i> | the parameter values associated with the points to approximate                                      |
| <i>dim</i>     | the spatial dimension of the points (usually 3)                                                     |
| <i>aepsge</i>  | the geometric tolerance to work with                                                                |
| <i>in</i>      | the number of control points of the resulting spline curve                                          |
| <i>ik</i>      | the order of the resulting spline curve (pol. degree + 1)                                           |

**29.10.2.3** `Go::ApproxCurve::ApproxCurve ( const std::vector< double > & points, const std::vector< double > & parvals, int dim, double aepsge, int in, int ik, const std::vector< double > & knots )`

Constructor where the user specifies a set of parameterized points, a tolerance and the spline space to use for constructing the curve.

#### Parameters

|                |                                                                                                     |
|----------------|-----------------------------------------------------------------------------------------------------|
| <i>points</i>  | vector containing the points to approximate. These are stored consecutively in "xyzxyzxyz-fashion". |
| <i>parvals</i> | the parameter values associated with the points to approximate                                      |
| <i>dim</i>     | the spatial dimension of the points (usually 3)                                                     |
| <i>aepsge</i>  | the geometric tolerance to work with                                                                |
| <i>in</i>      | the number of control points of the resulting spline curve                                          |
| <i>ik</i>      | the order of the resulting spline curve (pol. degree + 1)                                           |
| <i>knots</i>   | specifies the knotvector of the resulting spline curve.                                             |

**29.10.2.4** `Go::ApproxCurve::~~ApproxCurve ( )`

Destructor.

**29.10.2.5** `Go::ApproxCurve::ApproxCurve ( ) [protected]`

Default constructor.

### 29.10.3 Member Function Documentation

**29.10.3.1** `shared_ptr<SplineCurve> Go::ApproxCurve::getApproxCurve ( double & maxdist, double & avdist, int max_iter = 5 )`

When everything else is set, this function can be used to fetch the approximating curve

## Return values

|                |                                                                             |
|----------------|-----------------------------------------------------------------------------|
| <i>maxdist</i> | report the maximum distance between the generated curve and the data points |
| <i>avdist</i>  | report the average distance between the generated curve and the datapoints  |

## Parameters

|                 |                                                 |
|-----------------|-------------------------------------------------|
| <i>max_iter</i> | specify the maximum number of iterations to use |
|-----------------|-------------------------------------------------|

## Returns

a shared pointer to the generated [SplineCurve](#), approximating the points as specified.

## 29.10.3.2 void Go::ApproxCurve::setC1Approx ( double fac )

Approximate C1 continuity with a given importance  $0 \leq \text{fac} < 1$ .

## 29.10.3.3 void Go::ApproxCurve::setEndPoints ( const std::vector&lt; Point &gt; &amp; start\_point, const std::vector&lt; Point &gt; &amp; end\_point )

The user may decide upon end tangents of approximation curve. If these are not set, the final end tangents result from smoothing equation. The user may specifically decide the start and end points and tangents of the approximation curve (if these are not set, this information will result from the smoothing equation). This function lets the user specify the start and end points and tangents.

## Parameters

|                    |                                                                                                                |
|--------------------|----------------------------------------------------------------------------------------------------------------|
| <i>start_point</i> | this vector should contain one or two elements: the start point and optionally the start tangent of the curve. |
| <i>end_point</i>   | this vector should contain one or two elements: the end point and optionally the end tangent of the curve.     |

## 29.10.3.4 void Go::ApproxCurve::setSmooth ( double w )

Set smoothing weight  $0 < w < 1$

## Parameters

|          |                      |
|----------|----------------------|
| <i>w</i> | the smoothing weight |
|----------|----------------------|

## 29.10.3.5 void Go::ApproxCurve::unsetSmooth ( )

Unset smoothing weight (set it to zero)

The documentation for this class was generated from the following file:

- [gotools-core/include/GoTools/creators/ApproxCurve.h](#)

## 29.11 Go::ApproxSurf Class Reference

```
#include <ApproxSurf.h>
```

### Public Member Functions

- [ApproxSurf](#) (std::vector< shared\_ptr< [SplineCurve](#) > > &crvs, const std::vector< double > &points, const std::vector< double > &parvals, double domain[], int dim, double aepsge, int constdir=0, bool repar=true)
- [ApproxSurf](#) (shared\_ptr< [SplineSurface](#) > &srf, const std::vector< double > &points, const std::vector< double > &parvals, int dim, double aepsge, int constdir=0, bool approx\_orig=false, bool close\_belt=false, int nmb\_stabil=0, bool repar=true)
- [~ApproxSurf](#) ()
  - Destructor.*
- void [setSmoothingWeight](#) (double smooth)
- void [setFixBoundary](#) (bool fix\_boundary)
- void [edgeFix](#) (int edge\_fix[])
- void [setNormalConditions](#) (const std::vector< double > &points, const std::vector< double > &parvals, int nmb\_stabil=0)
- void [setC1Approx](#) (double fac1, double fac2)
  - Approximate C1 continuity with a given impartance  $0 \leq \text{fac} < 1$ .*
- shared\_ptr< [SplineSurface](#) > [getApproxSurf](#) (double &maxdist, double &avdist, int &nmb\_out\_eps, int max←\_iter=4, int keep\_init=0)
  - Fetch the approximating surface.*
- int [reParam](#) ()
- bool [getDoRefine](#) ()
- void [setDoRefine](#) (bool refine)

### Protected Member Functions

- [ApproxSurf](#) ()
  - Default constructor.*

#### 29.11.1 Detailed Description

This class can generate a B-spline surface that approximates a set of points for a given accuracy.

Definition at line 71 of file ApproxSurf.h.

#### 29.11.2 Constructor & Destructor Documentation

- 29.11.2.1 [Go::ApproxSurf::ApproxSurf](#) ( std::vector< shared\_ptr< [SplineCurve](#) > > &crvs, const std::vector< double > & points, const std::vector< double > & parvals, double domain[], int dim, double aepsge, int constdir = 0, bool repar = true )

Constructor where the user specifies the boundary curves of the surface to generate, a parameter domain for the surface, the points to approximate and their parameter values, as well as the geometric tolerance. The two spline bases of the generated tensor product spline surface will be determined by unifying bases of opposing boundary curves.

## Parameters

|                 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|-----------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>crvs</i>     | the boundary curves of the surface to be generated. This vector should contain exactly <i>four</i> curves, whose endpoints are connected so that they form a loop.                                                                                                                                                                                                                                                                                                                                                                                                           |
| <i>points</i>   | vector containing the coordinates of the points that this surface should interpolate. They are stored in "xyzxyz...-fashion".                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| <i>parvals</i>  | vector containing the parameter values of the points given in the 'points' vector. They are stored in "uvuv...-fashion".                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| <i>domain</i>   | pointer to an array of four doubles specifying the parametric domain for the surface to be generated. They should be stored as "u_min, u_max, v_min, v_max".                                                                                                                                                                                                                                                                                                                                                                                                                 |
| <i>dim</i>      | spatial dimension of the points (usually 3).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| <i>aepsge</i>   | geometric tolerance to use internally                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| <i>constdir</i> | The points <i>will</i> be reparameterized internally according to their spatial position with respect to the surface that shall be generated. However, they might be reparameterized in both their u and v parameters, only in their u parameters or only in their v parameters. The user can specify this with 'constdir'. If 'constdir' is set to 0, the points will be reparameterized in both u and v. If 'constdir' is set to 1, they will only be reparameterized in the v parameter. If 'constdir' is set to 2, they will only be reparameterized in the u parameter. |

29.11.2.2 `Go::ApproxSurf::ApproxSurf ( shared_ptr< SplineSurface > & srf, const std::vector< double > & points, const std::vector< double > & parvals, int dim, double aepsge, int constdir = 0, bool approx_orig = false, bool close_belt = false, int nmb_stabil = 0, bool repar = true )`

Constructor where the user specifies a spline surface that should be modified, the points to approximate and their parameter values, as well as the geometric tolerance. The surface that is given as argument is not copied internally, only pointed to, so it *will* be modified.

## Parameters

|                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|-------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>srf</i>        | the surface that will be modified to approximate the points. Assumed to contain k-regular knots.                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| <i>points</i>     | vector containing the coordinates of the points that this surface should interpolate. They are stored in "xyzxyz...-fashion".                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| <i>parvals</i>    | vector containing the parameter values of the points given in the 'points' vector. They are stored in "uvuv...-fashion".                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| <i>dim</i>        | spatial dimension of the points (usually 3).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| <i>aepsge</i>     | geometric tolerance to use internally                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| <i>constdir</i>   | The points <i>will</i> be reparameterized internally according to their spatial position with respect to the surface that shall be generated. However, they might be reparameterized in both their u and v parameters, only in their u parameters or only in their v parameters. The user can specify this with 'constdir'. If 'constdir' is set to 0, the points will be reparameterized in both u and v. If 'constdir' is set to 1, they will only be reparameterized in the v parameter. If 'constdir' is set to 2, they will only be reparameterized in the u parameter. |
| <i>close_belt</i> | Indicates if only coefficients close to the sampling points should be modified                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |

29.11.2.3 `Go::ApproxSurf::~~ApproxSurf ( )`

Destructor.

29.11.2.4 `Go::ApproxSurf::ApproxSurf ( )` [protected]

Default constructor.

### 29.11.3 Member Function Documentation

#### 29.11.3.1 void Go::ApproxSurf::edgeFix ( int *edge\_fix*[ ] ) [inline]

Decide whether specific edges of the surface's boundary should be kept fixed (i.e. unchanged by approximation process), as well as a certain number of cross- derivatives across these curves.

##### Parameters

|                 |                                                                                                                                                                                                                                                                                                                                                                                                                                |
|-----------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>edge_fix</i> | pointer to an array of four integers specifying to which extent each surface edge should be kept fixed during the approximation process. A value of 0 means that it will not be kept fixed, 1 means that its position will be kept fixed, 2 means that its position and cross-tangent will be kept fixed, etc. The integers are associated with the surface edges starting with the edge 'v=vmin' and moving counterclockwise. |
|-----------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

Definition at line 181 of file ApproxSurf.h.

#### 29.11.3.2 shared\_ptr<SplineSurface> Go::ApproxSurf::getApproxSurf ( double & *maxdist*, double & *avdist*, int & *nmb\_out\_eps*, int *max\_iter* = 4, int *keep\_init* = 0 )

Fetch the approximating surface.

When everything else is set, this function can be used to run the approximation process and fetch the approximated surface.

##### Return values

|                |                                                                                  |
|----------------|----------------------------------------------------------------------------------|
| <i>maxdist</i> | report the maximum distance between the approximated surface and the data points |
| <i>avdist</i>  | report the average distance between the approximated surface and the datapoints  |

##### Parameters

|                    |                                                                                                                                                                               |
|--------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>nmb_out_eps</i> | report the number of data points that were found to lie outside the geometric tolerance (as specified by the 'aepsge' argument to the <a href="#">ApproxSurf</a> constructor. |
| <i>max_iter</i>    | maximum number of iterations                                                                                                                                                  |

##### Returns

a shared pointer to the generated [SplineCurve](#), approximating the points as specified.

#### 29.11.3.3 bool Go::ApproxSurf::getDoRefine ( ) [inline]

Check whether or not the spline space will be refined during the approximation iterations

Definition at line 232 of file ApproxSurf.h.

#### 29.11.3.4 int Go::ApproxSurf::reParam ( )

Reparameterize the data points by a closest point match against the current surface.

29.11.3.5 void Go::ApproxSurf::setC1Approx ( double *fac1*, double *fac2* )

Approximate C1 continuity with a given importance  $0 \leq \text{fac} < 1$ .

29.11.3.6 void Go::ApproxSurf::setDoRefine ( bool *refine* ) [inline]

Set whether or not the spline space is to be refined during the approximation iterations

Definition at line 239 of file ApproxSurf.h.

29.11.3.7 void Go::ApproxSurf::setFixBoundary ( bool *fix\_boundary* ) [inline]

Decide whether or not the total boundary of the surface should be kept fixed (i.e. unchanged by approximation process). Default is true. Cross derivatives will not be kept fixed. (If you want to keep cross derivatives fixed, use the [edgeFix\(\)](#) member function instead).

#### Parameters

|                     |                                                                                                         |
|---------------------|---------------------------------------------------------------------------------------------------------|
| <i>fix_boundary</i> | if 'true' the boundary of the surface will not be modified, if 'false' it will be open to modification. |
|---------------------|---------------------------------------------------------------------------------------------------------|

Definition at line 164 of file ApproxSurf.h.

29.11.3.8 void Go::ApproxSurf::setNormalConditions ( const std::vector< double > & *points*, const std::vector< double > & *parvals*, int *nmb\_stabil* = 0 ) [inline]

Forces the surface to approximate certain normals at certain parameter values.

#### Parameters

|                |                                                                                                                             |
|----------------|-----------------------------------------------------------------------------------------------------------------------------|
| <i>points</i>  | this vector contains the normals that should be approximated. They are stored in 'xyzxyz... fashion'.                       |
| <i>parvals</i> | this vector contains the parameter values of the normals that should be approximated. They are stored in 'uvuv... fashion'. |

Definition at line 193 of file ApproxSurf.h.

29.11.3.9 void Go::ApproxSurf::setSmoothingWeight ( double *smooth* ) [inline]

Sets the smoothing weight to something other than the default (1e-9). The value should lie in the unit interval, typically close to 0.

#### Parameters

|               |                           |
|---------------|---------------------------|
| <i>smooth</i> | the new smoothing weight. |
|---------------|---------------------------|

Definition at line 152 of file ApproxSurf.h.

The documentation for this class was generated from the following file:

- [gotools-core/include/GoTools/creators/ApproxSurf.h](#)

## 29.12 Go::Array< T, Dim > Class Template Reference

```
#include <Array.h>
```

### Public Member Functions

- [Array \(\)](#)  
*Default constructor, does not initialize elements.*
- [Array \(T x\)](#)
- [Array \(T x, T y\)](#)
- [Array \(T x, T y, T z\)](#)
- [Array \(T x, T y, T z, T w\)](#)
- [Array \(const Array &v\)](#)  
*Copy constructor.*
- `template<typename RandomAccessIterator >`  
[Array \(RandomAccessIterator start\)](#)
- `template<typename RandomAccessIterator >`  
`void setValue (RandomAccessIterator from)`
- `template<typename U >`  
[Array \(const Array< U, Dim > &v\)](#)
- `template<typename RandomAccessIterator >`  
`void setValueConvert (RandomAccessIterator from)`
- [Array & operator= \(const Array &v\)](#)  
*Assignment operator.*
- `void read (std::istream &is)`
- `void write (std::ostream &os) const`
- `const T & x () const`
- `T & x ()`
- `const T & y () const`
- `T & y ()`
- `const T & z () const`
- `T & z ()`
- `const T & operator\[\] (int i) const`  
*Read-only index access.*
- `T & operator\[\] (int i)`  
*Index access.*
- `const T * begin () const`  
*Get a read-only start iterator.*
- `T * begin ()`  
*Get a start iterator.*
- `const T * end () const`  
*Get a read-only end iterator.*
- `T * end ()`  
*Get a end iterator.*
- `int size () const`  
*Get the dimension of the [Array](#).*



- [T length2 \(\) const](#)
- [T length \(\) const](#)
- [T lengthInf \(\) const](#)
- [T dist2 \(const Array &v\) const](#)
- [T dist \(const Array &v\) const](#)
- [T distInf \(const Array &v\) const](#)
- [void normalize \(\)](#)
- [Array operator+ \(const Array &v\) const](#)  
*The sum of two vectors.*
- [bool operator== \(const Array &v\) const](#)  
*Add a vector to this vector.*
- [Array & operator+= \(const Array &v\)](#)  
*Add a vector to this vector.*
- [Array operator- \(const Array &v\) const](#)  
*The difference between two vectors.*
- [Array & operator-= \(const Array &v\)](#)  
*Subtract a vector from this vector.*
- [Array operator\\* \(T d\) const](#)  
*The product of a vector and a scalar.*
- [Array & operator\\*= \(T d\)](#)  
*Multiply this vector by a scalar.*
- [Array operator/ \(double d\) const](#)  
*A vector divided by a scalar.*
- [Array & operator/= \(double d\)](#)  
*Divide this vector with a scalar.*
- [Array operator- \(\) const](#)  
*The negation of a vector.*
- [T operator\\* \(const Array &v\) const](#)
- [Array operator% \(const Array &v\) const](#)
- [Array cross \(const Array &v\) const](#)
- [T cosAngle \(const Array &v\) const](#)
- [T angle \(const Array &v\) const](#)
- [void zero \(\)](#)  
*Sets the vector to the zero vector.*

### 29.12.1 Detailed Description

```
template<typename T, int Dim>
class Go::Array< T, Dim >
```

Compile-time sized array. Encapsulates an array with full value semantics, i.e. copying and assignment works as expected. The class also has some vector algebra functionality, such as scalar product, multiplication by scalars etc. In such contexts the objects are talked about as 'vectors' and not 'Arrays'.

Definition at line 61 of file Array.h.

### 29.12.2 Constructor & Destructor Documentation

29.12.2.1 `template<typename T, int Dim> Go::Array< T, Dim >::Array ( ) [inline]`

Default constructor, does not initialize elements.

Definition at line 68 of file Array.h.

29.12.2.2 `template<typename T, int Dim> Go::Array< T, Dim >::Array ( T x ) [inline], [explicit]`

Constructor filling the 'Dim'-tuple with the same value. (J.O. 080211) I don't think this one should interfere with other constructors, particularly since there doesn't seem to be support for an M==1 case already...?!

Definition at line 74 of file Array.h.

29.12.2.3 `template<typename T, int Dim> Go::Array< T, Dim >::Array ( T x, T y ) [inline]`

Constructor taking 2 arguments, only compiles if [Array](#) dimension argument is 2.

Definition at line 82 of file Array.h.

29.12.2.4 `template<typename T, int Dim> Go::Array< T, Dim >::Array ( T x, T y, T z ) [inline]`

Constructor taking 3 arguments, only compiles if [Array](#) dimension argument is 3.

Definition at line 90 of file Array.h.

29.12.2.5 `template<typename T, int Dim> Go::Array< T, Dim >::Array ( T x, T y, T z, T w ) [inline]`

Constructor taking 4 arguments, only compiles if [Array](#) dimension argument is 4.

Definition at line 99 of file Array.h.

29.12.2.6 `template<typename T, int Dim> Go::Array< T, Dim >::Array ( const Array< T, Dim > & v ) [inline]`

Copy constructor.

Definition at line 108 of file Array.h.

29.12.2.7 `template<typename T, int Dim> template<typename RandomAccessIterator > Go::Array< T, Dim >::Array ( RandomAccessIterator start ) [inline], [explicit]`

Constructor taking a pointer or other random access iterator and copying all elements into the [Array](#).

Definition at line 116 of file Array.h.

29.12.2.8 `template<typename T, int Dim> template<typename U > Go::Array< T, Dim >::Array ( const Array< U, Dim > & v ) [inline], [explicit]`

Constructor that takes an [Array](#) of a compatible type.

Definition at line 132 of file Array.h.

### 29.12.3 Member Function Documentation

29.12.3.1 `template<typename T, int Dim> T Go::Array< T, Dim >::angle ( const Array< T, Dim > & v ) const`  
[inline]

The angle between this and another vector.

Definition at line 411 of file Array.h.

29.12.3.2 `template<typename T, int Dim> const T* Go::Array< T, Dim >::begin ( ) const` [inline]

Get a read-only start iterator.

Definition at line 198 of file Array.h.

29.12.3.3 `template<typename T, int Dim> T* Go::Array< T, Dim >::begin ( )` [inline]

Get a start iterator.

Definition at line 200 of file Array.h.

29.12.3.4 `template<typename T, int Dim> T Go::Array< T, Dim >::cosAngle ( const Array< T, Dim > & v ) const`  
[inline]

The cosine of the angle between this and another vector.

Definition at line 399 of file Array.h.

29.12.3.5 `template<typename T, int Dim> Array Go::Array< T, Dim >::cross ( const Array< T, Dim > & v ) const`  
[inline]

The cross product of two vectors. Only compiles if dimension is 3.

Definition at line 392 of file Array.h.

29.12.3.6 `template<typename T, int Dim> T Go::Array< T, Dim >::dist ( const Array< T, Dim > & v ) const`  
[inline]

Get the euclidian length of the difference between this vector and another vector.

Definition at line 254 of file Array.h.

29.12.3.7 `template<typename T, int Dim> T Go::Array< T, Dim >::dist2 ( const Array< T, Dim > & v ) const`  
[inline]

Get the square of the euclidian length of the difference between this vector and another vector.

Definition at line 241 of file Array.h.

**29.12.3.8** `template<typename T, int Dim> T Go::Array< T, Dim >::distInf ( const Array< T, Dim > & v ) const [inline]`

Get the infinity-norm (or max-norm) length of the difference between this vector and another vector.

Definition at line 265 of file Array.h.

**29.12.3.9** `template<typename T, int Dim> const T* Go::Array< T, Dim >::end ( ) const [inline]`

Get a read-only end iterator.

Definition at line 202 of file Array.h.

**29.12.3.10** `template<typename T, int Dim> T* Go::Array< T, Dim >::end ( ) [inline]`

Get a end iterator.

Definition at line 204 of file Array.h.

**29.12.3.11** `template<typename T, int Dim> T Go::Array< T, Dim >::length ( ) const [inline]`

Get the euclidian length of the vector.

Definition at line 220 of file Array.h.

**29.12.3.12** `template<typename T, int Dim> T Go::Array< T, Dim >::length2 ( ) const [inline]`

Get the square of the euclidian length of the vector.

Definition at line 211 of file Array.h.

**29.12.3.13** `template<typename T, int Dim> T Go::Array< T, Dim >::lengthInf ( ) const [inline]`

Get the infinity-norm (or max-norm) length of the vector.

Definition at line 230 of file Array.h.

**29.12.3.14** `template<typename T, int Dim> void Go::Array< T, Dim >::normalize ( ) [inline]`

Normalize this vector, i.e. divide every element by `length()`.

Definition at line 278 of file Array.h.

```
29.12.3.15 template<typename T, int Dim> Array Go::Array< T, Dim >::operator%(const Array< T, Dim > & v) const
[inline]
```

The cross product of two vectors. Only compiles if dimension is 3.

Definition at line 376 of file Array.h.

```
29.12.3.16 template<typename T, int Dim> Array Go::Array< T, Dim >::operator*(T d) const [inline]
```

The product of a vector and a scalar.

Definition at line 325 of file Array.h.

```
29.12.3.17 template<typename T, int Dim> T Go::Array< T, Dim >::operator*(const Array< T, Dim > & v) const
[inline]
```

The scalar product (or inner product, or dot product) of two vectors.

Definition at line 366 of file Array.h.

```
29.12.3.18 template<typename T, int Dim> Array& Go::Array< T, Dim >::operator*=(T d) [inline]
```

Multiply this vector by a scalar.

Definition at line 333 of file Array.h.

```
29.12.3.19 template<typename T, int Dim> Array Go::Array< T, Dim >::operator+ (const Array< T, Dim > & v) const
[inline]
```

The sum of two vectors.

Definition at line 284 of file Array.h.

```
29.12.3.20 template<typename T, int Dim> Array& Go::Array< T, Dim >::operator+=(const Array< T, Dim > & v)
[inline]
```

Add a vector to this vector.

Definition at line 300 of file Array.h.

```
29.12.3.21 template<typename T, int Dim> Array Go::Array< T, Dim >::operator- (const Array< T, Dim > & v) const
[inline]
```

The difference between two vectors.

Definition at line 308 of file Array.h.

29.12.3.22 `template<typename T, int Dim> Array Go::Array< T, Dim >::operator- ( ) const [inline]`

The negation of a vector.

Definition at line 356 of file Array.h.

29.12.3.23 `template<typename T, int Dim> Array& Go::Array< T, Dim >::operator-= ( const Array< T, Dim > & v ) [inline]`

Subtract a vector from this vector.

Definition at line 316 of file Array.h.

29.12.3.24 `template<typename T, int Dim> Array Go::Array< T, Dim >::operator/ ( double d ) const [inline]`

A vector divided by a scalar.

Definition at line 341 of file Array.h.

29.12.3.25 `template<typename T, int Dim> Array& Go::Array< T, Dim >::operator/= ( double d ) [inline]`

Divide this vector with a scalar.

Definition at line 348 of file Array.h.

29.12.3.26 `template<typename T, int Dim> Array& Go::Array< T, Dim >::operator= ( const Array< T, Dim > & v ) [inline]`

Assignment operator.

Definition at line 147 of file Array.h.

29.12.3.27 `template<typename T, int Dim> bool Go::Array< T, Dim >::operator==( const Array< T, Dim > & v ) const [inline]`

Add a vector to this vector.

Definition at line 292 of file Array.h.

29.12.3.28 `template<typename T, int Dim> const T& Go::Array< T, Dim >::operator[]( int i ) const [inline]`

Read-only index access.

Definition at line 193 of file Array.h.

29.12.3.29 `template<typename T, int Dim> T& Go::Array< T, Dim >::operator[]( int i ) [inline]`

Index access.

Definition at line 195 of file Array.h.

29.12.3.30 `template<typename T, int Dim> void Go::Array< T, Dim >::read ( std::istream & is ) [inline]`

Reads an [Array](#) elementwise from a standard istream.

Definition at line 157 of file Array.h.

29.12.3.31 `template<typename T, int Dim> template<typename RandomAccessIterator > void Go::Array< T, Dim >::setValue ( RandomAccessIterator from ) [inline]`

Takes a pointer or other random access iterator and copies all elements into the [Array](#).

Definition at line 124 of file Array.h.

29.12.3.32 `template<typename T, int Dim> template<typename RandomAccessIterator > void Go::Array< T, Dim >::setValueConvert ( RandomAccessIterator from ) [inline]`

Takes a pointer or other random access iterator with compatible value type and copies all elements into the [Array](#).

Definition at line 141 of file Array.h.

29.12.3.33 `template<typename T, int Dim> int Go::Array< T, Dim >::size ( ) const [inline]`

Get the dimension of the [Array](#).

Definition at line 207 of file Array.h.

29.12.3.34 `template<typename T, int Dim> void Go::Array< T, Dim >::write ( std::ostream & os ) const [inline]`

Writes an [Array](#) elementwise to a standard ostream. Precision is set to 16 by this function.

Definition at line 165 of file Array.h.

29.12.3.35 `template<typename T, int Dim> const T& Go::Array< T, Dim >::x ( ) const [inline]`

Access to first element. (Deprecated, kept for backward compatibility)

Definition at line 175 of file Array.h.

29.12.3.36 `template<typename T, int Dim> T& Go::Array< T, Dim >::x ( ) [inline]`

Access to first element. (Deprecated, kept for backward compatibility)

Definition at line 178 of file Array.h.

29.12.3.37 `template<typename T, int Dim> const T& Go::Array< T, Dim >::y ( ) const [inline]`

Access to first element. (Deprecated, kept for backward compatibility)

Definition at line 181 of file Array.h.

29.12.3.38 `template<typename T, int Dim> T& Go::Array< T, Dim >::y ( ) [inline]`

Access to first element. (Deprecated, kept for backward compatibility)

Definition at line 184 of file Array.h.

29.12.3.39 `template<typename T, int Dim> const T& Go::Array< T, Dim >::z ( ) const [inline]`

Access to first element. (Deprecated, kept for backward compatibility)

Definition at line 187 of file Array.h.

29.12.3.40 `template<typename T, int Dim> T& Go::Array< T, Dim >::z ( ) [inline]`

Access to first element. (Deprecated, kept for backward compatibility)

Definition at line 190 of file Array.h.

29.12.3.41 `template<typename T, int Dim> void Go::Array< T, Dim >::zero ( ) [inline]`

Sets the vector to the zero vector.

Definition at line 421 of file Array.h.

The documentation for this class was generated from the following file:

- [gotools-core/include/GoTools/utis/Array.h](#)

## 29.13 NEWMAT::ArrayLengthSpecifier Class Reference

```
#include <newmat.h>
```



## Public Member Functions

- `int Value () const`
- `ArrayLengthSpecifier (int l)`

### 29.13.1 Detailed Description

Definition at line 216 of file `newmat.h`.

### 29.13.2 Constructor & Destructor Documentation

29.13.2.1 `NEWMAT::ArrayLengthSpecifier::ArrayLengthSpecifier ( int l ) [inline]`

Definition at line 221 of file `newmat.h`.

### 29.13.3 Member Function Documentation

29.13.3.1 `int NEWMAT::ArrayLengthSpecifier::Value ( ) const [inline]`

Definition at line 220 of file `newmat.h`.

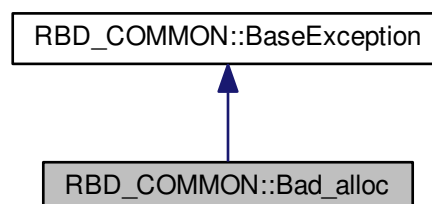
The documentation for this class was generated from the following file:

- `newmat/include/newmat.h`

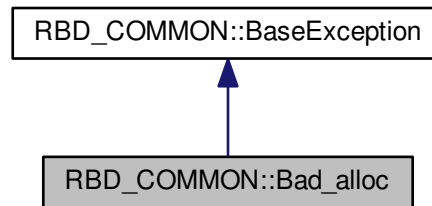
## 29.14 RBD\_COMMON::Bad\_alloc Class Reference

```
#include <myexcept.h>
```

Inheritance diagram for `RBD_COMMON::Bad_alloc`:



Collaboration diagram for RBD\_COMMON::Bad\_alloc:



- static unsigned long [Select](#)
- [Bad\\_alloc](#) (const char \*a\_what=0)

## Additional Inherited Members

### 29.14.1 Detailed Description

Definition at line 427 of file myexcept.h.

### 29.14.2 Constructor & Destructor Documentation

#### 29.14.2.1 `Bad_alloc::Bad_alloc ( const char * a_what = 0 )`

Definition at line 468 of file myexcept.cpp.

### 29.14.3 Member Data Documentation

#### 29.14.3.1 `unsigned long Bad_alloc::Select [static]`

Definition at line 430 of file myexcept.h.

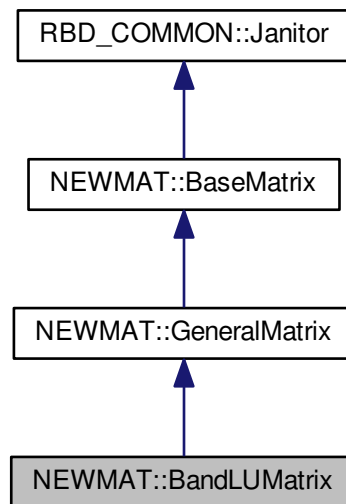
The documentation for this class was generated from the following files:

- [newmat/include/myexcept.h](#)
- [newmat/src/myexcept.cpp](#)

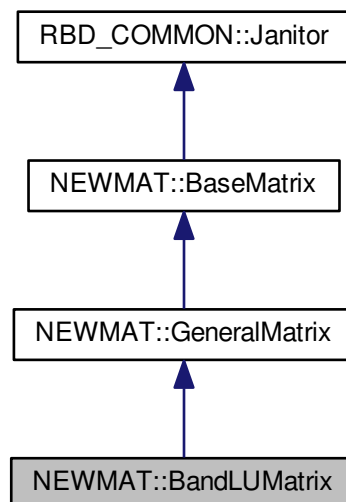
## 29.15 NEWMAT::BandLUMatrix Class Reference

```
#include <newmat.h>
```

Inheritance diagram for NEWMAT::BandLUMatrix:



Collaboration diagram for NEWMAT::BandLUMatrix:



## Public Member Functions

- [BandLUMatrix](#) ([const BaseMatrix](#) &)
- [MatrixType](#) [Type](#) () [const](#)
- [void](#) [lubksb](#) ([Real](#) \*, [int](#)=0)
- [~BandLUMatrix](#) ()
- [GeneralMatrix](#) \* [MakeSolver](#) ()
- [LogAndSign](#) [LogDeterminant](#) () [const](#)
- [void](#) [Solver](#) ([MatrixColX](#) &, [const MatrixColX](#) &)
- [void](#) [GetRow](#) ([MatrixRowCol](#) &)
- [void](#) [GetCol](#) ([MatrixRowCol](#) &)
- [void](#) [GetCol](#) ([MatrixColX](#) &c)
- [void](#) [operator=](#) ([const BaseMatrix](#) &)
- [void](#) [operator=](#) ([const BandLUMatrix](#) &m)
- [void](#) [CleanUp](#) ()
- [bool](#) [IsEqual](#) ([const GeneralMatrix](#) &) [const](#)
- [bool](#) [IsSingular](#) () [const](#)

## Additional Inherited Members

### 29.15.1 Detailed Description

Definition at line 1070 of file newmat.h.

### 29.15.2 Constructor & Destructor Documentation

#### 29.15.2.1 [BandLUMatrix::BandLUMatrix](#) ( [const BaseMatrix](#) & *m* )

Definition at line 225 of file bandmat.cpp.

#### 29.15.2.2 [BandLUMatrix::~~BandLUMatrix](#) ( )

Definition at line 243 of file bandmat.cpp.

### 29.15.3 Member Function Documentation

#### 29.15.3.1 [void](#) [BandLUMatrix::CleanUp](#) ( ) [[virtual](#)]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 844 of file newmat4.cpp.

#### 29.15.3.2 [void](#) [BandLUMatrix::GetCol](#) ( [MatrixRowCol](#) & ) [[virtual](#)]

Implements [NEWMAT::GeneralMatrix](#).

Definition at line 292 of file newmatex.cpp.

29.15.3.3 void NEWMAT::BandLUMatrix::GetCol ( MatrixColX & c ) [inline],[virtual]

Implements [NEWMAT::GeneralMatrix](#).

Definition at line 1090 of file newmat.h.

29.15.3.4 void BandLUMatrix::GetRow ( MatrixRowCol & ) [virtual]

Implements [NEWMAT::GeneralMatrix](#).

Definition at line 290 of file newmatex.cpp.

29.15.3.5 bool BandLUMatrix::IsEqual ( const GeneralMatrix & A ) const [virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 1028 of file newmat7.cpp.

29.15.3.6 bool NEWMAT::BandLUMatrix::IsSingular ( ) const [inline]

Definition at line 1095 of file newmat.h.

29.15.3.7 LogAndSign BandLUMatrix::LogDeterminant ( ) const [virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 254 of file bandmat.cpp.

29.15.3.8 void BandLUMatrix::lubksb ( Real \* B, int mini = 0 )

Definition at line 313 of file bandmat.cpp.

29.15.3.9 GeneralMatrix\* NEWMAT::BandLUMatrix::MakeSolver ( ) [inline],[virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 1085 of file newmat.h.

29.15.3.10 void BandLUMatrix::operator= ( const BaseMatrix & )

Definition at line 294 of file newmatex.cpp.

29.15.3.11 `void NEWMAT::BandLUMatrix::operator= ( const BandLUMatrix & m ) [inline]`

Definition at line 1092 of file newmat.h.

29.15.3.12 `void BandLUMatrix::Solver ( MatrixColX & mcout, const MatrixColX & mcin ) [virtual]`

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 340 of file bandmat.cpp.

29.15.3.13 `MatrixType BandLUMatrix::Type ( ) const [virtual]`

Implements [NEWMAT::GeneralMatrix](#).

Definition at line 251 of file bandmat.cpp.

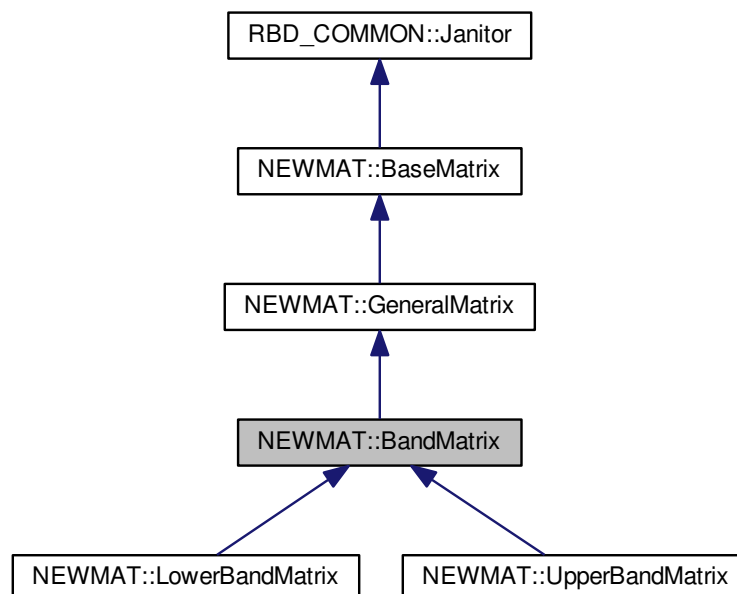
The documentation for this class was generated from the following files:

- [newmat/include/newmat.h](#)
- [newmat/src/bandmat.cpp](#)
- [newmat/src/newmat4.cpp](#)
- [newmat/src/newmat7.cpp](#)
- [newmat/src/newmatex.cpp](#)

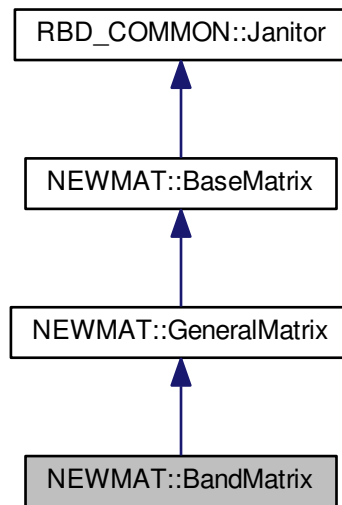
## 29.16 NEWMAT::BandMatrix Class Reference

```
#include <newmat.h>
```

Inheritance diagram for NEWMAT::BandMatrix:



Collaboration diagram for NEWMAT::BandMatrix:



## Public Member Functions

- [BandMatrix](#) ()
- [~BandMatrix](#) ()
- [BandMatrix](#) (int n, int lb, int ub)
- [BandMatrix](#) (const [BaseMatrix](#) &)
- void [operator=](#) (const [BaseMatrix](#) &)
- void [operator=](#) (Real f)
- void [operator=](#) (const [BandMatrix](#) &m)
- [MatrixType](#) [Type](#) () const
- [Real](#) & [operator\(\)](#) (int, int)
- [Real](#) & [element](#) (int, int)
- [Real](#) [operator\(\)](#) (int, int) const
- [Real](#) [element](#) (int, int) const
- [BandMatrix](#) (const [BandMatrix](#) &gm)
- [LogAndSign](#) [LogDeterminant](#) () const
- [GeneralMatrix](#) \* [MakeSolver](#) ()
- [Real](#) [Trace](#) () const
- [Real](#) [SumSquare](#) () const
- [Real](#) [SumAbsoluteValue](#) () const
- [Real](#) [Sum](#) () const
- [Real](#) [MaximumAbsoluteValue](#) () const
- [Real](#) [MinimumAbsoluteValue](#) () const
- [Real](#) [Maximum](#) () const
- [Real](#) [Minimum](#) () const
- void [GetRow](#) ([MatrixRowCol](#) &)
- void [GetCol](#) ([MatrixRowCol](#) &)
- void [GetCol](#) ([MatrixColX](#) &)
- void [RestoreCol](#) ([MatrixRowCol](#) &)

- void [RestoreCol](#) ([MatrixColX](#) &c)
- void [NextRow](#) ([MatrixRowCol](#) &)
- virtual void [ReSize](#) (int, int, int)
- void [ReSize](#) (const [GeneralMatrix](#) &A)
- bool [SameStorageType](#) (const [GeneralMatrix](#) &A) const
- void [ReSizeForAdd](#) (const [GeneralMatrix](#) &A, const [GeneralMatrix](#) &B)
- void [ReSizeForSP](#) (const [GeneralMatrix](#) &A, const [GeneralMatrix](#) &B)
- [MatrixBandWidth](#) [BandWidth](#) () const
- void [SetParameters](#) (const [GeneralMatrix](#) \*)
- [MatrixInput](#) operator<< ([Real](#))
- [MatrixInput](#) operator<< (int f)
- void operator<< (const [Real](#) \*r)
- void operator<< (const [BaseMatrix](#) &X)

## Public Attributes

- int [lower](#)
- int [upper](#)

## Protected Member Functions

- void [CornerClear](#) () const
- short [SimpleAddOK](#) (const [GeneralMatrix](#) \*gm)

## Additional Inherited Members

### 29.16.1 Detailed Description

Definition at line 889 of file newmat.h.

### 29.16.2 Constructor & Destructor Documentation

#### 29.16.2.1 [NEWMAT::BandMatrix::BandMatrix](#) ( ) [inline]

Definition at line 897 of file newmat.h.

#### 29.16.2.2 [NEWMAT::BandMatrix::~~BandMatrix](#) ( ) [inline]

Definition at line 898 of file newmat.h.

#### 29.16.2.3 [NEWMAT::BandMatrix::BandMatrix](#) ( int *n*, int *lb*, int *ub* ) [inline]

Definition at line 899 of file newmat.h.



29.16.2.4 NEWMAT::BandMatrix::BandMatrix ( const BaseMatrix & )

29.16.2.5 NEWMAT::BandMatrix::BandMatrix ( const BandMatrix & gm ) [inline]

Definition at line 914 of file newmat.h.

### 29.16.3 Member Function Documentation

29.16.3.1 MatrixBandWidth BandMatrix::BandWidth ( ) const [virtual]

Reimplemented from [NEWMAT::BaseMatrix](#).

Definition at line 412 of file newmat4.cpp.

29.16.3.2 void BandMatrix::CornerClear ( ) const [protected]

Definition at line 156 of file bandmat.cpp.

29.16.3.3 Real& NEWMAT::BandMatrix::element ( int , int )

29.16.3.4 Real NEWMAT::BandMatrix::element ( int , int ) const

29.16.3.5 void NEWMAT::BandMatrix::GetCol ( MatrixRowCol & ) [virtual]

Implements [NEWMAT::GeneralMatrix](#).

29.16.3.6 void NEWMAT::BandMatrix::GetCol ( MatrixColX & ) [virtual]

Implements [NEWMAT::GeneralMatrix](#).

29.16.3.7 void BandMatrix::GetRow ( MatrixRowCol & mrc ) [virtual]

Implements [NEWMAT::GeneralMatrix](#).

Definition at line 566 of file newmat3.cpp.

29.16.3.8 LogAndSign BandMatrix::LogDeterminant ( ) const [virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Reimplemented in [NEWMAT::LowerBandMatrix](#), and [NEWMAT::UpperBandMatrix](#).

Definition at line 397 of file bandmat.cpp.

29.16.3.9 **GeneralMatrix \* BandMatrix::MakeSolver ( )** [virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Reimplemented in [NEWMAT::LowerBandMatrix](#), and [NEWMAT::UpperBandMatrix](#).

Definition at line 264 of file bandmat.cpp.

29.16.3.10 **Real NEWMAT::BandMatrix::Maximum ( ) const** [inline],[virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 927 of file newmat.h.

29.16.3.11 **Real NEWMAT::BandMatrix::MaximumAbsoluteValue ( ) const** [inline],[virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 923 of file newmat.h.

29.16.3.12 **Real NEWMAT::BandMatrix::Minimum ( ) const** [inline],[virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 928 of file newmat.h.

29.16.3.13 **Real NEWMAT::BandMatrix::MinimumAbsoluteValue ( ) const** [inline],[virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 925 of file newmat.h.

29.16.3.14 **void BandMatrix::NextRow ( MatrixRowCol & mrc )** [virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 578 of file newmat3.cpp.

29.16.3.15 **Real& NEWMAT::BandMatrix::operator() ( int , int )**

29.16.3.16 **Real NEWMAT::BandMatrix::operator() ( int , int ) const**

29.16.3.17 **MatrixInput BandMatrix::operator<< ( Real )**

Definition at line 528 of file newmat5.cpp.

29.16.3.18 **MatrixInput** NEWMAT::BandMatrix::operator<<( int *f* ) [inline]

Definition at line 1785 of file newmat.h.

29.16.3.19 void NEWMAT::BandMatrix::operator<<( const Real \* *r* )

29.16.3.20 void NEWMAT::BandMatrix::operator<<( const BaseMatrix & *X* ) [inline]

Definition at line 947 of file newmat.h.

29.16.3.21 void NEWMAT::BandMatrix::operator=( const BaseMatrix & )

29.16.3.22 void NEWMAT::BandMatrix::operator=( Real *f* ) [inline]

Definition at line 903 of file newmat.h.

29.16.3.23 void NEWMAT::BandMatrix::operator=( const BandMatrix & *m* ) [inline]

Definition at line 904 of file newmat.h.

29.16.3.24 virtual void NEWMAT::BandMatrix::ReSize( int, int, int ) [virtual]

Reimplemented in [NEWMAT::LowerBandMatrix](#), and [NEWMAT::UpperBandMatrix](#).

29.16.3.25 void BandMatrix::ReSize( const GeneralMatrix & *A* ) [virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Reimplemented in [NEWMAT::LowerBandMatrix](#), and [NEWMAT::UpperBandMatrix](#).

Definition at line 102 of file bandmat.cpp.

29.16.3.26 void BandMatrix::ReSizeForAdd( const GeneralMatrix & *A*, const GeneralMatrix & *B* ) [virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 122 of file bandmat.cpp.

29.16.3.27 void BandMatrix::ReSizeForSP( const GeneralMatrix & *A*, const GeneralMatrix & *B* ) [virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 135 of file bandmat.cpp.

29.16.3.28 void NEWMAT::BandMatrix::RestoreCol ( MatrixRowCol & ) [virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

29.16.3.29 void NEWMAT::BandMatrix::RestoreCol ( MatrixColIX & c ) [inline],[virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 933 of file newmat.h.

29.16.3.30 bool BandMatrix::SameStorageType ( const GeneralMatrix & A ) const [virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 115 of file bandmat.cpp.

29.16.3.31 void BandMatrix::SetParameters ( const GeneralMatrix \* gmx ) [virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 38 of file bandmat.cpp.

29.16.3.32 short BandMatrix::SimpleAddOK ( const GeneralMatrix \* gm ) [protected],[virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 63 of file bandmat.cpp.

29.16.3.33 Real NEWMAT::BandMatrix::Sum ( ) const [inline],[virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 921 of file newmat.h.

29.16.3.34 Real NEWMAT::BandMatrix::SumAbsoluteValue ( ) const [inline],[virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 919 of file newmat.h.

29.16.3.35 Real NEWMAT::BandMatrix::SumSquare ( ) const [inline],[virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 918 of file newmat.h.

29.16.3.36 `Real BandMatrix::Trace ( ) const [virtual]`

Reimplemented from [NEWMAT::BaseMatrix](#).

Definition at line 580 of file `newmat8.cpp`.

29.16.3.37 `MatrixType BandMatrix::Type ( ) const [virtual]`

Implements [NEWMAT::GeneralMatrix](#).

Reimplemented in [NEWMAT::LowerBandMatrix](#), and [NEWMAT::UpperBandMatrix](#).

Definition at line 393 of file `newmat4.cpp`.

## 29.16.4 Member Data Documentation

29.16.4.1 `int NEWMAT::BandMatrix::lower`

Definition at line 896 of file `newmat.h`.

29.16.4.2 `int NEWMAT::BandMatrix::upper`

Definition at line 896 of file `newmat.h`.

The documentation for this class was generated from the following files:

- [newmat/include/newmat.h](#)
- [newmat/src/bandmat.cpp](#)
- [newmat/src/newmat3.cpp](#)
- [newmat/src/newmat4.cpp](#)
- [newmat/src/newmat5.cpp](#)
- [newmat/src/newmat8.cpp](#)

## 29.17 Go::BaryCoordSystem< Dim > Class Template Reference

```
#include <BaryCoordSystem.h>
```

### Public Member Functions

- [BaryCoordSystem \( \)](#)  
*Empty default constructor.*
- [BaryCoordSystem \(const Array< double, Dim > \\*corners\)](#)
- void [setCorners \(const Array< double, Dim > \\*corners\)](#)  
*Set corners.*
- [const Array< double, Dim > & corner \(int i\) const](#)  
*Returns the corner points.*
- `template<class T >`  
[Array< T, Dim > baryToCart \(const Array< T, Dim+1 > &bary\\_pt\) const](#)
- `template<class T >`  
[Array< T, Dim+1 > cartToBary \(const Array< T, Dim > &cart\\_pt\) const](#)
- void [read \(std::istream &is\)](#)  
*Read from input stream.*
- void [write \(std::ostream &os\) const](#)  
*Write to output stream.*

### 29.17.1 Detailed Description

```
template<int Dim>
class Go::BaryCoordSystem< Dim >
```

Encapsulates a barycentric coordinate system. It is designed to work in any dimension, but at the time of writing, only dimension 2 and 3 is supported by the functions in the included header "Volumes.h"

You specify the dimension with a template parameter:

```
BaryCoordSystem<2> bc(corner_pointer);
```

... or by using the supplied typedefs:

```
BaryCoordSystem2D bc(corner_pointer);
```

This file also contains the small function called `triangle_area_signed()`.

Definition at line 68 of file `BaryCoordSystem.h`.

### 29.17.2 Constructor & Destructor Documentation

29.17.2.1 `template<int Dim> Go::BaryCoordSystem< Dim >::BaryCoordSystem ( ) [inline]`

Empty default constructor.

Definition at line 71 of file `BaryCoordSystem.h`.

29.17.2.2 `template<int Dim> Go::BaryCoordSystem< Dim >::BaryCoordSystem ( const Array< double, Dim > * corners ) [inline]`

Constructor. Takes an array of points that will become the corners of the coordinate simplex.

Definition at line 75 of file `BaryCoordSystem.h`.

### 29.17.3 Member Function Documentation

29.17.3.1 `template<int Dim> template<class T > Array< T, Dim > Go::BaryCoordSystem< Dim >::baryToCart ( const Array< T, Dim+1 > & bary_pt ) const [inline]`

Input is a barycentric point, output is the corresponding cartesian point.

Definition at line 135 of file `BaryCoordSystem.h`.

29.17.3.2 `template<int Dim> template<class T > Array< T, Dim+1 > Go::BaryCoordSystem< Dim >::cartToBary ( const Array< T, Dim > & cart_pt ) const [inline]`

Input is a cartesian point, output is the corresponding barycentric point.

Definition at line 148 of file `BaryCoordSystem.h`.

29.17.3.3 `template<int Dim> const Array<double,Dim>& Go::BaryCoordSystem< Dim >::corner ( int i ) const`  
`[inline]`

Returns the corner points.

Definition at line 89 of file BaryCoordSystem.h.

29.17.3.4 `template<int Dim> void Go::BaryCoordSystem< Dim >::read ( std::istream & is )` `[inline]`

Read from input stream.

Definition at line 105 of file BaryCoordSystem.h.

29.17.3.5 `template<int Dim> void Go::BaryCoordSystem< Dim >::setCorners ( const Array< double, Dim > * corners )` `[inline]`

Set corners.

Definition at line 82 of file BaryCoordSystem.h.

29.17.3.6 `template<int Dim> void Go::BaryCoordSystem< Dim >::write ( std::ostream & os ) const` `[inline]`

Write to output stream.

Definition at line 117 of file BaryCoordSystem.h.

The documentation for this class was generated from the following file:

- [gtools-core/include/GoTools/utills/BaryCoordSystem.h](#)

## 29.18 Go::BaryCoordSystemTriangle3D Class Reference

```
#include <BaryCoordSystemTriangle3D.h>
```

### Public Member Functions

- [BaryCoordSystemTriangle3D \(\)](#)  
*Empty default constructor.*
- [BaryCoordSystemTriangle3D \(const Array< double, 3 > \\*corners\)](#)
- `template<typename T >`  
[Array< T, 3 > baryToCart \(const Array< T, 3 > &bary\\_pt\) const](#)
- `template<typename T >`  
[Array< T, 3 > cartToBary \(const Array< T, 3 > &cart\\_pt\) const](#)

### 29.18.1 Detailed Description

A barycentric coordinate system for a triangle (2-manifold) embedded in 3D. Note that this differs from what can be expressed using the [BaryCoordSystem](#) template in that for the latter, the dimension of the simplex and the dimension of the space is equal.

Definition at line 58 of file `BaryCoordSystemTriangle3D.h`.

### 29.18.2 Constructor & Destructor Documentation

29.18.2.1 `Go::BaryCoordSystemTriangle3D::BaryCoordSystemTriangle3D ( ) [inline]`

Empty default constructor.

Definition at line 61 of file `BaryCoordSystemTriangle3D.h`.

29.18.2.2 `Go::BaryCoordSystemTriangle3D::BaryCoordSystemTriangle3D ( const Array< double, 3 > * corners ) [inline]`

Constructor. Takes an array of points that will become the corners of the coordinate simplex.

Definition at line 65 of file `BaryCoordSystemTriangle3D.h`.

### 29.18.3 Member Function Documentation

29.18.3.1 `template<typename T > Array<T, 3> Go::BaryCoordSystemTriangle3D::baryToCart ( const Array< T, 3 > & bary_pt ) const [inline]`

Input is a barycentric point, output is the corresponding cartesian point.

Definition at line 76 of file `BaryCoordSystemTriangle3D.h`.

29.18.3.2 `template<typename T > Array<T, 3> Go::BaryCoordSystemTriangle3D::cartToBary ( const Array< T, 3 > & cart_pt ) const [inline]`

Input is a cartesian point, output is the corresponding barycentric point.

Definition at line 90 of file `BaryCoordSystemTriangle3D.h`.

The documentation for this class was generated from the following file:

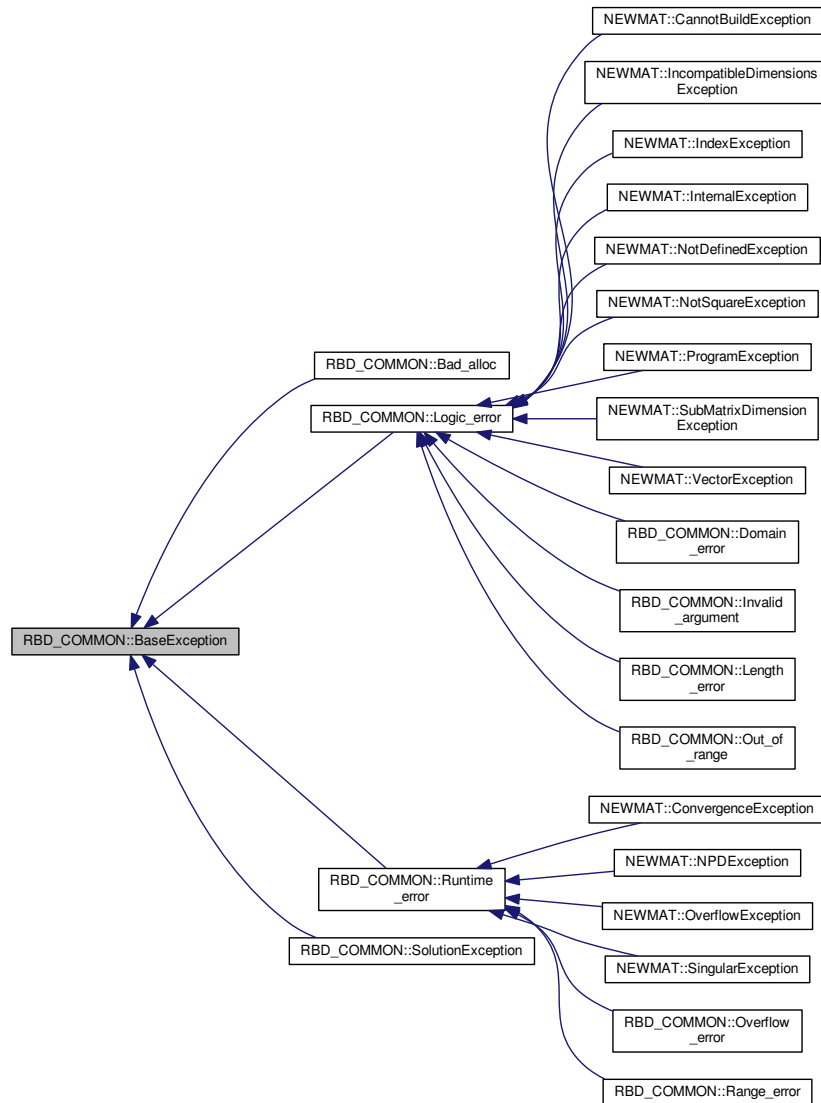
- [gotools-core/include/GoTools/utis/BaryCoordSystemTriangle3D.h](#)



## 29.19 RBD\_COMMON::BaseException Class Reference

```
#include <myexcept.h>
```

Inheritance diagram for RBD\_COMMON::BaseException:



### Static Public Member Functions

- static `const char *` `what ()`
- static `char *` `what_error`
- static `int` `SoFar`
- static `int` `LastOne`
- static `unsigned long` `Select`
- static `void` `AddMessage (const char *a_what)`
- static `void` `AddInt (int value)`
- `BaseException (const char *a_what=0)`

### 29.19.1 Detailed Description

Definition at line 81 of file myexcept.h.

### 29.19.2 Constructor & Destructor Documentation

#### 29.19.2.1 BaseException::BaseException ( const char \* *a\_what* = 0 )

Definition at line 55 of file myexcept.cpp.

### 29.19.3 Member Function Documentation

#### 29.19.3.1 void BaseException::AddInt ( int *value* ) [static]

Definition at line 88 of file myexcept.cpp.

#### 29.19.3.2 void BaseException::AddMessage ( const char \* *a\_what* ) [static]

Definition at line 73 of file myexcept.cpp.

#### 29.19.3.3 static const char\* RBD\_COMMON::BaseException::what ( ) [inline],[static]

Definition at line 93 of file myexcept.h.

### 29.19.4 Member Data Documentation

#### 29.19.4.1 int BaseException::LastOne [static],[protected]

Definition at line 86 of file myexcept.h.

#### 29.19.4.2 unsigned long BaseException::Select [static]

Definition at line 91 of file myexcept.h.

#### 29.19.4.3 int BaseException::SoFar [static],[protected]

Definition at line 85 of file myexcept.h.

29.19.4.4 `char * BaseException::what_error` [static], [protected]

Definition at line 84 of file `myexcept.h`.

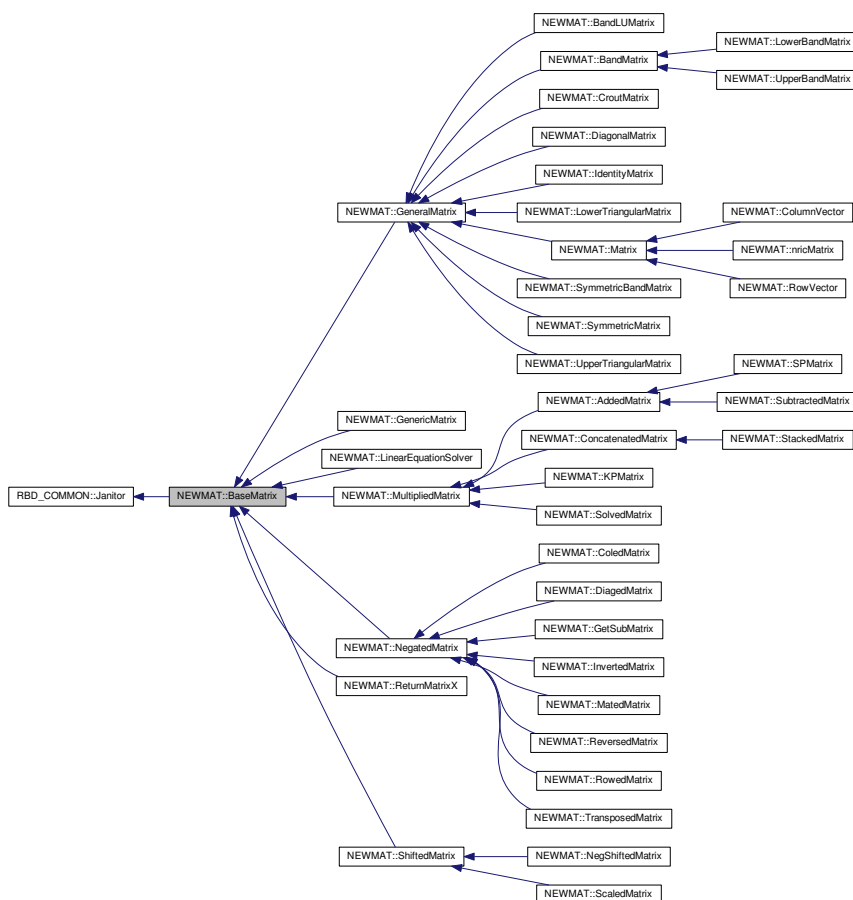
The documentation for this class was generated from the following files:

- [newmat/include/myexcept.h](#)
- [newmat/src/myexcept.cpp](#)

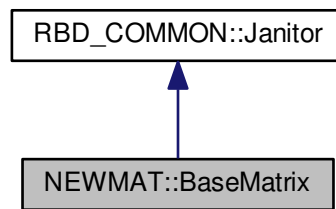
## 29.20 NEWMAT::BaseMatrix Class Reference

```
#include <newmat.h>
```

Inheritance diagram for `NEWMAT::BaseMatrix`:



Collaboration diagram for NEWMAT::BaseMatrix:



## Public Member Functions

- virtual [GeneralMatrix](#) \* [Evaluate](#) ([MatrixType](#) mt=[MatrixTypeUnSp](#))=0
- [AddedMatrix](#) [operator+](#) ([const BaseMatrix](#) &) [const](#)
- [MultipliedMatrix](#) [operator\\*](#) ([const BaseMatrix](#) &) [const](#)
- [SubtractedMatrix](#) [operator-](#) ([const BaseMatrix](#) &) [const](#)
- [ConcatenatedMatrix](#) [operator|](#) ([const BaseMatrix](#) &) [const](#)
- [StackedMatrix](#) [operator&](#) ([const BaseMatrix](#) &) [const](#)
- [ShiftedMatrix](#) [operator+](#) ([Real](#)) [const](#)
- [ScaledMatrix](#) [operator\\*](#) ([Real](#)) [const](#)
- [ScaledMatrix](#) [operator/](#) ([Real](#)) [const](#)
- [ShiftedMatrix](#) [operator-](#) ([Real](#)) [const](#)
- [TransposedMatrix](#) [t](#) () [const](#)
- [NegatedMatrix](#) [operator-](#) () [const](#)
- [ReversedMatrix](#) [Reverse](#) () [const](#)
- [InvertedMatrix](#) [i](#) () [const](#)
- [RowedMatrix](#) [AsRow](#) () [const](#)
- [ColedMatrix](#) [AsColumn](#) () [const](#)
- [DiagedMatrix](#) [AsDiagonal](#) () [const](#)
- [MatedMatrix](#) [AsMatrix](#) ([int](#), [int](#)) [const](#)
- [GetSubMatrix](#) [SubMatrix](#) ([int](#), [int](#), [int](#), [int](#)) [const](#)
- [GetSubMatrix](#) [SymSubMatrix](#) ([int](#), [int](#)) [const](#)
- [GetSubMatrix](#) [Row](#) ([int](#)) [const](#)
- [GetSubMatrix](#) [Rows](#) ([int](#), [int](#)) [const](#)
- [GetSubMatrix](#) [Column](#) ([int](#)) [const](#)
- [GetSubMatrix](#) [Columns](#) ([int](#), [int](#)) [const](#)
- [Real](#) [AsScalar](#) () [const](#)
- virtual [LogAndSign](#) [LogDeterminant](#) () [const](#)
- [Real](#) [Determinant](#) () [const](#)
- virtual [Real](#) [SumSquare](#) () [const](#)
- [Real](#) [NormFrobenius](#) () [const](#)
- virtual [Real](#) [SumAbsoluteValue](#) () [const](#)
- virtual [Real](#) [Sum](#) () [const](#)
- virtual [Real](#) [MaximumAbsoluteValue](#) () [const](#)
- virtual [Real](#) [MaximumAbsoluteValue1](#) ([int](#) &i) [const](#)
- virtual [Real](#) [MaximumAbsoluteValue2](#) ([int](#) &i, [int](#) &j) [const](#)
- virtual [Real](#) [MinimumAbsoluteValue](#) () [const](#)

- virtual [Real MinimumAbsoluteValue1](#) (int &i) const
- virtual [Real MinimumAbsoluteValue2](#) (int &i, int &j) const
- virtual [Real Maximum](#) () const
- virtual [Real Maximum1](#) (int &i) const
- virtual [Real Maximum2](#) (int &i, int &j) const
- virtual [Real Minimum](#) () const
- virtual [Real Minimum1](#) (int &i) const
- virtual [Real Minimum2](#) (int &i, int &j) const
- virtual [Real Trace](#) () const
- [Real Norm1](#) () const
- [Real NormInfinity](#) () const
- virtual [MatrixBandWidth BandWidth](#) () const
- virtual void [CleanUp](#) ()
- void [IEQND](#) () const

### Protected Member Functions

- virtual int [search](#) (const [BaseMatrix](#) \*) const =0

### Friends

- class [GeneralMatrix](#)
- class [Matrix](#)
- class [nricMatrix](#)
- class [RowVector](#)
- class [ColumnVector](#)
- class [SymmetricMatrix](#)
- class [UpperTriangularMatrix](#)
- class [LowerTriangularMatrix](#)
- class [DiagonalMatrix](#)
- class [CroutMatrix](#)
- class [BandMatrix](#)
- class [LowerBandMatrix](#)
- class [UpperBandMatrix](#)
- class [SymmetricBandMatrix](#)
- class [AddedMatrix](#)
- class [MultipliedMatrix](#)
- class [SubtractedMatrix](#)
- class [SPMatrix](#)
- class [KPMatrix](#)
- class [ConcatenatedMatrix](#)
- class [StackedMatrix](#)
- class [SolvedMatrix](#)
- class [ShiftedMatrix](#)
- class [NegShiftedMatrix](#)
- class [ScaledMatrix](#)
- class [TransposedMatrix](#)
- class [ReversedMatrix](#)
- class [NegatedMatrix](#)
- class [InvertedMatrix](#)
- class [RowedMatrix](#)
- class [ColedMatrix](#)
- class [DiagedMatrix](#)
- class [MatedMatrix](#)
- class [GetSubMatrix](#)
- class [ReturnMatrixX](#)
- class [LinearEquationSolver](#)
- class [GenericMatrix](#)

### 29.20.1 Detailed Description

Definition at line 275 of file newmat.h.

### 29.20.2 Member Function Documentation

#### 29.20.2.1 ColedMatrix BaseMatrix::AsColumn ( ) const

Definition at line 500 of file newmat6.cpp.

#### 29.20.2.2 DiagedMatrix BaseMatrix::AsDiagonal ( ) const

Definition at line 503 of file newmat6.cpp.

#### 29.20.2.3 MatedMatrix BaseMatrix::AsMatrix ( int nrx, int ncx ) const

Definition at line 506 of file newmat6.cpp.

#### 29.20.2.4 RowedMatrix BaseMatrix::AsRow ( ) const

Definition at line 497 of file newmat6.cpp.

#### 29.20.2.5 Real BaseMatrix::AsScalar ( ) const

Definition at line 254 of file newmat6.cpp.

#### 29.20.2.6 MatrixBandWidth BaseMatrix::BandWidth ( ) const [virtual]

Reimplemented in [NEWMAT::GetSubMatrix](#), [NEWMAT::ReturnMatrixX](#), [NEWMAT::MatedMatrix](#), [NEWMAT::DiagedMatrix](#), [NEWMAT::ColedMatrix](#), [NEWMAT::RowedMatrix](#), [NEWMAT::InvertedMatrix](#), [NEWMAT::TransposedMatrix](#), [NEWMAT::NegatedMatrix](#), [NEWMAT::ScaledMatrix](#), [NEWMAT::SolvedMatrix](#), [NEWMAT::ConcatenatedMatrix](#), [NEWMAT::KPMatrix](#), [NEWMAT::SPMatrix](#), [NEWMAT::AddedMatrix](#), [NEWMAT::MultipliedMatrix](#), [NEWMAT::GenericMatrix](#), [NEWMAT::IdentityMatrix](#), [NEWMAT::SymmetricBandMatrix](#), [NEWMAT::BandMatrix](#), [NEWMAT::DiagonalMatrix](#), [NEWMAT::LowerTriangularMatrix](#), and [NEWMAT::UpperTriangularMatrix](#).

Definition at line 402 of file newmat4.cpp.

#### 29.20.2.7 virtual void NEWMAT::BaseMatrix::Cleanup ( ) [inline],[virtual]

Reimplemented from [RBD\\_COMMON::Janitor](#).

Reimplemented in [NEWMAT::LinearEquationSolver](#), [NEWMAT::GenericMatrix](#), [NEWMAT::BandLUMatrix](#), [NEWMAT::CroutMatrix](#), [NEWMAT::ColumnVector](#), [NEWMAT::RowVector](#), [NEWMAT::nricMatrix](#), and [NEWMAT::GeneralMatrix](#).

Definition at line 364 of file newmat.h.

**29.20.2.8** `GetSubMatrix` `BaseMatrix::Column ( int first_col ) const`

Definition at line 108 of file `submat.cpp`.

**29.20.2.9** `GetSubMatrix` `BaseMatrix::Columns ( int first_col, int last_col ) const`

Definition at line 127 of file `submat.cpp`.

**29.20.2.10** `Real` `BaseMatrix::Determinant ( ) const`

Definition at line 709 of file `newmat8.cpp`.

**29.20.2.11** `virtual GeneralMatrix*` `NEWMAT::BaseMatrix::Evaluate ( MatrixType mt = MatrixTypeUnSp )` `[pure virtual]`

Implemented in `NEWMAT::LinearEquationSolver`, `NEWMAT::GetSubMatrix`, `NEWMAT::ReturnMatrixX`, `NEWMAT::MatedMatrix`, `NEWMAT::DiagedMatrix`, `NEWMAT::ColedMatrix`, `NEWMAT::RowedMatrix`, `NEWMAT::InvertedMatrix`, `NEWMAT::ReversedMatrix`, `NEWMAT::TransposedMatrix`, `NEWMAT::NegatedMatrix`, `NEWMAT::ScaledMatrix`, `NEWMAT::NegShiftedMatrix`, `NEWMAT::ShiftedMatrix`, `NEWMAT::SubtractedMatrix`, `NEWMAT::SolvedMatrix`, `NEWMAT::StackedMatrix`, `NEWMAT::ConcatenatedMatrix`, `NEWMAT::KPMatrix`, `NEWMAT::SPMatrix`, `NEWMAT::AddedMatrix`, `NEWMAT::MultipliedMatrix`, `NEWMAT::GenericMatrix`, and `NEWMAT::GeneralMatrix`.

**29.20.2.12** `InvertedMatrix` `BaseMatrix::i ( ) const`

Definition at line 493 of file `newmat6.cpp`.

**29.20.2.13** `void` `BaseMatrix::IEQND ( ) const`

Definition at line 296 of file `newmatex.cpp`.

**29.20.2.14** `LogAndSign` `BaseMatrix::LogDeterminant ( ) const` `[virtual]`

Reimplemented in `NEWMAT::IdentityMatrix`, `NEWMAT::BandLUMatrix`, `NEWMAT::SymmetricBandMatrix`, `NEWMAT::LowerBandMatrix`, `NEWMAT::UpperBandMatrix`, `NEWMAT::BandMatrix`, `NEWMAT::CroutMatrix`, `NEWMAT::DiagonalMatrix`, `NEWMAT::LowerTriangularMatrix`, `NEWMAT::UpperTriangularMatrix`, and `NEWMAT::GeneralMatrix`.

Definition at line 680 of file `newmat8.cpp`.

**29.20.2.15** `Real` `BaseMatrix::Maximum ( ) const` `[virtual]`

Reimplemented in `NEWMAT::SymmetricBandMatrix`, `NEWMAT::BandMatrix`, and `NEWMAT::GeneralMatrix`.

Definition at line 488 of file `newmat8.cpp`.

29.20.2.16 **Real BaseMatrix::Maximum1 ( int & i ) const** [virtual]

Reimplemented in [NEWMAT::GeneralMatrix](#).

Definition at line 494 of file newmat8.cpp.

29.20.2.17 **Real BaseMatrix::Maximum2 ( int & i, int & j ) const** [virtual]

Reimplemented in [NEWMAT::Matrix](#), and [NEWMAT::GeneralMatrix](#).

Definition at line 500 of file newmat8.cpp.

29.20.2.18 **Real BaseMatrix::MaximumAbsoluteValue ( ) const** [virtual]

Reimplemented in [NEWMAT::SymmetricBandMatrix](#), [NEWMAT::BandMatrix](#), and [NEWMAT::GeneralMatrix](#).

Definition at line 452 of file newmat8.cpp.

29.20.2.19 **Real BaseMatrix::MaximumAbsoluteValue1 ( int & i ) const** [virtual]

Reimplemented in [NEWMAT::GeneralMatrix](#).

Definition at line 458 of file newmat8.cpp.

29.20.2.20 **Real BaseMatrix::MaximumAbsoluteValue2 ( int & i, int & j ) const** [virtual]

Reimplemented in [NEWMAT::Matrix](#), and [NEWMAT::GeneralMatrix](#).

Definition at line 464 of file newmat8.cpp.

29.20.2.21 **Real BaseMatrix::Minimum ( ) const** [virtual]

Reimplemented in [NEWMAT::SymmetricBandMatrix](#), [NEWMAT::BandMatrix](#), and [NEWMAT::GeneralMatrix](#).

Definition at line 506 of file newmat8.cpp.

29.20.2.22 **Real BaseMatrix::Minimum1 ( int & i ) const** [virtual]

Reimplemented in [NEWMAT::GeneralMatrix](#).

Definition at line 512 of file newmat8.cpp.

29.20.2.23 **Real BaseMatrix::Minimum2 ( int & i, int & j ) const** [virtual]

Reimplemented in [NEWMAT::Matrix](#), and [NEWMAT::GeneralMatrix](#).

Definition at line 518 of file newmat8.cpp.



29.20.2.24 **Real** BaseMatrix::MinimumAbsoluteValue ( ) const [virtual]

Reimplemented in [NEWMAT::SymmetricBandMatrix](#), [NEWMAT::BandMatrix](#), and [NEWMAT::GeneralMatrix](#).

Definition at line 470 of file newmat8.cpp.

29.20.2.25 **Real** BaseMatrix::MinimumAbsoluteValue1 ( int & i ) const [virtual]

Reimplemented in [NEWMAT::GeneralMatrix](#).

Definition at line 476 of file newmat8.cpp.

29.20.2.26 **Real** BaseMatrix::MinimumAbsoluteValue2 ( int & i, int & j ) const [virtual]

Reimplemented in [NEWMAT::Matrix](#), and [NEWMAT::GeneralMatrix](#).

Definition at line 482 of file newmat8.cpp.

29.20.2.27 **Real** BaseMatrix::Norm1 ( ) const

Definition at line 770 of file newmat7.cpp.

29.20.2.28 **Real** BaseMatrix::NormFrobenius ( ) const

Definition at line 437 of file newmat8.cpp.

29.20.2.29 **Real** BaseMatrix::NormInfinity ( ) const

Definition at line 782 of file newmat7.cpp.

29.20.2.30 **StackedMatrix** BaseMatrix::operator& ( const BaseMatrix & bm ) const

Definition at line 460 of file newmat6.cpp.

29.20.2.31 **MultipliedMatrix** NEWMAT::BaseMatrix::operator\* ( const BaseMatrix & ) const

29.20.2.32 **ScaledMatrix** NEWMAT::BaseMatrix::operator\* ( Real ) const

29.20.2.33 **AddedMatrix** NEWMAT::BaseMatrix::operator+ ( const BaseMatrix & ) const

29.20.2.34 **ShiftedMatrix** NEWMAT::BaseMatrix::operator+ ( Real ) const

29.20.2.35 **SubtractedMatrix** NEWMAT::BaseMatrix::operator- ( const BaseMatrix & ) const

29.20.2.36 **ShiftedMatrix** NEWMAT::BaseMatrix::operator- ( Real ) const

29.20.2.37 **NegatedMatrix** NEWMAT::BaseMatrix::operator- ( ) const

29.20.2.38 **ScaledMatrix** BaseMatrix::operator/ ( Real f ) const

Definition at line 478 of file newmat6.cpp.

29.20.2.39 **ConcatenatedMatrix** `BaseMatrix::operator| ( const BaseMatrix & bm ) const`

Definition at line 457 of file newmat6.cpp.

29.20.2.40 **ReversedMatrix** `BaseMatrix::Reverse ( ) const`

Definition at line 490 of file newmat6.cpp.

29.20.2.41 **GetSubMatrix** `BaseMatrix::Row ( int first_row ) const`

Definition at line 69 of file submat.cpp.

29.20.2.42 **GetSubMatrix** `BaseMatrix::Rows ( int first_row, int last_row ) const`

Definition at line 88 of file submat.cpp.

29.20.2.43 **virtual int** `NEWMAT::BaseMatrix::search ( const BaseMatrix * ) const` [protected], [pure virtual]

Implemented in [NEWMAT::NegatedMatrix](#), [NEWMAT::ShiftedMatrix](#), [NEWMAT::MultipliedMatrix](#), and [NEWMAT::GeneralMatrix](#).

29.20.2.44 **GetSubMatrix** `BaseMatrix::SubMatrix ( int first_row, int last_row, int first_col, int last_col ) const`

Definition at line 27 of file submat.cpp.

29.20.2.45 **Real** `BaseMatrix::Sum ( ) const` [virtual]

Reimplemented in [NEWMAT::IdentityMatrix](#), [NEWMAT::SymmetricBandMatrix](#), [NEWMAT::BandMatrix](#), [NEWMAT::SymmetricMatrix](#), and [NEWMAT::GeneralMatrix](#).

Definition at line 446 of file newmat8.cpp.

29.20.2.46 **Real** `BaseMatrix::SumAbsoluteValue ( ) const` [virtual]

Reimplemented in [NEWMAT::IdentityMatrix](#), [NEWMAT::SymmetricBandMatrix](#), [NEWMAT::BandMatrix](#), [NEWMAT::SymmetricMatrix](#), and [NEWMAT::GeneralMatrix](#).

Definition at line 440 of file newmat8.cpp.

29.20.2.47 **Real** `BaseMatrix::SumSquare ( ) const` [virtual]

Reimplemented in [NEWMAT::IdentityMatrix](#), [NEWMAT::SymmetricBandMatrix](#), [NEWMAT::BandMatrix](#), [NEWMAT::SymmetricMatrix](#), and [NEWMAT::GeneralMatrix](#).

Definition at line 431 of file newmat8.cpp.

29.20.2.48 **GetSubMatrix** BaseMatrix::SymSubMatrix ( int *first\_row*, int *last\_row* ) const

Definition at line 49 of file submat.cpp.

29.20.2.49 **TransposedMatrix** BaseMatrix::t ( ) const

Definition at line 484 of file newmat6.cpp.

29.20.2.50 **Real** BaseMatrix::Trace ( ) const [virtual]

Reimplemented in [NEWMAT::IdentityMatrix](#), [NEWMAT::SymmetricBandMatrix](#), [NEWMAT::BandMatrix](#), [NEWMAT::DiagonalMatrix](#), [NEWMAT::LowerTriangularMatrix](#), [NEWMAT::UpperTriangularMatrix](#), [NEWMAT::SymmetricMatrix](#), and [NEWMAT::Matrix](#).

Definition at line 607 of file newmat8.cpp.

### 29.20.3 Friends And Related Function Documentation

29.20.3.1 friend class **AddedMatrix** [friend]

Definition at line 386 of file newmat.h.

29.20.3.2 friend class **BandMatrix** [friend]

Definition at line 382 of file newmat.h.

29.20.3.3 friend class **ColedMatrix** [friend]

Definition at line 402 of file newmat.h.

29.20.3.4 friend class **ColumnVector** [friend]

Definition at line 376 of file newmat.h.

29.20.3.5 friend class **ConcatenatedMatrix** [friend]

Definition at line 391 of file newmat.h.

29.20.3.6 friend class **CroutMatrix** [friend]

Definition at line 381 of file newmat.h.

29.20.3.7 friend class **DiagedMatrix** [[friend](#)]

Definition at line 403 of file newmat.h.

29.20.3.8 friend class **DiagonalMatrix** [[friend](#)]

Definition at line 380 of file newmat.h.

29.20.3.9 friend class **GeneralMatrix** [[friend](#)]

Definition at line 372 of file newmat.h.

29.20.3.10 friend class **GenericMatrix** [[friend](#)]

Definition at line 408 of file newmat.h.

29.20.3.11 friend class **GetSubMatrix** [[friend](#)]

Definition at line 405 of file newmat.h.

29.20.3.12 friend class **InvertedMatrix** [[friend](#)]

Definition at line 400 of file newmat.h.

29.20.3.13 friend class **KPMatrix** [[friend](#)]

Definition at line 390 of file newmat.h.

29.20.3.14 friend class **LinearEquationSolver** [[friend](#)]

Definition at line 407 of file newmat.h.

29.20.3.15 friend class **LowerBandMatrix** [[friend](#)]

Definition at line 383 of file newmat.h.

29.20.3.16 friend class **LowerTriangularMatrix** [[friend](#)]

Definition at line 379 of file newmat.h.

29.20.3.17 friend class **MatedMatrix** [friend]

Definition at line 404 of file newmat.h.

29.20.3.18 friend class **Matrix** [friend]

Definition at line 373 of file newmat.h.

29.20.3.19 friend class **MultipliedMatrix** [friend]

Definition at line 387 of file newmat.h.

29.20.3.20 friend class **NegatedMatrix** [friend]

Definition at line 399 of file newmat.h.

29.20.3.21 friend class **NegShiftedMatrix** [friend]

Definition at line 395 of file newmat.h.

29.20.3.22 friend class **nrncMatrix** [friend]

Definition at line 374 of file newmat.h.

29.20.3.23 friend class **ReturnMatrixX** [friend]

Definition at line 406 of file newmat.h.

29.20.3.24 friend class **ReversedMatrix** [friend]

Definition at line 398 of file newmat.h.

29.20.3.25 friend class **RowedMatrix** [friend]

Definition at line 401 of file newmat.h.

29.20.3.26 friend class **RowVector** [friend]

Definition at line 375 of file newmat.h.

29.20.3.27 friend class **ScaledMatrix** [friend]

Definition at line 396 of file newmat.h.

29.20.3.28 friend class **ShiftedMatrix** [friend]

Definition at line 394 of file newmat.h.

29.20.3.29 friend class **SolvedMatrix** [friend]

Definition at line 393 of file newmat.h.

29.20.3.30 friend class **SPMatrix** [friend]

Definition at line 389 of file newmat.h.

29.20.3.31 friend class **StackedMatrix** [friend]

Definition at line 392 of file newmat.h.

29.20.3.32 friend class **SubtractedMatrix** [friend]

Definition at line 388 of file newmat.h.

29.20.3.33 friend class **SymmetricBandMatrix** [friend]

Definition at line 385 of file newmat.h.

29.20.3.34 friend class **SymmetricMatrix** [friend]

Definition at line 377 of file newmat.h.

29.20.3.35 friend class **TransposedMatrix** [friend]

Definition at line 397 of file newmat.h.

29.20.3.36 friend class **UpperBandMatrix** [friend]

Definition at line 384 of file newmat.h.

29.20.3.37 friend class **UpperTriangularMatrix** [friend]

Definition at line 378 of file newmat.h.

The documentation for this class was generated from the following files:

- [newmat/include/newmat.h](#)
- [newmat/src/newmat4.cpp](#)
- [newmat/src/newmat6.cpp](#)
- [newmat/src/newmat7.cpp](#)
- [newmat/src/newmat8.cpp](#)
- [newmat/src/newmatex.cpp](#)
- [newmat/src/submat.cpp](#)

## 29.21 Go::BaseSurface Class Reference

```
#include <EvalOffsetSurface.h>
```

### 29.21.1 Detailed Description

Definition at line 54 of file EvalOffsetSurface.h.

The documentation for this class was generated from the following file:

- [compositemodel/include/GoTools/compositemodel/EvalOffsetSurface.h](#)

## 29.22 Go::BasisDerivs Struct Reference

```
#include <SplineVolume.h>
```

### Public Member Functions

- void [prepareDerivs](#) (double u, double v, double w, int idx\_u, int idx\_v, int idx\_w, int size)  
*Constructor.*

### Public Attributes

- [double param](#) [3]  
*Parameter tripple in which the basis functions are evaluated.*
- int [left\\_idx](#) [3]
- [std::vector< double > basisValues](#)  
*The value of all basis functions, size equal to (degree\_u+1)\*(degree\_v+1)\*(degree\_w+1)*
- [std::vector< double > basisDerivs\\_u](#)  
*the derivative of all basis functions in u direction, same size as previous*
- [std::vector< double > basisDerivs\\_v](#)  
*the derivative of all basis functions in v direction, same size as previous*
- [std::vector< double > basisDerivs\\_w](#)  
*the derivative of all basis functions in w direction, same size as previous*

### 29.22.1 Detailed Description

Structure for storage of results of grid evaluation of the basis function of s spline volume. Evaluation of position and first derivatives in one parameter value

Definition at line 87 of file SplineVolume.h.

### 29.22.2 Member Function Documentation

29.22.2.1 `void Go::BasisDerivs::prepareDerivs ( double u, double v, double w, int idx_u, int idx_v, int idx_w, int size )`  
[inline]

Constructor.

Definition at line 105 of file SplineVolume.h.

### 29.22.3 Member Data Documentation

29.22.3.1 `std::vector< double > Go::BasisDerivs::basisDerivs_u`

the derivative of all basis functions in u direction, same size as previous

Definition at line 98 of file SplineVolume.h.

29.22.3.2 `std::vector< double > Go::BasisDerivs::basisDerivs_v`

the derivative of all basis functions in v direction, same size as previous

Definition at line 100 of file SplineVolume.h.

29.22.3.3 `std::vector< double > Go::BasisDerivs::basisDerivs_w`

the derivative of all basis functions in w direction, same size as previous

Definition at line 102 of file SplineVolume.h.

29.22.3.4 `std::vector< double > Go::BasisDerivs::basisValues`

The value of all basis functions, size equal to  $(\text{degree\_u}+1)*(\text{degree\_v}+1)*(\text{degree\_w}+1)$

Definition at line 96 of file SplineVolume.h.

29.22.3.5 `int Go::BasisDerivs::left_idx[3]`

Index of the knot interval where the parameter value is situated for all parameter directions. The indices of the non-zero basis functions are `left_idx[i]-order[i]+1, ..., left_idx[i]` for  $i=0,1,2$

Definition at line 94 of file SplineVolume.h.



## 29.22.3.6 double Go::BasisDerivs::param[3]

Parameter tripple in which the basis functions are evaluated.

Definition at line 90 of file SplineVolume.h.

The documentation for this struct was generated from the following file:

- [trivariate/include/GoTools/trivariate/SplineVolume.h](#)

## 29.23 Go::BasisDerivs2 Struct Reference

```
#include <SplineVolume.h>
```

### Public Member Functions

- void [prepareDerivs](#) (double u, double v, double w, int idx\_u, int idx\_v, int idx\_w, int size)  
*Constructor.*

### Public Attributes

- double [param](#) [3]  
*Parameter tripple in which the basis functions are evaluated.*
- int [left\\_idx](#) [3]
- std::vector< double > [basisValues](#)  
*The value of all basis functions, size equal to (degree\_u+1)\*(degree\_v+1)\*(degree\_w+1)*
- std::vector< double > [basisDerivs\\_u](#)  
*the derivative of all basis functions in u direction, same size as previous*
- std::vector< double > [basisDerivs\\_v](#)  
*the derivative of all basis functions in v direction, same size as previous*
- std::vector< double > [basisDerivs\\_w](#)  
*the derivative of all basis functions in w direction, same size as previous*
- std::vector< double > [basisDerivs\\_uu](#)  
*the second derivative of all basis functions twice in u direction, same size as previous*
- std::vector< double > [basisDerivs\\_uv](#)  
*the second derivative of all basis functions in u and v direction, same size as previous*
- std::vector< double > [basisDerivs\\_uw](#)  
*the second derivative of all basis functions in u and w direction, same size as previous*
- std::vector< double > [basisDerivs\\_vv](#)  
*the second derivative of all basis functions twice in v direction, same size as previous*
- std::vector< double > [basisDerivs\\_vw](#)  
*the second derivative of all basis functions in v and w direction, same size as previous*
- std::vector< double > [basisDerivs\\_ww](#)  
*the second derivative of all basis functions twice in w direction, same size as previous*

### 29.23.1 Detailed Description

Structure for storage of results of grid evaluation of the basis function of a spline volume. Evaluation of position, first and second derivatives in one parameter value

Definition at line 124 of file SplineVolume.h.

### 29.23.2 Member Function Documentation

29.23.2.1 `void Go::BasisDerivs2::prepareDerivs ( double u, double v, double w, int idx_u, int idx_v, int idx_w, int size )`  
[inline]

Constructor.

Definition at line 156 of file SplineVolume.h.

### 29.23.3 Member Data Documentation

29.23.3.1 `std::vector< double > Go::BasisDerivs2::basisDerivs_u`

the derivative of all basis functions in u direction, same size as previous

Definition at line 136 of file SplineVolume.h.

29.23.3.2 `std::vector< double > Go::BasisDerivs2::basisDerivs_uu`

the second derivative of all basis functions twice in u direction, same size as previous

Definition at line 143 of file SplineVolume.h.

29.23.3.3 `std::vector< double > Go::BasisDerivs2::basisDerivs_uv`

the second derivative of all basis functions in u and v direction, same size as previous

Definition at line 145 of file SplineVolume.h.

29.23.3.4 `std::vector< double > Go::BasisDerivs2::basisDerivs_uw`

the second derivative of all basis functions in u and w direction, same size as previous

Definition at line 147 of file SplineVolume.h.

29.23.3.5 `std::vector< double > Go::BasisDerivs2::basisDerivs_v`

the derivative of all basis functions in v direction, same size as previous

Definition at line 138 of file SplineVolume.h.

**29.23.3.6** `std::vector< double > Go::BasisDerivs2::basisDerivs_vv`

the second derivative of all basis functions twice in v direction, same size as previous

Definition at line 149 of file SplineVolume.h.

**29.23.3.7** `std::vector< double > Go::BasisDerivs2::basisDerivs_vw`

the second derivative of all basis functions in v and w direction, same size as previous

Definition at line 151 of file SplineVolume.h.

**29.23.3.8** `std::vector< double > Go::BasisDerivs2::basisDerivs_w`

the derivative of all basis functions in w direction, same size as previous

Definition at line 140 of file SplineVolume.h.

**29.23.3.9** `std::vector< double > Go::BasisDerivs2::basisDerivs_ww`

the second derivative of all basis functions twice in w direction, same size as previous

Definition at line 153 of file SplineVolume.h.

**29.23.3.10** `std::vector< double > Go::BasisDerivs2::basisValues`

The value of all basis functions, size equal to  $(\text{degree\_u}+1)*(\text{degree\_v}+1)*(\text{degree\_w}+1)$

Definition at line 133 of file SplineVolume.h.

**29.23.3.11** `int Go::BasisDerivs2::left_idx[3]`

Index of the knot interval where the parameter value is situated for all parameter directions. The indices of the non-zero basis functions are `left_idx[i]-order[i]+1, ..., left_idx[i]` for `i=0,1,2`

Definition at line 131 of file SplineVolume.h.

**29.23.3.12** `double Go::BasisDerivs2::param[3]`

Parameter tripple in which the basis functions are evaluated.

Definition at line 127 of file SplineVolume.h.

The documentation for this struct was generated from the following file:

- [trivariate/include/GoTools/trivariate/SplineVolume.h](#)

## 29.24 Go::BasisDerivsSf Struct Reference

```
#include <SplineSurface.h>
```

### Public Member Functions

- void [prepareDerivs](#) (double u, double v, int idx\_u, int idx\_v, int size)

### Public Attributes

- double [param](#) [2]  
*Parameter double in which the basis functions are evaluated.*
- int [left\\_idx](#) [2]
- std::vector< double > [basisValues](#)  
*The value of all basis functions, size equal to (degree\_u+1)\*(degree\_v+1)*
- std::vector< double > [basisDerivs\\_u](#)  
*the derivative of all basis functions in u direction, same size as previous*
- std::vector< double > [basisDerivs\\_v](#)  
*the derivative of all basis functions in v direction, same size as previous*

### 29.24.1 Detailed Description

Structure for storage of results of grid evaluation of the basis function of a spline surface. Position and first derivatives

Definition at line 83 of file SplineSurface.h.

### 29.24.2 Member Function Documentation

29.24.2.1 void [Go::BasisDerivsSf::prepareDerivs](#) ( double u, double v, int idx\_u, int idx\_v, int size ) `[inline]`

Definition at line 98 of file SplineSurface.h.

### 29.24.3 Member Data Documentation

29.24.3.1 std::vector< double > [Go::BasisDerivsSf::basisDerivs\\_u](#)

the derivative of all basis functions in u direction, same size as previous

Definition at line 94 of file SplineSurface.h.

29.24.3.2 std::vector< double > [Go::BasisDerivsSf::basisDerivs\\_v](#)

the derivative of all basis functions in v direction, same size as previous

Definition at line 96 of file SplineSurface.h.

### 29.24.3.3 `std::vector< double > Go::BasisDerivsSf::basisValues`

The value of all basis functions, size equal to  $(\text{degree\_u}+1)*(\text{degree\_v}+1)$

Definition at line 92 of file `SplineSurface.h`.

### 29.24.3.4 `int Go::BasisDerivsSf::left_idx[2]`

Index of the knot interval where the parameter value is situated for all parameter directions. The indices of the non-zero basis functions are `left_idx[i]-order[i]+1, ..., left_idx[i]` for  $i=0,1$

Definition at line 90 of file `SplineSurface.h`.

### 29.24.3.5 `double Go::BasisDerivsSf::param[2]`

Parameter double in which the basis functions are evaluated.

Definition at line 86 of file `SplineSurface.h`.

The documentation for this struct was generated from the following file:

- [gotools-core/include/GoTools/geometry/SplineSurface.h](#)

## 29.25 Go::BasisDerivsSf2 Struct Reference

```
#include <SplineSurface.h>
```

### Public Member Functions

- void [prepareDerivs](#) (`double` u, `double` v, int idx\_u, int idx\_v, int size)

### Public Attributes

- `double` [param](#) [2]  
*Parameter double in which the basis functions are evaluated.*
- int [left\\_idx](#) [2]
- `std::vector< double >` [basisValues](#)  
*The value of all basis functions, size equal to  $(\text{degree\_u}+1)*(\text{degree\_v}+1)$*
- `std::vector< double >` [basisDerivs\\_u](#)  
*the derivative of all basis functions in u direction, same size as previous*
- `std::vector< double >` [basisDerivs\\_v](#)  
*the derivative of all basis functions in v direction, same size as previous*
- `std::vector< double >` [basisDerivs\\_uu](#)  
*the second derivative of all basis functions twice in u direction, same size as previous*
- `std::vector< double >` [basisDerivs\\_uv](#)  
*the second derivative of all basis functions in u and v direction, same size as previous*
- `std::vector< double >` [basisDerivs\\_vv](#)  
*the second derivative of all basis functions twice in v direction, same size as previous*

### 29.25.1 Detailed Description

Structure for storage of results of grid evaluation of the basis function of a spline surface. Position, first and second derivatives

Definition at line 112 of file SplineSurface.h.

### 29.25.2 Member Function Documentation

29.25.2.1 void Go::BasisDerivsSf2::prepareDerivs ( double u, double v, int idx\_u, int idx\_v, int size ) [inline]

Definition at line 135 of file SplineSurface.h.

### 29.25.3 Member Data Documentation

29.25.3.1 std::vector< double > Go::BasisDerivsSf2::basisDerivs\_u

the derivative of all basis functions in u direction, same size as previous

Definition at line 124 of file SplineSurface.h.

29.25.3.2 std::vector< double > Go::BasisDerivsSf2::basisDerivs\_uu

the second derivative of all basis functions twice in u direction, same size as previous

Definition at line 129 of file SplineSurface.h.

29.25.3.3 std::vector< double > Go::BasisDerivsSf2::basisDerivs\_uv

the second derivative of all basis functions in u and v direction, same size as previous

Definition at line 131 of file SplineSurface.h.

29.25.3.4 std::vector< double > Go::BasisDerivsSf2::basisDerivs\_v

the derivative of all basis functions in v direction, same size as previous

Definition at line 126 of file SplineSurface.h.

29.25.3.5 std::vector< double > Go::BasisDerivsSf2::basisDerivs\_vv

the second derivative of all basis functions twice in v direction, same size as previous

Definition at line 133 of file SplineSurface.h.

### 29.25.3.6 `std::vector< double > Go::BasisDerivsSf2::basisValues`

The value of all basis functions, size equal to  $(\text{degree\_u}+1)*(\text{degree\_v}+1)$

Definition at line 121 of file `SplineSurface.h`.

### 29.25.3.7 `int Go::BasisDerivsSf2::left_idx[2]`

Index of the knot interval where the parameter value is situated for all parameter directions. The indices of the non-zero basis functions are `left_idx[i]-order[i]+1, ..., left_idx[i]` for  $i=0,1$

Definition at line 119 of file `SplineSurface.h`.

### 29.25.3.8 `double Go::BasisDerivsSf2::param[2]`

Parameter double in which the basis functions are evaluated.

Definition at line 115 of file `SplineSurface.h`.

The documentation for this struct was generated from the following file:

- [gotools-core/include/GoTools/geometry/SplineSurface.h](#)

## 29.26 Go::BasisPts Struct Reference

```
#include <SplineVolume.h>
```

### Public Member Functions

- void [preparePts](#) ([double](#) u, [double](#) v, [double](#) w, int idx\_u, int idx\_v, int idx\_w, int size)  
*Constructor.*

### Public Attributes

- [double](#) [param](#) [3]  
*Parameter tripple in which the basis functions are evaluated.*
- int [left\\_idx](#) [3]
- `std::vector< double >` [basisValues](#)  
*The value of all basis functions, size equal to  $(\text{degree\_u}+1)*(\text{degree\_v}+1)*(\text{degree\_w}+1)$*

### 29.26.1 Detailed Description

Structure for storage of results of grid evaluation of the basis function of a spline volume. Positional evaluation information in one parameter value

Definition at line 60 of file `SplineVolume.h`.

## 29.26.2 Member Function Documentation

29.26.2.1 void Go::BasisPts::preparePts ( double *u*, double *v*, double *w*, int *idx\_u*, int *idx\_v*, int *idx\_w*, int *size* )  
[inline]

Constructor.

Definition at line 72 of file SplineVolume.h.

## 29.26.3 Member Data Documentation

29.26.3.1 std::vector< double > Go::BasisPts::basisValues

The value of all basis functions, size equal to  $(\text{degree\_u}+1)*(\text{degree\_v}+1)*(\text{degree\_w}+1)$

Definition at line 69 of file SplineVolume.h.

29.26.3.2 int Go::BasisPts::left\_idx[3]

Index of the knot interval where the parameter value is situated for all parameter directions. The indices of the non-zero basis functions are  $\text{left\_idx}[i]-\text{order}[i]+1, \dots, \text{left\_idx}[i]$  for  $i=0,1,2$

Definition at line 67 of file SplineVolume.h.

29.26.3.3 double Go::BasisPts::param[3]

Parameter tripple in which the basis functions are evaluated.

Definition at line 63 of file SplineVolume.h.

The documentation for this struct was generated from the following file:

- [trivariate/include/GoTools/trivariate/SplineVolume.h](#)

## 29.27 Go::BasisPtsSf Struct Reference

```
#include <SplineSurface.h>
```

### Public Member Functions

- void [preparePts](#) (double *u*, double *v*, int *idx\_u*, int *idx\_v*, int *size*)



## Public Attributes

- `double param` [2]  
*Parameter tripple in which the basis functions are evaluated.*
- `int left_idx` [2]
- `std::vector< double > basisValues`  
*The value of all basis functions, size equal to  $(degree\_u+1)*(degree\_v+1)*(degree\_w+1)$*

### 29.27.1 Detailed Description

Structure for storage of results of grid evaluation of the basis function of a spline surface. Positional evaluation information in one parameter value

Definition at line 60 of file `SplineSurface.h`.

### 29.27.2 Member Function Documentation

29.27.2.1 `void Go::BasisPtsSf::preparePts ( double u, double v, int idx_u, int idx_v, int size ) [inline]`

Definition at line 71 of file `SplineSurface.h`.

### 29.27.3 Member Data Documentation

29.27.3.1 `std::vector< double > Go::BasisPtsSf::basisValues`

The value of all basis functions, size equal to  $(degree\_u+1)*(degree\_v+1)*(degree\_w+1)$

Definition at line 69 of file `SplineSurface.h`.

29.27.3.2 `int Go::BasisPtsSf::left_idx[2]`

Index of the knot interval where the parameter value is situated for all parameter directions. The indices of the non-zero basis functions are `left_idx[i]-order[i]+1, ..., left_idx[i]` for  $i=0,1$

Definition at line 67 of file `SplineSurface.h`.

29.27.3.3 `double Go::BasisPtsSf::param[2]`

Parameter tripple in which the basis functions are evaluated.

Definition at line 63 of file `SplineSurface.h`.

The documentation for this struct was generated from the following file:

- `gotools-core/include/GoTools/geometry/SplineSurface.h`

## 29.28 Go::BdCondFunctor Class Reference

```
#include <BdCondFunctor.h>
```

### Public Member Functions

- [BdCondFunctor](#) ()
- virtual [~BdCondFunctor](#) ()
- virtual [Point evaluate](#) ([const Point](#) &geom\_pos)=0

*Functor evaluation in point.*

### 29.28.1 Detailed Description

Definition at line 50 of file [BdCondFunctor.h](#).

### 29.28.2 Constructor & Destructor Documentation

29.28.2.1 [Go::BdCondFunctor::BdCondFunctor](#) ( )

29.28.2.2 virtual [Go::BdCondFunctor::~~BdCondFunctor](#) ( ) [virtual]

### 29.28.3 Member Function Documentation

29.28.3.1 virtual [Point](#) [Go::BdCondFunctor::evaluate](#) ( [const Point](#) & *geom\_pos* ) [pure virtual]

Functor evaluation in point.

The documentation for this class was generated from the following file:

- [isogeometric\\_model/include/GoTools/isogeometric\\_model/BdCondFunctor.h](#)

## 29.29 Go::BernsteinMulti Class Reference

```
#include <BernsteinMulti.h>
```

## Public Member Functions

- [BernsteinMulti \(\)](#)  
*Default constructor.*
- [BernsteinMulti \(int m, int n, const std::vector< double > &coefs\)](#)
- [template<typename ForwardIterator > BernsteinMulti \(int m, int n, ForwardIterator begin, ForwardIterator end\)](#)
- [BernsteinMulti \(double coef\)](#)
- [int degreeU \(\) const](#)
- [int degreeV \(\) const](#)
- [const double & operator\[\] \(int num\) const](#)  
*Access function for coefficients.*
- [double & operator\[\] \(int num\)](#)  
*Access function for coefficients.*
- [std::vector< double >::iterator coefsBegin \(\)](#)  
*Access function for coefficients.*
- [std::vector< double >::const\\_iterator coefsBegin \(\) const](#)  
*Access function for coefficients.*
- [std::vector< double >::iterator coefsEnd \(\)](#)  
*Access function for coefficients.*
- [std::vector< double >::const\\_iterator coefsEnd \(\) const](#)  
*Access function for coefficients.*
- [double operator\(\) \(double u, double v\) const](#)
- [double operator\(\) \(const Vector2D &pt\) const](#)
- [double operator\(\) \(const std::vector< double > &pt\) const](#)
- [bool isZero \(const double eps=0.0\) const](#)
- [bool isStrictlyPositive \(const double eps=0.0\) const](#)
- [bool isStrictlyNegative \(const double eps=0.0\) const](#)
- [bool isNonNegative \(const double eps=0.0\) const](#)
- [double norm \(\) const](#)
- [void normalize \(\)](#)
- [double mean \(\) const](#)  
*Calculates the mean value of the coefficients.*
- [BernsteinMulti deriv \(int der1, int der2\) const](#)
- [BernsteinMulti detHess \(\) const](#)
- [BernsteinMulti traceHess \(\) const](#)
- [double blossom \(const std::vector< double > &uvec, const std::vector< double > &vvec\) const](#)
- [BernsteinMulti pickDomain \(double u0, double u1, double v0, double v1\) const](#)
- [BernsteinPoly pickLine \(Array< double, 2 > a, Array< double, 2 > b\) const](#)
- [void degreeElevate \(int du, int dv\)](#)
- [BernsteinMulti & operator\\*= \(const BernsteinMulti &multi\)](#)  
*Multiplication with another polynomial.*
- [BernsteinMulti & operator\\*= \(double c\)](#)  
*Multiplication with a scalar.*
- [BernsteinMulti & operator+= \(const BernsteinMulti &multi\)](#)  
*Addition with another polynomial.*
- [BernsteinMulti & operator+= \(double c\)](#)  
*Addition with a scalar.*
- [BernsteinMulti & operator-= \(const BernsteinMulti &multi\)](#)  
*Subtraction with another polynomial.*
- [BernsteinMulti & operator-= \(double c\)](#)  
*Subtraction with a scalar.*
- [BernsteinMulti & operator/= \(double c\)](#)

*Division with a scalar.*

- BernsteinPoly bindU (double u) const
- BernsteinPoly bindV (double v) const
- void read (std::istream &is)

*Read from an input stream.*

- void write (std::ostream &os) const

*Write to an output stream.*

### 29.29.1 Detailed Description

Class that implements bivariate tensor product Bernstein polynomials on the domain  $[0,1] \times [0,1]$ .

The formula for a degree (m,n) polynomial is

$$p(u, v) = \sum_{i=0}^m \sum_{j=0}^n B_i(u) B_j(v) c_{ij}$$

where the basis functions in the first parameter direction are defined by

$$B_i(u) = \binom{m}{i} u^i (1-u)^{m-i}$$

The coefficients  $c_{ij}$  are stored in the following order:  $c_{00}, c_{10}, c_{20}, \dots, c_{mn}$ .

Definition at line 71 of file BernsteinMulti.h.

### 29.29.2 Constructor & Destructor Documentation

29.29.2.1 Go::BernsteinMulti::BernsteinMulti ( ) [inline]

Default constructor.

Definition at line 74 of file BernsteinMulti.h.

29.29.2.2 Go::BernsteinMulti::BernsteinMulti ( int m, int n, const std::vector< double > &coefs ) [inline]

Constructor.

Parameters

|         |                                          |
|---------|------------------------------------------|
| $m$     | degree in the first parameter direction  |
| $n$     | degree in the second parameter direction |
| $coefs$ | vector of Bernstein coefficients         |

Definition at line 79 of file BernsteinMulti.h.

29.29.2.3 `template<typename ForwardIterator > Go::BernsteinMulti::BernsteinMulti ( int m, int n, ForwardIterator begin, ForwardIterator end ) [inline]`

Constructor

Parameters

|              |                                                            |
|--------------|------------------------------------------------------------|
| <i>m</i>     | degree in the first parameter direction                    |
| <i>n</i>     | degree in the second parameter direction                   |
| <i>begin</i> | iterator to start of container with Bernstein coefficients |
| <i>end</i>   | iterator to end of container                               |

Definition at line 93 of file BernsteinMulti.h.

29.29.2.4 `Go::BernsteinMulti::BernsteinMulti ( double coef ) [inline],[explicit]`

Constructs a constant polynomial.

Parameters

|             |                                      |
|-------------|--------------------------------------|
| <i>coef</i> | the constant value of the polynomial |
|-------------|--------------------------------------|

Definition at line 98 of file BernsteinMulti.h.

## 29.29.3 Member Function Documentation

29.29.3.1 `BernsteinPoly Go::BernsteinMulti::bindU ( double u ) const`

Function that "binds" the u-parameter.

Returns

a [BernsteinPoly](#) in the v-direction for a constant value of u.

29.29.3.2 `BernsteinPoly Go::BernsteinMulti::bindV ( double v ) const`

Function that "binds" the v-parameter.

Returns

a [BernsteinPoly](#) in the u-direction for a constant value of v.

29.29.3.3 `double Go::BernsteinMulti::blossom ( const std::vector< double > & uvec, const std::vector< double > & vvec ) const`

Evaluates the blossom of the polynomial. The blossom of the polynomial  $p$  of degree  $(m,n)$  is a multi-affine  $(m+n)$ -variate function  $B_p$  that satisfies

$$B_p(\underbrace{u, \dots, u}_m, \underbrace{v, \dots, v}_n) = p(u, v)$$

and is symmetric in the first  $m$  and last  $n$  arguments separately.

## Parameters

|             |                                                                                          |
|-------------|------------------------------------------------------------------------------------------|
| <i>uvec</i> | the vector of values at which the blossom is evaluated in the first parameter direction  |
| <i>vvec</i> | the vector of values at which the blossom is evaluated in the second parameter direction |

## Returns

the value of the blossom at (*uvec*,*vvec*)

**29.29.3.4** `std::vector<double>::iterator Go::BernsteinMulti::coefsBegin ( ) [inline]`

Access function for coefficients.

Definition at line 117 of file BernsteinMulti.h.

**29.29.3.5** `std::vector<double>::const_iterator Go::BernsteinMulti::coefsBegin ( ) const [inline]`

Access function for coefficients.

Definition at line 120 of file BernsteinMulti.h.

**29.29.3.6** `std::vector<double>::iterator Go::BernsteinMulti::coefsEnd ( ) [inline]`

Access function for coefficients.

Definition at line 123 of file BernsteinMulti.h.

**29.29.3.7** `std::vector<double>::const_iterator Go::BernsteinMulti::coefsEnd ( ) const [inline]`

Access function for coefficients.

Definition at line 126 of file BernsteinMulti.h.

**29.29.3.8** `void Go::BernsteinMulti::degreeElevate ( int du, int dv )`

Degree elevation.

## Parameters

|           |                                                                                                        |
|-----------|--------------------------------------------------------------------------------------------------------|
| <i>du</i> | the polynomial degree in which you want to represent this polynomial in the first parameter direction  |
| <i>dv</i> | the polynomial degree in which you want to represent this polynomial in the second parameter direction |

29.29.3.9 `int Go::BernsteinMulti::degreeU ( ) const [inline]`

Get the degree in the first parameter direction.

#### Returns

the degree in the first parameter direction

Definition at line 103 of file BernsteinMulti.h.

29.29.3.10 `int Go::BernsteinMulti::degreeV ( ) const [inline]`

Get the degree in the second parameter direction.

#### Returns

the degree in the second parameter direction

Definition at line 107 of file BernsteinMulti.h.

29.29.3.11 `BernsteinMulti Go::BernsteinMulti::deriv ( int der1, int der2 ) const`

Finds the derivative of the polynomial in terms of a new [BernsteinMulti](#).

#### Parameters

|             |                                                                      |
|-------------|----------------------------------------------------------------------|
| <i>der1</i> | the order of derivatives requested in the first parameter direction  |
| <i>der2</i> | the order of derivatives requested in the second parameter direction |

#### Returns

a new polynomial q given by

$$q(u, v) = D^{(der1, der2)}p(u, v).$$

29.29.3.12 `BernsteinMulti Go::BernsteinMulti::detHess ( ) const`

Calculates the determinant of the Hessian as a [BernsteinMulti](#) Describes Gaussian curvature of the graph of the polynomial

29.29.3.13 `bool Go::BernsteinMulti::isNonNegative ( const double eps = 0.0 ) const`

Check if polynomial is non-negative

#### Parameters

|            |                                                              |
|------------|--------------------------------------------------------------|
| <i>eps</i> | the threshold value with which every coefficient is compared |
|------------|--------------------------------------------------------------|

**Returns**

true if for every coefficient  $c$ ,  $c \geq -eps$ , false otherwise

**29.29.3.14** `bool Go::BernsteinMulti::isStrictlyNegative ( const double eps = 0.0 ) const`

Check if polynomial is strictly negative

**Parameters**

|                  |                                                              |
|------------------|--------------------------------------------------------------|
| <code>eps</code> | the threshold value with which every coefficient is compared |
|------------------|--------------------------------------------------------------|

**Returns**

true if for every coefficient  $c$ ,  $c < -eps$ , false otherwise

**29.29.3.15** `bool Go::BernsteinMulti::isStrictlyPositive ( const double eps = 0.0 ) const`

Check if polynomial is strictly positive

**Parameters**

|                  |                                                              |
|------------------|--------------------------------------------------------------|
| <code>eps</code> | the threshold value with which every coefficient is compared |
|------------------|--------------------------------------------------------------|

**Returns**

true if for every coefficient  $c$ ,  $c > eps$ , false otherwise

**29.29.3.16** `bool Go::BernsteinMulti::isZero ( const double eps = 0.0 ) const`

Check if the polynomial is the zero function

**Parameters**

|                  |                                                              |
|------------------|--------------------------------------------------------------|
| <code>eps</code> | the threshold value with which every coefficient is compared |
|------------------|--------------------------------------------------------------|

**Returns**

true if for every coefficient  $c$ ,  $|c| \leq eps$ , false otherwise

**29.29.3.17** `double Go::BernsteinMulti::mean ( ) const`

Calculates the mean value of the coefficients.



29.29.3.18 `double Go::BernsteinMulti::norm ( ) const`

Calculates the norm of the polynomial, defined as the sum of absolute values of the coefficients divided by the number of coefficients. This norm is not the same as the L1 norm of the polynomial on the domain  $[0,1] \times [0,1]$ , unless the polynomial does not change sign. In any case this norm is greater than or equal to the L1 norm of the polynomial.

29.29.3.19 `void Go::BernsteinMulti::normalize ( )`

Normalizes the polynomial by dividing all coefficients with `norm()`, which is close to (but not identical to) the L1 norm of the polynomial.

See also

[norm\(\)](#)

29.29.3.20 `double Go::BernsteinMulti::operator() ( double u, double v ) const`

Evaluation operator, based on the de Casteljau algorithm.

Parameters

|          |                  |
|----------|------------------|
| <i>u</i> | first parameter  |
| <i>v</i> | second parameter |

Returns

the value of the polynomial at (*u*,*v*)

29.29.3.21 `double Go::BernsteinMulti::operator() ( const Vector2D & pt ) const` `[inline]`

Evaluation operator - as above, but taking the arguments in a Vector2D.

Parameters

|           |                 |
|-----------|-----------------|
| <i>pt</i> | parameter point |
|-----------|-----------------|

Returns

the value of the polynomial at *pt*

Definition at line 138 of file BernsteinMulti.h.

29.29.3.22 `double Go::BernsteinMulti::operator() ( const std::vector< double > & pt ) const` `[inline]`

Evaluation operator - as above, but taking the arguments in a `std::vector`.

## Parameters

|      |                 |
|------|-----------------|
| $pt$ | parameter point |
|------|-----------------|

## Returns

the value of the polynomial at  $pt$

Definition at line 144 of file BernsteinMulti.h.

**29.29.3.23 BernsteinMulti& Go::BernsteinMulti::operator\*= ( const BernsteinMulti & multi )**

Multiplication with another polynomial.

**29.29.3.24 BernsteinMulti& Go::BernsteinMulti::operator\*= ( double c )**

Multiplication with a scalar.

**29.29.3.25 BernsteinMulti& Go::BernsteinMulti::operator+= ( const BernsteinMulti & multi )**

Addition with another polynomial.

**29.29.3.26 BernsteinMulti& Go::BernsteinMulti::operator+= ( double c )**

Addition with a scalar.

**29.29.3.27 BernsteinMulti& Go::BernsteinMulti::operator-= ( const BernsteinMulti & multi )**

Subtraction with another polynomial.

**29.29.3.28 BernsteinMulti& Go::BernsteinMulti::operator-= ( double c )**

Subtraction with a scalar.

**29.29.3.29 BernsteinMulti& Go::BernsteinMulti::operator/= ( double c )**

Division with a scalar.

**29.29.3.30 const double& Go::BernsteinMulti::operator[] ( int num ) const [inline]**

Access function for coefficients.

Definition at line 111 of file BernsteinMulti.h.

**29.29.3.31** `double& Go::BernsteinMulti::operator[]( int num ) [inline]`

Access function for coefficients.

Definition at line 114 of file BernsteinMulti.h.

**29.29.3.32** `BernsteinMulti Go::BernsteinMulti::pickDomain ( double u0, double u1, double v0, double v1 ) const`

Finds a new [BernsteinMulti](#) representing the original on some rectangular domain. The new polynomial is defined on  $[0,1] \times [0,1]$ . If  $p$  is the old polynomial and  $q$  is the new one,  $q$  is defined by

$$q(u, v) = p((1 - u) \cdot u_0 + u \cdot u_1, (1 - v) \cdot v_0 + v \cdot v_1).$$

The function is implemented by blossoming.

#### Parameters

|      |                                                             |
|------|-------------------------------------------------------------|
| $u0$ | the start of the interval in the first parameter direction  |
| $u1$ | the end of the interval in the first parameter direction    |
| $v0$ | the start of the interval in the second parameter direction |
| $v1$ | the end of the interval in the second parameter direction   |

#### Returns

a polynomial  $q$  as specified above

**29.29.3.33** `BernsteinPoly Go::BernsteinMulti::pickLine ( Array< double, 2 > a, Array< double, 2 > b ) const`

Routine that picks out the line with endpoints  $a$  and  $b$  as a new [BernsteinPoly](#). The new polynomial is defined on  $[0,1]$ . If  $p$  is the old bivariate polynomial and  $q$  is the new one,  $q$  is defined by

$$q(t) = p((1 - t)a_0 + tb_0, (1 - t)a_1 + tv_1).$$

The function is implemented by blossoming.

**29.29.3.34** `void Go::BernsteinMulti::read ( std::istream & is )`

Read from an input stream.

**29.29.3.35** `BernsteinMulti Go::BernsteinMulti::traceHess ( ) const`

Calculates the trace of the Hessian as a BernsteinMulti Describes mean curvature (modulo a factor 2) of the graph of the polynomial

**29.29.3.36** `void Go::BernsteinMulti::write ( std::ostream & os ) const`

Write to an output stream.

The documentation for this class was generated from the following file:

- [implicitization/include/GoTools/implicitization/BernsteinMulti.h](#)

## 29.30 Go::BernsteinPoly Class Reference

```
#include <BernsteinPoly.h>
```

### Public Member Functions

- [BernsteinPoly \(\)](#)  
*Default constructor.*
- [BernsteinPoly \(const std::vector< double > &coefs\)](#)
- [template<typename ForwardIterator > BernsteinPoly \(ForwardIterator begin, ForwardIterator end\)](#)
- [BernsteinPoly \(double coef\)](#)
- [int degree \(\) const](#)
- [const double & operator\[\] \(int num\) const](#)  
*Access function for coefficients.*
- [double & operator\[\] \(int num\)](#)  
*Access function for coefficients.*
- [std::vector< double >::iterator coefsBegin \(\)](#)  
*Access function for coefficients.*
- [std::vector< double >::const\\_iterator coefsBegin \(\) const](#)  
*Access function for coefficients.*
- [std::vector< double >::iterator coefsEnd \(\)](#)  
*Access function for coefficients.*
- [std::vector< double >::const\\_iterator coefsEnd \(\) const](#)  
*Access function for coefficients.*
- [double operator\(\) \(double t\) const](#)
- [bool isZero \(const double eps=0.0\) const](#)
- [bool isStrictlyPositive \(const double eps=0.0\) const](#)
- [double norm \(\) const](#)
- [void normalize \(\)](#)
- [BernsteinPoly deriv \(int der\) const](#)
- [double integral \(double a=0.0, double b=1.0\) const](#)
- [double blossom \(const std::vector< double > &tvec\) const](#)
- [BernsteinPoly pickInterval \(double a, double b\) const](#)
- [void degreeElevate \(int d\)](#)
- [BernsteinPoly & operator\\*= \(const BernsteinPoly &poly\)](#)  
*Multiplication with another polynomial.*
- [BernsteinPoly & operator\\*= \(double c\)](#)  
*Multiplication with a scalar.*
- [BernsteinPoly & operator+= \(const BernsteinPoly &poly\)](#)  
*Addition with another polynomial.*
- [BernsteinPoly & operator+= \(double c\)](#)  
*Addition with a scalar.*
- [BernsteinPoly & operator-= \(const BernsteinPoly &poly\)](#)  
*Subtraction with another polynomial.*
- [BernsteinPoly & operator-= \(double c\)](#)  
*Subtraction with a scalar.*
- [BernsteinPoly & operator/= \(double c\)](#)  
*Division with a scalar.*
- [void read \(std::istream &is\)](#)  
*Read from input stream.*
- [void write \(std::ostream &os\) const](#)  
*Write to output stream.*

### 29.30.1 Detailed Description

Class that implements Bernstein polynomials on the interval [0,1]. The formula for Bernstein polynomials is

$$p(t) = \sum_{i=0}^n \binom{n}{i} t^i (1-t)^{n-i} c_i.$$

The coefficients  $c_i$  are stored in a vector of doubles.

Definition at line 58 of file BernsteinPoly.h.

### 29.30.2 Constructor & Destructor Documentation

#### 29.30.2.1 Go::BernsteinPoly::BernsteinPoly ( ) [inline]

Default constructor.

Definition at line 61 of file BernsteinPoly.h.

#### 29.30.2.2 Go::BernsteinPoly::BernsteinPoly ( const std::vector< double > & *coefs* ) [inline],[explicit]

Constructor

Parameters

|              |                                                           |
|--------------|-----------------------------------------------------------|
| <i>coefs</i> | a vector of doubles containing the Bernstein coefficients |
|--------------|-----------------------------------------------------------|

Definition at line 65 of file BernsteinPoly.h.

#### 29.30.2.3 template<typename ForwardIterator > Go::BernsteinPoly::BernsteinPoly ( ForwardIterator *begin*, ForwardIterator *end* ) [inline]

Constructor

Parameters

|              |                                                                      |
|--------------|----------------------------------------------------------------------|
| <i>begin</i> | iterator to start of container containing the Bernstein coefficients |
| <i>end</i>   | iterator to end of same container                                    |

Definition at line 72 of file BernsteinPoly.h.

#### 29.30.2.4 Go::BernsteinPoly::BernsteinPoly ( double *coef* ) [inline],[explicit]

Constructor for a constant polynomial

## Parameters

|             |                                      |
|-------------|--------------------------------------|
| <i>coef</i> | the constant value of the polynomial |
|-------------|--------------------------------------|

Definition at line 76 of file BernsteinPoly.h.

### 29.30.3 Member Function Documentation

#### 29.30.3.1 `double Go::BernsteinPoly::blossom ( const std::vector< double > & tvec ) const`

Evaluates the blossom of the polynomial. The blossom of the polynomial  $p$  of degree  $n$  is a multi-affine  $n$ -variate symmetric function  $B_p$  that satisfies  $B_p(t, \dots, t) = p(t)$ .

## Parameters

|             |                                                         |
|-------------|---------------------------------------------------------|
| <i>tvec</i> | the vector of values at which the blossom is evaluated. |
|-------------|---------------------------------------------------------|

## Returns

the value of the blossom at *tvec*

#### 29.30.3.2 `std::vector<double>::iterator Go::BernsteinPoly::coefsBegin ( ) [inline]`

Access function for coefficients.

Definition at line 91 of file BernsteinPoly.h.

#### 29.30.3.3 `std::vector<double>::const_iterator Go::BernsteinPoly::coefsBegin ( ) const [inline]`

Access function for coefficients.

Definition at line 94 of file BernsteinPoly.h.

#### 29.30.3.4 `std::vector<double>::iterator Go::BernsteinPoly::coefsEnd ( ) [inline]`

Access function for coefficients.

Definition at line 97 of file BernsteinPoly.h.

#### 29.30.3.5 `std::vector<double>::const_iterator Go::BernsteinPoly::coefsEnd ( ) const [inline]`

Access function for coefficients.

Definition at line 100 of file BernsteinPoly.h.

29.30.3.6 `int Go::BernsteinPoly::degree ( ) const [inline]`

Get the degree of the polynomial

Returns

the degree of the polynomial

Definition at line 81 of file BernsteinPoly.h.

29.30.3.7 `void Go::BernsteinPoly::degreeElevate ( int d )`

Degree elevation.

Parameters

|          |                                                                       |
|----------|-----------------------------------------------------------------------|
| <i>d</i> | the polynomial degree in which you want to represent this polynomial. |
|----------|-----------------------------------------------------------------------|

29.30.3.8 `BernsteinPoly Go::BernsteinPoly::deriv ( int der ) const`

Calculates the derivative in terms of a new [BernsteinPoly](#).

Parameters

|            |                                    |
|------------|------------------------------------|
| <i>der</i> | the order of derivatives requested |
|------------|------------------------------------|

Returns

a new polynomial q given by

$$q(t) = p^{(der)}(t).$$

29.30.3.9 `double Go::BernsteinPoly::integral ( double a = 0.0, double b = 1.0 ) const`

Calculates the integral of the polynomial over an interval. The default interval is [0,1].

29.30.3.10 `bool Go::BernsteinPoly::isStrictlyPositive ( const double eps = 0.0 ) const`

Check if the polynomial is strictly positive. NOTE: If the return value is 'true', then the polynomial is guaranteed to be strictly positive. If the return value is 'false', the situation is undetermined and the polynomial may still be strictly positive.

Parameters

|            |                                                              |
|------------|--------------------------------------------------------------|
| <i>eps</i> | the threshold value with which every coefficient is compared |
|------------|--------------------------------------------------------------|

**Returns**

true if for every coefficient  $c$ ,  $c > eps$ , false otherwise

**29.30.3.11** `bool Go::BernsteinPoly::isZero ( const double eps = 0.0 ) const`

Check if the polynomial is the zero function

**Parameters**

|                  |                                                              |
|------------------|--------------------------------------------------------------|
| <code>eps</code> | the threshold value with which every coefficient is compared |
|------------------|--------------------------------------------------------------|

**Returns**

true if for every coefficient  $c$ ,  $|c| \leq eps$ , false otherwise

**29.30.3.12** `double Go::BernsteinPoly::norm ( ) const`

Calculates a norm, defined as the sum of the absolute values of the coefficients divided by the number of coefficients. This norm is not the same as the L1 norm of the polynomial (on the interval  $[0,1]$ ), unless the polynomial does not change sign. In any case this norm is greater than or equal to the L1 norm of the polynomial.

**29.30.3.13** `void Go::BernsteinPoly::normalize ( )`

Normalizes the polynomial by dividing all coefficients with `norm()`, which is close to (but not identical to) the L1 norm of the polynomial.

**See also**

[norm\(\)](#)

**29.30.3.14** `double Go::BernsteinPoly::operator() ( double t ) const`

Evaluation operator, based on the de Casteljau algorithm.

**Parameters**

|                |                                                          |
|----------------|----------------------------------------------------------|
| <code>t</code> | the parameter value at which the polynomial is evaluated |
|----------------|----------------------------------------------------------|

**Returns**

the value of the polynomial at  $t$

**29.30.3.15** `BernsteinPoly& Go::BernsteinPoly::operator*= ( const BernsteinPoly & poly )`

Multiplication with another polynomial.



29.30.3.16 **BernsteinPoly**& Go::BernsteinPoly::operator\*=( double c )

Multiplication with a scalar.

29.30.3.17 **BernsteinPoly**& Go::BernsteinPoly::operator+=( const BernsteinPoly & poly )

Addition with another polynomial.

29.30.3.18 **BernsteinPoly**& Go::BernsteinPoly::operator+=( double c )

Addition with a scalar.

29.30.3.19 **BernsteinPoly**& Go::BernsteinPoly::operator-=( const BernsteinPoly & poly )

Subtraction with another polynomial.

29.30.3.20 **BernsteinPoly**& Go::BernsteinPoly::operator-=( double c )

Subtraction with a scalar.

29.30.3.21 **BernsteinPoly**& Go::BernsteinPoly::operator/=( double c )

Division with a scalar.

29.30.3.22 **const double&** Go::BernsteinPoly::operator[]( int num ) const `[inline]`

Access function for coefficients.

Definition at line 85 of file BernsteinPoly.h.

29.30.3.23 **double&** Go::BernsteinPoly::operator[]( int num ) `[inline]`

Access function for coefficients.

Definition at line 88 of file BernsteinPoly.h.

29.30.3.24 **BernsteinPoly** Go::BernsteinPoly::pickInterval( double a, double b ) const

Finds a new [BernsteinPoly](#) representing the original on some interval. The new polynomial is defined on [0,1]. If p is the old polynomial and q is the new one, q is defined by

$$q(t) = p(a(1 - t) + tb).$$

## Parameters

|          |                           |
|----------|---------------------------|
| <i>a</i> | the start of the interval |
| <i>b</i> | the end of the interval   |

## Returns

a polynomial  $q$  as specified above

29.30.3.25 void Go::BernsteinPoly::read ( std::istream & *is* )

Read from input stream.

29.30.3.26 void Go::BernsteinPoly::write ( std::ostream & *os* ) const

Write to output stream.

The documentation for this class was generated from the following file:

- [implicitization/include/GoTools/implicitization/BernsteinPoly.h](#)

## 29.31 Go::BernsteinTetrahedralPoly Class Reference

```
#include <BernsteinTetrahedralPoly.h>
```

### Public Member Functions

- [BernsteinTetrahedralPoly \(\)](#)  
*Default constructor.*
- [BernsteinTetrahedralPoly \(int deg, const std::vector< double > &coefs\)](#)
- [template<typename ForwardIterator > BernsteinTetrahedralPoly \(int deg, ForwardIterator begin, ForwardIterator end\)](#)
- [BernsteinTetrahedralPoly \(double coef\)](#)
- [int degree \(\) const](#)
- [const double & operator\[\] \(int num\) const](#)  
*Access function for coefficients.*
- [double & operator\[\] \(int num\)](#)  
*Access function for coefficients.*
- [template<typename T > T operator\(\) \(const Array< T, 4 > &u\) const](#)
- [double norm \(\) const](#)
- [void normalize \(\)](#)
- [void deriv \(int der, const Vector4D &d, BernsteinTetrahedralPoly &bt\) const](#)
- [template<typename T > T blossom \(const std::vector< Array< T, 4 > > &uvec\) const](#)
- [BernsteinPoly pickLine \(const Array< double, 4 > &a, const Array< double, 4 > &b\) const](#)
- [BernsteinTetrahedralPoly & operator\\*=\(const BernsteinTetrahedralPoly &poly\)](#)

- Multiplication with another polynomial.*

  - [BernsteinTetrahedralPoly](#) & `operator*=(double c)`

*Multiplication with a scalar.*

  - [BernsteinTetrahedralPoly](#) & `operator+=(const BernsteinTetrahedralPoly &poly)`

*Addition with another polynomial.*

  - [BernsteinTetrahedralPoly](#) & `operator+=(double c)`

*Addition with a scalar.*

  - [BernsteinTetrahedralPoly](#) & `operator-=(const BernsteinTetrahedralPoly &poly)`

*Subtraction with another polynomial.*

  - [BernsteinTetrahedralPoly](#) & `operator-=(double c)`

*Subtraction with a scalar.*

  - [BernsteinTetrahedralPoly](#) & `operator/=(double c)`

*Division with a scalar.*

  - `void read` (`std::istream &is`)

*Read from an input stream.*

  - `void write` (`std::ostream &os`) `const`

*Write to an output stream.*

### 29.31.1 Detailed Description

Class that implements Bernstein polynomials on a tetrahedron. The formula for a tetrahedral Bernstein polynomial of degree  $n$  is

$$p(\beta_0, \beta_1, \beta_2, \beta_3) = \sum_{i+j+k+l=n} \frac{n!}{i!j!k!l!} \beta_0^i \beta_1^j \beta_2^k \beta_3^l c_{(i,j,k)}.$$

The coefficients  $c_{ijk}$  are stored in the following order:  $c_{(n,0,0,0)}$ ,  $c_{(n-1,1,0,0)}$ ,  $c_{(n-2,2,0,0)}$ ,  $\dots$ ,  $c_{(0,n,0,0)}$ ,  $c_{(n-1,0,1,0)}$ ,  $c_{(n-2,1,1,0)}$ ,  $\dots$ ,  $c_{(0,0,0,n)}$ . That is, a reverse lexicographical ordering with the most significant index being the last one.

Definition at line 71 of file `BernsteinTetrahedralPoly.h`.

### 29.31.2 Constructor & Destructor Documentation

**29.31.2.1** `Go::BernsteinTetrahedralPoly::BernsteinTetrahedralPoly ( )` [`inline`]

Default constructor.

Definition at line 74 of file `BernsteinTetrahedralPoly.h`.

**29.31.2.2** `Go::BernsteinTetrahedralPoly::BernsteinTetrahedralPoly ( int deg, const std::vector< double > &coefs )` [`inline`]

Constructor.

Parameters

|              |                                  |
|--------------|----------------------------------|
| <i>deg</i>   | degree of the polynomial         |
| <i>coefs</i> | vector of Bernstein coefficients |

Definition at line 78 of file BernsteinTetrahedralPoly.h.

```
29.31.2.3 template<typename ForwardIterator > Go::BernsteinTetrahedralPoly::BernsteinTetrahedralPoly (int deg,
ForwardIterator begin, ForwardIterator end) [inline]
```

Constructor.

#### Parameters

|              |                                                       |
|--------------|-------------------------------------------------------|
| <i>deg</i>   | degree of the polynomial                              |
| <i>begin</i> | iterator to start of vector of Bernstein coefficients |
| <i>end</i>   | iterator to end of vector of Bernstein coefficients   |

Definition at line 86 of file BernsteinTetrahedralPoly.h.

```
29.31.2.4 Go::BernsteinTetrahedralPoly::BernsteinTetrahedralPoly (double coef) [inline],[explicit]
```

Constructor for a constant polynomial.

#### Parameters

|             |                                      |
|-------------|--------------------------------------|
| <i>coef</i> | the constant value of the polynomial |
|-------------|--------------------------------------|

Definition at line 91 of file BernsteinTetrahedralPoly.h.

### 29.31.3 Member Function Documentation

```
29.31.3.1 template<typename T > T Go::BernsteinTetrahedralPoly::blossom (const std::vector< Array< T, 4 > > & uvec
) const [inline]
```

Evaluates the blossom of the polynomial. The blossom of the polynomial  $p$  of degree  $n$  is a multi-affine, symmetric  $n$ -variate function  $B_p$  that satisfies

$$B_p(b, \dots, b) = p(b).$$

#### Parameters

|             |                                                                                                                        |
|-------------|------------------------------------------------------------------------------------------------------------------------|
| <i>uvec</i> | the vector of values at which the blossom is evaluated, each value is a domain point given in barycentric coordinates. |
|-------------|------------------------------------------------------------------------------------------------------------------------|

Definition at line 172 of file BernsteinTetrahedralPoly.h.

```
29.31.3.2 int Go::BernsteinTetrahedralPoly::degree () const [inline]
```

Get the degree

**Returns**

the degree of the polynomial

Definition at line 96 of file BernsteinTetrahedralPoly.h.

**29.31.3.3** `void Go::BernsteinTetrahedralPoly::deriv ( int der, const Vector4D & d, BernsteinTetrahedralPoly & btp ) const`

Calculates the *der*'th derivative in the *d*-direction in terms of a new [BernsteinTetrahedralPoly](#). The direction is given in barycentric coordinates, so it is a vector whose elements sum to zero.

**Parameters**

|            |                               |
|------------|-------------------------------|
| <i>der</i> | order of derivative requested |
| <i>d</i>   | direction of derivative       |

**Return values**

|            |                           |
|------------|---------------------------|
| <i>btp</i> | the derivative polynomial |
|------------|---------------------------|

**29.31.3.4** `double Go::BernsteinTetrahedralPoly::norm ( ) const`

Calculates the norm of the polynomial, defined as the sum of absolute values of the coefficients divided by the number of coefficients. This norm is not the same as the L1 norm of the polynomial over the tetrahedron, but is an upper bound for it.

**29.31.3.5** `void Go::BernsteinTetrahedralPoly::normalize ( )`

Normalizes the polynomial by dividing all coefficients with [norm\(\)](#), which is related to (but not identical to) the L1 norm of the polynomial.

**See also**

[norm\(\)](#)

**29.31.3.6** `template<typename T > T Go::BernsteinTetrahedralPoly::operator() ( const Array< T, 4 > & u ) const`  
`[inline]`

Evaluation operator, based on the tetrahedral de Casteljau algorithm. Generalized from 'tri\_decas' in Farin: "Curves and Surfaces for CAGD".

**Parameters**

|          |                                                  |
|----------|--------------------------------------------------|
| <i>u</i> | parameter point given in barycentric coordinates |
|----------|--------------------------------------------------|

**Returns**

the value of the polynomial at  $u$

Definition at line 112 of file BernsteinTetrahedralPoly.h.

**29.31.3.7** `BernsteinTetrahedralPoly& Go::BernsteinTetrahedralPoly::operator*=( const BernsteinTetrahedralPoly & poly )`

Multiplication with another polynomial.

**29.31.3.8** `BernsteinTetrahedralPoly& Go::BernsteinTetrahedralPoly::operator*=( double c )`

Multiplication with a scalar.

**29.31.3.9** `BernsteinTetrahedralPoly& Go::BernsteinTetrahedralPoly::operator+=( const BernsteinTetrahedralPoly & poly )`

Addition with another polynomial.

**29.31.3.10** `BernsteinTetrahedralPoly& Go::BernsteinTetrahedralPoly::operator+=( double c )`

Addition with a scalar.

**29.31.3.11** `BernsteinTetrahedralPoly& Go::BernsteinTetrahedralPoly::operator-=( const BernsteinTetrahedralPoly & poly )`

Subtraction with another polynomial.

**29.31.3.12** `BernsteinTetrahedralPoly& Go::BernsteinTetrahedralPoly::operator-=( double c )`

Subtraction with a scalar.

**29.31.3.13** `BernsteinTetrahedralPoly& Go::BernsteinTetrahedralPoly::operator/=( double c )`

Division with a scalar.

**29.31.3.14** `const double& Go::BernsteinTetrahedralPoly::operator[]( int num ) const [inline]`

Access function for coefficients.

Definition at line 100 of file BernsteinTetrahedralPoly.h.

29.31.3.15 `double& Go::BernsteinTetrahedralPoly::operator[] ( int num ) [inline]`

Access function for coefficients.

Definition at line 103 of file BernsteinTetrahedralPoly.h.

29.31.3.16 `BernsteinPoly Go::BernsteinTetrahedralPoly::pickLine ( const Array< double, 4 > & a, const Array< double, 4 > & b ) const`

Routine that picks out the line with endpoints a and b as a new [BernsteinPoly](#). The new polynomial is defined on [0,1]. If p is the old bivariate polynomial and q is the new one, q is defined by

$$q(t) = p((1-t)a_0 + tb_0, (1-t)a_1 + tb_1).$$

The function is implemented by blossoming.

#### Parameters

|          |                                                           |
|----------|-----------------------------------------------------------|
| <i>a</i> | the starting point of the line in barycentric coordinates |
| <i>b</i> | the endpoint of the line in barycentric coordinates       |

29.31.3.17 `void Go::BernsteinTetrahedralPoly::read ( std::istream & is )`

Read from an input stream.

29.31.3.18 `void Go::BernsteinTetrahedralPoly::write ( std::ostream & os ) const`

Write to an output stream.

The documentation for this class was generated from the following file:

- [implicitization/include/GoTools/implicitization/BernsteinTetrahedralPoly.h](#)

## 29.32 Go::BernsteinTriangularPoly Class Reference

```
#include <BernsteinTriangularPoly.h>
```

### Public Member Functions

- [BernsteinTriangularPoly \(\)](#)  
*Default constructor.*
- [BernsteinTriangularPoly \(int deg, const std::vector< double > &coefs\)](#)
- [template<typename ForwardIterator > BernsteinTriangularPoly \(int deg, ForwardIterator begin, ForwardIterator end\)](#)
- [BernsteinTriangularPoly \(double coef\)](#)
- [int degree \(\) const](#)

- `const double operator[] (int num) const`  
*Access function for coefficients.*
- `double & operator[] (int num)`  
*Access function for coefficients.*
- `template<typename T >`  
`T operator() (const Array< T, 3 > &u) const`
- `double norm () const`
- `void normalize ()`
- `void deriv (int der, const Vector3D &d, BernsteinTriangularPoly &btp) const`
- `template<typename T >`  
`T blossom (const std::vector< Array< T, 3 > > &uvec) const`
- `BernsteinTriangularPoly & operator*= (const BernsteinTriangularPoly &poly)`  
*Multiplication with another polynomial.*
- `BernsteinTriangularPoly & operator*= (double c)`  
*Multiplication with a scalar.*
- `BernsteinTriangularPoly & operator+= (const BernsteinTriangularPoly &poly)`  
*Addition with another polynomial.*
- `BernsteinTriangularPoly & operator+= (double c)`  
*Addition with a scalar.*
- `BernsteinTriangularPoly & operator-= (const BernsteinTriangularPoly &poly)`  
*Subtraction with another polynomial.*
- `BernsteinTriangularPoly & operator-= (double c)`  
*Subtraction with a scalar.*
- `BernsteinTriangularPoly & operator/= (double c)`  
*Division with a scalar.*
- `void read (std::istream &is)`  
*Read from an input stream.*
- `void write (std::ostream &os) const`  
*Write to an output stream.*

### 29.32.1 Detailed Description

Class that implements Bernstein polynomials on a triangle. The formula for a triangular Bernstein polynomial of degree  $n$  is

$$p(\beta_0, \beta_1, \beta_2) = \sum_{i+j+k=n} \frac{n!}{i!j!k!} \beta_0^i \beta_1^j \beta_2^k c_{(i,j,k)}.$$

The coefficients  $c_{ijk}$  are stored in the following order:  $c_{(n,0,0)}$ ,  $c_{(n-1,1,0)}$ ,  $c_{(n-2,2,0)}$ ,  $\dots$ ,  $c_{(0,n,0)}$ ,  $c_{(n-1,0,1)}$ ,  $c_{(n-2,1,1)}$ ,  $\dots$ ,  $c_{(0,0,n)}$ .

Definition at line 66 of file `BernsteinTriangularPoly.h`.

### 29.32.2 Constructor & Destructor Documentation

29.32.2.1 `Go::BernsteinTriangularPoly::BernsteinTriangularPoly ( ) [inline]`

Default constructor.

Definition at line 69 of file `BernsteinTriangularPoly.h`.

29.32.2.2 `Go::BernsteinTriangularPoly::BernsteinTriangularPoly ( int deg, const std::vector< double > &coefs ) [inline]`

Constructor.



## Parameters

|              |                                  |
|--------------|----------------------------------|
| <i>deg</i>   | degree of the polynomial         |
| <i>coefs</i> | vector of Bernstein coefficients |

Definition at line 73 of file BernsteinTriangularPoly.h.

```
29.32.2.3 template<typename ForwardIterator > Go::BernsteinTriangularPoly::BernsteinTriangularPoly (int deg,
ForwardIterator begin, ForwardIterator end) [inline]
```

Constructor.

## Parameters

|              |                                                       |
|--------------|-------------------------------------------------------|
| <i>deg</i>   | degree of the polynomial                              |
| <i>begin</i> | iterator to start of vector of Bernstein coefficients |
| <i>end</i>   | iterator to end of vector of Bernstein coefficients   |

Definition at line 81 of file BernsteinTriangularPoly.h.

```
29.32.2.4 Go::BernsteinTriangularPoly::BernsteinTriangularPoly (double coef) [inline],[explicit]
```

Constructor for a constant polynomial.

## Parameters

|             |                                      |
|-------------|--------------------------------------|
| <i>coef</i> | the constant value of the polynomial |
|-------------|--------------------------------------|

Definition at line 86 of file BernsteinTriangularPoly.h.

### 29.32.3 Member Function Documentation

```
29.32.3.1 template<typename T > T Go::BernsteinTriangularPoly::blossom (const std::vector< Array< T, 3 > > & uvec)
const [inline]
```

Evaluates the blossom of the polynomial. The blossom of the polynomial  $p$  of degree  $n$  is a multi-affine, symmetric  $n$ -variate function  $B_p$  that satisfies

$$B_p(b, \dots, b) = p(b).$$

## Parameters

|             |                                                                                                                        |
|-------------|------------------------------------------------------------------------------------------------------------------------|
| <i>uvec</i> | the vector of values at which the blossom is evaluated, each value is a domain point given in barycentric coordinates. |
|-------------|------------------------------------------------------------------------------------------------------------------------|

Definition at line 164 of file BernsteinTriangularPoly.h.

29.32.3.2 `int Go::BernsteinTriangularPoly::degree ( ) const` `[inline]`

Get the degree

Returns

the degree of the polynomial

Definition at line 91 of file BernsteinTriangularPoly.h.

29.32.3.3 `void Go::BernsteinTriangularPoly::deriv ( int der, const Vector3D & d, BernsteinTriangularPoly & btp ) const`

Calculates the *der*'th derivative in the *d*-direction in terms of a new [BernsteinTriangularPoly](#). The direction is given in barycentric coordinates, so it is a vector whose elements sum to zero.

Parameters

|            |                               |
|------------|-------------------------------|
| <i>der</i> | order of derivative requested |
| <i>d</i>   | direction of derivative       |

Return values

|            |                           |
|------------|---------------------------|
| <i>btp</i> | the derivative polynomial |
|------------|---------------------------|

29.32.3.4 `double Go::BernsteinTriangularPoly::norm ( ) const`

Calculates the norm of the polynomial, defined as the sum of absolute values of the coefficients divided by the number of coefficients. This norm is not the same as the L1 norm of the polynomial over the triangle, but is an upper bound for it.

29.32.3.5 `void Go::BernsteinTriangularPoly::normalize ( )`

Normalizes the polynomial by dividing all coefficients with [norm\(\)](#), which is related to (but not identical to) the L1 norm of the polynomial.

See also

[norm\(\)](#)

29.32.3.6 `template<typename T> T Go::BernsteinTriangularPoly::operator() ( const Array< T, 3 > & u ) const`  
`[inline]`

Evaluation operator, based on the triangular de Casteljau algorithm. Adapted from 'tri\_decas' in Farin: "Curves and Surfaces for CAGD".

## Parameters

|     |                                                  |
|-----|--------------------------------------------------|
| $u$ | parameter point given in barycentric coordinates |
|-----|--------------------------------------------------|

## Returns

the value of the polynomial at  $u$

Definition at line 107 of file BernsteinTriangularPoly.h.

**29.32.3.7 BernsteinTriangularPoly& Go::BernsteinTriangularPoly::operator\*=( const BernsteinTriangularPoly & *poly* )**

Multiplication with another polynomial.

**29.32.3.8 BernsteinTriangularPoly& Go::BernsteinTriangularPoly::operator\*=( double *c* )**

Multiplication with a scalar.

**29.32.3.9 BernsteinTriangularPoly& Go::BernsteinTriangularPoly::operator+=( const BernsteinTriangularPoly & *poly* )**

Addition with another polynomial.

**29.32.3.10 BernsteinTriangularPoly& Go::BernsteinTriangularPoly::operator+=( double *c* )**

Addition with a scalar.

**29.32.3.11 BernsteinTriangularPoly& Go::BernsteinTriangularPoly::operator-=( const BernsteinTriangularPoly & *poly* )**

Subtraction with another polynomial.

**29.32.3.12 BernsteinTriangularPoly& Go::BernsteinTriangularPoly::operator-=( double *c* )**

Subtraction with a scalar.

**29.32.3.13 BernsteinTriangularPoly& Go::BernsteinTriangularPoly::operator/=( double *c* )**

Division with a scalar.

29.32.3.14 `const double Go::BernsteinTriangularPoly::operator[] ( int num ) const` `[inline]`

Access function for coefficients.

Definition at line 95 of file BernsteinTriangularPoly.h.

29.32.3.15 `double& Go::BernsteinTriangularPoly::operator[] ( int num )` `[inline]`

Access function for coefficients.

Definition at line 98 of file BernsteinTriangularPoly.h.

29.32.3.16 `void Go::BernsteinTriangularPoly::read ( std::istream & is )`

Read from an input stream.

29.32.3.17 `void Go::BernsteinTriangularPoly::write ( std::ostream & os ) const`

Write to an output stream.

The documentation for this class was generated from the following file:

- [implicitization/include/GoTools/implicitization/BernsteinTriangularPoly.h](#)

## 29.33 Go::BezierTriangle< N > Class Template Reference

Not documented.

```
#include <BezierTriangle.h>
```

### Public Member Functions

- [BezierTriangle](#) ()  
*Default constructor.*
- [BezierTriangle](#) (int deg, [const Array](#)< [double](#), N > \*cp)  
*Constructor.*
- [Array](#)< [double](#), N > [eval](#) ([Vector3D](#) beta)  
*Not documented.*
- [Array](#)< [double](#), N > [evalderiv](#) ([Vector3D](#) beta, [Vector3D](#) dir)  
*Not documented.*
- void [interpolate](#) (int deg, [const Vector3D](#) \*nodes, [const Array](#)< [double](#), N > \*values)  
*Not documented.*

### 29.33.1 Detailed Description

```
template<int N>
class Go::BezierTriangle< N >
```

Not documented.

Definition at line 124 of file BezierTriangle.h.

### 29.33.2 Constructor & Destructor Documentation

29.33.2.1 `template<int N> Go::BezierTriangle< N >::BezierTriangle ( ) [inline]`

Default constructor.

Definition at line 128 of file BezierTriangle.h.

29.33.2.2 `template<int N> Go::BezierTriangle< N >::BezierTriangle ( int deg, const Array< double, N > * cp ) [inline]`

Constructor.

Definition at line 131 of file BezierTriangle.h.

### 29.33.3 Member Function Documentation

29.33.3.1 `template<int N> Array<double, N> Go::BezierTriangle< N >::eval ( Vector3D beta ) [inline]`

Not documented.

Definition at line 145 of file BezierTriangle.h.

29.33.3.2 `template<int N> Array<double, N> Go::BezierTriangle< N >::evalderiv ( Vector3D beta, Vector3D dir ) [inline]`

Not documented.

Definition at line 155 of file BezierTriangle.h.

29.33.3.3 `template<int N> void Go::BezierTriangle< N >::interpolate ( int deg, const Vector3D * nodes, const Array< double, N > * values ) [inline]`

Not documented.

Definition at line 171 of file BezierTriangle.h.

The documentation for this class was generated from the following file:

- [implicitization/include/GoTools/implicitization/BezierTriangle.h](#)

## 29.34 Go::Binomial Class Reference

```
#include <Binomial.h>
```

### Public Types

- typedef std::vector< double >::iterator iter  
*Useful iterator.*

### Public Member Functions

- Binomial ()  
*Default constructor.*
- Binomial (int n)
- double operator() (int n, int i)
- iter operator[] (int n)
- double trinomial (int n, int i, int j)
- double quadrimomial (int n, int i, int j, int k)

#### 29.34.1 Detailed Description

Class that computes the binomial coefficients.

Definition at line 54 of file Binomial.h.

#### 29.34.2 Member Typedef Documentation

##### 29.34.2.1 typedef std::vector<double>::iterator Go::Binomial::iter

Useful iterator.

Definition at line 57 of file Binomial.h.

#### 29.34.3 Constructor & Destructor Documentation

##### 29.34.3.1 Go::Binomial::Binomial( ) [inline]

Default constructor.

Definition at line 60 of file Binomial.h.

##### 29.34.3.2 Go::Binomial::Binomial( int n ) [inline]

Constructor that creates an initial Pascal's triangle to some specified level.

## Parameters

|     |                                                     |
|-----|-----------------------------------------------------|
| $n$ | the level up to which you want to make the triangle |
|-----|-----------------------------------------------------|

Definition at line 71 of file Binomial.h.

### 29.34.4 Member Function Documentation

#### 29.34.4.1 `double Go::Binomial::operator()( int $n$ , int $i$ ) [inline]`

Evaluates a binomial coefficient. If the current table is too small, the table will be expanded accordingly.

## Parameters

|     |            |
|-----|------------|
| $n$ | an integer |
| $i$ | an integer |

## Returns

$$\binom{n}{i}$$

Definition at line 82 of file Binomial.h.

#### 29.34.4.2 `iter Go::Binomial::operator[]( int $n$ ) [inline]`

Gives an iterator to the first element of the vector containing the  $n$ 'th line of Pascal's triangle. Expands the table if necessary.

## Parameters

|     |            |
|-----|------------|
| $n$ | an integer |
|-----|------------|

## Returns

an iterator `iter` such that `iter[i] = operator()(n,i)`.

Definition at line 96 of file Binomial.h.

#### 29.34.4.3 `double Go::Binomial::quadrinomial ( int $n$ , int $i$ , int $j$ , int $k$ ) [inline]`

Evaluates a quadrinomial coefficient. This function should be considered a temporary solution.

## Parameters

|     |            |
|-----|------------|
| $n$ | an integer |
| $i$ | an integer |
| $j$ | an integer |
| $k$ | an integer |

## Returns

$$\frac{n!}{i!j!k!(n-i-j-k)!}$$

Definition at line 127 of file Binomial.h.

**29.34.4.4** `double Go::Binomial::trinomial ( int  $n$ , int  $i$ , int  $j$  ) [inline]`

Evaluates a trinomial coefficient. This function should be considered a temporary solution.

## Parameters

|     |            |
|-----|------------|
| $n$ | an integer |
| $i$ | an integer |
| $j$ | an integer |

## Returns

$$\frac{n!}{i!j!(n-i-j)!}$$

Definition at line 110 of file Binomial.h.

The documentation for this class was generated from the following file:

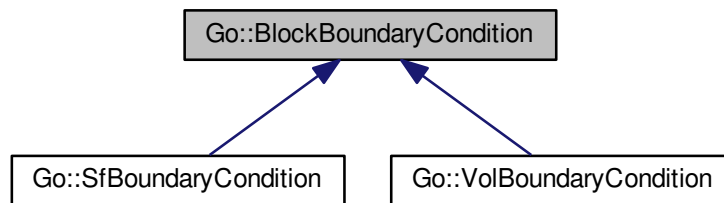
- `implicitization/include/GoTools/implicitization/Binomial.h`

## 29.35 Go::BlockBoundaryCondition Class Reference

```
#include <BlockBoundaryCondition.h>
```



Inheritance diagram for Go::BlockBoundaryCondition:



## Public Member Functions

- [BlockBoundaryCondition](#) ([BdConditionType](#) type)
- virtual [~BlockBoundaryCondition](#) ()
- virtual void [getCoefficientsEnumeration](#) (std::vector< int > &local\_enumeration)=0
- virtual void [getCoefficientsEnumeration](#) (std::vector< int > &local\_enumeration\_bd, std::vector< int > &local\_enumeration\_bd2)=0
- [BdConditionType](#) [getBdConditionType](#) () const
- bool [isDirichlet](#) () const
- int [getSolutionSpaceldx](#) () const
- virtual void [getBdCoefficients](#) (std::vector< std::pair< int, [Point](#) > > &coefs)=0
- virtual void [getBdCoefficients](#) (std::vector< std::pair< int, [Point](#) > > &coefs\_bd, std::vector< std::pair< int, [Point](#) > > &coefs\_inner)=0
- virtual void [update](#) ()=0
- virtual void [updateBoundaryValue](#) ([BdCondFuncor](#) \*fbd)=0
- virtual [tpTolerances](#) [getTolerances](#) () const =0

## Protected Attributes

- [BdConditionType](#) type\_

### 29.35.1 Detailed Description

Definition at line 58 of file [BlockBoundaryCondition.h](#).

### 29.35.2 Constructor & Destructor Documentation

29.35.2.1 [Go::BlockBoundaryCondition::BlockBoundaryCondition](#) ( [BdConditionType](#) type )

29.35.2.2 virtual [Go::BlockBoundaryCondition::~~BlockBoundaryCondition](#) ( ) [virtual]

### 29.35.3 Member Function Documentation

29.35.3.1 virtual void [Go::BlockBoundaryCondition::getBdCoefficients](#) ( std::vector< std::pair< int, [Point](#) > > & coefs ) [pure virtual]

Implemented in [Go::SfBoundaryCondition](#), and [Go::VolBoundaryCondition](#).

29.35.3.2 `virtual void Go::BlockBoundaryCondition::getBdCoefficients ( std::vector< std::pair< int, Point > > & coefs_bd, std::vector< std::pair< int, Point > > & coefs_inner ) [pure virtual]`

Implemented in [Go::SfBoundaryCondition](#), and [Go::VolBoundaryCondition](#).

29.35.3.3 `BdConditionType Go::BlockBoundaryCondition::getBdConditionType ( ) const`

29.35.3.4 `virtual void Go::BlockBoundaryCondition::getCoefficientsEnumeration ( std::vector< int > & local_enumeration ) [pure virtual]`

Implemented in [Go::SfBoundaryCondition](#), and [Go::VolBoundaryCondition](#).

29.35.3.5 `virtual void Go::BlockBoundaryCondition::getCoefficientsEnumeration ( std::vector< int > & local_enumeration_bd, std::vector< int > & local_enumeration_bd2 ) [pure virtual]`

Implemented in [Go::SfBoundaryCondition](#), and [Go::VolBoundaryCondition](#).

29.35.3.6 `int Go::BlockBoundaryCondition::getSolutionSpaceldx ( ) const`

29.35.3.7 `virtual tpTolerances Go::BlockBoundaryCondition::getTolerances ( ) const [pure virtual]`

Implemented in [Go::SfBoundaryCondition](#), and [Go::VolBoundaryCondition](#).

29.35.3.8 `bool Go::BlockBoundaryCondition::isDirichlet ( ) const`

29.35.3.9 `virtual void Go::BlockBoundaryCondition::update ( ) [pure virtual]`

Implemented in [Go::VolBoundaryCondition](#), and [Go::SfBoundaryCondition](#).

29.35.3.10 `virtual void Go::BlockBoundaryCondition::updateBoundaryValue ( BdCondFuncor * fbd ) [pure virtual]`

Implemented in [Go::SfBoundaryCondition](#), and [Go::VolBoundaryCondition](#).

## 29.35.4 Member Data Documentation

29.35.4.1 `BdConditionType Go::BlockBoundaryCondition::type_ [protected]`

Definition at line 109 of file `BlockBoundaryCondition.h`.

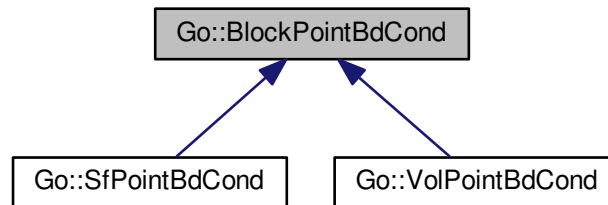
The documentation for this class was generated from the following file:

- [isogeometric\\_model/include/GoTools/isogeometric\\_model/BlockBoundaryCondition.h](#)

## 29.36 Go::BlockPointBdCond Class Reference

```
#include <BlockPointBdCond.h>
```

Inheritance diagram for Go::BlockPointBdCond:



### Public Member Functions

- [BlockPointBdCond](#) ()
- virtual [~BlockPointBdCond](#) ()
- virtual int [getSolutionSpaceIdx](#) () const
- virtual void [getCoefficientsEnumeration](#) (std::vector< int > &local\_enumeration) const =0
- virtual [Point](#) [getConditionValue](#) () const =0
- virtual void [getInterpolationFactors](#) (std::vector< std::pair< int, double > > &factors) const =0

### 29.36.1 Detailed Description

Definition at line 51 of file [BlockPointBdCond.h](#).

### 29.36.2 Constructor & Destructor Documentation

29.36.2.1 [Go::BlockPointBdCond::BlockPointBdCond](#) ( )

29.36.2.2 virtual [Go::BlockPointBdCond::~~BlockPointBdCond](#) ( ) [virtual]

### 29.36.3 Member Function Documentation

29.36.3.1 virtual void [Go::BlockPointBdCond::getCoefficientsEnumeration](#) ( std::vector< int > & local\_enumeration ) const  
[pure virtual]

Implemented in [Go::SfPointBdCond](#), and [Go::VolPointBdCond](#).

29.36.3.2 `virtual Point Go::BlockPointBdCond::getConditionValue ( ) const` [pure virtual]

Implemented in [Go::SfPointBdCond](#), and [Go::VolPointBdCond](#).

29.36.3.3 `virtual void Go::BlockPointBdCond::getInterpolationFactors ( std::vector< std::pair< int, double > > & factors ) const` [pure virtual]

Implemented in [Go::SfPointBdCond](#), and [Go::VolPointBdCond](#).

29.36.3.4 `virtual int Go::BlockPointBdCond::getSolutionSpaceldx ( ) const` [virtual]

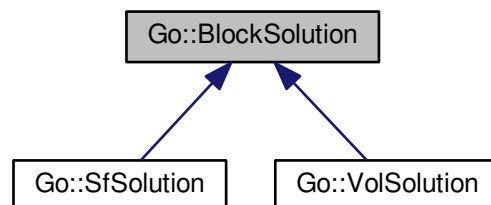
The documentation for this class was generated from the following file:

- [isogeometric\\_model/include/GoTools/isogeometric\\_model/BlockPointBdCond.h](#)

## 29.37 Go::BlockSolution Class Reference

```
#include <BlockSolution.h>
```

Inheritance diagram for `Go::BlockSolution`:



### Public Member Functions

- `virtual ~BlockSolution ()`
- `virtual SfSolution * asSfSolution ()`
- `virtual VolSolution * asVolSolution ()`
- `virtual int getNmbOfPointBdConditions () const =0`
- `virtual bool matchingSplineSpace (BlockSolution *other) const =0`
- `virtual void getMatchingCoefficients (BlockSolution *other, std::vector< std::pair< int, int > > &enumeration, int match_pos=0) const =0`
- `virtual void getBoundaryCoefficients (int boundary, std::vector< int > &enumeration) const =0`
- `virtual void getBoundaryCoefficients (int boundary, std::vector< int > &enumeration_bd, std::vector< int > &enumeration_b2) const =0`
- `virtual void makeMatchingSplineSpace (BlockSolution *other)=0`

- virtual void `increaseDegree` (int new\_degree, int paddir)=0
- virtual void `insertKnots` (const std::vector< int > &knot\_intervals, int paddir)=0
- virtual void `insertKnots` (const std::vector< double > &knots, int paddir)=0
- virtual void `erasePreEvaluatedBasisFunctions` ()=0
- virtual void `performPreEvaluation` (std::vector< std::vector< double > > &Gauss\_par)=0
- virtual `double` `getJacobian` (std::vector< int > &index\_of\_Gauss\_point) const =0
- virtual void `valuesInGaussPoint` (const std::vector< int > &index\_of\_Gauss\_point, std::vector< Point > &derivs) const =0
- virtual void `setSolutionCoefficients` (const std::vector< double > &coefs)=0
- virtual int `nmbCoefs` () const =0
- virtual int `nmbCoefs` (int paddir) const =0
- virtual int `degree` (int paddir) const =0
- virtual std::vector< double > `knots` (int paddir) const =0
- virtual std::vector< double > `distinctKnots` (int paddir) const =0
- virtual `BsplineBasis` `basis` (int paddir) const =0
- virtual int `dimension` () const =0
- virtual void `updateConditions` ()=0
- virtual `tpTolerances` `getTolerances` () const =0

### 29.37.1 Detailed Description

Definition at line 60 of file `BlockSolution.h`.

### 29.37.2 Constructor & Destructor Documentation

29.37.2.1 virtual `Go::BlockSolution::~~BlockSolution` ( ) [virtual]

### 29.37.3 Member Function Documentation

29.37.3.1 virtual `SfSolution*` `Go::BlockSolution::asSfSolution` ( ) [virtual]

Reimplemented in [Go::SfSolution](#).

29.37.3.2 virtual `VolSolution*` `Go::BlockSolution::asVolSolution` ( ) [virtual]

Reimplemented in [Go::VolSolution](#).

29.37.3.3 virtual `BsplineBasis` `Go::BlockSolution::basis` ( int paddir ) const [pure virtual]

Implemented in [Go::VolSolution](#), and [Go::SfSolution](#).

29.37.3.4 virtual int `Go::BlockSolution::degree` ( int paddir ) const [pure virtual]

Implemented in [Go::VolSolution](#), and [Go::SfSolution](#).

29.37.3.5 `virtual int Go::BlockSolution::dimension ( ) const [pure virtual]`

Implemented in [Go::VolSolution](#), and [Go::SfSolution](#).

29.37.3.6 `virtual std::vector<double> Go::BlockSolution::distinctKnots ( int pardir ) const [pure virtual]`

Implemented in [Go::VolSolution](#), and [Go::SfSolution](#).

29.37.3.7 `virtual void Go::BlockSolution::erasePreEvaluatedBasisFunctions ( ) [pure virtual]`

Implemented in [Go::VolSolution](#), and [Go::SfSolution](#).

29.37.3.8 `virtual void Go::BlockSolution::getBoundaryCoefficients ( int boundary, std::vector< int > & enumeration ) const [pure virtual]`

Implemented in [Go::VolSolution](#), and [Go::SfSolution](#).

29.37.3.9 `virtual void Go::BlockSolution::getBoundaryCoefficients ( int boundary, std::vector< int > & enumeration_bd, std::vector< int > & enumeration_b2 ) const [pure virtual]`

Implemented in [Go::VolSolution](#), and [Go::SfSolution](#).

29.37.3.10 `virtual double Go::BlockSolution::getJacobian ( std::vector< int > & index_of_Gauss_point ) const [pure virtual]`

Implemented in [Go::VolSolution](#), and [Go::SfSolution](#).

29.37.3.11 `virtual void Go::BlockSolution::getMatchingCoefficients ( BlockSolution * other, std::vector< std::pair< int, int > > & enumeration, int match_pos = 0 ) const [pure virtual]`

Implemented in [Go::VolSolution](#), and [Go::SfSolution](#).

29.37.3.12 `virtual int Go::BlockSolution::getNmbOfPointBdConditions ( ) const [pure virtual]`

Implemented in [Go::VolSolution](#), and [Go::SfSolution](#).

29.37.3.13 `virtual tpTolerances Go::BlockSolution::getTolerances ( ) const [pure virtual]`

Implemented in [Go::VolSolution](#), and [Go::SfSolution](#).

29.37.3.14 `virtual void Go::BlockSolution::increaseDegree ( int new_degree, int pardir ) [pure virtual]`

Implemented in [Go::VolSolution](#), and [Go::SfSolution](#).

29.37.3.15 `virtual void Go::BlockSolution::insertKnots ( const std::vector< int > & knot_intervals, int pardir )` [pure virtual]

Implemented in [Go::VolSolution](#), and [Go::SfSolution](#).

29.37.3.16 `virtual void Go::BlockSolution::insertKnots ( const std::vector< double > & knots, int pardir )` [pure virtual]

Implemented in [Go::VolSolution](#), and [Go::SfSolution](#).

29.37.3.17 `virtual std::vector<double> Go::BlockSolution::knots ( int pardir ) const` [pure virtual]

Implemented in [Go::VolSolution](#), and [Go::SfSolution](#).

29.37.3.18 `virtual void Go::BlockSolution::makeMatchingSplineSpace ( BlockSolution * other )` [pure virtual]

Implemented in [Go::VolSolution](#), and [Go::SfSolution](#).

29.37.3.19 `virtual bool Go::BlockSolution::matchingSplineSpace ( BlockSolution * other ) const` [pure virtual]

Implemented in [Go::VolSolution](#), and [Go::SfSolution](#).

29.37.3.20 `virtual int Go::BlockSolution::nmbCoefs ( ) const` [pure virtual]

Implemented in [Go::VolSolution](#), and [Go::SfSolution](#).

29.37.3.21 `virtual int Go::BlockSolution::nmbCoefs ( int pardir ) const` [pure virtual]

Implemented in [Go::VolSolution](#), and [Go::SfSolution](#).

29.37.3.22 `virtual void Go::BlockSolution::performPreEvaluation ( std::vector< std::vector< double > > & Gauss_par )` [pure virtual]

Implemented in [Go::VolSolution](#), and [Go::SfSolution](#).

29.37.3.23 `virtual void Go::BlockSolution::setSolutionCoefficients ( const std::vector< double > & coefs )` [pure virtual]

Implemented in [Go::VolSolution](#), and [Go::SfSolution](#).

29.37.3.24 `virtual void Go::BlockSolution::updateConditions ( )` [pure virtual]

Implemented in [Go::VolSolution](#), and [Go::SfSolution](#).

```
29.37.3.25 virtual void Go::BlockSolution::valuesInGaussPoint (const std::vector< int > & index_of_Gauss_point,
std::vector< Point > & derivs) const [pure virtual]
```

Implemented in [Go::VolSolution](#), and [Go::SfSolution](#).

The documentation for this class was generated from the following file:

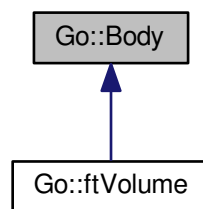
- [isogeometric\\_model/include/GoTools/isogeometric\\_model/BlockSolution.h](#)

## 29.38 Go::Body Class Reference

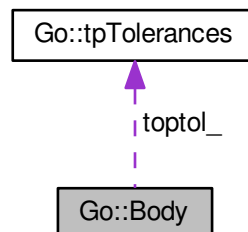
A boundary represented solid.

```
#include <Body.h>
```

Inheritance diagram for Go::Body:



Collaboration diagram for Go::Body:





## Public Member Functions

- [Body](#) ()  
*Default constructor.*
- [Body](#) (std::vector< shared\_ptr< [SurfaceModel](#) > > &shells, int material\_id=-1)  
*Constructor with multiple shells.*
- [Body](#) (shared\_ptr< [SurfaceModel](#) > shell, int material\_id=-1)  
*Constructor with one outer shell.*
- [~Body](#) ()  
*Destructor.*
- void [addshell](#) (shared\_ptr< [SurfaceModel](#) > shell)
- shared\_ptr< [SurfaceModel](#) > [getOuterShell](#) () const  
*Fetch the outer shell belonging to this solid.*
- std::vector< shared\_ptr< [SurfaceModel](#) > > [getAllShells](#) () const
- int [nmbOfShells](#) () const  
*The number of shells belonging to this solid.*
- shared\_ptr< [SurfaceModel](#) > [getShell](#) (int idx) const
- void [setMaterial](#) (int material\_id)  
*Material information.*
- bool [hasMaterialInfo](#) () const
- int [getMaterial](#) () const
- int [nmbOfFaces](#) () const  
*Total number of faces.*
- std::vector< shared\_ptr< [Vertex](#) > > [vertices](#) () const  
*Fetch all vertices.*
- virtual [BoundingBox](#) [boundingBox](#) () const  
*The bounding box corresponding to this solid.*
- virtual bool [areNeighbours](#) ([Body](#) \*other, shared\_ptr< [ftSurface](#) > &bd\_face1, shared\_ptr< [ftSurface](#) > &bd\_face2, int adj\_idx=0) const  
*Check if two bodies are neighbours.*
- void [getAdjacentBodies](#) (std::vector< [Body](#) \* > &neighbours)
- void [eraseBodyAdjacency](#) ()  
*Remove adjacency information to other bodies.*
- bool [isInside](#) (const [Point](#) &pnt) const  
*Check if a point lies inside this body.*
- shared\_ptr< [SurfaceModel](#) > [getShell](#) ([ftSurface](#) \*face) const  
*Find the shell containing a given face (if any)*
- [tpTolerances](#) [getTolerances](#) ()  
*Return topology tolerances.*

## Protected Member Functions

- void [addBodyPointers](#) ()

## Protected Attributes

- std::vector< shared\_ptr< [SurfaceModel](#) > > [shells\\_](#)  
*Outer and possibly inner void shells.*
- int [material\\_id\\_](#)  
*Possibility to store material information (-1 = not set)*
- [tpTolerances](#) [toptol\\_](#)  
*Tolerances used in topology analysis.*

### 29.38.1 Detailed Description

A boundary represented solid.

Definition at line 53 of file Body.h.

### 29.38.2 Constructor & Destructor Documentation

#### 29.38.2.1 Go::Body::Body ( )

Default constructor.

#### 29.38.2.2 Go::Body::Body ( std::vector< shared\_ptr< SurfaceModel > > & shells, int material\_id = -1 )

Constructor with multiple shells.

#### 29.38.2.3 Go::Body::Body ( shared\_ptr< SurfaceModel > shell, int material\_id = -1 )

Constructor with one outer shell.

#### 29.38.2.4 Go::Body::~~Body ( )

Destructor.

### 29.38.3 Member Function Documentation

#### 29.38.3.1 void Go::Body::addBodyPointers ( ) [protected]

#### 29.38.3.2 void Go::Body::addshell ( shared\_ptr< SurfaceModel > shell )

Add another shell to the solid description. Used in connection with building the solid

#### 29.38.3.3 virtual bool Go::Body::areNeighbours ( Body \* other, shared\_ptr< ftSurface > & bd\_face1, shared\_ptr< ftSurface > & bd\_face2, int adj\_idx = 0 ) const [virtual]

Check if two bodies are neighbours.

#### 29.38.3.4 virtual BoundingBox Go::Body::boundingBox ( ) const [virtual]

The bounding box corresponding to this solid.

Reimplemented in [Go::ftVolume](#).

29.38.3.5 void Go::Body::eraseBodyAdjacency ( )

Remove adjacency information to other bodies.

29.38.3.6 void Go::Body::getAdjacentBodies ( std::vector< Body \* > & neighbours )

Fetch all bodies adjacent to this one. This function is relevant in an assembly system or in an isogeometric setting

29.38.3.7 std::vector<shared\_ptr<SurfaceModel> > Go::Body::getAllShells ( ) const [inline]

Fetch all shells belonging to this solid. The first shell corresponds to the outer boundary of the solid, any subsequent shells are void

Definition at line 96 of file Body.h.

29.38.3.8 int Go::Body::getMaterial ( ) const [inline]

Definition at line 129 of file Body.h.

29.38.3.9 shared\_ptr<SurfaceModel> Go::Body::getOuterShell ( ) const [inline]

Fetch the outer shell belonging to this solid.

Definition at line 84 of file Body.h.

29.38.3.10 shared\_ptr<SurfaceModel> Go::Body::getShell ( int idx ) const [inline]

Fetch a specified shell. Index zero corresponds to the outer shell

Definition at line 109 of file Body.h.

29.38.3.11 shared\_ptr<SurfaceModel> Go::Body::getShell ( ftSurface \* face ) const

Find the shell containing a given face (if any)

29.38.3.12 tpTolerances Go::Body::getTolerances ( ) [inline]

Return topology tolerances.

Definition at line 161 of file Body.h.

29.38.3.13 bool Go::Body::hasMaterialInfo ( ) const [inline]

Definition at line 124 of file Body.h.

29.38.3.14 `bool Go::Body::isInside ( const Point & pnt ) const`

Check if a point lies inside this body.

29.38.3.15 `int Go::Body::nmbOfFaces ( ) const`

Total number of faces.

29.38.3.16 `int Go::Body::nmbOfShells ( ) const [inline]`

The number of shells belonging to this solid.

Definition at line 102 of file Body.h.

29.38.3.17 `void Go::Body::setMaterial ( int material_id ) [inline]`

Material information.

Definition at line 119 of file Body.h.

29.38.3.18 `std::vector<shared_ptr<Vertex> > Go::Body::vertices ( ) const`

Fetch all vertices.

## 29.38.4 Member Data Documentation

29.38.4.1 `int Go::Body::material_id_ [protected]`

Possibility to store material information (-1 = not set)

Definition at line 175 of file Body.h.

29.38.4.2 `std::vector<shared_ptr<SurfaceModel> > Go::Body::shells_ [protected]`

Outer and possibly inner void shells.

Definition at line 172 of file Body.h.

29.38.4.3 `tpTolerances Go::Body::toptol_ [protected]`

Tolerances used in topology analysis.

Definition at line 181 of file Body.h.

The documentation for this class was generated from the following file:

- `compositemodel/include/GoTools/compositemodel/Body.h`

## 29.39 bool Class Reference

```
#include <boolean.h>
```

### Public Member Functions

- [bool \(const int b\)](#)
- [bool \(const void \\*b\)](#)
- [bool \(\)](#)
- [operator int \(\) const](#)
- [int operator! \(\) const](#)

### 29.39.1 Detailed Description

Definition at line 13 of file boolean.h.

### 29.39.2 Constructor & Destructor Documentation

**29.39.2.1** `bool::bool ( const int b )` [\[inline\]](#)

Definition at line 17 of file boolean.h.

**29.39.2.2** `bool::bool ( const void * b )` [\[inline\]](#)

Definition at line 18 of file boolean.h.

**29.39.2.3** `bool::bool ( )` [\[inline\]](#)

Definition at line 19 of file boolean.h.

### 29.39.3 Member Function Documentation

**29.39.3.1** `bool::operator int ( )const` [\[inline\]](#)

Definition at line 20 of file boolean.h.

**29.39.3.2** `int bool::operator! ( )const` [\[inline\]](#)

Definition at line 21 of file boolean.h.

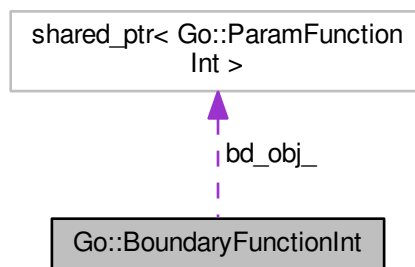
The documentation for this class was generated from the following file:

- [newmat/include/boolean.h](#)

## 29.40 Go::BoundaryFunctionInt Struct Reference

```
#include <BoundaryFunctionInt.h>
```

Collaboration diagram for Go::BoundaryFunctionInt:



### Public Member Functions

- [BoundaryFunctionInt](#) ([shared\\_ptr< ParamFunctionInt >](#) *bd\_obj*, *int dir*, *double par*)
- *int* [getDir](#) ()
- *double* [getPar](#) ()
- [shared\\_ptr< ParamFunctionInt >](#) [getObject](#) ()

### Public Attributes

- [shared\\_ptr< ParamFunctionInt >](#) *bd\_obj\_*
- *int* *pardir\_*
- *double* *par\_*

#### 29.40.1 Detailed Description

This struct is a helper struct that bundles boundary information for an object of type [ParamFunctionInt](#).

Definition at line 57 of file [BoundaryFunctionInt.h](#).

#### 29.40.2 Constructor & Destructor Documentation

29.40.2.1 [Go::BoundaryFunctionInt::BoundaryFunctionInt](#) ([shared\\_ptr< ParamFunctionInt >](#) *bd\_obj*, *int dir*, *double par*) [*inline*]

Constructor

## Parameters

|               |                                                     |
|---------------|-----------------------------------------------------|
| <i>bd_obj</i> | shared pointer to the object of interest            |
| <i>dir</i>    | parameter direction that specifies the boundary     |
| <i>par</i>    | the value of the relevant parameter at the boundary |

Definition at line 68 of file BoundaryFunctionInt.h.

### 29.40.3 Member Function Documentation

#### 29.40.3.1 `int Go::BoundaryFunctionInt::getDir ( ) [inline]`

Get the parameter direction that specifies the boundary

## Returns

the parameter direction

Definition at line 78 of file BoundaryFunctionInt.h.

#### 29.40.3.2 `shared_ptr<ParamFunctionInt> Go::BoundaryFunctionInt::getObject ( ) [inline]`

Get the object which the boundary belongs to

## Returns

shared pointer to the object

Definition at line 88 of file BoundaryFunctionInt.h.

#### 29.40.3.3 `double Go::BoundaryFunctionInt::getPar ( ) [inline]`

Get the value of the relevant parameter at the boundary

## Returns

the parameter value

Definition at line 83 of file BoundaryFunctionInt.h.

### 29.40.4 Member Data Documentation

#### 29.40.4.1 `shared_ptr<ParamFunctionInt> Go::BoundaryFunctionInt::bd_obj_`

Definition at line 59 of file BoundaryFunctionInt.h.

#### 29.40.4.2 double Go::BoundaryFunctionInt::par\_

Definition at line 62 of file BoundaryFunctionInt.h.

#### 29.40.4.3 int Go::BoundaryFunctionInt::pdir\_

Definition at line 60 of file BoundaryFunctionInt.h.

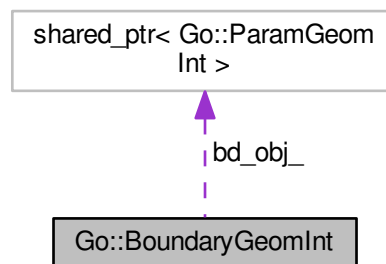
The documentation for this struct was generated from the following file:

- [intersections/include/GoTools/intersections/BoundaryFunctionInt.h](#)

## 29.41 Go::BoundaryGeomInt Struct Reference

```
#include <BoundaryGeomInt.h>
```

Collaboration diagram for Go::BoundaryGeomInt:



### Public Member Functions

- [BoundaryGeomInt](#) (shared\_ptr< [ParamGeomInt](#) > bd\_obj, int dir, double par)
- [~BoundaryGeomInt](#) ()  
*Destructor.*
- int [getDir](#) ()
- double [getPar](#) ()
- shared\_ptr< [ParamGeomInt](#) > [getObject](#) ()

### Public Attributes

- shared\_ptr< [ParamGeomInt](#) > [bd\\_obj\\_](#)
- int [pdir\\_](#)
- double [par\\_](#)



### 29.41.1 Detailed Description

This struct is a helper struct that bundles boundary information for an object of type [ParamGeomInt](#).

Definition at line 56 of file `BoundaryGeomInt.h`.

### 29.41.2 Constructor & Destructor Documentation

**29.41.2.1** `Go::BoundaryGeomInt::BoundaryGeomInt ( shared_ptr< ParamGeomInt > bd_obj, int dir, double par )`  
[`inline`]

Constructor

Parameters

|               |                                                     |
|---------------|-----------------------------------------------------|
| <i>bd_obj</i> | shared pointer to the object of interest            |
| <i>dir</i>    | parameter direction that specifies the boundary     |
| <i>par</i>    | the value of the relevant parameter at the boundary |

Definition at line 67 of file `BoundaryGeomInt.h`.

**29.41.2.2** `Go::BoundaryGeomInt::~~BoundaryGeomInt ( )` [`inline`]

Destructor.

Definition at line 76 of file `BoundaryGeomInt.h`.

### 29.41.3 Member Function Documentation

**29.41.3.1** `int Go::BoundaryGeomInt::getDir ( )` [`inline`]

Get the parameter direction that specifies the boundary

Returns

the parameter direction

Definition at line 80 of file `BoundaryGeomInt.h`.

**29.41.3.2** `shared_ptr<ParamGeomInt> Go::BoundaryGeomInt::getObject ( )` [`inline`]

Get the object which the boundary belongs to

Returns

shared pointer to the object

Definition at line 90 of file `BoundaryGeomInt.h`.

29.41.3.3 **double** Go::BoundaryGeomInt::getPar ( ) [inline]

Get the value of the relevant parameter at the boundary

#### Returns

the parameter value

Definition at line 85 of file BoundaryGeomInt.h.

### 29.41.4 Member Data Documentation

29.41.4.1 **shared\_ptr<ParamGeomInt>** Go::BoundaryGeomInt::bd\_obj\_

Definition at line 58 of file BoundaryGeomInt.h.

29.41.4.2 **double** Go::BoundaryGeomInt::par\_

Definition at line 61 of file BoundaryGeomInt.h.

29.41.4.3 **int** Go::BoundaryGeomInt::paddir\_

Definition at line 59 of file BoundaryGeomInt.h.

The documentation for this struct was generated from the following file:

- [intersections/include/GoTools/intersections/BoundaryGeomInt.h](#)

## 29.42 Go::BoundaryIntersectionData Struct Reference

```
#include <IntersectionPool.h>
```

### Public Attributes

- **int** [dir](#) [2]  
*Running parameter direction in object 1 and 2.*
- **double** [par](#) [2]  
*Parameter value for the fixed parameter in object 1 and 2 (-1 if the object does not have a fixed parameter, i.e. when the object is a curve).*
- **std::vector< shared\_ptr< [IntersectionPoint](#) > >** [pts](#)  
*Chain of IntersectionPoints making up the intersection.*

### 29.42.1 Detailed Description

Helper struct to be used with [IntersectionPool::intersectAlongCommonBoundary\(\)](#)

Definition at line 968 of file IntersectionPool.h.

### 29.42.2 Member Data Documentation

#### 29.42.2.1 int Go::BoundaryIntersectionData::dir[2]

Running parameter direction in object 1 and 2.

Definition at line 972 of file IntersectionPool.h.

#### 29.42.2.2 double Go::BoundaryIntersectionData::par[2]

Parameter value for the fixed parameter in object 1 and 2 (-1 if the object does not have a fixed parameter, i.e. when the object is a curve).

Definition at line 977 of file IntersectionPool.h.

#### 29.42.2.3 std::vector<shared\_ptr<IntersectionPoint> > Go::BoundaryIntersectionData::pts

Chain of IntersectionPoints making up the intersection.

Definition at line 980 of file IntersectionPool.h.

The documentation for this struct was generated from the following file:

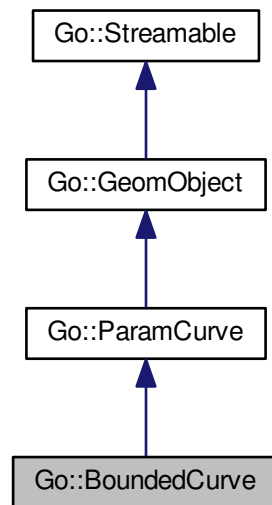
- intersections/include/GoTools/intersections/[IntersectionPool.h](#)

## 29.43 Go::BoundedCurve Class Reference

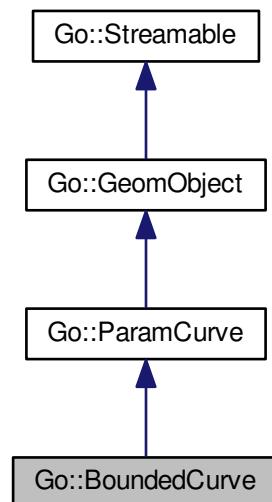
A bounded curve. Both parameter values and end points may be given to define the boundaries. Assuming that both points prefer parameter, or both points prefer points. Typically used to bound infinite curves, for instance lines.

```
#include <BoundedCurve.h>
```

Inheritance diagram for Go::BoundedCurve:



Collaboration diagram for Go::BoundedCurve:



## Public Member Functions

- [BoundedCurve \(\)](#)

- [BoundedCurve](#) ([shared\\_ptr](#)< [ParamCurve](#) > [curve](#), [bool](#) [prefer\\_bd\\_par](#), [double](#) [start\\_par](#), [double](#) [end\\_par](#), [Point](#) [start\\_pt](#), [Point](#) [end\\_pt](#))
- [BoundedCurve](#) ([shared\\_ptr](#)< [ParamCurve](#) > [curve](#), [Point](#) [start\\_pt](#), [Point](#) [end\\_pt](#))
- [BoundedCurve](#) ([shared\\_ptr](#)< [ParamCurve](#) > [curve](#), [double](#) [start\\_par](#), [double](#) [end\\_par](#))
- virtual [~BoundedCurve](#) ()
  - virtual destructor - ensures safe inheritance*
- virtual void [read](#) ([std::istream](#) &[is](#))
- virtual void [write](#) ([std::ostream](#) &[os](#)) [const](#)
- virtual [BoundingBox](#) [boundingBox](#) () [const](#)
  - Return the object's bounding box.*
- virtual int [dimension](#) () [const](#)
  - Return the dimension of the space in which the object lies (usually 2 or 3)*
- virtual [ClassType](#) [instanceType](#) () [const](#)
  - Return the class type identifier of a given, derived instance of [GeomObject](#).*
- virtual [BoundedCurve](#) \* [clone](#) () [const](#)
- virtual void [point](#) ([Point](#) &[pt](#), [double](#) [tpar](#)) [const](#)
- virtual void [point](#) ([std::vector](#)< [Point](#) > &[pts](#), [double](#) [tpar](#), int [derivs](#), [bool](#) [from\\_right=true](#)) [const](#)
- virtual [double](#) [startparam](#) () [const](#)
- virtual [double](#) [endparam](#) () [const](#)
- virtual void [reverseParameterDirection](#) ([bool](#) [switchparam=false](#))
- virtual void [setParameterInterval](#) ([double](#) [t1](#), [double](#) [t2](#))
- virtual [SplineCurve](#) \* [geometryCurve](#) ()
- virtual [bool](#) [isDegenerate](#) ([double](#) [degenerate\\_epsilon](#))
- virtual [BoundedCurve](#) \* [subCurve](#) ([double](#) [from\\_par](#), [double](#) [to\\_par](#), [double](#) [fuzzy=DEFAULT\\_PARAMETER\\_↵](#)  
[R\\_EPSILON](#)) [const](#)
- virtual [DirectionCone](#) [directionCone](#) () [const](#)
- virtual void [appendCurve](#) ([ParamCurve](#) \*[cv](#), [bool](#) [reparam=true](#))
- virtual void [appendCurve](#) ([ParamCurve](#) \*[cv](#), int [continuity](#), [double](#) &[dist](#), [bool](#) [reparam=true](#))
- virtual void [closestPoint](#) ([const Point](#) &[pt](#), [double](#) [tmin](#), [double](#) [tmax](#), [double](#) &[clo\\_t](#), [Point](#) &[clo\\_pt](#), [double](#) &[clo\\_dist](#), [double const](#) \*[seed=0](#)) [const](#)
- virtual [double](#) [length](#) ([double](#) [tol](#))
- void [setParamBounds](#) ([double](#) [startpar](#), [double](#) [endpar](#))
- [shared\\_ptr](#)< [ParamCurve](#) > [underlyingCurve](#) () [const](#)
- virtual [bool](#) [isAxisRotational](#) ([Point](#) &[centre](#), [Point](#) &[axis](#), [Point](#) &[vec](#), [double](#) &[angle](#))
- virtual [bool](#) [isLinear](#) ([Point](#) &[dir](#), [double](#) [tol](#))
  - Check if the curve is linear.*
- virtual [bool](#) [isInPlane](#) ([const Point](#) &[loc](#), [const Point](#) &[axis](#), [double](#) [eps](#), [Point](#) &[normal](#)) [const](#)
  - Check if the curve lies in a plane passing through a given axis.*

### Static Public Member Functions

- static [ClassType](#) [classType](#) ()

### Additional Inherited Members

#### 29.43.1 Detailed Description

A bounded curve. Both parameter values and end points may be given to define the boundaries. Assuming that both points prefer parameter, or both points prefer points. Typically used to bound infinite curves, for instance lines.

Definition at line 56 of file [BoundedCurve.h](#).

## 29.43.2 Constructor & Destructor Documentation

### 29.43.2.1 `Go::BoundedCurve::BoundedCurve ( ) [inline]`

Default constructor. Constructs an uninitialized [Line](#) which can only be assigned to or read into.

Definition at line 62 of file `BoundedCurve.h`.

### 29.43.2.2 `Go::BoundedCurve::BoundedCurve ( shared_ptr< ParamCurve > curve, bool prefer_bd_par, double start_par, double end_par, Point start_pt, Point end_pt )`

Constructor. Input is start point and end point. Assumed to lie on curve (or at least close to it).

### 29.43.2.3 `Go::BoundedCurve::BoundedCurve ( shared_ptr< ParamCurve > curve, Point start_pt, Point end_pt )`

Constructor. Input is start point and end point. Assumed to lie on curve (or at least close to it). Only position given.

### 29.43.2.4 `Go::BoundedCurve::BoundedCurve ( shared_ptr< ParamCurve > curve, double start_par, double end_par )`

Constructor. Input is start point and end point. Assumed to lie on curve (or at least close to it). Only parameter value of curve given.

### 29.43.2.5 `virtual Go::BoundedCurve::~~BoundedCurve ( ) [virtual]`

virtual destructor - ensures safe inheritance

## 29.43.3 Member Function Documentation

### 29.43.3.1 `virtual void Go::BoundedCurve::appendCurve ( ParamCurve * cv, bool reparam = true ) [virtual]`

append a curve to this curve, with eventual reparametrization NB: This virtual member function currently only works for `SplineCurves` and `CurveOnSurfaces`. Moreover, 'this' curve and the 'cv' curve must be of the same type.

#### Parameters

|                |                                                          |
|----------------|----------------------------------------------------------|
| <i>cv</i>      | the curve to append to 'this' curve.                     |
| <i>reparam</i> | specify whether or not there should be reparametrization |

Implements [Go::ParamCurve](#).

### 29.43.3.2 `virtual void Go::BoundedCurve::appendCurve ( ParamCurve * cv, int continuity, double & dist, bool reparam = true ) [virtual]`

append a curve to this curve, with eventual reparametrization

## Parameters

|                   |                                                                                                       |
|-------------------|-------------------------------------------------------------------------------------------------------|
| <i>cv</i>         | the curve to append to 'this' curve.                                                                  |
| <i>continuity</i> | the required continuity at the transition. Can be $G^{(-1)}$ and upwards.                             |
| <i>dist</i>       | a measure of the local distortion around the transition in order to achieve the specified continuity. |
| <i>reparam</i>    | specify whether or not there should be reparametrization                                              |

Implements [Go::ParamCurve](#).

29.43.3.3 `virtual BoundingBox Go::BoundedCurve::boundingBox ( ) const [virtual]`

Return the object's bounding box.

Implements [Go::GeomObject](#).

29.43.3.4 `static ClassType Go::BoundedCurve::classType ( ) [static]`

29.43.3.5 `virtual BoundedCurve* Go::BoundedCurve::clone ( ) const [virtual]`

The clone-function is inherited from [GeomObject](#), but overridden here to get a covariant return type (for those compilers that allow this).

Implements [Go::ParamCurve](#).

29.43.3.6 `virtual void Go::BoundedCurve::closestPoint ( const Point & pt, double tmin, double tmax, double & clo_t, Point & clo_pt, double & clo_dist, double const * seed = 0 ) const [virtual]`

Compute the closest point from an interval of this curve to a specified point.

## Parameters

|                 |                                                                                                                                          |
|-----------------|------------------------------------------------------------------------------------------------------------------------------------------|
| <i>pt</i>       | point we want to find the closest point to                                                                                               |
| <i>tmin</i>     | start parameter of search interval                                                                                                       |
| <i>tmax</i>     | end parameter of search interval                                                                                                         |
| <i>clo_t</i>    | upon function return, 'clo_t' will contain the parameter value of the closest point found.                                               |
| <i>clo_pt</i>   | upon function return, 'clo_pt' will contain the position of the closest point found.                                                     |
| <i>clo_dist</i> | upon function return, 'clo_dist' will contain the distance between 'pt' and the closest point found.                                     |
| <i>seed</i>     | pointer to initial guess value, provided by the user (can be 0, for which the algorithm will determine a (hopefully) reasonable choice). |

Implements [Go::ParamCurve](#).

29.43.3.7 `virtual int Go::BoundedCurve::dimension ( ) const [virtual]`

Return the dimension of the space in which the object lies (usually 2 or 3)

Implements [Go::GeomObject](#).

29.43.3.8 `virtual DirectionCone Go::BoundedCurve::directionCone ( ) const [virtual]`

Creates a [DirectionCone](#) which covers all tangent directions of this curve.

**Returns**

the smallest [DirectionCone](#) containing all tangent directions of this curve.

Implements [Go::ParamCurve](#).

29.43.3.9 `virtual double Go::BoundedCurve::endparam ( ) const [virtual]`

Query the end parameter of the curve

**Returns**

the curve's end parameter

Implements [Go::ParamCurve](#).

29.43.3.10 `virtual SplineCurve* Go::BoundedCurve::geometryCurve ( ) [virtual]`

If the definition of this [ParamCurve](#) contains a [SplineCurve](#) describing its spatial shape, then this function will return a pointer to this [SplineCurve](#). Otherwise it will return a null pointer. The returned curve is NEWed, so the user is responsible for deleting it. This function may have side-effects.

**Returns**

a pointer to a [SplineCurve](#) representation of the [ParamCurve](#), if it exists. Null pointer otherwise.

Implements [Go::ParamCurve](#).

29.43.3.11 `virtual ClassType Go::BoundedCurve::instanceType ( ) const [virtual]`

Return the class type identifier of a given, derived instance of [GeomObject](#).

Implements [Go::GeomObject](#).

29.43.3.12 `virtual bool Go::BoundedCurve::isAxisRotational ( Point & centre, Point & axis, Point & vec, double & angle ) [virtual]`

Check if the curve is axis rotational. Only true if a connection to an axis rotational elementary curve exist The axis and rotational angle is only specified if the curve is actually rotational

Reimplemented from [Go::ParamCurve](#).

29.43.3.13 `virtual bool Go::BoundedCurve::isDegenerate ( double degenerate_epsilon ) [virtual]`

Query whether the curve is degenerate (collapsed into a single point).



## Parameters

|                           |                                                                                                                                                   |
|---------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>degenerate_epsilon</i> | the tolerance used in determine whether the curve is degenerate. A curve is considered degenerate if its total length is shorter than this value. |
|---------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------|

## Returns

`true` if the curve is degenerate, `false` otherwise.

Implements [Go::ParamCurve](#).

**29.43.3.14** `virtual bool Go::BoundedCurve::isInPlane ( const Point & loc, const Point & axis, double eps, Point & normal ) const` [virtual]

Check if the curve lies in a plane passing through a given axis.

Reimplemented from [Go::ParamCurve](#).

**29.43.3.15** `virtual bool Go::BoundedCurve::isLinear ( Point & dir, double tol )` [virtual]

Check if the curve is linear.

Reimplemented from [Go::ParamCurve](#).

**29.43.3.16** `virtual double Go::BoundedCurve::length ( double tol )` [virtual]

Compute the total length of this curve up to some tolerance

## Parameters

|            |                                                                                                                                 |
|------------|---------------------------------------------------------------------------------------------------------------------------------|
| <i>tol</i> | the relative tolerance when approximating the length, i.e. stop iteration when error becomes smaller than $tol/(curve\ length)$ |
|------------|---------------------------------------------------------------------------------------------------------------------------------|

## Returns

the length calculated

Implements [Go::ParamCurve](#).

**29.43.3.17** `virtual void Go::BoundedCurve::point ( Point & pt, double tpar ) const` [virtual]

Evaluate the curve's position at a given parameter

## Parameters

|             |                                                                      |
|-------------|----------------------------------------------------------------------|
| <i>pt</i>   | the evaluated position will be written to this <a href="#">Point</a> |
| <i>tpar</i> | the parameter for which we wish to evaluate the curve                |

Implements [Go::ParamCurve](#).

29.43.3.18 `virtual void Go::BoundedCurve::point ( std::vector< Point > & pts, double tpar, int derivs, bool from_right = true ) const` [virtual]

Evaluate the curve's position and a certain number of derivatives at a given parameter.

#### Parameters

|                   |                                                                                                                                                                                                                                                                                       |
|-------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>pts</i>        | the evaluated position and derivatives (tangent, curvature vector, etc.) will be written to this vector. The first entry will be the position, the second entry will be the first derivative, etc. The size of this vector must be set to 'derivs'+ 1 prior to calling this function. |
| <i>tpar</i>       | the parameter for which we want to evaluate the curve                                                                                                                                                                                                                                 |
| <i>derivs</i>     | the number of derivatives we want to have calculated                                                                                                                                                                                                                                  |
| <i>from_right</i> | specify whether we should calculate derivatives 'from the right' or 'from the left' (default is from the right). This matters only when the curve presents discontinuities in its derivatives.                                                                                        |

Implements [Go::ParamCurve](#).

29.43.3.19 `virtual void Go::BoundedCurve::read ( std::istream & is )` [virtual]

Read object from stream

#### Parameters

|           |                                  |
|-----------|----------------------------------|
| <i>is</i> | stream from which object is read |
|-----------|----------------------------------|

Implements [Go::Streamable](#).

29.43.3.20 `virtual void Go::BoundedCurve::reverseParameterDirection ( bool switchparam = false )` [virtual]

Set the parameter direction of the curve. The curve's parameter interval will always remain constant, but by flipping the parameter direction, the curve will be traced the opposite way when moving a parameter over the parameter interval.

#### Parameters

|                    |                                                                                                                                                   |
|--------------------|---------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>switchparam</i> | if true, and the curve is 2D, the x and y coordinates should be swapped. This is used when turning the orientation of bounded (trimmed) surfaces. |
|--------------------|---------------------------------------------------------------------------------------------------------------------------------------------------|

Implements [Go::ParamCurve](#).

29.43.3.21 `void Go::BoundedCurve::setParamBounds ( double startpar, double endpar )`

Set bounds for the parametrization of the [Line](#).

## Parameters

|                 |                 |
|-----------------|-----------------|
| <i>startpar</i> | start parameter |
| <i>endpar</i>   | end parameter   |

29.43.3.22 virtual void Go::BoundedCurve::setParameterInterval ( double *t1*, double *t2* ) [virtual]

Set bounds for the parametrization of the curve

## Parameters

|                 |                 |
|-----------------|-----------------|
| <i>startpar</i> | start parameter |
| <i>endpar</i>   | end parameter   |

Implements [Go::ParamCurve](#).

29.43.3.23 virtual double Go::BoundedCurve::startparam ( ) const [virtual]

Query the start parameter of the curve

## Returns

the curve's start parameter

Implements [Go::ParamCurve](#).

29.43.3.24 virtual BoundedCurve\* Go::BoundedCurve::subCurve ( double *from\_par*, double *to\_par*, double *fuzzy* = DEFAULT\_PARAMETER\_EPSILON ) const [virtual]

Returns a curve which is a part of this curve. The result is NEWed, so the user is responsible for deleting it. NB: It is not guaranteed that the [ParamCurve](#) that is returned is of the same type as the curve itself. Thus, the returned curve might be a [SplineCurve](#).

## Parameters

|                 |                                                                                                                                                                                                   |
|-----------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>from_par</i> | start value of parameter interval that will define the subcurve                                                                                                                                   |
| <i>to_par</i>   | end value of parameter interval that will define the subcurve                                                                                                                                     |
| <i>fuzzy</i>    | since subCurve works on those curves who are spline-based, this tolerance defines how close the start and end parameter must be to an existing knot in order to be considered <i>on</i> the knot. |

## Returns

a pointer to a new subcurve which represents the part of the curve between 'from\_par' and 'to\_par'. It will be spline-based and have a k-regular knotvector. The user is responsible for deleting this subcurve when it is no longer needed.

Implements [Go::ParamCurve](#).

29.43.3.25 `shared_ptr<ParamCurve> Go::BoundedCurve::underlyingCurve ( ) const` `[inline]`

Get a pointer to the underlying curve

#### Returns

shared pointer to the underlying curve

Definition at line 156 of file BoundedCurve.h.

29.43.3.26 `virtual void Go::BoundedCurve::write ( std::ostream & os ) const` `[virtual]`

Write object to stream

#### Parameters

|                 |                                   |
|-----------------|-----------------------------------|
| <code>os</code> | stream to which object is written |
|-----------------|-----------------------------------|

Implements [Go::Streamable](#).

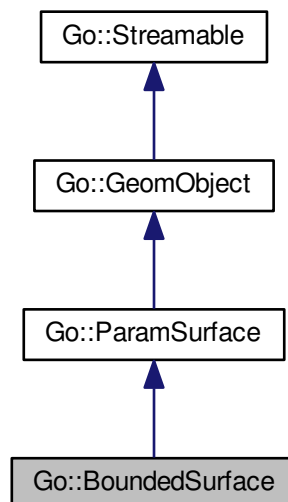
The documentation for this class was generated from the following file:

- [gotools-core/include/GoTools/geometry/BoundedCurve.h](#)

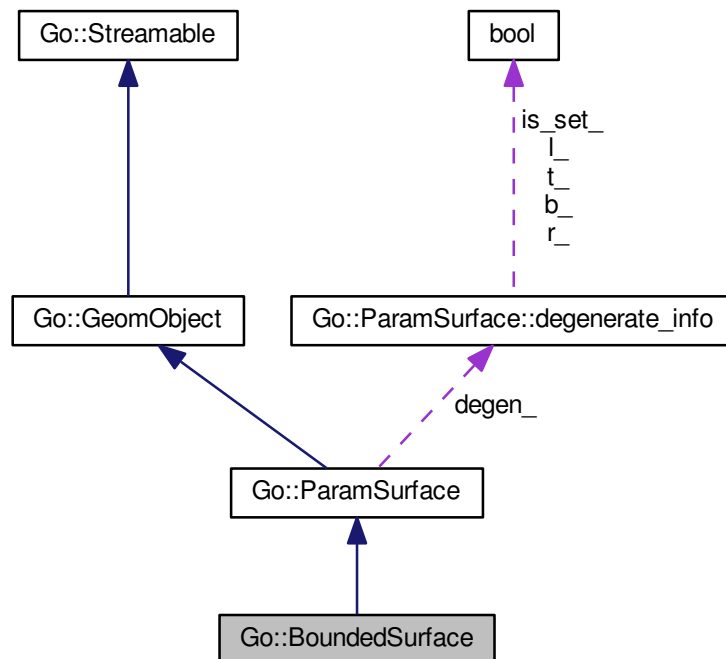
## 29.44 Go::BoundedSurface Class Reference

```
#include <BoundedSurface.h>
```

Inheritance diagram for Go::BoundedSurface:



Collaboration diagram for Go::BoundedSurface:



## Public Member Functions

- [BoundedSurface](#) ()  
*Create an empty [BoundedSurface](#) that can be assigned or [read\(\)](#) into.*
- [BoundedSurface](#) (shared\_ptr< [ParamSurface](#) > surf, std::vector< shared\_ptr< [CurveOnSurface](#) > > loop, double space\_epsilon, bool fix\_trim\_cvts=true)
- [BoundedSurface](#) (shared\_ptr< [ParamSurface](#) > surf, std::vector< std::vector< shared\_ptr< [CurveOnSurface](#) > > > loops, double space\_epsilon, bool fix\_trim\_cvts=true)
- [BoundedSurface](#) (shared\_ptr< [ParamSurface](#) > surf, std::vector< std::vector< shared\_ptr< [CurveOnSurface](#) > > > loops, std::vector< double > space\_epsilons, bool fix\_trim\_cvts=true)
- [BoundedSurface](#) (shared\_ptr< [ParamSurface](#) > surf, double space\_epsilon)  
*Create a bounded surface from a non-trimmed one.*
- [BoundedSurface](#) (shared\_ptr< [ParamSurface](#) > surf, std::vector< [CurveLoop](#) > &loops)  
*Create a bounded surface from a non-trimmed one.*
- [BoundedSurface](#) (shared\_ptr< [ParamSurface](#) > surf, std::vector< shared\_ptr< [CurveLoop](#) > > &loops)  
*Create a bounded surface from a non-trimmed one.*
- virtual [~BoundedSurface](#) ()  
*Virtual destructor ensures safe inheritance.*
- virtual void [read](#) (std::istream &is)  
*read this [BoundedSurface](#) from a stream*
- void [read](#) (std::istream &is, bool fix\_trim\_cvts)
- virtual void [write](#) (std::ostream &os) const  
*write this [BoundedSurface](#) to a stream*

- virtual [BoundingBox](#) boundingBox () const  
*Return the object's bounding box.*
- virtual int dimension () const  
*Return the dimension of the space in which the object lies (usually 2 or 3)*
- virtual [ClassType](#) instanceType () const  
*Return the class type identifier of type [BoundedSurface](#).*
- virtual [BoundedSurface](#) \* clone () const  
*clone this [BoundedSurface](#) and return a pointer to the clone*
- virtual [SplineSurface](#) \* asSplineSurface ()  
*Return the spline surface represented by this surface, if any.*
- virtual [DirectionCone](#) normalCone () const
- virtual [DirectionCone](#) tangentCone (bool paddir\_is\_u) const
- virtual const [CurveBoundedDomain](#) & parameterDomain () const
- virtual [RectDomain](#) containingDomain () const
- virtual bool inDomain (double u, double v, double eps=1.0e-4) const  
*Check if a parameter pair lies inside the domain of this surface.*
- virtual int inDomain2 (double u, double v, double eps=1.0e-4) const
- virtual bool onBoundary (double u, double v, double eps=1.0e-4) const  
*Check if a parameter pair lies at the boundary of this surface.*
- virtual [Point](#) closestInDomain (double u, double v) const
- virtual [CurveLoop](#) outerBoundaryLoop (double degenerate\_epsilon=DEFAULT\_SPACE\_EPSILON) const
- virtual std::vector< [CurveLoop](#) > allBoundaryLoops (double degenerate\_epsilon=DEFAULT\_SPACE\_EPSILON) const
- std::vector< [CurveLoop](#) > absolutelyAllBoundaryLoops () const
- virtual void point ([Point](#) &pt, double upar, double vpar) const
- virtual void point (std::vector< [Point](#) > &pts, double upar, double vpar, int derivs, bool u\_from\_right=true, bool v\_from\_right=true, double resolution=1.0e-12) const
- virtual void normal ([Point](#) &n, double upar, double vpar) const
- virtual void evalGrid (int num\_u, int num\_v, double umin, double umax, double vmin, double vmax, std::vector< double > &points, double nodata\_val=-9999) const
- virtual [Point](#) getInternalPoint (double &u, double &v) const
- virtual std::vector< shared\_ptr< [ParamCurve](#) > > constParamCurves (double parameter, bool paddir\_is\_u) const
- virtual std::vector< shared\_ptr< [ParamSurface](#) > > subSurfaces (double from\_upar, double from\_vpar, double to\_upar, double to\_vpar, double fuzzy=DEFAULT\_PARAMETER\_EPSILON) const
- virtual double nextSegmentVal (int dir, double par, bool forward, double tol) const
- virtual void closestPoint (const [Point](#) &pt, double &clo\_u, double &clo\_v, [Point](#) &clo\_pt, double &clo\_dist, double epsilon, const [RectDomain](#) \*domain\_of\_interest=NULL, double \*seed=0) const
- virtual void closestBoundaryPoint (const [Point](#) &pt, double &clo\_u, double &clo\_v, [Point](#) &clo\_pt, double &clo\_dist, double epsilon, const [RectDomain](#) \*domain\_of\_interest=NULL, double \*seed=0) const
- virtual void getBoundaryInfo ([Point](#) &pt1, [Point](#) &pt2, double epsilon, [SplineCurve](#) \*&cv, [SplineCurve](#) \*&crosscv, double knot\_tol=1e-05) const
- void getBoundaryInfo ([Point](#) &pt1, [Point](#) &pt2, std::vector< shared\_ptr< [CurveOnSurface](#) > > &bd\_cvs) const
- virtual void turnOrientation ()  
*Turns the direction of the normal of the surface.*
- virtual void reverseParameterDirection (bool direction\_is\_u)
- void makeBoundaryCurvesG1 (double kink)
- void removeSmallBoundaryCurves (double gap, double neighbour, double kink)
- virtual void swapParameterDirection ()  
*Swaps the two parameter directions.*
- virtual double area (double tol) const
- virtual void setParameterDomain (double u1, double u2, double v1, double v2)
- void splitSingleLoops ()

- `shared_ptr< ParamSurface > underlyingSurface ()`
- `shared_ptr< const ParamSurface > underlyingSurface () const`
- `bool hasUnderlyingSpline (shared_ptr< SplineSurface > &srf)`
- `int numberOfLoops () const`  
*Get the number of boundary loops that describe the trimmed surface.*
- `shared_ptr< CurveLoop > loop (int idx)`  
*Get a shared pointer to a specific boundary loop.*
- `SplineCurve * constParamCurve (double parameter, bool direction_is_u) const`
- `virtual bool isDegenerate (bool &b, bool &r, bool &t, bool &l, double tolerance) const`
- `virtual void getDegenerateCorners (std::vector< Point > &deg_corners, double tol) const`  
*Check for paralell and anti paralell partial derivatives in surface corners.*
- `virtual void getCornerPoints (std::vector< std::pair< Point, Point > > &corners) const`
- `virtual void setIterator (IteratorType type)`
- `virtual bool isIsoTrimmed (double tol) const`  
*Check if the current surface is trimmed along constant parameter curves.*
- `shared_ptr< ParamSurface > getIsoTrimSurface (double tol) const`  
*Fetch.*
- `bool orientationIsSet ()`  
*Check if the loop orientation is tested and corrected.*
- `bool orientationOK ()`
- `void setOrientationOK ()`  
*Remove history of loop orientation fixes.*
- `void turnLoopOrientation (int idx)`  
*Turn orientation of specified loop, and remember turning it.*
- `bool isValid (int &valid_state) const`
- `bool fixInvalidSurface (double &max_loop_gap, double max_tol_mult=1.0)`
- `void analyzeLoops ()`
- `void removeMismatchCurves (double max_tol_mult)`
- `double maxLoopSfDist (int loop_ind, int nmb_seg_samples=100)`
- `double maxLoopGap ()`  
*We measure the largest distance between loop segments.*
- `Point getSurfaceParameter (int loop_idx, int cv_idx, double bd_par) const`
- `bool simplifyBdLoops (double tol, double ang_tol, double &max_dist)`  
*Simplify boundary loops by reducing the number of curves if possible.*
- `bool makeUnderlyingSpline ()`
- `bool allIsSpline () const`
- `BoundedSurface * allSplineCopy () const`
- `virtual bool isAxisRotational (Point &centre, Point &axis, Point &vec, double &angle)`
- `virtual bool isPlanar (Point &normal, double tol)`  
*This surface is planar if the underlying surface is.*
- `virtual bool isLinear (Point &dir1, Point &dir2, double tol)`  
*Check if the surface is linear in one or both parameter directions.*
- `double getEpsGeo () const`  
*Run through the boundary loops, returning the smallest epsgeo.*
- `bool closeToUnderlyingBoundary (double upar, double vpar, double domain_fraction=0.01) const`
- `virtual int ElementOnBoundary (int elem_ix, double eps)`
- `virtual int ElementBoundaryStatus (int elem_ix, double eps)`

### Static Public Member Functions

- `static ClassType classType ()`  
*Return the class type identifier of type `BoundedSurface`.*

## Friends

- void [GeometryTools::setParameterDomain](#) (std::vector< shared\_ptr< [BoundedSurface](#) > &sfs, double u1, double u2, double v1, double v2)

## Additional Inherited Members

### 29.44.1 Detailed Description

The class representing trimmed surfaces in [Go](#). Surface should be connected.

Definition at line 59 of file [BoundedSurface.h](#).

### 29.44.2 Constructor & Destructor Documentation

#### 29.44.2.1 [Go::BoundedSurface::BoundedSurface](#) ( )

Create an empty [BoundedSurface](#) that can be assigned or [read\(\)](#) into.

#### 29.44.2.2 [Go::BoundedSurface::BoundedSurface](#) ( shared\_ptr< [ParamSurface](#) > surf, std::vector< shared\_ptr< [CurveOnSurface](#) > > loop, double space\_epsilon, bool fix\_trim\_cvs = true )

Create a [BoundedSurface](#) by specifying the underlying surface and a loop of curves that specifies the trimming of the surface.

#### Parameters

|                      |                                                                                                                                                                                                                                                                                                |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>surf</i>          | the created <a href="#">BoundedSurface</a> will represent a trimmed version of this surface                                                                                                                                                                                                    |
| <i>loop</i>          | a vector of <a href="#">CurveOnSurface</a> s that together describe the boundary that defines the trimming of the surface. The curves in this vector should all lie on the surface in question, and when placed head-to-tail they should form a closed loop with counterclockwise orientation. |
| <i>space_epsilon</i> | geometrical tolerance used when treating the loops.                                                                                                                                                                                                                                            |
| <i>fix_trim_cvs</i>  | the constructor may alter loop curves if they are not consistent.                                                                                                                                                                                                                              |

#### 29.44.2.3 [Go::BoundedSurface::BoundedSurface](#) ( shared\_ptr< [ParamSurface](#) > surf, std::vector< std::vector< shared\_ptr< [CurveOnSurface](#) > > > loops, double space\_epsilon, bool fix\_trim\_cvs = true )

Create a [BoundedSurface](#) by specifying the underlying surface and a number of loops of curves that specify the trimming of the surface.

#### Parameters

|              |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|--------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>surf</i>  | the created <a href="#">BoundedSurface</a> will represent a trimmed version of this surface                                                                                                                                                                                                                                                                                                                                                                                                    |
| <i>loops</i> | each entry in 'loop is a vector of <a href="#">CurveOnSurface</a> s that describe a closed loop forming a part of the trimmed surface's boundary. (Since the surface may have internal holes, more than one loop might be required to describe its boundary). The curve loops should all lie on the surface in question. The first entry in this vector describes the outermost boundary, which should be oriented counterclockwise. The other entries represent holes, and should be oriented |
|              | clockwise.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |



## Parameters

|                      |                                                                   |
|----------------------|-------------------------------------------------------------------|
| <i>space_epsilon</i> | geometrical tolerance used when treating the loops.               |
| <i>fix_trim_cvs</i>  | the constructor may alter loop curves if they are not consistent. |

29.44.2.4 `Go::BoundedSurface::BoundedSurface ( shared_ptr< ParamSurface > surf, std::vector< std::vector< shared_ptr< CurveOnSurface > > > loops, std::vector< double > space_emptons, bool fix_trim_cvs = true )`

Create a [BoundedSurface](#) by specifying the underlying surface and a number of loops of curves that specify the trimming of the surface.

## Parameters

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>surf</i>          | the created <a href="#">BoundedSurface</a> will represent a trimmed version of this surface                                                                                                                                                                                                                                                                                                                                                                                                                 |
| <i>loops</i>         | each entry in 'loops' is a vector of <a href="#">CurveOnSurface</a> s that describe a closed loop forming a part of the trimmed surface's boundary. (Since the surface may have internal holes, more than one loop might be required to describe its boundary). The curve loops should all lie on the surface in question. The first entry in this vector describes the outermost boundary, which should be oriented counterclockwise. The other entries represent holes, and should be oriented clockwise. |
| <i>space_emptons</i> | geometrical tolerances used when treating the loops.                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| <i>fix_trim_cvs</i>  | the constructor may alter loop curves if they are not consistent.                                                                                                                                                                                                                                                                                                                                                                                                                                           |

29.44.2.5 `Go::BoundedSurface::BoundedSurface ( shared_ptr< ParamSurface > surf, double space_epsilon )`

Create a bounded surface from a non-trimmed one.

29.44.2.6 `Go::BoundedSurface::BoundedSurface ( shared_ptr< ParamSurface > surf, std::vector< CurveLoop > & loops )`

Create a bounded surface from a non-trimmed one.

29.44.2.7 `Go::BoundedSurface::BoundedSurface ( shared_ptr< ParamSurface > surf, std::vector< shared_ptr< CurveLoop > > & loops )`

Create a bounded surface from a non-trimmed one.

29.44.2.8 `virtual Go::BoundedSurface::~~BoundedSurface ( ) [virtual]`

Virtual destructor ensures safe inheritance.

### 29.44.3 Member Function Documentation

#### 29.44.3.1 `std::vector<CurveLoop> Go::BoundedSurface::absolutelyAllBoundaryLoops ( ) const`

Returns the anticlockwise outer boundary loop of the surface, together with clockwise loops of any interior boundaries, such that the surface always is 'to the left of' the loops. This function works like [allBoundaryLoops\(\)](#), except that it includes degenerate edges.

#### Returns

vector containing CurveLoops. The first of these describe the outer boundary of the surface (counterclockwise) whereas the others describe boundaries of interior holes (clockwise).

#### 29.44.3.2 `virtual std::vector<CurveLoop> Go::BoundedSurface::allBoundaryLoops ( double degenerate_epsilon = DEFAULT_SPACE_EPSILON ) const [virtual]`

Returns the anticlockwise outer boundary loop of the surface, together with clockwise loops of any interior boundaries, such that the surface always is 'to the left of' the loops.

#### Parameters

|                           |                                                            |
|---------------------------|------------------------------------------------------------|
| <i>degenerate_epsilon</i> | edges whose length is smaller than this value are ignored. |
|---------------------------|------------------------------------------------------------|

#### Returns

a vector containing CurveLoops. The first of these describe the outer boundary of the surface (counterclockwise), whereas the others describe boundaries of interior holes (clockwise).

Implements [Go::ParamSurface](#).

#### 29.44.3.3 `bool Go::BoundedSurface::allIsSpline ( ) const`

Test if underlying surface and all loop curves are [CurveOnSurface](#) where the space curves are [SplineCurve](#).

#### 29.44.3.4 `BoundedSurface* Go::BoundedSurface::allSplineCopy ( ) const`

Return copy where underlying surface and all loop curves are [CurveOnSurface](#) where the space curves are [SplineCurve](#).

#### 29.44.3.5 `void Go::BoundedSurface::analyzeLoops ( )`

Checking all boundary\_loops\_ to see if they fulfill requirements. Routine sets valid\_state\_. Must be called after surface is altered.

#### 29.44.3.6 `virtual double Go::BoundedSurface::area ( double tol ) const [virtual]`

Compute the total area of this surface up to some tolerance

## Parameters

|            |                                                                                                                               |
|------------|-------------------------------------------------------------------------------------------------------------------------------|
| <i>tol</i> | the relative tolerance when approximating the area, i.e. stop iteration when error becomes smaller than $tol/(surface\ area)$ |
|------------|-------------------------------------------------------------------------------------------------------------------------------|

## Returns

the area calculated NB! Intermediate solution with lower accuracy

Implements [Go::ParamSurface](#).

**29.44.3.7** `virtual SplineSurface* Go::BoundedSurface::asSplineSurface ( ) [inline],[virtual]`

Return the spline surface represented by this surface, if any.

Reimplemented from [Go::ParamSurface](#).

Definition at line 175 of file BoundedSurface.h.

**29.44.3.8** `virtual BoundingBox Go::BoundedSurface::boundingBox ( ) const [virtual]`

Return the object's bounding box.

Implements [Go::GeomObject](#).

**29.44.3.9** `static ClassType Go::BoundedSurface::classType ( ) [inline],[static]`

Return the class type identifier of type [BoundedSurface](#).

Definition at line 168 of file BoundedSurface.h.

**29.44.3.10** `virtual BoundedSurface* Go::BoundedSurface::clone ( ) const [virtual]`

clone this [BoundedSurface](#) and return a pointer to the clone

Implements [Go::ParamSurface](#).

**29.44.3.11** `virtual void Go::BoundedSurface::closestBoundaryPoint ( const Point & pt, double & clo_u, double & clo_v, Point & clo_pt, double & clo_dist, double epsilon, const RectDomain * domain_of_interest = NULL, double * seed = 0 ) const [virtual]`

Iterates to the closest point to pt on the boundary of the surface.

## See also

[closestPoint\(\)](#)

Implements [Go::ParamSurface](#).

29.44.3.12 `virtual Point Go::BoundedSurface::closestInDomain ( double u, double v ) const` [virtual]

Fetch the parameter value in the parameter domain of the surface closest to the parameter pair (u,v)

Implements [Go::ParamSurface](#).

29.44.3.13 `virtual void Go::BoundedSurface::closestPoint ( const Point & pt, double & clo_u, double & clo_v, Point & clo_pt, double & clo_dist, double epsilon, const RectDomain * domain_of_interest = NULL, double * seed = 0 ) const` [virtual]

Iterates to the closest point to pt on the surface.

## Parameters

|                           |                                                                                                                              |
|---------------------------|------------------------------------------------------------------------------------------------------------------------------|
| <i>pt</i>                 | the point to find the closest point to                                                                                       |
| <i>clo_u</i>              | u parameter of the closest point                                                                                             |
| <i>clo_v</i>              | v parameter of the closest point                                                                                             |
| <i>clo_pt</i>             | the geometric position of the closest point                                                                                  |
| <i>clo_dist</i>           | the distance between pt and clo_pt                                                                                           |
| <i>epsilon</i>            | parameter tolerance (will in any case not be higher than sqrt(machine_precision) x magnitude of solution)                    |
| <i>domain_of_interest</i> | pointer to parameter domain in which to search for closest point. If a NULL pointer is used, the entire surface is searched. |
| <i>seed</i>               | pointer to parameter values where iteration starts.                                                                          |

Reimplemented from [Go::ParamSurface](#).

29.44.3.14 **bool** Go::BoundedSurface::closeToUnderlyingBoundary ( **double** *upar*, **double** *vpar*, **double** *domain\_fraction* = 0.01 ) **const**

29.44.3.15 **SplineCurve\*** Go::BoundedSurface::constParamCurve ( **double** *parameter*, **bool** *direction\_is\_u* ) **const**

Get the space-curve resulting from fixing one of the surface's parameters and moving the other along its allowed range (inside the trimmed domain). If this results in several disjoint curves, an exception is thrown.

## Parameters

|                                      |                                                                                                                                            |
|--------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------|
| <i>parameter</i>                     | the parameter value of the fixed parameter                                                                                                 |
| <i>direction_is_u</i> ↔<br><i>_u</i> | if 'true' then the "free" parameter will be the first one, and the second parameter will be fixed. If 'false', it is the other way around. |

## Returns

a newly created [SplineCurve](#) representing the requested space-curve. The ownership is assumed by the user.

29.44.3.16 **virtual std::vector<shared\_ptr<ParamCurve>>** Go::BoundedSurface::constParamCurves ( **double** *parameter*, **bool** *pardir\_is\_u* ) **const** [virtual]

Get the curve(s) obtained by intersecting the surface with one of its constant parameter curves. For surfaces without holes, this will be the parameter curve itself; for surfaces with interior holes this may be a collection of several, disjoint curves.

## Parameters

|                                   |                                                                                                                                              |
|-----------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------|
| <i>parameter</i>                  | parameter value for the constant parameter (either u or v)                                                                                   |
| <i>pardir_is_u</i> ↔<br><i>_u</i> | specify whether the <i>moving</i> parameter (as opposed to the <i>constant</i> parameter) is the first ('true') or the second ('false') one. |

**Returns**

a vector containing shared pointers to the obtained, newly constructed constant-parameter curves.

Implements [Go::ParamSurface](#).

29.44.3.17 `virtual RectDomain Go::BoundedSurface::containingDomain ( ) const [virtual]`

Get a rectangular parameter domain that is guaranteed to contain the surface's [parameterDomain\(\)](#). It may be the same. There is no guarantee that this is the smallest domain containing the actual domain.

**Returns**

a [RectDomain](#) that is guaranteed to include the surface's total parameter domain.

Implements [Go::ParamSurface](#).

29.44.3.18 `virtual int Go::BoundedSurface::dimension ( ) const [virtual]`

Return the dimension of the space in which the object lies (usually 2 or 3)

Implements [Go::GeomObject](#).

29.44.3.19 `virtual int Go::BoundedSurface::ElementBoundaryStatus ( int elem_ix, double eps ) [virtual]`

Check if a polynomial element (for spline surfaces) intersects the (trimming) boundaries of this [ftSurface](#), is inside or outside

**Parameters**

|                |                                                                                                          |
|----------------|----------------------------------------------------------------------------------------------------------|
| <i>elem_ix</i> | Element index counted according to distinct knot values. Sequence of coordinates: x runs fastest, then y |
| <i>eps</i>     | Intersection tolerance                                                                                   |

**Returns**

-1: Not a spline surface or element index out of range 0: Outside trimmed volume 1: On boundary (intersection with boundary found) 2: Internal to trimmed surfaces Note that a touch with the boundaries of the underlying surface is not considered a boundary intersection while touching a trimming curve is seen as an intersection

Reimplemented from [Go::ParamSurface](#).

29.44.3.20 `virtual int Go::BoundedSurface::ElementOnBoundary ( int elem_ix, double eps ) [virtual]`

Check if a polynomial element (for spline surfaces) intersects the (trimming) boundaries of this surface

## Parameters

|                |                                                                                                          |
|----------------|----------------------------------------------------------------------------------------------------------|
| <i>elem_ix</i> | Element index counted according to distinct knot values. Sequence of coordinates: x runs fastest, then y |
| <i>eps</i>     | Intersection tolerance                                                                                   |

## Returns

-1: Not a spline surface or element index out of range  
 0: Not on boundary or touching a boundary curve  
 1: On boundary (intersection with boundary found)  
 Note that a touch with the boundaries of the underlying surfaces is not considered a boundary intersection while touching a trimming curve is seen as an intersection

Reimplemented from [Go::ParamSurface](#).

```
29.44.3.21 virtual void Go::BoundedSurface::evalGrid (int num_u, int num_v, double umin, double umax, double vmin,
double vmax, std::vector< double > & points, double nodata_val = -9999) const [virtual]
```

Evaluate points in a grid The nodata value is applicable for bounded surfaces and grid points outside the trimming loop(s) will get this value

Reimplemented from [Go::ParamSurface](#).

```
29.44.3.22 bool Go::BoundedSurface::fixInvalidSurface (double & max_loop_gap, double max_tol_mult = 1.0)
```

We try to fix the invalid loops. Return value: true if the loops are valid. Returned max\_gap is the largest gap between end segments in the loops. Assuming that max\_tol\_mult >= 1.0. The routine is allowed to alter the eps to eps\*max\_tol\_mult.

```
29.44.3.23 virtual void Go::BoundedSurface::getBoundaryInfo (Point & pt1, Point & pt2, double epsilon, SplineCurve
*& cv, SplineCurve *& crosscv, double knot_tol = 1e-05) const [virtual]
```

Get the boundary curve segment between two points on the boundary, as well as the cross-tangent curve. If the given points are not positioned on the same boundary (within a certain tolerance), no curves will be created. **NB:** This function has not yet been implemented!

## Parameters

|                 |                                                                                                                                                                                                                                                                                                                                   |
|-----------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>pt1</i>      | the first point on the boundary, given by the user                                                                                                                                                                                                                                                                                |
| <i>pt2</i>      | the second point on the boundary, given by the user                                                                                                                                                                                                                                                                               |
| <i>epsilon</i>  | the tolerance used when determining whether the given points are lying on a boundary, and if they do, whether they both lie on the <i>same</i> boundary.                                                                                                                                                                          |
| <i>cv</i>       | upon return, this will point to a newly created curve representing the boundary curve between 'pt1' and 'pt2'. The user assumes ownership of this object and is responsible for its deletion. No curve is created if the given points are not found to lie on the same boundary.                                                  |
| <i>crosscv</i>  | upon return, this will point to a newly created curve representing the cross-boundary curve between 'pt1' and 'pt2' The user assumes ownership of this object and is responsible for its deletion. The direction is outwards from the surface. No curve is created if the given points are not found to lie on the same boundary. |
| <i>knot_tol</i> | tolerance used when working with the knot-vector, to specify how close a parameter value must be to a knot in order to be considered 'on' the knot.                                                                                                                                                                               |

Implements [Go::ParamSurface](#).

29.44.3.24 `void Go::BoundedSurface::getBoundaryInfo ( Point & pt1, Point & pt2, std::vector< shared_ptr< CurveOnSurface > > & bd_cvs ) const`

Get the boundary curve segment between two points on the same boundary loop. If the given points are not positioned on the same boundary loop (within a certain tolerance), no curves will be returned.

#### Parameters

|            |                                                     |
|------------|-----------------------------------------------------|
| <i>pt1</i> | the first point on the boundary, given by the user  |
| <i>pt2</i> | the second point on the boundary, given by the user |

#### Return values

|               |                                                                                                                                              |
|---------------|----------------------------------------------------------------------------------------------------------------------------------------------|
| <i>bd_cvs</i> | upon return, this will contain shared pointers to curves that, taken consecutively, describe the requested boundary segment in its entirety. |
|---------------|----------------------------------------------------------------------------------------------------------------------------------------------|

29.44.3.25 `virtual void Go::BoundedSurface::getCornerPoints ( std::vector< std::pair< Point, Point > > & corners ) const [virtual]`

Return surface corners, i.e joints between trimming curves, geometric and parametric points in that sequence

Implements [Go::ParamSurface](#).

29.44.3.26 `virtual void Go::BoundedSurface::getDegenerateCorners ( std::vector< Point > & deg_corners, double tol ) const [virtual]`

Check for parallel and anti parallel partial derivatives in surface corners.

Implements [Go::ParamSurface](#).

29.44.3.27 `double Go::BoundedSurface::getEpsGeo ( ) const`

Run through the boundary loops, returning the smallest epsgeo.

29.44.3.28 `virtual Point Go::BoundedSurface::getInternalPoint ( double & u, double & v ) const [virtual]`

Fetch an arbitrary internal point in the surface Used for localization purposes

Reimplemented from [Go::ParamSurface](#).

29.44.3.29 `shared_ptr<ParamSurface> Go::BoundedSurface::getIsoTrimSurface ( double tol ) const`

Fetch.



29.44.3.30 **Point** Go::BoundedSurface::getSurfaceParameter ( int *loop\_idx*, int *cv\_idx*, double *bd\_par* ) const

Given a parameter value corresponding to on specified curve in a specified boundary loop, return the corresponding surface parameter

29.44.3.31 **bool** Go::BoundedSurface::hasUnderlyingSpline ( shared\_ptr< SplineSurface > & *srf* )

Check if the final underlying surface is a spline surface and in that case return this surface

29.44.3.32 **virtual bool** Go::BoundedSurface::inDomain ( double *u*, double *v*, double *eps* = 1.0e-4 ) const  
[virtual]

Check if a parameter pair lies inside the domain of this surface.

Implements [Go::ParamSurface](#).

29.44.3.33 **virtual int** Go::BoundedSurface::inDomain2 ( double *u*, double *v*, double *eps* = 1.0e-4 ) const  
[virtual]

Check if a parameter pair lies inside the domain of this surface return value = 0: outside = 1: internal = 2: at the boundary

Implements [Go::ParamSurface](#).

29.44.3.34 **virtual ClassType** Go::BoundedSurface::instanceType ( ) const [virtual]

Return the class type identifier of type [BoundedSurface](#).

Implements [Go::GeomObject](#).

29.44.3.35 **virtual bool** Go::BoundedSurface::isAxisRotational ( Point & *centre*, Point & *axis*, Point & *vec*, double & *angle* ) [virtual]

Check if the surface is axis rotational. Only true if a connection to an axis rotational elementary surface exist The axis and rotational angle is only specified if the surface is actually rotational

Reimplemented from [Go::ParamSurface](#).

29.44.3.36 **virtual bool** Go::BoundedSurface::isDegenerate ( bool & *b*, bool & *r*, bool & *t*, bool & *l*, double *tolerance* ) const [virtual]

Query whether any of the four boundaries of the *underlying surface* are degenerate (zero length) within a certain tolerance. In the below, we refer to 'u' as the first parameter and 'v' as the second.

## Parameters

|                  |                                                                                                    |
|------------------|----------------------------------------------------------------------------------------------------|
| <i>b</i>         | 'true' upon return of function if the boundary ( $v = v_{\min}$ ) is degenerate                    |
| <i>r</i>         | 'true' upon return of function if the boundary ( $v = v_{\max}$ ) is degenerate                    |
| <i>t</i>         | 'true' upon return of function if the boundary ( $u = u_{\min}$ ) is degenerate                    |
| <i>l</i>         | 'true' upon return of function if the boundary ( $u = u_{\max}$ ) is degenerate                    |
| <i>tolerance</i> | boundaries are considered degenerate if their length is shorter than this value, given by the user |

## Returns

'true' if at least one boundary curve was found to be degenerate, 'false' otherwise.

Reimplemented from [Go::ParamSurface](#).

**29.44.3.37** `virtual bool Go::BoundedSurface::isAlsoTrimmed ( double tol ) const` [virtual]

Check if the current surface is trimmed along constant parameter curves.

Reimplemented from [Go::ParamSurface](#).

**29.44.3.38** `virtual bool Go::BoundedSurface::isLinear ( Point & dir1, Point & dir2, double tol )` [virtual]

Check if the surface is linear in one or both parameter directions.

Reimplemented from [Go::ParamSurface](#).

**29.44.3.39** `virtual bool Go::BoundedSurface::isPlanar ( Point & normal, double tol )` [virtual]

This surface is planar if the underlying surface is.

Reimplemented from [Go::ParamSurface](#).

**29.44.3.40** `bool Go::BoundedSurface::isValid ( int & valid_state ) const`

The boundary loops may be outside the loop tolerance, or the loops do not fulfill the loop requirements (1 outer loop which is ccw, cw loops inside, loops should be simple and disjoint).

**29.44.3.41** `shared_ptr<CurveLoop> Go::BoundedSurface::loop ( int idx )` [inline]

Get a shared pointer to a specific boundary loop.

Definition at line 532 of file BoundedSurface.h.

**29.44.3.42** `void Go::BoundedSurface::makeBoundaryCurvesG1 ( double kink )`

This function processes all the curves that participate in defining the surface's (trimmed) boundary. Those curves that are not G1 within a certain tolerance are split into several curves, so that all G1-discontinuities will end up *between* consecutive curve segments.

## Parameters

|             |                                                 |
|-------------|-------------------------------------------------|
| <i>kink</i> | the tolerance to use for checking G1 continuity |
|-------------|-------------------------------------------------|

29.44.3.43 `bool Go::BoundedSurface::makeUnderlyingSpline ( )`

Test if underlying surface is spline, if not try to convert it to spline /return wether underlying surface is spline afterwards

29.44.3.44 `double Go::BoundedSurface::maxLoopGap ( )`

We measure the largest distance between loop segments.

29.44.3.45 `double Go::BoundedSurface::maxLoopSfDist ( int loop_ind, int nmb_seg_samples = 100 )`

We measure the largest distance from loop to the surface. If the loops are defined by curves in the parameter domain, then it is trivially 0.0. Useful for testing whether the tolerance makes any sense.

29.44.3.46 `virtual double Go::BoundedSurface::nextSegmentVal ( int dir, double par, bool forward, double tol ) const [virtual]`

Determine the parameter value of the start of the 'next segment' from a parameter value, along a given parameter direction. A 'segment' is here defined as a parameter interval in which there will be no discontinuities in derivatives or other artifacts. For spline objects, a segment will typically be the interval between two consecutive, non-coincident knots.

## Parameters

|                |                                                                                                      |
|----------------|------------------------------------------------------------------------------------------------------|
| <i>dir</i>     | the parameter direction in which we search for the next segment (0 or 1)                             |
| <i>par</i>     | the parameter value starting from which we search for the start value of the next segment            |
| <i>forward</i> | define whether we shall move forward ('true') or backwards when searching along this parameter       |
| <i>tol</i>     | tolerance used for determining whether the 'par' is already located <i>on</i> the next segment value |

## Returns

the value of the start value of the next segment (or the end of the previous segment, if we are moving backwards...)

Implements [Go::ParamSurface](#).

29.44.3.47 `virtual void Go::BoundedSurface::normal ( Point & n, double upar, double vpar ) const [virtual]`

Evaluates the surface normal for a given parameter pair

## Parameters

|             |                                                      |
|-------------|------------------------------------------------------|
| <i>n</i>    | the computed normal will be written to this variable |
| <i>upar</i> | the first parameter                                  |
| <i>vpar</i> | the second parameter                                 |

Implements [Go::ParamSurface](#).

29.44.3.48 `virtual DirectionCone Go::BoundedSurface::normalCone ( ) const [virtual]`

Creates a [DirectionCone](#) covering all normals to this surface.

## Returns

a [DirectionCone](#) (not necessarily the smallest) containing all normals to this surface.

Implements [Go::ParamSurface](#).

29.44.3.49 `int Go::BoundedSurface::numberOfLoops ( ) const [inline]`

Get the number of boundary loops that describe the trimmed surface.

Definition at line 528 of file BoundedSurface.h.

29.44.3.50 `virtual bool Go::BoundedSurface::onBoundary ( double u, double v, double eps = 1.0e-4 ) const [virtual]`

Check if a parameter pair lies at the boundary of this surface.

Implements [Go::ParamSurface](#).

29.44.3.51 `bool Go::BoundedSurface::orientationIsSet ( ) [inline]`

Check if the loop orientation is tested and corrected.

Definition at line 583 of file BoundedSurface.h.

29.44.3.52 `bool Go::BoundedSurface::orientationOK ( ) [inline]`

Check the status of the loop orientation. If a fix has been performed or the loop is not OK, the return value will be false

Definition at line 590 of file BoundedSurface.h.

29.44.3.53 `virtual CurveLoop Go::BoundedSurface::outerBoundaryLoop ( double degenerate_epsilon = DEFAULT_SPACE_EPSILON ) const [virtual]`

Returns the anticlockwise, outer boundary loop of the surface.

## Parameters

|                           |                                                            |
|---------------------------|------------------------------------------------------------|
| <i>degenerate_epsilon</i> | edges whose length is smaller than this value are ignored. |
|---------------------------|------------------------------------------------------------|

## Returns

a [CurveLoop](#) describing the anticlockwise, outer boundary loop of the surface.

Implements [Go::ParamSurface](#).

29.44.3.54 `virtual const CurveBoundedDomain& Go::BoundedSurface::parameterDomain ( ) const` [virtual]

Return the parameter domain of the surface. This may be a simple rectangular domain ([RectDomain](#)) or any other subclass of [Domain](#) (such as [CurveBoundedDomain](#), found in the `sisl_dependent` module).

## Returns

a [Domain](#) object describing the parametric domain of the surface

Implements [Go::ParamSurface](#).

29.44.3.55 `virtual void Go::BoundedSurface::point ( Point & pt, double upar, double vpar ) const` [virtual]

Evaluates the surface's position for a given parameter pair.

## Parameters

|             |                                              |
|-------------|----------------------------------------------|
| <i>pt</i>   | the result of the evaluation is written here |
| <i>upar</i> | the first parameter                          |
| <i>vpar</i> | the second parameter                         |

Implements [Go::ParamSurface](#).

29.44.3.56 `virtual void Go::BoundedSurface::point ( std::vector< Point > & pts, double upar, double vpar, int derivs, bool u_from_right = true, bool v_from_right = true, double resolution = 1.0e-12 ) const` [virtual]

Evaluates the surface's position and a certain number of derivatives for a given parameter pair.

## Parameters

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|-------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>pts</i>  | the vector containing the evaluated values. Its size must be set by the user prior to calling this function, and should be equal to $(derivs+1) * (derivs+2) / 2$ . Upon completion of the function, its first entry is the surface's position at the given parameter pair. Then, if 'derivs' > 0, the two next entries will be the surface tangents along the first and second parameter direction. The next three entries are the second- and cross derivatives, in the order (du2, dudv, dv2), and similar for even higher derivatives. |
| <i>upar</i> | the first parameter                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |

## Parameters

|                     |                                                                                                                                                                  |
|---------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>vpar</i>         | the second parameter                                                                                                                                             |
| <i>derivs</i>       | number of requested derivatives                                                                                                                                  |
| <i>u_from_right</i> | specify whether derivatives along the first parameter are to be calculated from the right ('true', default) or from the left ('false')                           |
| <i>v_from_right</i> | specify whether derivatives along the second parameter are to be calculated from the right ('true', default) or from the left ('false')                          |
| <i>resolution</i>   | tolerance used when determining whether parameters are located at special values of the parameter domain (in particular; knot values in case of spline objects). |

Implements [Go::ParamSurface](#).

29.44.3.57 virtual void `Go::BoundedSurface::read ( std::istream & is )` [virtual]

read this [BoundedSurface](#) from a stream

Implements [Go::Streamable](#).

29.44.3.58 void `Go::BoundedSurface::read ( std::istream & is, bool fix_trim_cvs )`

29.44.3.59 void `Go::BoundedSurface::removeMismatchCurves ( double max_tol_mult )`

If both parameter and space curve are given for a segment, and they do not match, one of them is removed, unless we may alter the tolerance slightly (at most to  $\text{epsgeo} * \text{max\_tol\_mult}$ ).

29.44.3.60 void `Go::BoundedSurface::removeSmallBoundaryCurves ( double gap, double neighbour, double kink )`

This function processes all the curves that participate in defining the surface's (trimmed) boundary. Those curves that are smaller than a given tolerance is merged with one of the adjacent curves if possible

## Parameters

|                  |                                                             |
|------------------|-------------------------------------------------------------|
| <i>gap</i>       | positional tolerance used to check for possibility of merge |
| <i>neighbour</i> | tolerance used for checking if a curve is small             |
| <i>kink</i>      | angular tolerance used to check for possibility of merge    |

29.44.3.61 virtual void `Go::BoundedSurface::reverseParameterDirection ( bool direction_is_u )` [virtual]

Reverses the direction of the basis in input direction. **NB:** This function has not yet been implemented!

## Parameters

|                       |                                                                                                                       |
|-----------------------|-----------------------------------------------------------------------------------------------------------------------|
| <i>direction_is_u</i> | if 'true', the first parameter direction will be reversed, otherwise, the second parameter direction will be reversed |
|-----------------------|-----------------------------------------------------------------------------------------------------------------------|

Implements [Go::ParamSurface](#).

29.44.3.62 `virtual void Go::BoundedSurface::setIterator ( IteratorType type ) [inline], [virtual]`

Set type of closest point iterator type == Iterator\_parametric - use conjugate gradient iteration type == Iterator\_↔  
geometric - sisl type geometric closest point iteration

Reimplemented from [Go::ParamSurface](#).

Definition at line 570 of file BoundedSurface.h.

29.44.3.63 `void Go::BoundedSurface::setOrientationOK ( ) [inline]`

Remove history of loop orientation fixes.

Definition at line 601 of file BoundedSurface.h.

29.44.3.64 `virtual void Go::BoundedSurface::setParameterDomain ( double u1, double u2, double v1, double v2 ) [virtual]`

Change the parameter domain of the underlying surface, and modify the boundary loops with respect to this change

#### Parameters

|           |                                     |
|-----------|-------------------------------------|
| <i>u1</i> | new start value of first parameter  |
| <i>u2</i> | new end value of first parameter    |
| <i>v1</i> | new start value of second parameter |
| <i>v2</i> | new end value of second parameter   |

Reimplemented from [Go::ParamSurface](#).

29.44.3.65 `bool Go::BoundedSurface::simplifyBdLoops ( double tol, double ang_tol, double & max_dist )`

Simplify boundary loops by reducing the number of curves if possible.

29.44.3.66 `void Go::BoundedSurface::splitSingleLoops ( )`

Split all boundary loops defined by only *curve* into three parts. (This somewhat exotic member function is included due to its handiness with the [GoTools](#) topology analysator).

29.44.3.67 `virtual std::vector<shared_ptr<ParamSurface>> Go::BoundedSurface::subSurfaces ( double from_upar, double from_vpar, double to_upar, double to_vpar, double fuzzy = DEFAULT_PARAMETER_EPSILON ) const [virtual]`

Get the surface(s) obtained by cropping the parameter domain of this surface between given values for the first and second parameter. In general, for surfaces with no interior holes, the result will be *one* surface; however, for surfaces with interior holes, the result might be *several disjoint* surfaces.

## Parameters

|                  |                                                                       |
|------------------|-----------------------------------------------------------------------|
| <i>from_upar</i> | lower value for the first parameter in the subdomain                  |
| <i>from_vpar</i> | lower value for the second parameter in the subdomain                 |
| <i>to_upar</i>   | upper value for the first parameter in the subdomain                  |
| <i>to_vpar</i>   | upper value for the second parameter in the subdomain                 |
| <i>fuzzy</i>     | tolerance used when determining intersection with interior boundaries |

## Returns

a vector contained shared pointers to the obtained, newly constructed sub-surfaces.

Implements [Go::ParamSurface](#).

29.44.3.68 virtual void `Go::BoundedSurface::swapParameterDirection ( )` [virtual]

Swaps the two parameter directions.

Implements [Go::ParamSurface](#).

29.44.3.69 virtual `DirectionCone` `Go::BoundedSurface::tangentCone ( bool paddir_is_u ) const` [virtual]

Creates a [DirectionCone](#) covering all tangents to this surface along a given parameter direction.

## Parameters

|                    |                                                                                                                                                                                                  |
|--------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>paddir_is_u</i> | if 'true', then the <a href="#">DirectionCone</a> will be defined on basis of the surface's tangents along the first parameter direction. Otherwise the second parameter direction will be used. |
|--------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

## Returns

a [DirectionCone](#) (not necessarily the smallest) containing all tangents to this surface along the specified parameter direction.

Implements [Go::ParamSurface](#).

29.44.3.70 void `Go::BoundedSurface::turnLoopOrientation ( int idx )`

Turn orientation of specified loop, and remember turning it.

29.44.3.71 virtual void `Go::BoundedSurface::turnOrientation ( )` [virtual]

Turns the direction of the normal of the surface.

Implements [Go::ParamSurface](#).



29.44.3.72 `shared_ptr<ParamSurface> Go::BoundedSurface::underlyingSurface ( ) [inline]`

Get a pointer to the underlying surface

#### Returns

shared pointer to the underlying surface

Definition at line 515 of file BoundedSurface.h.

29.44.3.73 `shared_ptr<const ParamSurface> Go::BoundedSurface::underlyingSurface ( ) const [inline]`

Get a pointer to the underlying surface

#### Returns

shared pointer to the underlying surface

Definition at line 520 of file BoundedSurface.h.

29.44.3.74 `virtual void Go::BoundedSurface::write ( std::ostream & os ) const [virtual]`

write this [BoundedSurface](#) to a stream

Implements [Go::Streamable](#).

## 29.44.4 Friends And Related Function Documentation

29.44.4.1 `void GeometryTools::setParameterDomain ( std::vector< shared_ptr< BoundedSurface > > & sfs, double u1, double u2, double v1, double v2 ) [friend]`

The documentation for this class was generated from the following file:

- [gtools-core/include/GoTools/geometry/BoundedSurface.h](#)

## 29.45 Go::BoundingBox Class Reference

```
#include <BoundingBox.h>
```

## Public Member Functions

- [BoundingBox](#) ()
- [BoundingBox](#) (int dim)
- [BoundingBox](#) (const [Point](#) &low, const [Point](#) &high)
- [~BoundingBox](#) ()
  - Do not inherit from this class – nonvirtual destructor.*
- void [setFromPoints](#) (const [Point](#) &low, const [Point](#) &high)
- template<typename FloatType >  
void [setFromArray](#) (const FloatType \*start, const FloatType \*end, int dim)
- template<typename ForwardIterator >  
void [setFromArray](#) (ForwardIterator start, ForwardIterator end, int dim)
- void [setFromPoints](#) (const std::vector< [Point](#) > &points)
- void [read](#) (std::istream &is)
  - Read a bounding box from a standard istream.*
- void [write](#) (std::ostream &os) const
  - Write a bounding box to a standard ostream.*
- int [dimension](#) () const
  - The dimension of the bounding box.*
- const [Point](#) & [low](#) () const
  - The lower bound of the bounding box.*
- const [Point](#) & [high](#) () const
  - The upper bound of the bounding box.*
- bool [containsPoint](#) (const [Point](#) &pt, double tol=0.0) const
- bool [overlaps](#) (const [BoundingBox](#) &box, double tol=0.0) const
- bool [getOverlap](#) (const [BoundingBox](#) &box, double &overlap, double tol=0.0) const
- bool [containsBox](#) (const [BoundingBox](#) &box, double tol=0.0) const
- void [addUnionWith](#) (const [Point](#) &pt)
- void [addUnionWith](#) (const [BoundingBox](#) &box)
- std::vector< [Point](#) > [lineIntersect](#) (const [Point](#) &p1, const [Point](#) &dir) const
  - Compute the intersections with this box and a line.*
- bool [valid](#) () const
  - Is the bounding box initialized?*
- void [unset](#) ()
  - Unset bounding box.*
- void [check](#) () const

### 29.45.1 Detailed Description

Axis-aligned bounding box. A [BoundingBox](#) object can be an axis-aligned box in any number of dimensions.

Definition at line 56 of file [BoundingBox.h](#).

### 29.45.2 Constructor & Destructor Documentation

#### 29.45.2.1 `Go::BoundingBox::BoundingBox ( ) [inline]`

The default constructor makes an uninitialized object, with dimension zero.

Definition at line 61 of file [BoundingBox.h](#).

29.45.2.2 `Go::BoundingBox::BoundingBox ( int dim ) [inline], [explicit]`

Creates a [BoundingBox](#) of the specified dimension, but apart from that still uninitialized.

Definition at line 64 of file `BoundingBox.h`.

29.45.2.3 `Go::BoundingBox::BoundingBox ( const Point & low, const Point & high ) [inline]`

Creates a [BoundingBox](#) with the [Point](#) *low* specifying the lower bound in all dimensions and *high* specifying the upper bound.

Definition at line 69 of file `BoundingBox.h`.

29.45.2.4 `Go::BoundingBox::~~BoundingBox ( )`

Do not inherit from this class – nonvirtual destructor.

### 29.45.3 Member Function Documentation

29.45.3.1 `void Go::BoundingBox::addUnionWith ( const Point & pt )`

After the call, the bounding box will contain both this box and the given point.

29.45.3.2 `void Go::BoundingBox::addUnionWith ( const BoundingBox & box )`

After the call, the bounding box will contain both initial boxes.

29.45.3.3 `void Go::BoundingBox::check ( ) const`

Check that box validity. Call [valid\(\)](#) to find out if check succeeded.

29.45.3.4 `bool Go::BoundingBox::containsBox ( const BoundingBox & box, double tol = 0.0 ) const`

Returns true if this box contain the box passed as a parameter, if enlarged by *tol* in all directions.

29.45.3.5 `bool Go::BoundingBox::containsPoint ( const Point & pt, double tol = 0.0 ) const`

Returns true if the point *pt* is inside the box, or within *tol* of the boundary.

29.45.3.6 `int Go::BoundingBox::dimension ( ) const [inline]`

The dimension of the bounding box.

Definition at line 149 of file `BoundingBox.h`.

29.45.3.7 **bool** Go::BoundingBox::getOverlap ( **const** BoundingBox & *box*, **double** & *overlap*, **double** *tol* = 0.0 ) **const**

29.45.3.8 **const Point&** Go::BoundingBox::high ( ) **const** [inline]

The upper bound of the bounding box.

Definition at line 154 of file BoundingBox.h.

29.45.3.9 **std::vector<Point>** Go::BoundingBox::lineIntersect ( **const** Point & *p1*, **const** Point & *dir* ) **const**

Compute the intersections with this box and a line.

29.45.3.10 **const Point&** Go::BoundingBox::low ( ) **const** [inline]

The lower bound of the bounding box.

Definition at line 152 of file BoundingBox.h.

29.45.3.11 **bool** Go::BoundingBox::overlaps ( **const** BoundingBox & *box*, **double** *tol* = 0.0 ) **const**

Returns true if the two boxes overlap, or are a distance less than *tol* apart.

29.45.3.12 **void** Go::BoundingBox::read ( **std::istream** & *is* )

Read a bounding box from a standard istream.

29.45.3.13 **template<typename FloatType > void** Go::BoundingBox::setFromArray ( **const** FloatType \* *start*, **const** FloatType \* *end*, **int** *dim* ) [inline]

Given an array of *dim*-dimensional points stored as doubles or floats, makes the smallest bounding box containing all points in the array.

Definition at line 82 of file BoundingBox.h.

29.45.3.14 **template<typename ForwardIterator > void** Go::BoundingBox::setFromArray ( **ForwardIterator** *start*, **ForwardIterator** *end*, **int** *dim* ) [inline]

Given an array of *dim*-dimensional points stored as doubles or floats, makes the smallest bounding box containing all points in the array.

Definition at line 104 of file BoundingBox.h.

29.45.3.15 **void** Go::BoundingBox::setFromPoints ( **const** Point & *low*, **const** Point & *high* )

Makes the bounding box have lower bounds as specified in *low* and upper bounds as specified in *high*.

29.45.3.16 void Go::BoundingBox::setFromPoints ( const std::vector< Point > & points ) [inline]

Given a vector of dim-dimensional points, makes the smallest bounding box containing all points in the array.

Definition at line 125 of file BoundingBox.h.

29.45.3.17 void Go::BoundingBox::unset ( ) [inline]

Unset bounding box.

Definition at line 182 of file BoundingBox.h.

29.45.3.18 bool Go::BoundingBox::valid ( ) const [inline]

Is the bounding box initialized?

Definition at line 179 of file BoundingBox.h.

29.45.3.19 void Go::BoundingBox::write ( std::ostream & os ) const

Write a bounding box to a standard ostream.

The documentation for this class was generated from the following file:

- [gtools-core/include/GoTools/Utils/BoundingBox.h](#)

## 29.46 Go::boxStructuring::BoundingBoxStructure Class Reference

```
#include <ClosestPointUtils.h>
```

### Public Member Functions

- void [addBox](#) (shared\_ptr< [SubSurfaceBoundingBox](#) > box)  
*Add information of a specific parameter sub segment of a surface to the structure.*
- void [addSurface](#) (shared\_ptr< [SurfaceData](#) > surface)  
*Add a surface from the surface model to the structure.*
- int [n\\_boxes](#) () const  
*Get the number of segments in the structure.*
- int [n\\_surfaces](#) () const  
*Get the number of surfaces in the structure.*
- shared\_ptr< [SubSurfaceBoundingBox](#) > [getBox](#) (int i) const  
*Get a specific segment in the structure.*
- shared\_ptr< [SurfaceData](#) > [getSurface](#) (int i) const  
*Get a specific surface in the structure.*
- int [n\\_voxels\\_x](#) () const  
*Get the number of voxels in first coordinate direction.*

- `int n_voxels_y () const`  
*Get the number of voxels in second coordinate direction.*
- `int n_voxels_z () const`  
*Get the number of voxels in third coordinate direction.*
- `Point big_vox_low () const`  
*Get the lower left corner of the entire voxel structure.*
- `double voxel_length () const`  
*Get the common length of all sides in the voxels.*
- `std::vector< int > boxes_in_voxel (int i, int j, int k) const`
- `void setSurfaceCopies (int nmb_copies)`  
*Set number of surface copies.*
- `void BuildVoxelStructure (BoundingBox bigbox, double volume_reduction)`
- `bool closestPoint (int box_idx, bool any_tested, double best_dist, bool isInside, const Point &pt, double &clo_u, double &clo_v, Point &clo_pt, double &clo_dist, double epsilon, const RectDomain *domain_of_interest, double *seed) const`

### 29.46.1 Detailed Description

Class for the preprocessed information of a surface model The geometry space is split into voxels, axis aligned boxes of the same cubic shape (equal side length in all directions) The union of the voxels is rectangular, and holds all the surfaces

Definition at line 358 of file ClosestPointUtils.h.

### 29.46.2 Member Function Documentation

**29.46.2.1** `void Go::boxStructuring::BoundingBoxStructure::addBox ( shared_ptr< SubSurfaceBoundingBox > box )`  
[inline]

Add information of a specific parameter sub segment of a surface to the structure.

Definition at line 364 of file ClosestPointUtils.h.

**29.46.2.2** `void Go::boxStructuring::BoundingBoxStructure::addSurface ( shared_ptr< SurfaceData > surface )`  
[inline]

Add a surface from the surface model to the structure.

Definition at line 370 of file ClosestPointUtils.h.

**29.46.2.3** `Point Go::boxStructuring::BoundingBoxStructure::big_vox_low ( ) const` [inline]

Get the lower left corner of the entire voxel structure.

Definition at line 419 of file ClosestPointUtils.h.

29.46.2.4 `std::vector<int> Go::boxStructuring::BoundingBoxStructure::boxes_in_voxel ( int i, int j, int k ) const`  
[inline]

Get all the segments that might hit a given voxel (those where the bounding boxes hit the voxel)

Definition at line 432 of file ClosestPointUtils.h.

29.46.2.5 `void Go::boxStructuring::BoundingBoxStructure::BuildVoxelStructure ( BoundingBox bigbox, double volume_reduction )` [inline]

Creates the voxel structure with information about the segment bounding boxes that hit each voxel *bigbox* is the entire bounding box of the surface structure, the voxel structure must include *bigbox* *volume\_reduction* gives the ratio of the volume of the *bigbox* structure. It will be a lower limit on the number of voxels

Definition at line 450 of file ClosestPointUtils.h.

29.46.2.6 `bool Go::boxStructuring::BoundingBoxStructure::closestPoint ( int box_idx, bool any_tested, double best_dist, bool isInside, const Point & pt, double & clo_u, double & clo_v, Point & clo_pt, double & clo_dist, double epsilon, const RectDomain * domain_of_interest, double * seed ) const` [inline]

Test for `closestPoint`. Only used by the old code, `closestVectorsOld()` Will be removed if we know `closestVectors()` is safe

Definition at line 492 of file ClosestPointUtils.h.

29.46.2.7 `shared_ptr<SubSurfaceBoundingBox> Go::boxStructuring::BoundingBoxStructure::getBox ( int i ) const`  
[inline]

Get a specific segment in the structure.

Definition at line 389 of file ClosestPointUtils.h.

29.46.2.8 `shared_ptr<SurfaceData> Go::boxStructuring::BoundingBoxStructure::getSurface ( int i ) const` [inline]

Get a specific surface in the structure.

Definition at line 395 of file ClosestPointUtils.h.

29.46.2.9 `int Go::boxStructuring::BoundingBoxStructure::n_boxes ( ) const` [inline]

Get the number of segments in the structure.

Definition at line 377 of file ClosestPointUtils.h.

29.46.2.10 `int Go::boxStructuring::BoundingBoxStructure::n_surfaces ( ) const` [inline]

Get the number of surfaces in the structure.

Definition at line 383 of file ClosestPointUtils.h.

29.46.2.11 `int Go::boxStructuring::BoundingBoxStructure::n_voxels_x ( ) const [inline]`

Get the number of voxels in first coordinate direction.

Definition at line 401 of file `ClosestPointUtils.h`.

29.46.2.12 `int Go::boxStructuring::BoundingBoxStructure::n_voxels_y ( ) const [inline]`

Get the number of voxels in second coordinate direction.

Definition at line 407 of file `ClosestPointUtils.h`.

29.46.2.13 `int Go::boxStructuring::BoundingBoxStructure::n_voxels_z ( ) const [inline]`

Get the number of voxels in third coordinate direction.

Definition at line 413 of file `ClosestPointUtils.h`.

29.46.2.14 `void Go::boxStructuring::BoundingBoxStructure::setSurfaceCopies ( int nmb_copies ) [inline]`

Set number of surface copies.

Definition at line 438 of file `ClosestPointUtils.h`.

29.46.2.15 `double Go::boxStructuring::BoundingBoxStructure::voxel_length ( ) const [inline]`

Get the common length of all sides in the voxels.

Definition at line 425 of file `ClosestPointUtils.h`.

The documentation for this class was generated from the following file:

- [gotools-core/include/GoTools/Utils/ClosestPointUtils.h](#)

## 29.47 Go::LRSplineSurface::BSKey Struct Reference

```
#include <LRSplineSurface.h>
```

### Public Member Functions

- [bool operator< \(const BSKey &rhs\) const](#)



## Public Attributes

- [double u\\_min](#)
- [double v\\_min](#)
- [double u\\_max](#)
- [double v\\_max](#)
- [int u\\_mult1](#)
- [int v\\_mult1](#)
- [int u\\_mult2](#)
- [int v\\_mult2](#)

### 29.47.1 Detailed Description

Definition at line 105 of file LRSplineSurface.h.

### 29.47.2 Member Function Documentation

29.47.2.1 **bool** Go::LRSplineSurface::BSKey::operator< ( const BSKey & rhs ) const [inline]

Definition at line 189 of file LRSplineSurface\_impl.h.

### 29.47.3 Member Data Documentation

29.47.3.1 **double** Go::LRSplineSurface::BSKey::u\_max

Definition at line 107 of file LRSplineSurface.h.

29.47.3.2 **double** Go::LRSplineSurface::BSKey::u\_min

Definition at line 107 of file LRSplineSurface.h.

29.47.3.3 **int** Go::LRSplineSurface::BSKey::u\_mult1

Definition at line 108 of file LRSplineSurface.h.

29.47.3.4 **int** Go::LRSplineSurface::BSKey::u\_mult2

Definition at line 108 of file LRSplineSurface.h.

29.47.3.5 **double** Go::LRSplineSurface::BSKey::v\_max

Definition at line 107 of file LRSplineSurface.h.

29.47.3.6 `double Go::LRSplineSurface::BSKey::v_min`

Definition at line 107 of file LRSplineSurface.h.

29.47.3.7 `int Go::LRSplineSurface::BSKey::v_mult1`

Definition at line 108 of file LRSplineSurface.h.

29.47.3.8 `int Go::LRSplineSurface::BSKey::v_mult2`

Definition at line 108 of file LRSplineSurface.h.

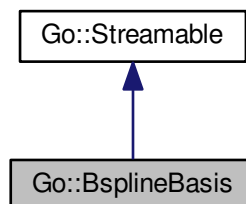
The documentation for this struct was generated from the following files:

- [Irsplines2D/include/GoTools/Irsplines2D/LRSplineSurface.h](#)
- [Irsplines2D/include/GoTools/Irsplines2D/LRSplineSurface\\_impl.h](#)

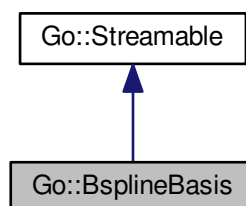
## 29.48 Go::BsplineBasis Class Reference

```
#include <BsplineBasis.h>
```

Inheritance diagram for Go::BsplineBasis:



Collaboration diagram for Go::BsplineBasis:



## Public Member Functions

- [BsplineBasis](#) ()
  - Default constructor, making an empty basis.*
- `template<typename RandomIterator >`  
[BsplineBasis](#) (int number, int [order](#), RandomIterator knotstart)
- `template<typename RandomIterator >`  
[BsplineBasis](#) (int [order](#), RandomIterator knotstart, RandomIterator knotend)
- virtual `~BsplineBasis` ()
  - Virtual destructor, enables safe inheritance.*
- `template<typename RandomIterator >`  
 void [setData](#) (int number, int [order](#), RandomIterator knotstart)
- void [swap](#) ([BsplineBasis](#) &other)
- virtual void [read](#) (std::istream &is)
- virtual void [write](#) (std::ostream &os) [const](#)
- virtual void [read\\_bin](#) (std::istream &is)
- virtual void [write\\_bin](#) (std::ostream &os) [const](#)
- int [knotInterval](#) (double t) [const](#)
- std::vector< [double](#) > [computeBasisValues](#) (double t, int derivs=0) [const](#)
- void [computeBasisValues](#) (double t, [double](#) \*basisvals\_start, int derivs=0, [double](#) resolution=1.0e-12) [const](#)
- void [computeBasisValues](#) ([const](#) [double](#) \*parvals\_start, [const](#) [double](#) \*parvals\_end, [double](#) \*basisvals\_start, int \*knotinter\_start, int derivs=0) [const](#)
- std::vector< [double](#) > [computeBasisValuesLeft](#) (double tval, int derivs) [const](#)
- void [computeBasisValuesLeft](#) ([const](#) [double](#) \*parvals\_start, [const](#) [double](#) \*parvals\_end, [double](#) \*basisvals\_start, int \*knotinter\_start, int derivs) [const](#)
- void [reverseParameterDirection](#) ()
- void [rescale](#) ([double](#) new\_start, [double](#) new\_end)
- void [insertKnot](#) ([double](#) apar)
- void [increaseOrder](#) (int [order](#))
- int [knotIntervalFuzzy](#) ([double](#) &t, [double](#) tol=DEFAULT\_PARAMETER\_EPSILON) [const](#)
- void [insertKnot](#) ([const](#) std::vector< [double](#) > &new\_knots)
- void [removeKnot](#) ([double](#) old\_knot)
- int [lastKnotInterval](#) () [const](#)
- [double](#) [grevilleParameter](#) (int index) [const](#)
- int [knotMultiplicity](#) ([const](#) [double](#) parval) [const](#)
- void [knotMultiplicities](#) (std::vector< int > &multiplicities) [const](#)
- int [endMultiplicity](#) (bool atstart) [const](#)
- void [knotsSimple](#) (std::vector< [double](#) > &result) [const](#)
- int [numElem](#) () [const](#)
- void [cNDiscontinuities](#) (std::vector< [double](#) > &cNDisconts, int depth) [const](#)
- int [getMinContinuity](#) () [const](#)
  - The minimum continuity at inner knots.*
- [bool](#) [isKreg](#) () [const](#)
- void [coefsAffectingParam](#) ([double](#) tpar, int &first\_coef, int &last\_coef) [const](#)
- int [numCoefs](#) () [const](#)
- int [order](#) () [const](#)
- [double](#) [startparam](#) () [const](#)
- [double](#) [endparam](#) () [const](#)
- std::vector< [double](#) >::const\_iterator [begin](#) () [const](#)
- std::vector< [double](#) >::const\_iterator [end](#) () [const](#)
- std::vector< [double](#) >::iterator [begin](#) ()
- std::vector< [double](#) >::iterator [end](#) ()
- std::vector< [double](#) > [getKnots](#) ()
- void [check](#) () [const](#)

- Checks the validity of the object. Throws at error.*

  - `bool isOK () const`

*Checks the validity of the object. Does not throw.*

  - `bool indistinctKnots (double tol, std::vector< double > &first_knotval) const`

*Look for indistinct knots.*

  - `bool sameSplineSpace (const BsplineBasis &other, double tol=DEFAULT_PARAMETER_EPSILON) const`

*Check equality of two spline spaces.*

  - `std::vector< double > missingKnots (const BsplineBasis &other, double tol=DEFAULT_PARAMETER_EPSILON) const`

*Return knots existing in the other spline space, but not in this one.*

  - `BsplineBasis subBasis (double tmin, double tmax, double knot_diff_tol=1e-05) const`

*Return the basis for the subspace limited by the interval [tmin,tmax].*

  - `BsplineBasis extendedBasis (int order) const`

## Related Functions

(Note that these are not member functions.)

- void `computeBasisValuesLeft (double tval, double *basisvals_start, int derivs, double resolution=1.0e-12) const`

### 29.48.1 Detailed Description

Class representing a B-spline basis of a spline space. This basis is defined by its order, its dimension (number of basis functions) and its knotvector.

Definition at line 63 of file BsplineBasis.h.

### 29.48.2 Constructor & Destructor Documentation

#### 29.48.2.1 `Go::BsplineBasis::BsplineBasis ( ) [inline]`

Default constructor, making an empty basis.

Definition at line 67 of file BsplineBasis.h.

#### 29.48.2.2 `template<typename RandomIterator > Go::BsplineBasis::BsplineBasis ( int number, int order, RandomIterator knotstart ) [inline]`

Constructor

Parameters

|                  |                                                                                                                                                     |
|------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>number</i>    | Number of basis functions in space                                                                                                                  |
| <i>order</i>     | order of the basis functions (polynomial degree + 1)                                                                                                |
| <i>knotstart</i> | iterator pointing to start of defining knotvector The length of the knotvector is deduced automatically, and should be equal to 'order' + 'number'. |

Definition at line 79 of file BsplineBasis.h.

**29.48.2.3** `template<typename RandomIterator > Go::BsplineBasis::BsplineBasis ( int order, RandomIterator knotstart, RandomIterator knotend ) [inline]`

Constructor

Parameters

|                  |                                                                                                                                                                                     |
|------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>order</i>     | order of the basis functions (polynomial degree + 1)                                                                                                                                |
| <i>knotstart</i> | iterator pointing to start of defining knotvector                                                                                                                                   |
| <i>knotend</i>   | iterator pointing to one-past-end of defining knotvector. The number of basis functions will be deduced automatically by subtracting the 'order' from the length of the knotvector. |

Definition at line 96 of file BsplineBasis.h.

**29.48.2.4** `virtual Go::BsplineBasis::~BsplineBasis ( ) [virtual]`

Virtual destructor, enables safe inheritance.

### 29.48.3 Member Function Documentation

**29.48.3.1** `std::vector<double>::const_iterator Go::BsplineBasis::begin ( ) const [inline]`

Get a const-iterator to the beginning of the knotvector

Returns

an iterator to the beginning of the knotvector

Definition at line 349 of file BsplineBasis.h.

**29.48.3.2** `std::vector<double>::iterator Go::BsplineBasis::begin ( ) [inline]`

Get an iterator to the beginning of the knotvector

Returns

an iterator to the beginning of the knotvector

Definition at line 359 of file BsplineBasis.h.

**29.48.3.3** `void Go::BsplineBasis::check ( ) const`

Checks the validity of the object. Throws at error.

**29.48.3.4** `void Go::BsplineBasis::cNDiscontinuities ( std::vector< double > & cNDisconts, int depth ) const`

Find the first knot, the last knot, and all knots where not all B-splines have  $C^N$ -continuity, i.e. knots with multiplicity at least the order of the B-splines minus N

## Parameters

|                   |                                                             |
|-------------------|-------------------------------------------------------------|
| <i>cNdisconts</i> | upon function return, this vector holds all the knots found |
| <i>depth</i>      | the value of N, the degree of smoothness                    |

29.48.3.5 `void Go::BsplineBasis::coefsAffectingParam ( double tpar, int & first_coef, int & last_coef ) const`

Get the range of basis functions that are involved when evaluating at a given parameter.

## Parameters

|                   |                                                                |
|-------------------|----------------------------------------------------------------|
| <i>tpar</i>       | the parameter where we want to determine the influential range |
| <i>first_coef</i> | on function completion, will point to start of relevant range  |
| <i>last_coef</i>  | on function completion, will point to end of relevant range    |

29.48.3.6 `std::vector<double> Go::BsplineBasis::computeBasisValues ( double t, int derivs = 0 ) const`

Create a vector containing the basis values in a given parameter.

## Parameters

|               |                                                                                         |
|---------------|-----------------------------------------------------------------------------------------|
| <i>t</i>      | the parameter at which to evaluate the basis functions                                  |
| <i>derivs</i> | the number of function derivatives to calculate for each nonzero basis function at 't'. |

## Returns

a vector containing `order()` \* ('derivs'+1) values, which are the function values and derivatives up to order 'derivs' of the nonzero basis functions at parameter value 't'. The order in which these values are stored is as follows: first all function values for the 'order' nonzero basis functions, then their first derivatives, then their second, etc..

29.48.3.7 `void Go::BsplineBasis::computeBasisValues ( double t, double * basisvals_start, int derivs = 0, double resolution = 1.0e-12 ) const`

Space for the basis values is allocated outside and passed in. Needed space is equal to `order()*('derivs'+1)` doubles.

## Parameters

|                        |                                                                                                                                                                                                              |
|------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>t</i>               | the parameter at which to evaluate the basis function                                                                                                                                                        |
| <i>basisvals_start</i> | pointer to the memory area where the result will be written. This must be allocated by the user. The order in which the result values are written is similar to <code>computeBasisValues(double, int)</code> |
| <i>derivs</i>          | number of derivatives that should be evaluated for each nonzero basis function ( <code>derivs</code> = 0 => only function values will be computed).                                                          |
| <i>resolution</i>      | accuracy for determining whether a given parametric value lies exactly 'on' a knot.                                                                                                                          |

29.48.3.8 `void Go::BsplineBasis::computeBasisValues ( const double * parvals_start, const double * parvals_end, double * basisvals_start, int * knotinter_start, int derivs = 0 ) const`

Compute basis values for many points simultaneously.

#### Parameters

|                        |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>parvals_start</i>   | pointer to the start of list of parameters where you want to evaluate the basis functions                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| <i>parvals_end</i>     | pointer to one-past-end of list of parameters where you want to evaluate the basis functions                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| <i>basisvals_start</i> | pointer to range where the result of the evaluations will be written. The necessary memory is equal to 'derivs' multiplied by <code>order()</code> multiplied by length of parameter list, and should be allocated by user before calling the function. The order in which these values are stored is: first all values concerning the first parameter in the input range, then all values concerning the second parameter in the input range, etc. For each such parameter, the concerned values are stored in the order described in <code>computeBasisValues(double, int)</code> |
| <i>knotinter_start</i> | pointer to range where the indexes to the parameter intervals in which the evaluated parameter values are located. ie. the index at <code>knotinter_start[i]</code> refer to the parameter interval where the i'th parameter in the range [ <code>parvals_start</code> , <code>parvals_end</code> ] is located. The total size of this range is therefore equal to ( <code>parvals_end</code> - <code>parvals_start</code> ), and should be allocated by the user.                                                                                                                  |
| <i>derivs</i>          | number of derivatives that should be evaluated for each nonzero basis function ( <code>derivs = 0</code> => only function values will be computed).                                                                                                                                                                                                                                                                                                                                                                                                                                 |

29.48.3.9 `std::vector<double> Go::BsplineBasis::computeBasisValuesLeft ( double tval, int derivs ) const`

This function is similar to `computeBasisValues(double, int)`, except that the values are calculated from the left, as opposed to the default right-evaluation.

#### See also

[computeBasisValues\(\)](#)

29.48.3.10 `void Go::BsplineBasis::computeBasisValuesLeft ( const double * parvals_start, const double * parvals_end, double * basisvals_start, int * knotinter_start, int derivs ) const`

This function is similar to `computeBasisValues(const double*, const double*, double*, int*, int)`, except that the values are calculated from the left, as opposed to the default right-evaluation.

29.48.3.11 `std::vector<double>::const_iterator Go::BsplineBasis::end ( ) const` `[inline]`

Get a const-iterator to the end of the knotvector

#### Returns

an iterator to the end of the knotvector

Definition at line 354 of file `BsplineBasis.h`.

29.48.3.12 `std::vector<double>::iterator Go::BsplineBasis::end ( )` `[inline]`

Get an iterator to the end of the knotvector

#### Returns

an iterator to the end of the knotvector

Definition at line 363 of file BsplineBasis.h.

29.48.3.13 `int Go::BsplineBasis::endMultiplicity ( bool atstart )` `const`

return the multiplicity of the first or the last knot

#### Parameters

|                |                                                                                                                                            |
|----------------|--------------------------------------------------------------------------------------------------------------------------------------------|
| <i>atstart</i> | if this is 'true' then the multiplicity of the first knot will be returned. Otherwise, the multiplicity of the last knot will be returned. |
|----------------|--------------------------------------------------------------------------------------------------------------------------------------------|

#### Returns

the corresponding knot multiplicity

29.48.3.14 `double Go::BsplineBasis::endparam ( )` `const` `[inline]`

Query the end value of the parameter range

#### Returns

the end value of the parameter range

Definition at line 344 of file BsplineBasis.h.

29.48.3.15 `BsplineBasis Go::BsplineBasis::extendedBasis ( int order )` `const`

Make a basis function with the same inner knots, but with increased degree NB! The continuity of the space is increased If the basis is periodic, the new knots will be inserted at the start and end of the parameter interval, i.e. the continuity at the seam will be reduced Order is equal to polynomial degree + 1 The function throws if the specified order is less than the existing one

29.48.3.16 `std::vector<double> Go::BsplineBasis::getKnots ( )` `[inline]`

Definition at line 366 of file BsplineBasis.h.

29.48.3.17 `int Go::BsplineBasis::getMinContinuity ( )` `const`

The minimum continuity at inner knots.

29.48.3.18 `double Go::BsplineBasis::grevilleParameter ( int index )` `const` `[inline]`

Get the greville parameter starting from a certain knot



## Parameters

|              |                                                                             |
|--------------|-----------------------------------------------------------------------------|
| <i>index</i> | the index of the knot from which the greville parameter should be computed. |
|--------------|-----------------------------------------------------------------------------|

## Returns

the corresponding greville parameter

Definition at line 417 of file BsplineBasis.h.

**29.48.3.19** void Go::BsplineBasis::increaseOrder ( int *order* )

Adapt the knot vector to an increased polynomial order The given order is the expected new polynomial order of the basis

**29.48.3.20** bool Go::BsplineBasis::indistinctKnots ( double *tol*, std::vector< double > & *first\_knotval* ) const

Look for indistinct knots.

**29.48.3.21** void Go::BsplineBasis::insertKnot ( double *apar* )

Insert a knot into the knotvector.

## Parameters

|             |                                  |
|-------------|----------------------------------|
| <i>apar</i> | parameter value of the new knot. |
|-------------|----------------------------------|

**29.48.3.22** void Go::BsplineBasis::insertKnot ( const std::vector< double > & *new\_knots* )

Insert several knots into the knotvector

## Parameters

|                  |                                                                 |
|------------------|-----------------------------------------------------------------|
| <i>new_knots</i> | a STL vector containing the new knots to insert into the vector |
|------------------|-----------------------------------------------------------------|

**29.48.3.23** bool Go::BsplineBasis::isKreg ( ) const `[inline]`

query whether the basis has a k-regular knotvector (start and end knot has multiplicity equal to [order\(\)](#)).

## Returns

whether this basis has a k-regular knotvector or not.

Definition at line 314 of file BsplineBasis.h.

29.48.3.24 `bool Go::BsplineBasis::isOK ( ) const`

Checks the validity of the object. Does not throw.

29.48.3.25 `int Go::BsplineBasis::knotInterval ( double t ) const`

Find the interval in which the parameter value '*t*' lies.

#### Parameters

|          |                              |
|----------|------------------------------|
| <i>t</i> | the parameter value to test. |
|----------|------------------------------|

29.48.3.26 `int Go::BsplineBasis::knotIntervalFuzzy ( double & t, double tol = DEFAULT_PARAMETER_EPSILON ) const`

Find the interval in which a given parameter lies.

#### Parameters

|            |                                                                                                                                                                                                               |
|------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>t</i>   | the parameter we want to locate within an interval                                                                                                                                                            |
| <i>tol</i> | if ' <i>t</i> ' is within DEFAULT_PARAMETER_EPSILON of an existing knot, ' <i>t</i> ' will be changed to that knot. To some callers of this function, that may be the primary wanted effect of this function. |

29.48.3.27 `void Go::BsplineBasis::knotMultiplicities ( std::vector< int > & multiplicities ) const`

Get the vector of knot multiplicities. To get the vector of corresponding knot values, use [knotsSimple\(\)](#).

#### Parameters

|                       |                                             |
|-----------------------|---------------------------------------------|
| <i>multiplicities</i> | the resulting vector of knot multiplicities |
|-----------------------|---------------------------------------------|

29.48.3.28 `int Go::BsplineBasis::knotMultiplicity ( const double parval ) const`

If there is a knot in the knotvector with the value '*parval*', its multiplicity is returned. Otherwise, 0 is returned.

#### Parameters

|               |                                                                     |
|---------------|---------------------------------------------------------------------|
| <i>parval</i> | the parameter value for which we want to know the knot multiplicity |
|---------------|---------------------------------------------------------------------|

#### Returns

the multiplicity in case the knot exists, 0 otherwise.

29.48.3.29 void Go::BsplineBasis::knotsSimple ( std::vector< double > & *result* ) const

Store all knots without multiplicity into one array

#### Parameters

|               |                                                       |
|---------------|-------------------------------------------------------|
| <i>result</i> | upon function return, this vector holds all the knots |
|---------------|-------------------------------------------------------|

29.48.3.30 int Go::BsplineBasis::lastKnotInterval ( ) const [inline]

Return the index of the left knot of the last knot interval accessed for some reason (evaluation, etc.) in this basis.

Definition at line 411 of file BsplineBasis.h.

29.48.3.31 std::vector<double> Go::BsplineBasis::missingKnots ( const BsplineBasis & *other*, double *tol* = DEFAULT\_PARAMETER\_EPSILON ) const

Return knots existing in the other spline space, but not in this one.

29.48.3.32 int Go::BsplineBasis::numCoefs ( ) const [inline]

Query the number of basis functions

#### Returns

the number of basis functions

Definition at line 329 of file BsplineBasis.h.

29.48.3.33 int Go::BsplineBasis::numElem ( ) const

Query the number of elements, number of non-degenerate knot intervals, in the knot vector

29.48.3.34 int Go::BsplineBasis::order ( ) const [inline]

Query the order of the basis functions

#### Returns

the order of the basis functions

Definition at line 334 of file BsplineBasis.h.

29.48.3.35 virtual void Go::BsplineBasis::read ( std::istream & *is* ) [virtual]

Read the [BsplineBasis](#) from a stream (ASCII mode).

## Parameters

|           |                                                                 |
|-----------|-----------------------------------------------------------------|
| <i>is</i> | the stream from which the <a href="#">BsplineBasis</a> is read. |
|-----------|-----------------------------------------------------------------|

Implements [Go::Streamable](#).

29.48.3.36 `virtual void Go::BsplineBasis::read_bin ( std::istream & is ) [virtual]`

Read the [BsplineBasis](#) from a stream (binary mode).

## Parameters

|           |                                                                 |
|-----------|-----------------------------------------------------------------|
| <i>is</i> | the stream from which the <a href="#">BsplineBasis</a> is read. |
|-----------|-----------------------------------------------------------------|

29.48.3.37 `void Go::BsplineBasis::removeKnot ( double old_knot )`

Remove a knot from the knotvector.

## Parameters

|                 |                                                                                                                                         |
|-----------------|-----------------------------------------------------------------------------------------------------------------------------------------|
| <i>old_knot</i> | Parameter value indicating the knot to be removed. The corresponding knot is the largest one which is less than or equal to 'old_knot'. |
|-----------------|-----------------------------------------------------------------------------------------------------------------------------------------|

29.48.3.38 `void Go::BsplineBasis::rescale ( double new_start, double new_end )`

Rescales the knotvector so that parameter values lay in the specified interval.

## Parameters

|                  |                              |
|------------------|------------------------------|
| <i>new_start</i> | start of specified interval. |
| <i>new_end</i>   | end of specified interval.   |

29.48.3.39 `void Go::BsplineBasis::reverseParameterDirection ( )`

We want the new knot vector to be a mirror image of the old one, translated so that it starts and ends in the same values as before.

29.48.3.40 `bool Go::BsplineBasis::sameSplineSpace ( const BsplineBasis & other, double tol = DEFAULT_PARAMETER_EPSILON ) const`

Check equality of two spline spaces.

29.48.3.41 `template<typename RandomIterator > void Go::BsplineBasis::setData ( int number, int order, RandomIterator knotstart ) [inline]`

Change the defining data of an already-existing [BsplineBasis](#).

#### Parameters

|                  |                                                    |
|------------------|----------------------------------------------------|
| <i>number</i>    | Number of basis functions in space                 |
| <i>order</i>     | of the basis functions (polynomial degree + 1)     |
| <i>knotstart</i> | iterator pointing to start of defining knotvector. |

Definition at line 115 of file BsplineBasis.h.

29.48.3.42 `double Go::BsplineBasis::startparam ( ) const [inline]`

Query the start value of the parameter range

#### Returns

the start value of the parameter range

Definition at line 339 of file BsplineBasis.h.

29.48.3.43 `BsplineBasis Go::BsplineBasis::subBasis ( double tmin, double tmax, double knot_diff_tol = 1e-05 ) const`

Return the basis for the subspace limited by the interval [tmin,tmax].

29.48.3.44 `void Go::BsplineBasis::swap ( BsplineBasis & other )`

Quick swap of two BsplineBasises.

#### Parameters

|              |                                                |
|--------------|------------------------------------------------|
| <i>other</i> | the <a href="#">BsplineBasis</a> to swap with. |
|--------------|------------------------------------------------|

29.48.3.45 `virtual void Go::BsplineBasis::write ( std::ostream & os ) const [virtual]`

Write the [BsplineBasis](#) to a stream (ASCII mode).

#### Parameters

|           |                                                                 |
|-----------|-----------------------------------------------------------------|
| <i>os</i> | the stream to which the <a href="#">BsplineBasis</a> is written |
|-----------|-----------------------------------------------------------------|

Implements [Go::Streamable](#).

29.48.3.46 `virtual void Go::BsplineBasis::write_bin ( std::ostream & os ) const` [virtual]

Write the [BsplineBasis](#) to a stream (binary mode).

#### Parameters

|                 |                                                                 |
|-----------------|-----------------------------------------------------------------|
| <code>os</code> | the stream to which the <a href="#">BsplineBasis</a> is written |
|-----------------|-----------------------------------------------------------------|

## 29.48.4 Friends And Related Function Documentation

29.48.4.1 `void computeBasisValuesLeft ( double tval, double * basisvals_start, int derivs, double resolution = 1.0e-12 ) const` [related]

This function is similar to `computeBasisValues(double, double*, int)`, except that the values are calculated from the left, as opposed to the default right-evaluation.

#### Parameters

|                         |                                                                                                                                                                  |
|-------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>resolution</code> | tolerance deciding whether a parameter value is situated "directly on" a knot. (In such situations, it might matter whether we evaluate from left or from left). |
|-------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------|

#### See also

[computeBasisValues\(\) \(\)](#)

The documentation for this class was generated from the following file:

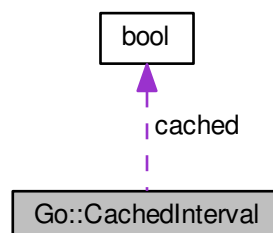
- [gotools-core/include/GoTools/geometry/BsplineBasis.h](#)

## 29.49 Go::CachedInterval Struct Reference

Helper struct for saving bracketed bounds of influence areas.

```
#include <IntersectionPointUtils.h>
```

Collaboration diagram for `Go::CachedInterval`:



## Public Member Functions

- [CachedInterval](#) ()

## Public Attributes

- [double](#) `inside`
- [double](#) `outside`
- [bool](#) `cached`

### 29.49.1 Detailed Description

Helper struct for saving bracketed bounds of influence areas.

Definition at line 95 of file `IntersectionPointUtils.h`.

### 29.49.2 Constructor & Destructor Documentation

#### 29.49.2.1 `Go::CachedInterval::CachedInterval ( )` [`inline`]

Definition at line 97 of file `IntersectionPointUtils.h`.

### 29.49.3 Member Data Documentation

#### 29.49.3.1 `bool` `Go::CachedInterval::cached`

Definition at line 101 of file `IntersectionPointUtils.h`.

#### 29.49.3.2 `double` `Go::CachedInterval::inside`

Definition at line 99 of file `IntersectionPointUtils.h`.

#### 29.49.3.3 `double` `Go::CachedInterval::outside`

Definition at line 100 of file `IntersectionPointUtils.h`.

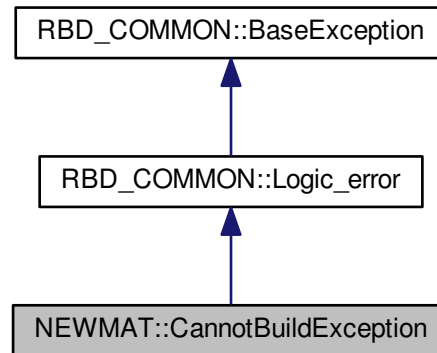
The documentation for this struct was generated from the following file:

- [intersections/include/GoTools/intersections/IntersectionPointUtils.h](#)

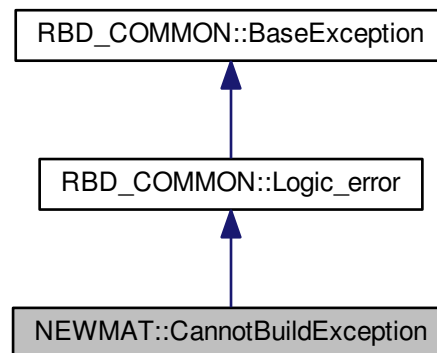
## 29.50 NEWMAT::CannotBuildException Class Reference

```
#include <newmat.h>
```

Inheritance diagram for NEWMAT::CannotBuildException:



Collaboration diagram for NEWMAT::CannotBuildException:



### Public Member Functions

- [CannotBuildException](#) (const char \*matrix)

### Static Public Attributes

- static unsigned long [Select](#)



## Additional Inherited Members

### 29.50.1 Detailed Description

Definition at line 1684 of file newmat.h.

### 29.50.2 Constructor & Destructor Documentation

#### 29.50.2.1 CannotBuildException::CannotBuildException ( `const char * matrix` )

Definition at line 190 of file newmatex.cpp.

### 29.50.3 Member Data Documentation

#### 29.50.3.1 unsigned long CannotBuildException::Select `[static]`

Definition at line 1687 of file newmat.h.

The documentation for this class was generated from the following files:

- [newmat/include/newmat.h](#)
- [newmat/src/newmatex.cpp](#)

## 29.51 Go::CellDivision Class Reference

```
#include <CellDivision.h>
```

### Public Member Functions

- [CellDivision](#) ()
- [CellDivision](#) (std::vector< [ftSurface](#) \* > faces, int n1, int n2)
- [CellDivision](#) (std::vector< [ftSurface](#) \* > faces, int n1, int n2, int n3)
- [CellDivision](#) (std::vector< [ftSurface](#) \* > faces, std::vector< int > n)
- [~CellDivision](#) ()
- int [numCells](#) () const
- const [ftCell](#) & [getCell](#) (int idx) const
- void [addFace](#) ([ftSurface](#) \*face)
- void [removeFace](#) ([ftSurface](#) \*face)
- void [constructCells](#) ()
- void [constructCells](#) (int n1, int n2)
- void [constructCells](#) (int n1, int n2, int n3)
- void [constructCells](#) (std::vector< int > n)
- [BoundingBox](#) [big\\_box](#) () const

### 29.51.1 Detailed Description

Used internally in [SurfaceModel](#). [CellDivision](#) divides the space occupied by a surface model into a number of equal boxes. The division is intended to improve the performance in closest point and intersection computations by giving the means to perform an initial interception test before the actual computations are started

Definition at line 106 of file CellDivision.h.

### 29.51.2 Constructor & Destructor Documentation

29.51.2.1 `Go::CellDivision::CellDivision ( )` [\[inline\]](#)

Definition at line 109 of file CellDivision.h.

29.51.2.2 `Go::CellDivision::CellDivision ( std::vector< ftSurface * > faces, int n1, int n2 )`

29.51.2.3 `Go::CellDivision::CellDivision ( std::vector< ftSurface * > faces, int n1, int n2, int n3 )`

29.51.2.4 `Go::CellDivision::CellDivision ( std::vector< ftSurface * > faces, std::vector< int > n )`

29.51.2.5 `Go::CellDivision::~~CellDivision ( )`

### 29.51.3 Member Function Documentation

29.51.3.1 `void Go::CellDivision::addFace ( ftSurface * face )`

29.51.3.2 `BoundingBox Go::CellDivision::big_box ( ) const` [\[inline\]](#)

Definition at line 138 of file CellDivision.h.

29.51.3.3 `void Go::CellDivision::constructCells ( )`

29.51.3.4 `void Go::CellDivision::constructCells ( int n1, int n2 )`

29.51.3.5 `void Go::CellDivision::constructCells ( int n1, int n2, int n3 )`

29.51.3.6 `void Go::CellDivision::constructCells ( std::vector< int > n )`

29.51.3.7 `const ftCell& Go::CellDivision::getCell ( int idx ) const` [\[inline\]](#)

Definition at line 126 of file CellDivision.h.

29.51.3.8 `int Go::CellDivision::numCells ( ) const` [\[inline\]](#)

Definition at line 121 of file CellDivision.h.

29.51.3.9 void Go::CellDivision::removeFace ( ftSurface \* face )

The documentation for this class was generated from the following file:

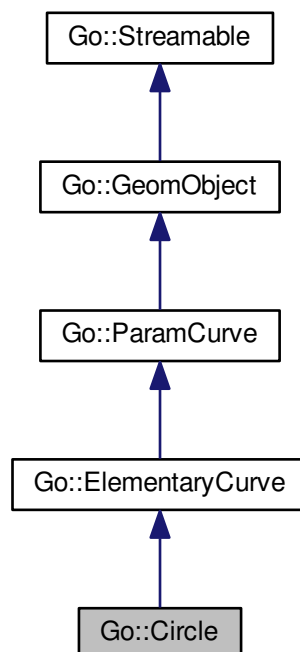
- [compositemodel/include/GoTools/compositemodel/CellDivision.h](#)

## 29.52 Go::Circle Class Reference

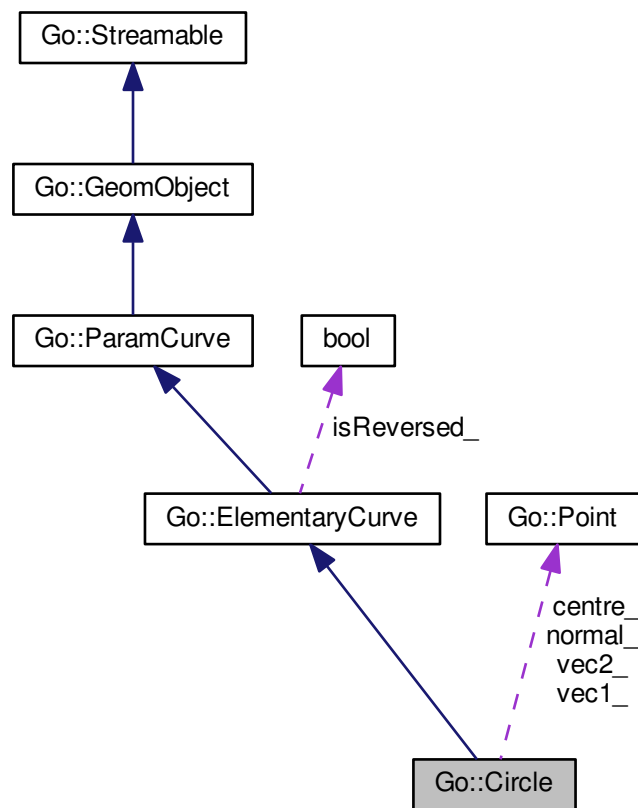
Class that represents a circle. It is a subclass of [ElementaryCurve](#) and thus has a parametrization.

```
#include <Circle.h>
```

Inheritance diagram for Go::Circle:



Collaboration diagram for Go::Circle:



## Public Member Functions

- [Circle](#) ()
- [Circle](#) (double radius, [Point](#) centre, [Point](#) normal, [Point](#) x\_axis, bool isReversed=false)
- virtual [~Circle](#) ()
  - virtual destructor - ensures safe inheritance*
- virtual void [read](#) (std::istream &is)
- virtual void [write](#) (std::ostream &os) const
- virtual [BoundingBox](#) boundingBox () const
  - Return the object's bounding box.*
- virtual int [dimension](#) () const
  - Return the dimension of the space in which the object lies (usually 2 or 3)*
- virtual [ClassType](#) instanceType () const
  - Return the class type identifier of a given, derived instance of [GeomObject](#).*
- virtual [Circle](#) \* [clone](#) () const
- virtual void [point](#) ([Point](#) &pt, double tpar) const
- virtual void [point](#) (std::vector< [Point](#) > &pts, double tpar, int derives, bool from\_right=true) const
- virtual double [startparam](#) () const
- virtual double [endparam](#) () const

- virtual void `setParameterInterval` (double t1, double t2)
- virtual `SplineCurve * geometryCurve` ()
- virtual `SplineCurve * createSplineCurve` () const
- virtual `bool isDegenerate` (double degenerate\_epsilon)
- virtual `Circle * subCurve` (double from\_par, double to\_par, double fuzzy=DEFAULT\_PARAMETER\_EPSILON) const
- virtual `DirectionCone directionCone` () const
- virtual void `appendCurve` (ParamCurve \*cv, bool reparam=true)
- virtual void `appendCurve` (ParamCurve \*cv, int continuity, double &dist, bool reparam=true)
- virtual void `closestPoint` (const Point &pt, double tmin, double tmax, double &clo\_t, Point &clo\_pt, double &clo\_dist, double const \*seed=0) const
- virtual `double length` (double tol)
- `Point getCentre` () const
- `Point getNormal` () const
- `Point getXAxis` () const
- `double getRadius` () const
- virtual void `setParamBounds` (double startpar, double endpar)
- virtual void `translateCurve` (const Point &dir)
- `bool isClosed` () const
- virtual `bool isAxisRotational` (Point &centre, Point &axis, Point &vec, double &angle)
- virtual void `swapParameters2D` ()
- virtual `bool isInPlane` (const Point &loc, const Point &axis, double eps, Point &normal) const  
*Check if the line lies in a plane passing through a given axis.*
- virtual `bool isInPlane` (const Point &norm, double eps, Point &pos) const  
*Check if the circle lies in a plane with a given normal.*

### Static Public Member Functions

- static `ClassType classType` ()

### Protected Member Functions

- void `setSpanningVectors` ()

### Protected Attributes

- `double radius_`
- `Point centre_`
- `Point normal_`
- `Point vec1_`
- `Point vec2_`
- `double startparam_`
- `double endparam_`

## 29.52.1 Detailed Description

Class that represents a circle. It is a subclass of `ElementaryCurve` and thus has a parametrization.

A `Circle` has a natural parametrization in terms of the angle  $t$ :  $\mathbf{p}(t) = \mathbf{C} + r(\cos(t) \mathbf{x} + \sin(t) \mathbf{y})$ , where  $\mathbf{C}$  is the centre,  $r$  is the radius, and  $\mathbf{x}$  and  $\mathbf{y}$  are the (local) axes. The parametrization is (in principle) bounded:  $0 \leq t \leq 2\pi$ . The dimension is either 2 or 3.

Definition at line 59 of file `Circle.h`.

## 29.52.2 Constructor & Destructor Documentation

### 29.52.2.1 `Go::Circle::Circle ( ) [inline]`

Default constructor. Constructs an uninitialized [Circle](#) which can only be assigned to or read into.

Definition at line 64 of file `Circle.h`.

### 29.52.2.2 `Go::Circle::Circle ( double radius, Point centre, Point normal, Point x_axis, bool isReversed = false )`

Constructor. Input is the radius, the centre, the normal to the plane of the circle, and the (approximate) direction of the x-axis. The dimension must be either 2 or 3.

### 29.52.2.3 `virtual Go::Circle::~~Circle ( ) [virtual]`

virtual destructor - ensures safe inheritance

## 29.52.3 Member Function Documentation

### 29.52.3.1 `virtual void Go::Circle::appendCurve ( ParamCurve * cv, bool reparam = true ) [virtual]`

append a curve to this curve, with eventual reparametrization NB: This virtual member function currently only works for `SplineCurves` and `CurveOnSurfaces`. Moreover, 'this' curve and the 'cv' curve must be of the same type.

#### Parameters

|                |                                                          |
|----------------|----------------------------------------------------------|
| <i>cv</i>      | the curve to append to 'this' curve.                     |
| <i>reparam</i> | specify whether or not there should be reparametrization |

Implements [Go::ParamCurve](#).

### 29.52.3.2 `virtual void Go::Circle::appendCurve ( ParamCurve * cv, int continuity, double & dist, bool reparam = true ) [virtual]`

append a curve to this curve, with eventual reparametrization

#### Parameters

|                   |                                                                                                       |
|-------------------|-------------------------------------------------------------------------------------------------------|
| <i>cv</i>         | the curve to append to 'this' curve.                                                                  |
| <i>continuity</i> | the required continuity at the transition. Can be $G^{(-1)}$ and upwards.                             |
| <i>dist</i>       | a measure of the local distortion around the transition in order to achieve the specified continuity. |
| <i>reparam</i>    | specify whether or not there should be reparametrization                                              |

Implements [Go::ParamCurve](#).

29.52.3.3 virtual **BoundingBox** Go::Circle::boundingBox ( ) const [virtual]

Return the object's bounding box.

Implements [Go::GeomObject](#).

29.52.3.4 static **ClassType** Go::Circle::classType ( ) [static]

29.52.3.5 virtual **Circle\*** Go::Circle::clone ( ) const [virtual]

The clone-function is herited from [GeomObject](#), but overridden here to get a covariant return type (for those compilers that allow this).

Implements [Go::ElementaryCurve](#).

29.52.3.6 virtual void Go::Circle::closestPoint ( const **Point** & *pt*, double *tmin*, double *tmax*, double & *clo\_t*, **Point** & *clo\_pt*, double & *clo\_dist*, double const \* *seed* = 0 ) const [virtual]

Compute the closest point from an interval of this curve to a specified point.

Parameters

|                 |                                                                                                                                          |
|-----------------|------------------------------------------------------------------------------------------------------------------------------------------|
| <i>pt</i>       | point we want to find the closest point to                                                                                               |
| <i>tmin</i>     | start parameter of search interval                                                                                                       |
| <i>tmax</i>     | end parameter of search interval                                                                                                         |
| <i>clo_t</i>    | upon function return, 'clo_t' will contain the parameter value of the closest point found.                                               |
| <i>clo_pt</i>   | upon function return, 'clo_pt' will contain the position of the closest point found.                                                     |
| <i>clo_dist</i> | upon function return, 'clo_dist' will contain the distance between 'pt' and the closest point found.                                     |
| <i>seed</i>     | pointer to initial guess value, provided by the user (can be 0, for which the algorithm will determine a (hopefully) reasonable choice). |

Implements [Go::ParamCurve](#).

29.52.3.7 virtual **SplineCurve\*** Go::Circle::createSplineCurve ( ) const [virtual]

Creates a [SplineCurve](#) representation of the circle. The spline representation is geometrically exact, but the parametrization is only an approximation. If `setParameterBounds()` have been used to set bounds on the parametrization, then the endpoints of the resulting spline curve *does* have correct parameter values.

Implements [Go::ElementaryCurve](#).

29.52.3.8 virtual int Go::Circle::dimension ( ) const [virtual]

Return the dimension of the space in which the object lies (usually 2 or 3)

Implements [Go::GeomObject](#).

29.52.3.9 virtual **DirectionCone** Go::Circle::directionCone ( ) const [virtual]

Creates a [DirectionCone](#) which covers all tangent directions of this curve.

**Returns**

the smallest [DirectionCone](#) containing all tangent directions of this curve.

Implements [Go::ParamCurve](#).

29.52.3.10 virtual **double** Go::Circle::endparam ( ) const [virtual]

Query the end parameter of the curve

**Returns**

the curve's end parameter

Implements [Go::ParamCurve](#).

29.52.3.11 virtual **SplineCurve\*** Go::Circle::geometryCurve ( ) [virtual]

If the definition of this [ParamCurve](#) contains a [SplineCurve](#) describing its spatial shape, then this function will return a pointer to this [SplineCurve](#). Otherwise it will return a null pointer. The returned curve is NEWed, so the user is responsible for deleting it. This function may have side-effects.

**Returns**

a pointer to a [SplineCurve](#) representation of the [ParamCurve](#), if it exists. Null pointer otherwise.

Implements [Go::ParamCurve](#).

29.52.3.12 **Point** Go::Circle::getCentre ( ) const [inline]

Definition at line 151 of file Circle.h.

29.52.3.13 **Point** Go::Circle::getNormal ( ) const [inline]

Definition at line 156 of file Circle.h.

29.52.3.14 **double** Go::Circle::getRadius ( ) const [inline]

Definition at line 166 of file Circle.h.



29.52.3.15 `Point Go::Circle::getXAxis ( ) const [inline]`

Definition at line 161 of file Circle.h.

29.52.3.16 `virtual ClassType Go::Circle::instanceType ( ) const [virtual]`

Return the class type identifier of a given, derived instance of [GeomObject](#).

Implements [Go::GeomObject](#).

29.52.3.17 `virtual bool Go::Circle::isAxisRotational ( Point & centre, Point & axis, Point & vec, double & angle ) [virtual]`

Check if the curve is axis rotational. Only true if a connection to an axis rotational elementary curve exist The axis and rotational angle is only specified if the curve is actually rotational

Reimplemented from [Go::ParamCurve](#).

29.52.3.18 `bool Go::Circle::isClosed ( ) const`

29.52.3.19 `virtual bool Go::Circle::isDegenerate ( double degenerate_epsilon ) [virtual]`

Query whether the curve is degenerate (collapsed into a single point).

#### Parameters

|                           |                                                                                                                                                   |
|---------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>degenerate_epsilon</i> | the tolerance used in determine whether the curve is degenerate. A curve is considered degenerate if its total length is shorter than this value. |
|---------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------|

#### Returns

`true` if the curve is degenerate, `false` otherwise.

Implements [Go::ParamCurve](#).

29.52.3.20 `virtual bool Go::Circle::isInPlane ( const Point & loc, const Point & axis, double eps, Point & normal ) const [virtual]`

Check if the line lies in a plane passing through a given axis.

Reimplemented from [Go::ParamCurve](#).

29.52.3.21 `virtual bool Go::Circle::isInPlane ( const Point & norm, double eps, Point & pos ) const [virtual]`

Check if the circle lies in a plane with a given normal.

Reimplemented from [Go::ParamCurve](#).

29.52.3.22 virtual double Go::Circle::length ( double *tol* ) [virtual]

Compute the total length of this curve up to some tolerance

## Parameters

|            |                                                                                                                                 |
|------------|---------------------------------------------------------------------------------------------------------------------------------|
| <i>tol</i> | the relative tolerance when approximating the length, i.e. stop iteration when error becomes smaller than $tol/(curve\ length)$ |
|------------|---------------------------------------------------------------------------------------------------------------------------------|

## Returns

the length calculated

Implements [Go::ParamCurve](#).

**29.52.3.23** `virtual void Go::Circle::point ( Point & pt, double tpar ) const [virtual]`

Evaluate the curve's position at a given parameter

## Parameters

|             |                                                                      |
|-------------|----------------------------------------------------------------------|
| <i>pt</i>   | the evaluated position will be written to this <a href="#">Point</a> |
| <i>tpar</i> | the parameter for which we wish to evaluate the curve                |

Implements [Go::ParamCurve](#).

**29.52.3.24** `virtual void Go::Circle::point ( std::vector< Point > & pts, double tpar, int derivs, bool from_right = true ) const [virtual]`

Evaluate the curve's position and a certain number of derivatives at a given parameter.

## Parameters

|                   |                                                                                                                                                                                                                                                                                       |
|-------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>pts</i>        | the evaluated position and derivatives (tangent, curvature vector, etc.) will be written to this vector. The first entry will be the position, the second entry will be the first derivative, etc. The size of this vector must be set to 'derivs'+ 1 prior to calling this function. |
| <i>tpar</i>       | the parameter for which we want to evaluate the curve                                                                                                                                                                                                                                 |
| <i>derivs</i>     | the number of derivatives we want to have calculated                                                                                                                                                                                                                                  |
| <i>from_right</i> | specify whether we should calculate derivatives 'from the right' or 'from the left' (default is from the right). This matters only when the curve presents discontinuities in its derivatives.                                                                                        |

Implements [Go::ParamCurve](#).

**29.52.3.25** `virtual void Go::Circle::read ( std::istream & is ) [virtual]`

Read object from stream

## Parameters

|           |                                  |
|-----------|----------------------------------|
| <i>is</i> | stream from which object is read |
|-----------|----------------------------------|

Implements [Go::Streamable](#).

**29.52.3.26** virtual void `Go::Circle::setParamBounds ( double startpar, double endpar )` [virtual]

Set bounds for the parametrization of the [Circle](#) other than the default  $[0, 2\pi]$ . Requirements for a valid parametrization are: 1) The first parameter must be strictly less than the second, 2) Parameter values must be in  $[-2\pi, 2\pi]$ , and 3) ( $endpar - startpar$ ) must not exceed  $2\pi$ .

Parameters

|                 |                 |
|-----------------|-----------------|
| <i>startpar</i> | start parameter |
| <i>endpar</i>   | end parameter   |

Implements [Go::ElementaryCurve](#).

**29.52.3.27** virtual void `Go::Circle::setParameterInterval ( double t1, double t2 )` [virtual]

[Circle](#) parametrized on  $[0, 2 * M\_PI]$ . Allowing class to be defined on a section of the circle.

Implements [Go::ParamCurve](#).

**29.52.3.28** void `Go::Circle::setSpanningVectors ( )` [protected]

**29.52.3.29** virtual double `Go::Circle::startparam ( )` const [virtual]

Query the start parameter of the curve

Returns

the curve's start parameter

Implements [Go::ParamCurve](#).

**29.52.3.30** virtual `Circle*` `Go::Circle::subCurve ( double from_par, double to_par, double fuzzy = DEFAULT_PARAMETER_EPSILON )` const [virtual]

Returns a curve which is a part of this curve. The result is NEWed, so the user is responsible for deleting it. NB: It is not guaranteed that the [ParamCurve](#) that is returned is of the same type as the curve itself. Thus, the returned curve might be a [SplineCurve](#).

Parameters

|                 |                                                                                                                                                                                                                |
|-----------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>from_par</i> | start value of parameter interval that will define the subcurve                                                                                                                                                |
| <i>to_par</i>   | end value of parameter interval that will define the subcurve                                                                                                                                                  |
| <i>fuzzy</i>    | since <code>subCurve</code> works on those curves who are spline-based, this tolerance defines how close the start and end parameter must be to an existing knot in order to be considered <i>on</i> the knot. |

**Returns**

a pointer to a new subcurve which represents the part of the curve between 'from\_par' and 'to\_par'. It will be spline-based and have a k-regular knotvector. The user is responsible for deleting this subcurve when it is no longer needed.

Implements [Go::ElementaryCurve](#).

**29.52.3.31** virtual void `Go::Circle::swapParameters2D ( )` [virtual]

If the curve is 2 dimensional, x and y coordinates will be swapped. Used when curve is a parameter curve.

Implements [Go::ElementaryCurve](#).

**29.52.3.32** virtual void `Go::Circle::translateCurve ( const Point & dir )` [virtual]

Implements [Go::ElementaryCurve](#).

**29.52.3.33** virtual void `Go::Circle::write ( std::ostream & os ) const` [virtual]

Write object to stream

**Parameters**

|                 |                                   |
|-----------------|-----------------------------------|
| <code>os</code> | stream to which object is written |
|-----------------|-----------------------------------|

Implements [Go::Streamable](#).

**29.52.4 Member Data Documentation**

**29.52.4.1** Point `Go::Circle::centre_` [protected]

Definition at line 205 of file Circle.h.

**29.52.4.2** double `Go::Circle::endparam_` [protected]

Definition at line 211 of file Circle.h.

**29.52.4.3** Point `Go::Circle::normal_` [protected]

Definition at line 206 of file Circle.h.

**29.52.4.4** double `Go::Circle::radius_` [protected]

Definition at line 204 of file Circle.h.

#### 29.52.4.5 `double Go::Circle::startparam_` [protected]

Definition at line 210 of file Circle.h.

#### 29.52.4.6 `Point Go::Circle::vec1_` [protected]

Definition at line 207 of file Circle.h.

#### 29.52.4.7 `Point Go::Circle::vec2_` [protected]

Definition at line 208 of file Circle.h.

The documentation for this class was generated from the following file:

- [gotools-core/include/GoTools/geometry/Circle.h](#)

## 29.53 `Go::ClosestPointCalculator` Class Reference

```
#include <IntersectionPoolUtils.h>
```

### Public Member Functions

- [ClosestPointCalculator](#) (shared\_ptr< [ParamObjectInt](#) > obj)
- [double compute](#) (const [Point](#) &pt, double \*unknown\_par, double eps)

#### 29.53.1 Detailed Description

Definition at line 262 of file IntersectionPoolUtils.h.

#### 29.53.2 Constructor & Destructor Documentation

##### 29.53.2.1 `Go::ClosestPointCalculator::ClosestPointCalculator ( shared_ptr< ParamObjectInt > obj )` [inline]

Definition at line 266 of file IntersectionPoolUtils.h.

#### 29.53.3 Member Function Documentation

##### 29.53.3.1 `double Go::ClosestPointCalculator::compute ( const Point & pt, double * unknown_par, double eps )` [inline]

Definition at line 283 of file IntersectionPoolUtils.h.

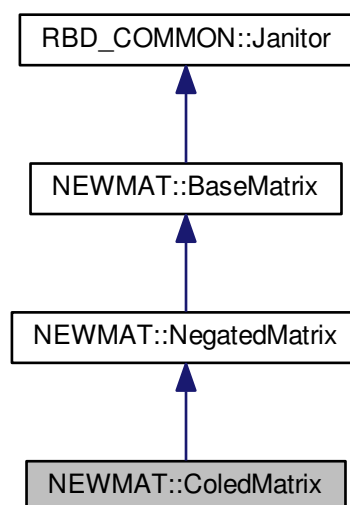
The documentation for this class was generated from the following file:

- [intersections/include/GoTools/intersections/IntersectionPoolUtils.h](#)

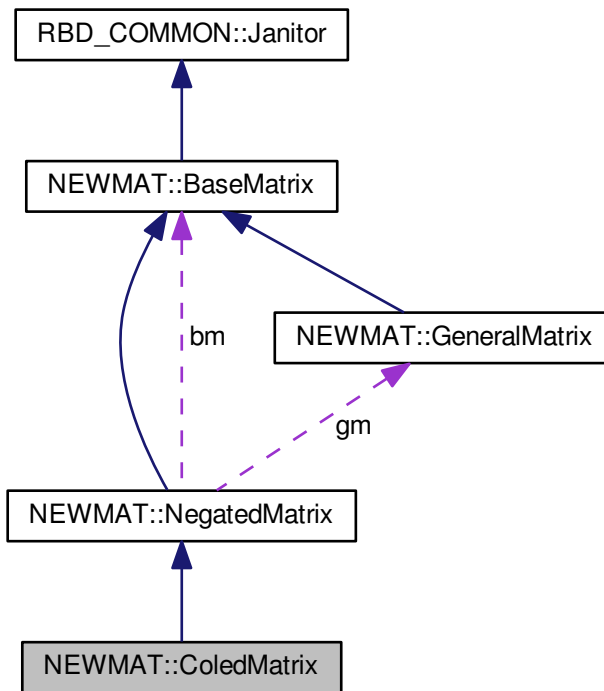
## 29.54 NEWMAT::ColedMatrix Class Reference

```
#include <newmat.h>
```

Inheritance diagram for NEWMAT::ColedMatrix:



Collaboration diagram for NEWMAT::ColedMatrix:



### Public Member Functions

- [~ColedMatrix \(\)](#)
- [GeneralMatrix \\* Evaluate \(MatrixType mt=MatrixTypeUnSp\)](#)
- [MatrixBandWidth BandWidth \(\) const](#)

### Friends

- class [BaseMatrix](#)

### Additional Inherited Members

#### 29.54.1 Detailed Description

Definition at line 1431 of file newmat.h.

#### 29.54.2 Constructor & Destructor Documentation

##### 29.54.2.1 NEWMAT::ColedMatrix::~~ColedMatrix ( ) [inline]

Definition at line 1436 of file newmat.h.



### 29.54.3 Member Function Documentation

29.54.3.1 **MatrixBandWidth** ColedMatrix::BandWidth ( ) const [virtual]

Reimplemented from [NEWMAT::NegatedMatrix](#).

Definition at line 482 of file newmat4.cpp.

29.54.3.2 **GeneralMatrix \* ColedMatrix::Evaluate ( MatrixType mt = MatrixTypeUnSp )** [virtual]

Reimplemented from [NEWMAT::NegatedMatrix](#).

Definition at line 302 of file newmat5.cpp.

### 29.54.4 Friends And Related Function Documentation

29.54.4.1 **friend class BaseMatrix** [friend]

Definition at line 1434 of file newmat.h.

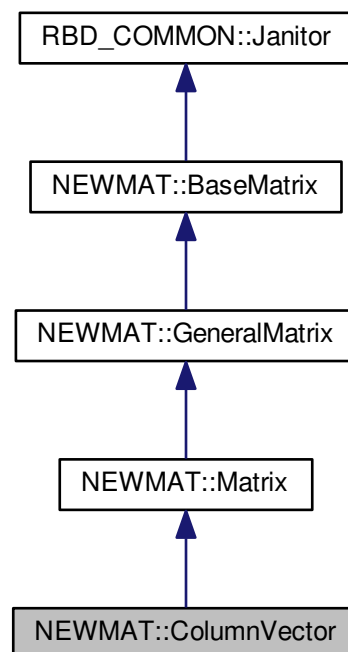
The documentation for this class was generated from the following files:

- [newmat/include/newmat.h](#)
- [newmat/src/newmat4.cpp](#)
- [newmat/src/newmat5.cpp](#)

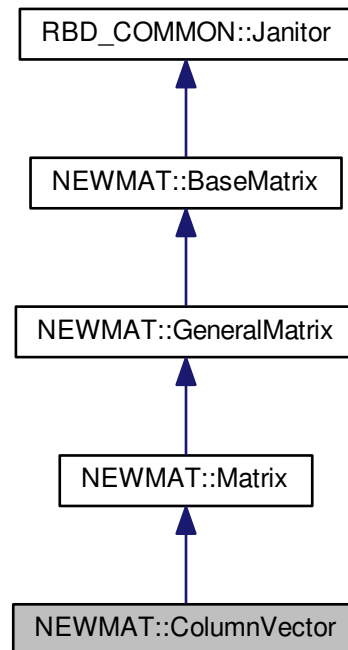
## 29.55 NEWMAT::ColumnVector Class Reference

```
#include <newmat.h>
```

Inheritance diagram for NEWMAT::ColumnVector:



Collaboration diagram for NEWMAT::ColumnVector:



## Public Member Functions

- [ColumnVector](#) ()
- [~ColumnVector](#) ()
- [ColumnVector](#) (ArrayLengthSpecifier n)
- [ColumnVector](#) (const BaseMatrix &)
- [ColumnVector](#) (const ColumnVector &gm)
- void [operator=](#) (const BaseMatrix &)
- void [operator=](#) (Real f)
- void [operator=](#) (const ColumnVector &m)
- [Real & operator\(\)](#) (int)
- [Real & element](#) (int)
- [Real operator\(\)](#) (int) const
- [Real element](#) (int) const
- [MatrixType Type](#) () const
- [GeneralMatrix \\* Transpose](#) (TransposedMatrix \*, MatrixType)
- void [ReSize](#) (int)
- void [ReSize](#) (int, int)
- void [ReSize](#) (const GeneralMatrix &A)
- [Real \\* nric](#) () const
- void [CleanUp](#) ()

## Additional Inherited Members

### 29.55.1 Detailed Description

Definition at line 830 of file newmat.h.

### 29.55.2 Constructor & Destructor Documentation

29.55.2.1 `NEWMAT::ColumnVector::ColumnVector ( )` [`inline`]

Definition at line 834 of file newmat.h.

29.55.2.2 `NEWMAT::ColumnVector::~~ColumnVector ( )` [`inline`]

Definition at line 835 of file newmat.h.

29.55.2.3 `NEWMAT::ColumnVector::ColumnVector ( ArrayLengthSpecifier n )` [`inline`]

Definition at line 836 of file newmat.h.

29.55.2.4 `NEWMAT::ColumnVector::ColumnVector ( const BaseMatrix & )`

29.55.2.5 `NEWMAT::ColumnVector::ColumnVector ( const ColumnVector & gm )` [`inline`]

Definition at line 838 of file newmat.h.

### 29.55.3 Member Function Documentation

29.55.3.1 `void ColumnVector::CleanUp ( )` [`virtual`]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 835 of file newmat4.cpp.

29.55.3.2 `Real& NEWMAT::ColumnVector::element ( int )`

29.55.3.3 `Real NEWMAT::ColumnVector::element ( int ) const`

29.55.3.4 `Real* NEWMAT::ColumnVector::nric ( ) const` [`inline`]

Definition at line 855 of file newmat.h.

29.55.3.5 `Real& NEWMAT::ColumnVector::operator()( int )`

29.55.3.6 `Real NEWMAT::ColumnVector::operator()( int ) const`

29.55.3.7 `void NEWMAT::ColumnVector::operator= ( const BaseMatrix & )`

29.55.3.8 `void NEWMAT::ColumnVector::operator= ( Real f ) [inline]`

Definition at line 840 of file newmat.h.

29.55.3.9 `void NEWMAT::ColumnVector::operator= ( const ColumnVector & m ) [inline]`

Definition at line 841 of file newmat.h.

29.55.3.10 `void NEWMAT::ColumnVector::ReSize ( int )`

29.55.3.11 `void NEWMAT::ColumnVector::ReSize ( int , int ) [virtual]`

Reimplemented from [NEWMAT::Matrix](#).

29.55.3.12 `void ColumnVector::ReSize ( const GeneralMatrix & A ) [virtual]`

Reimplemented from [NEWMAT::Matrix](#).

Definition at line 273 of file newmat4.cpp.

29.55.3.13 `GeneralMatrix * ColumnVector::Transpose ( TransposedMatrix *, MatrixType mt ) [virtual]`

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 60 of file newmat5.cpp.

29.55.3.14 `MatrixType ColumnVector::Type ( ) const [virtual]`

Reimplemented from [NEWMAT::Matrix](#).

Definition at line 391 of file newmat4.cpp.

The documentation for this class was generated from the following files:

- [newmat/include/newmat.h](#)
- [newmat/src/newmat4.cpp](#)
- [newmat/src/newmat5.cpp](#)

## 29.56 Go::CompleteEdgeNet Class Reference

Complete the edge net of a [SurfaceModel](#). Fetches the wire frame model corresponding to a surface model, and extends it such that the extended wire frame will become the wire frame model corresponding to a volume model have the given surface model as its outer boundary. The extension to the wireframe is represented as pairs of vertices where these vertices lie at the endpoints of the missing edges. NB! This solution is currently not expected to handle all configurations.

```
#include <CompleteEdgeNet.h>
```

### Public Member Functions

- [CompleteEdgeNet](#) (shared\_ptr< [SurfaceModel](#) > sfmodel, bool perform\_step2, bool smooth\_connections)
- [~CompleteEdgeNet](#) ()
  - Destructor.*
- bool perform (std::vector< std::pair< [Point](#), [Point](#) > > &corr\_vx\_pts)
- shared\_ptr< [SurfaceModel](#) > [getRegularizedModel](#) ()
- std::vector< std::pair< shared\_ptr< [Vertex](#) >, shared\_ptr< [Vertex](#) > > > [getMissingEdges](#) ()

#### 29.56.1 Detailed Description

Complete the edge net of a [SurfaceModel](#). Fetches the wire frame model corresponding to a surface model, and extends it such that the extended wire frame will become the wire frame model corresponding to a volume model have the given surface model as its outer boundary. The extension to the wireframe is represented as pairs of vertices where these vertices lie at the endpoints of the missing edges. NB! This solution is currently not expected to handle all configurations.

Definition at line 55 of file CompleteEdgeNet.h.

#### 29.56.2 Constructor & Destructor Documentation

**29.56.2.1** `Go::CompleteEdgeNet::CompleteEdgeNet ( shared_ptr< SurfaceModel > sfmodel, bool perform_step2, bool smooth_connections )`

Constructor. The method is applied on a [SurfaceModel](#)

##### Parameters

|                |                                           |
|----------------|-------------------------------------------|
| <i>sfmodel</i> | Pointer to a <a href="#">SurfaceModel</a> |
|----------------|-------------------------------------------|

**29.56.2.2** `Go::CompleteEdgeNet::~~CompleteEdgeNet ( )`

Destructor.

#### 29.56.3 Member Function Documentation

29.56.3.1 `std::vector<std::pair<shared_ptr<Vertex>, shared_ptr<Vertex> > >` `Go::CompleteEdgeNet::getMissingEdges ( )` `[inline]`

Fetch new edges represented by their end vertices

#### Returns

Vector of pointers to end vertices of the edges

Definition at line 81 of file `CompleteEdgeNet.h`.

29.56.3.2 `shared_ptr<SurfaceModel>` `Go::CompleteEdgeNet::getRegularizedModel ( )` `[inline]`

Fetch modified model (after regularization)

#### Returns

Pointer to the modified model

Definition at line 73 of file `CompleteEdgeNet.h`.

29.56.3.3 `bool` `Go::CompleteEdgeNet::perform ( std::vector< std::pair< Point, Point > > & corr_vx_pts )`

Apply algorithm for completing the edge net and create a starting ground for a block structured model of solids

#### Returns

Whether the edge net was completed or not.

The documentation for this class was generated from the following file:

- [compositemodel/include/GoTools/compositemodel/CompleteEdgeNet.h](#)

## 29.57 Go::ComplexityInfo Class Reference

```
#include <ComplexityInfo.h>
```

### Public Member Functions

- [ComplexityInfo \( \)](#)  
*Default constructor.*
- [~ComplexityInfo \( \)](#)  
*Destructor.*
- void [setConeOverlap \(double overlap\)](#)
- [double getConeOverlap \( \)](#)
- void [setBoxOverlap \(double overlap\)](#)
- [double getBoxOverlap \( \)](#)
- void [setNmbIntpts \(int nmbpts\)](#)
- [int getNmbIntpts \( \)](#)
- void [setNmbSingpts \(int nmbpts\)](#)
- [int getNmbSingpts \( \)](#)
- void [setComplex \( \)](#)  
*Set flag for complex case.*
- [bool isComplex \( \)](#)

### 29.57.1 Detailed Description

This class contains statistical information used to check whether an intersection problem is simplified during recursive subdivision

Definition at line 50 of file ComplexityInfo.h.

### 29.57.2 Constructor & Destructor Documentation

#### 29.57.2.1 `Go::ComplexityInfo::ComplexityInfo ( )` `[inline]`

Default constructor.

Definition at line 53 of file ComplexityInfo.h.

#### 29.57.2.2 `Go::ComplexityInfo::~~ComplexityInfo ( )` `[inline]`

Destructor.

Definition at line 61 of file ComplexityInfo.h.

### 29.57.3 Member Function Documentation

#### 29.57.3.1 `double Go::ComplexityInfo::getBoxOverlap ( )` `[inline]`

Get the overlap between two boxes

##### Returns

the box overlap

Definition at line 80 of file ComplexityInfo.h.

#### 29.57.3.2 `double Go::ComplexityInfo::getConeOverlap ( )` `[inline]`

Get the overlapping angle between two cones

##### Returns

the overlapping angle

Definition at line 70 of file ComplexityInfo.h.



29.57.3.3 `int Go::ComplexityInfo::getNmbIntpts ( ) [inline]`

Get the number of intersection points between two objects

**Returns**

the number of intersection points

Definition at line 90 of file ComplexityInfo.h.

29.57.3.4 `int Go::ComplexityInfo::getNmbSingpts ( ) [inline]`

Get the number of singular points

**Returns**

the number of singular points

Definition at line 100 of file ComplexityInfo.h.

29.57.3.5 `bool Go::ComplexityInfo::isComplex ( ) [inline]`

Get flag for complex case

**Returns**

`true` if complex, `false` otherwise

Definition at line 109 of file ComplexityInfo.h.

29.57.3.6 `void Go::ComplexityInfo::setBoxOverlap ( double overlap ) [inline]`

Set the overlap between two boxes

**Parameters**

|                |                 |
|----------------|-----------------|
| <i>overlap</i> | the box overlap |
|----------------|-----------------|

Definition at line 75 of file ComplexityInfo.h.

29.57.3.7 `void Go::ComplexityInfo::setComplex ( ) [inline]`

Set flag for complex case.

Definition at line 104 of file ComplexityInfo.h.

29.57.3.8 void Go::ComplexityInfo::setConeOverlap ( double *overlap* ) [inline]

Set the overlapping angle between two cones

Parameters

|                |                       |
|----------------|-----------------------|
| <i>overlap</i> | the overlapping angle |
|----------------|-----------------------|

Definition at line 65 of file ComplexityInfo.h.

29.57.3.9 void Go::ComplexityInfo::setNmbIntpts ( int *nmbpts* ) [inline]

Set the number of intersection points between two objects

Parameters

|               |                                   |
|---------------|-----------------------------------|
| <i>nmbpts</i> | the number of intersection points |
|---------------|-----------------------------------|

Definition at line 85 of file ComplexityInfo.h.

29.57.3.10 void Go::ComplexityInfo::setNmbSingpts ( int *nmbpts* ) [inline]

Set the number of singular points

Parameters

|               |                               |
|---------------|-------------------------------|
| <i>nmbpts</i> | the number of singular points |
|---------------|-------------------------------|

Definition at line 95 of file ComplexityInfo.h.

The documentation for this class was generated from the following file:

- intersections/include/GoTools/intersections/[ComplexityInfo.h](#)

## 29.58 Go::CompositeBox Class Reference

```
#include <CompositeBox.h>
```

### Public Member Functions

- `template<typename RandomAccessIterator >`  
[CompositeBox](#) (RandomAccessIterator start, int *dim*, int num\_u, int num\_v)  
*Constructor using setFromArray()*
- [CompositeBox](#) (const Point &low, const Point &high)

- `~CompositeBox ()`  
*Do not inherit from this class – nonvirtual destructor.*
- `void setFromPoints (const Point &low, const Point &high)`
- `template<typename RandomAccessIterator >`  
`void setFromArray (RandomAccessIterator start, int dim, int num_u, int num_v)`
- `void read (std::istream &is)`  
*Read a composite box from a standard istream.*
- `void write (std::ostream &os) const`  
*Write a composite box to a standard ostream.*
- `int dimension () const`  
*The dimension of the composite box.*
- `Point low (double toli=0.0, double tole=0.0) const`  
*The lower bound of the composite box.*
- `Point high (double toli=0.0, double tole=0.0) const`  
*The upper bound of the composite box.*
- `const BoundingBox & inner () const`  
*The inner box.*
- `const BoundingBox & edge () const`  
*The edge box.*
- `bool containsPoint (const Point &pt, double toli=0.0, double tole=0.0) const`
- `bool overlaps (const CompositeBox &box, double toli=0.0, double tole=0.0) const`
- `bool getOverlap (const CompositeBox &box, double &overlap, double toli=0.0, double tole=0.0) const`
- `bool containsBox (const CompositeBox &box, double toli=0.0, double tole=0.0) const`

### 29.58.1 Detailed Description

Composite of two bounding boxes. The inner box is supposed to be a bounding box for the interior of an object, while the edge box bounds the outer edges. When requests are made, the `CompositeBox` is treated as a single axis-aligned box, but tolerances can be given for both sub-boxes separately. A `CompositeBox` object can have any number of dimensions.

Definition at line 57 of file `CompositeBox.h`.

### 29.58.2 Constructor & Destructor Documentation

**29.58.2.1** `template<typename RandomAccessIterator > Go::CompositeBox::CompositeBox ( RandomAccessIterator start, int dim, int num_u, int num_v ) [inline]`

Constructor using `setFromArray()`

Definition at line 62 of file `CompositeBox.h`.

**29.58.2.2** `Go::CompositeBox::CompositeBox ( const Point & low, const Point & high ) [inline]`

Creates a `CompositeBox` with the `Point` `low` specifying the lower bound in all dimensions and `high` specifying the upper bound. The inner and edge boxes are equal.

Definition at line 73 of file `CompositeBox.h`.

### 29.58.2.3 Go::CompositeBox::~~CompositeBox ( )

Do not inherit from this class – nonvirtual destructor.

## 29.58.3 Member Function Documentation

### 29.58.3.1 bool Go::CompositeBox::containsBox ( const CompositeBox & *box*, double *toli* = 0.0, double *tole* = 0.0 ) const

Returns true if this box contain the box passed as a parameter, up to tolerances. Tolerances may be specified separately for inner and edge boxes.

### 29.58.3.2 bool Go::CompositeBox::containsPoint ( const Point & *pt*, double *toli* = 0.0, double *tole* = 0.0 ) const

Returns true if the point *pt* is inside the box, up to tolerances. Tolerances may be specified separately for inner and edge boxes.

### 29.58.3.3 int Go::CompositeBox::dimension ( ) const [inline]

The dimension of the composite box.

Definition at line 180 of file CompositeBox.h.

### 29.58.3.4 const BoundingBox& Go::CompositeBox::edge ( ) const [inline]

The edge box.

Definition at line 201 of file CompositeBox.h.

### 29.58.3.5 bool Go::CompositeBox::getOverlap ( const CompositeBox & *box*, double & *overlap*, double *toli* = 0.0, double *tole* = 0.0 ) const

Get the overlap between the two boxes. Tolerances may be specified separately for inner and edge boxes.

### 29.58.3.6 Point Go::CompositeBox::high ( double *toli* = 0.0, double *tole* = 0.0 ) const

The upper bound of the composite box.

### 29.58.3.7 const BoundingBox& Go::CompositeBox::inner ( ) const [inline]

The inner box.

Definition at line 196 of file CompositeBox.h.

29.58.3.8 **Point** Go::CompositeBox::low ( **double** *tol* = 0.0, **double** *tole* = 0.0 ) **const**

The lower bound of the composite box.

29.58.3.9 **bool** Go::CompositeBox::overlaps ( **const** CompositeBox & *box*, **double** *tol* = 0.0, **double** *tole* = 0.0 ) **const**

Returns true if the two boxes overlap, up to tolerances. Tolerances may be specified separately for inner and edge boxes.

29.58.3.10 **void** Go::CompositeBox::read ( **std::istream** & *is* )

Read a composite box from a standard istream.

29.58.3.11 **template**<typename RandomAccessIterator > **void** Go::CompositeBox::setFromArray ( RandomAccessIterator *start*, **int** *dim*, **int** *num\_u*, **int** *num\_v* ) **[inline]**

Given an array of dim-dimensional points stored as doubles or floats, makes the smallest composite box containing all points in the array. The array must be like a control point grid, with num\_u points in the fastest running direction, and num\_v points in the other direction. For curves, use num\_v = 1.

Definition at line 96 of file CompositeBox.h.

29.58.3.12 **void** Go::CompositeBox::setFromPoints ( **const** Point & *low*, **const** Point & *high* ) **[inline]**

Makes the bounding box have lower bounds as specified in low and upper bounds as specified in high.

Definition at line 82 of file CompositeBox.h.

29.58.3.13 **void** Go::CompositeBox::write ( **std::ostream** & *os* ) **const**

Write a composite box to a standard ostream.

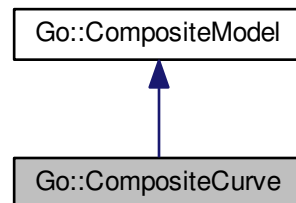
The documentation for this class was generated from the following file:

- [gotools-core/include/GoTools/Utils/CompositeBox.h](#)

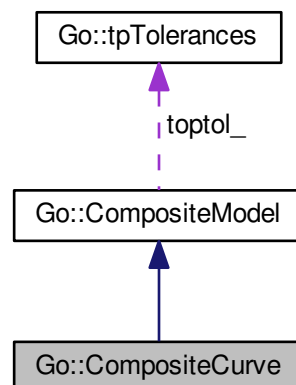
## 29.59 Go::CompositeCurve Class Reference

```
#include <CompositeCurve.h>
```

Inheritance diagram for Go::CompositeCurve:



Collaboration diagram for Go::CompositeCurve:



### Public Member Functions

- [CompositeCurve](#) ([double](#) gap, [double](#) neighbour, [double](#) kink, [double](#) bend, [std::vector](#)< [shared\\_ptr](#)< [ParamCurve](#) > > &curves)
  - [~CompositeCurve](#) ()
- Destructor.*
- [virtual CompositeCurve \\* clone](#) () [const](#)
  - [virtual int nmbEntities](#) () [const](#)
  - [shared\\_ptr](#)< [ParamCurve](#) > [getCurve](#) (int idx) [const](#)
  - [int getIndex](#) ([ParamCurve \\*curve](#)) [const](#)
  - [void parameterRange](#) ([double](#) &start, [double](#) &end) [const](#)

- `double getGlobalPar (int idx, double par) const`
- `void getLocalPar (double global_par, int &idx, double &local_par) const`
- `virtual void evaluate (int idx, double par[], Point &pnt) const`
- `virtual void evaluate (int idx, double par[], int nder, std::vector< Point > &der) const`
- `void evaluateCurve (double par, Point &pnt) const`
- `void evaluateCurve (double par, int nder, std::vector< Point > &der) const`
- `virtual void closestPoint (Point &pnt, Point &clo_pnt, int &idx, double clo_par[], double &dist)`
- `PointOnCurve closestPoint (Point &pnt)`
- `virtual void extremalPoint (Point &dir, Point &ext_pnt, int &idx, double ext_par[])`
- `virtual BoundingBox boundingBox ()`
- `virtual BoundingBox boundingBox (int idx) const`
- `virtual bool isDegenerate (int idx) const`
- `virtual double curvature (int idx, double *par) const`
- `virtual void turn (int idx)`
- `virtual void turn ()`
- `virtual shared_ptr< IntResultsModel > intersect (const ftLine &line)`
- `virtual shared_ptr< IntResultsModel > intersect_plane (const ftPlane &plane)`
- `bool hit (const Point &point, const Point &dir, PointOnCurve &result)`
- `void append (shared_ptr< ParamCurve > curve)`
- `virtual void tessellate (std::vector< shared_ptr< GeneralMesh > > &meshes) const`
- `virtual void tessellate (int resolution[], std::vector< shared_ptr< GeneralMesh > > &meshes) const`
- `void tessellate (const std::vector< shared_ptr< ParamCurve > > &curves, int resolution[], std::vector< shared_ptr< GeneralMesh > > &meshes) const`
- `virtual void tessellate (double density, std::vector< shared_ptr< GeneralMesh > > &meshes) const`
- `void tessellate (const std::vector< shared_ptr< ParamCurve > > &curves, double density, std::vector< shared_ptr< GeneralMesh > > &meshes) const`
- `virtual void tessellatedCtrPolygon (std::vector< shared_ptr< LineCloud > > &ctr_pol) const`
- `void tessellatedCtrPolygon (const std::vector< shared_ptr< ParamCurve > > &curves, std::vector< shared_ptr< LineCloud > > &ctr_pol) const`

## Additional Inherited Members

### 29.59.1 Detailed Description

A composite curve including topological information

Definition at line 66 of file CompositeCurve.h.

### 29.59.2 Constructor & Destructor Documentation

**29.59.2.1** `Go::CompositeCurve::CompositeCurve ( double gap, double neighbour, double kink, double bend, std::vector< shared_ptr< ParamCurve > > &curves )`

Constructor. The sequence of the input curves will be kept, but a permutation array will sort the curves according to continuity

#### Parameters

|                  |                                                                                                                 |
|------------------|-----------------------------------------------------------------------------------------------------------------|
| <i>gap</i>       | If the distance between two points is less than 'gap' they are viewed as identical.                             |
| <i>neighbour</i> | Maximum distance between curves viewed as adjacent.                                                             |
| <i>kink</i>      | If two adjacent curves meet with an angle less than 'kink', they are seen as G1 continuous. (angles in radians) |
| <i>bend</i>      | If two curves meet with an angle larger than 'bend', there is an intentional corner. (angles in radians)        |
| <i>curves</i>    | A vector of curves.                                                                                             |

### 29.59.2.2 `Go::CompositeCurve::~~CompositeCurve ( )`

Destructor.

## 29.59.3 Member Function Documentation

### 29.59.3.1 `void Go::CompositeCurve::append ( shared_ptr< ParamCurve > curve )`

Append a new curve to the composite curve. The curve is included in the topological structure.

#### Parameters

|              |                |
|--------------|----------------|
| <i>curve</i> | The new curve. |
|--------------|----------------|

### 29.59.3.2 `virtual BoundingBox Go::CompositeCurve::boundingBox ( ) [virtual]`

Bounding box of the entire composite curve

#### Returns

Bounding box

Implements [Go::CompositeModel](#).

### 29.59.3.3 `virtual BoundingBox Go::CompositeCurve::boundingBox ( int idx ) const [virtual]`

Bounding box corresponding to one curve

#### Parameters

|            |                |
|------------|----------------|
| <i>idx</i> | Index of curve |
|------------|----------------|

#### Returns

Bounding box of curve with index *idx*.

Implements [Go::CompositeModel](#).

### 29.59.3.4 `virtual CompositeCurve* Go::CompositeCurve::clone ( ) const [virtual]`

Make a copy of the current model

#### Returns

Pointer to the composite curve

Reimplemented from [Go::CompositeModel](#).



29.59.3.5 `virtual void Go::CompositeCurve::closestPoint ( Point & pnt, Point & clo_pnt, int & idx, double clo_par[], double & dist )` [virtual]

Closest point between a given point and this composite curve Returns one point

#### Parameters

|            |             |
|------------|-------------|
| <i>pnt</i> | Input point |
|------------|-------------|

#### Return values

|                   |                                                      |
|-------------------|------------------------------------------------------|
| <i>clo_pnt</i>    | Found closest point                                  |
| <i>idx</i>        | Index of curve where the closest point is found      |
| <i>clo_par</i> [] | Parameter value corresponding to the closest point   |
| <i>dist</i>       | Distance between input point and found closest point |

Implements [Go::CompositeModel](#).

29.59.3.6 `PointOnCurve Go::CompositeCurve::closestPoint ( Point & pnt )`

Closest point between a given point and this composite curve Returns one point

#### Parameters

|            |             |
|------------|-------------|
| <i>pnt</i> | Input point |
|------------|-------------|

#### Returns

Closest point

29.59.3.7 `virtual double Go::CompositeCurve::curvature ( int idx, double * par ) const` [virtual]

[Curvature](#) of a curve

#### Parameters

|            |                                               |
|------------|-----------------------------------------------|
| <i>idx</i> | Index of curve                                |
| <i>par</i> | Parameter value at which to compute curvature |

#### Returns

The curvature.

Implements [Go::CompositeModel](#).

29.59.3.8 virtual void Go::CompositeCurve::evaluate ( int *idx*, double *par*[], Point & *pnt* ) const [virtual]

Evaluate position

## Parameters

|              |                 |
|--------------|-----------------|
| <i>idx</i>   | Index of curve  |
| <i>par[]</i> | Parameter value |

## Return values

|            |        |
|------------|--------|
| <i>pnt</i> | Result |
|------------|--------|

Implements [Go::CompositeModel](#).

**29.59.3.9** `virtual void Go::CompositeCurve::evaluate ( int idx, double par[], int nder, std::vector< Point > & der ) const`  
 [virtual]

Evaluate position and a number of derivatives The sequence is position, first derivative, second derivative, etc.

## Parameters

|             |                                                   |
|-------------|---------------------------------------------------|
| <i>idx</i>  | Index of curve                                    |
| <i>par</i>  | Parameter value                                   |
| <i>nder</i> | Number of derivatives to compute, 0=only position |

## Return values

|            |        |
|------------|--------|
| <i>der</i> | Result |
|------------|--------|

Implements [Go::CompositeModel](#).

**29.59.3.10** `void Go::CompositeCurve::evaluateCurve ( double par, Point & pnt ) const`

Evaluate with respect to the composite curve. Evaluate position

## Parameters

|            |                 |
|------------|-----------------|
| <i>par</i> | Parameter value |
|------------|-----------------|

## Return values

|            |        |
|------------|--------|
| <i>pnt</i> | Result |
|------------|--------|

**29.59.3.11** `void Go::CompositeCurve::evaluateCurve ( double par, int nder, std::vector< Point > & der ) const`

Evaluate position and a number of derivatives The sequence is position, first derivative, second derivative, etc.

## Parameters

|             |                                                   |
|-------------|---------------------------------------------------|
| <i>par</i>  | Parameter value                                   |
| <i>nder</i> | Number of derivatives to compute, 0=only position |

## Return values

|            |        |
|------------|--------|
| <i>der</i> | Result |
|------------|--------|

29.59.3.12 `virtual void Go::CompositeCurve::extremalPoint ( Point & dir, Point & ext_pnt, int & idx, double ext_par[] )`  
[virtual]

Extremal point(s) in a given direction

## Parameters

|            |           |
|------------|-----------|
| <i>dir</i> | Direction |
|------------|-----------|

## Return values

|                  |                                                  |
|------------------|--------------------------------------------------|
| <i>ext_pnt</i>   | Found extremal point                             |
| <i>idx</i>       | Index of curve where the extremal point is found |
| <i>ext_par[]</i> | Parameter value of extremal point                |

Implements [Go::CompositeModel](#).

29.59.3.13 `shared_ptr<ParamCurve> Go::CompositeCurve::getCurve ( int idx ) const`

Return one curve. Note that the index corresponds to the sequence of which the curves are added to the [CompositeCurve](#), not the position in the composite curve

## Parameters

|            |                 |
|------------|-----------------|
| <i>idx</i> | Index of curve. |
|------------|-----------------|

## Returns

Pointer to the curve.

29.59.3.14 `double Go::CompositeCurve::getGlobalPar ( int idx, double par ) const`

Conversion between parameter settings

## Parameters

|            |                        |
|------------|------------------------|
| <i>idx</i> | Index of sub curve     |
| <i>par</i> | Parameter in sub curve |

## Returns

the global parameter corresponding to a sub curve parameter

**29.59.3.15** `int Go::CompositeCurve::getIndex ( ParamCurve * curve ) const`

Given a curve in the composite curve, return the index of this curve

## Parameters

|              |                  |
|--------------|------------------|
| <i>curve</i> | Pointer to curve |
|--------------|------------------|

## Returns

Index to curve

**29.59.3.16** `void Go::CompositeCurve::getLocalPar ( double global_par, int & idx, double & local_par ) const`

Return the sub curve parameter corresponding to a global parameter. At joints, the first sub curve parameter will be returned.

## Parameters

|                   |                                  |
|-------------------|----------------------------------|
| <i>global_par</i> | Parameter in composite curve     |
| <i>idx</i>        | Index of corresponding sub curve |

## Return values

|                  |                        |
|------------------|------------------------|
| <i>local_par</i> | Parameter in sub curve |
|------------------|------------------------|

**29.59.3.17** `bool Go::CompositeCurve::hit ( const Point & point, const Point & dir, PointOnCurve & result )`

Test if a line with direction 'dir' through the point 'point' hits this composite curve. If it hits, return 'true' and the intersection point closest to 'point'.

## Parameters

|              |                    |
|--------------|--------------------|
| <i>point</i> | Point on the line. |
| <i>dir</i>   | Line direction.    |

## Return values

|               |                             |
|---------------|-----------------------------|
| <i>result</i> | Closest intersection point. |
|---------------|-----------------------------|

## Returns

Whether the line hits or not.

29.59.3.18 `virtual shared_ptr<IntResultsModel> Go::CompositeCurve::intersect ( const ftLine & line ) [virtual]`

Intersection with a line. Expected output is points, probably one point. Curves can occur in special configurations. The function makes most sense in 2D, otherwise it is very dependent on the tolerance.

## Parameters

|             |           |
|-------------|-----------|
| <i>line</i> | The line. |
|-------------|-----------|

## Returns

Pointer to an [IntResultsModel](#).

Implements [Go::CompositeModel](#).

29.59.3.19 `virtual shared_ptr<IntResultsModel> Go::CompositeCurve::intersect_plane ( const ftPlane & plane ) [virtual]`

Intersection with a plane.

## Parameters

|              |            |
|--------------|------------|
| <i>plane</i> | The plane. |
|--------------|------------|

## Returns

Pointer to an [IntResultsModel](#).

Implements [Go::CompositeModel](#).

29.59.3.20 `virtual bool Go::CompositeCurve::isDegenerate ( int idx ) const [virtual]`

Whether one particular curve is degenerated

## Parameters

|            |                |
|------------|----------------|
| <i>idx</i> | Index of curve |
|------------|----------------|

**Returns**

Whether the curve is degenerated

Implements [Go::CompositeModel](#).

29.59.3.21 `virtual int Go::CompositeCurve::nmbEntities ( ) const [virtual]`

Number of simple entities

**Returns**

Number of simple entities

Implements [Go::CompositeModel](#).

29.59.3.22 `void Go::CompositeCurve::parameterRange ( double & start, double & end ) const`

Get the parameter range.

**Return values**

|              |                  |
|--------------|------------------|
| <i>start</i> | Parameter start. |
| <i>end</i>   | Parameter end.   |

29.59.3.23 `virtual void Go::CompositeCurve::tessellate ( std::vector< shared_ptr< GeneralMesh > > & meshes ) const [virtual]`

Tessellate with respect to a default parameter

**Return values**

|               |                   |
|---------------|-------------------|
| <i>meshes</i> | Tessellated model |
|---------------|-------------------|

Implements [Go::CompositeModel](#).

29.59.3.24 `virtual void Go::CompositeCurve::tessellate ( int resolution[], std::vector< shared_ptr< GeneralMesh > > & meshes ) const [virtual]`

Tessellate with respect to a given resolution

**Parameters**

|                     |                                                |
|---------------------|------------------------------------------------|
| <i>resolution[]</i> | All curves are tessellated with resolution[0]. |
|---------------------|------------------------------------------------|

## Return values

|               |                  |
|---------------|------------------|
| <i>meshes</i> | Tesselated model |
|---------------|------------------|

Implements [Go::CompositeModel](#).

29.59.3.25 `void Go::CompositeCurve::tessellate ( const std::vector< shared_ptr< ParamCurve > > & curves, int resolution[], std::vector< shared_ptr< GeneralMesh > > & meshes ) const`

Tesselate specified curves with respect to a given resolution

## Parameters

|                     |                                                      |
|---------------------|------------------------------------------------------|
| <i>curves</i>       | Specified curves                                     |
| <i>resolution[]</i> | Specified curves are tessellated with resolution[0]. |

## Return values

|               |                  |
|---------------|------------------|
| <i>meshes</i> | Tesselated model |
|---------------|------------------|

29.59.3.26 `virtual void Go::CompositeCurve::tessellate ( double density, std::vector< shared_ptr< GeneralMesh > > & meshes ) const [virtual]`

Tesselate with respect to a given tessellation density

## Parameters

|                |                      |
|----------------|----------------------|
| <i>density</i> | Tessellation density |
|----------------|----------------------|

## Return values

|               |                  |
|---------------|------------------|
| <i>meshes</i> | Tesselated model |
|---------------|------------------|

Implements [Go::CompositeModel](#).

29.59.3.27 `void Go::CompositeCurve::tessellate ( const std::vector< shared_ptr< ParamCurve > > & curves, double density, std::vector< shared_ptr< GeneralMesh > > & meshes ) const`

Tesselate specified curves with respect to a given tessellation density

## Parameters

|                |                      |
|----------------|----------------------|
| <i>curves</i>  | Specified curves     |
| <i>density</i> | Tessellation density |



## Return values

|               |                  |
|---------------|------------------|
| <i>meshes</i> | Tesselated model |
|---------------|------------------|

29.59.3.28 virtual void Go::CompositeCurve::tesselatedCtrPolygon ( std::vector< shared\_ptr< LineCloud > > & *ctr\_pol* ) const [virtual]

Return a tessellation of the control polygon of this composite curve

## Return values

|                |                                                    |
|----------------|----------------------------------------------------|
| <i>ctr_pol</i> | Tesselated control polygon of this composite curve |
|----------------|----------------------------------------------------|

Implements [Go::CompositeModel](#).

29.59.3.29 void Go::CompositeCurve::tesselatedCtrPolygon ( const std::vector< shared\_ptr< ParamCurve > > & *curves*, std::vector< shared\_ptr< LineCloud > > & *ctr\_pol* ) const

Return a tessellation of the control polygon of specified curves

## Parameters

|               |                  |
|---------------|------------------|
| <i>curves</i> | Specified curves |
|---------------|------------------|

## Return values

|                |                                               |
|----------------|-----------------------------------------------|
| <i>ctr_pol</i> | Tesselated control polygon of specified curve |
|----------------|-----------------------------------------------|

29.59.3.30 virtual void Go::CompositeCurve::turn ( int *idx* ) [virtual]

Turn parameter direction of one curve. An update in the topology structures is required.

## Parameters

|            |                |
|------------|----------------|
| <i>idx</i> | Index of curve |
|------------|----------------|

Implements [Go::CompositeModel](#).

29.59.3.31 virtual void Go::CompositeCurve::turn ( ) [virtual]

Turn parameter directions of all curves. An update in the topology structures is required.

Implements [Go::CompositeModel](#).

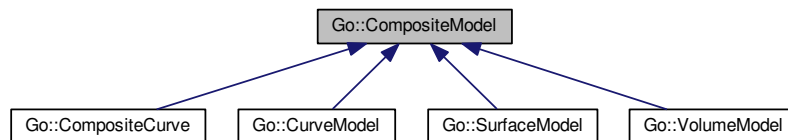
The documentation for this class was generated from the following file:

- compositemodel/include/GoTools/compositemodel/[CompositeCurve.h](#)

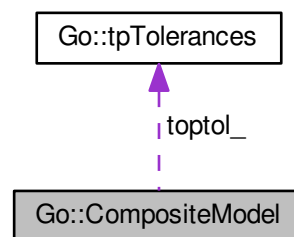
## 29.60 Go::CompositeModel Class Reference

```
#include <CompositeModel.h>
```

Inheritance diagram for Go::CompositeModel:



Collaboration diagram for Go::CompositeModel:



### Public Member Functions

- [CompositeModel](#) ([double](#) gap, [double](#) neighbour, [double](#) kink, [double](#) bend)
- virtual [~CompositeModel](#) ()  
*Destructor.*
- void [setTolerances](#) ([double](#) gap, [double](#) neighbour, [double](#) kink, [double](#) bend)
- [tpTolerances](#) [getTolerances](#) ()  
*Return topology tolerances.*
- virtual [SurfaceModel](#) \* [asSurfaceModel](#) ()
- virtual [CompositeModel](#) \* [clone](#) () const
- virtual int [nmbEntities](#) () const =0
- virtual void [evaluate](#) (int [idx](#), [double](#) par[], [Point](#) &pnt) const =0
- virtual void [evaluate](#) (int [idx](#), [double](#) par[], int [nder](#), std::vector< [Point](#) > &der) const =0
- virtual void [closestPoint](#) ([Point](#) &pnt, [Point](#) &clo\_pnt, int &idx, [double](#) clo\_par[], [double](#) &dist)=0
- virtual shared\_ptr< [IntResultsModel](#) > [intersect](#) (const [ftLine](#) &line)=0
- virtual shared\_ptr< [IntResultsModel](#) > [intersect\\_plane](#) (const [ftPlane](#) &plane)=0
- virtual void [extremalPoint](#) ([Point](#) &dir, [Point](#) &ext\_pnt, int &idx, [double](#) ext\_par[])=0
- virtual [BoundingBox](#) [boundingBox](#) ()=0
- virtual [BoundingBox](#) [boundingBox](#) (int [idx](#)) const =0

- virtual `bool isDegenerate (int idx) const =0`
- virtual `double curvature (int idx, double *par) const =0`
- virtual `void turn (int idx)=0`
- virtual `void turn ()=0`  
*Turn parameter directions of all entities.*
- virtual `void tessellate (std::vector< shared_ptr< GeneralMesh > > &meshes) const =0`
- virtual `void tessellate (int resolution[ ], std::vector< shared_ptr< GeneralMesh > > &meshes) const =0`
- virtual `void tessellate (double density, std::vector< shared_ptr< GeneralMesh > > &meshes) const =0`
- virtual `void tessellatedCtrPolygon (std::vector< shared_ptr< LineCloud > > &ctr_pol) const =0`

### Protected Member Functions

- `double boxVecDist (const BoundingBox &b, const Point &v) const`
- `bool boxExtreme (const BoundingBox &box, const Point &dir, const Point &curr_pnt) const`

### Protected Attributes

- `tpTolerances toptol_`
- `int closest_idx_`

#### 29.60.1 Detailed Description

Abstract base class for a surface model or a curve model.

Definition at line 72 of file CompositeModel.h.

#### 29.60.2 Constructor & Destructor Documentation

##### 29.60.2.1 Go::CompositeModel::CompositeModel ( double gap, double neighbour, double kink, double bend )

Constructor with topology tolerances

##### Parameters

|                  |                                                                                                                                                                                                                                                                                                       |
|------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>gap</i>       | If the distance between two points are less than 'gap' they are viewed as identical.                                                                                                                                                                                                                  |
| <i>neighbour</i> | Maximum distance between curves or surfaces viewed as adjacent.                                                                                                                                                                                                                                       |
| <i>kink</i>      | If two adjacent curves or surfaces meet with an angle less than 'kink', they are seen as G1 continuous. (Angles in radians)                                                                                                                                                                           |
| <i>bend</i>      | If two surfaces meet along a common boundary and corresponding surface normals form an angle which is larger than 'bend', there is an intentional sharp edge between the surfaces. Similarly if two curves meet with an angle larger than 'bend', there is an intentional corner. (Angles in radians) |

##### 29.60.2.2 virtual Go::CompositeModel::~~CompositeModel ( ) [virtual]

Destructor.

### 29.60.3 Member Function Documentation

29.60.3.1 `virtual SurfaceModel* Go::CompositeModel::asSurfaceModel ( ) [inline],[virtual]`

Return surface model pointer

#### Returns

Pointer to surface model

Reimplemented in [Go::SurfaceModel](#).

Definition at line 112 of file CompositeModel.h.

29.60.3.2 `virtual BoundingBox Go::CompositeModel::boundingBox ( ) [pure virtual]`

Bounding box of the entire model

#### Returns

Bounding box

Implemented in [Go::SurfaceModel](#), [Go::CompositeCurve](#), [Go::CurveModel](#), and [Go::VolumeModel](#).

29.60.3.3 `virtual BoundingBox Go::CompositeModel::boundingBox ( int idx ) const [pure virtual]`

Bounding box corresponding to one entity

#### Parameters

|            |                 |
|------------|-----------------|
| <i>idx</i> | Index of entity |
|------------|-----------------|

#### Returns

Bounding box

Implemented in [Go::SurfaceModel](#), [Go::CompositeCurve](#), [Go::CurveModel](#), and [Go::VolumeModel](#).

29.60.3.4 `bool Go::CompositeModel::boxExtreme ( const BoundingBox & box, const Point & dir, const Point & curr_pnt ) const [protected]`

29.60.3.5 `double Go::CompositeModel::boxVecDist ( const BoundingBox & b, const Point & v ) const [protected]`

29.60.3.6 `virtual CompositeModel* Go::CompositeModel::clone ( ) const [inline],[virtual]`

Make a copy of the current model

**Returns**

Pointer to new [CompositeModel](#)

Reimplemented in [Go::SurfaceModel](#), [Go::CompositeCurve](#), [Go::CurveModel](#), and [Go::VolumeModel](#).

Definition at line 119 of file `CompositeModel.h`.

```
29.60.3.7 virtual void Go::CompositeModel::closestPoint (Point & pnt, Point & clo_pnt, int & idx, double clo_par[],
double & dist) [pure virtual]
```

Compute one closest point

**Parameters**

|            |             |
|------------|-------------|
| <i>pnt</i> | Input point |
|------------|-------------|

**Return values**

|                   |                                                            |
|-------------------|------------------------------------------------------------|
| <i>clo_pnt</i>    | Found closest point                                        |
| <i>idx</i>        | Index of curve or surface where the closest point is found |
| <i>clo_par</i> [] | Parameter value corresponding to the closest point         |
| <i>dist</i>       | Distance between input point and found closest point       |

Implemented in [Go::SurfaceModel](#), [Go::CompositeCurve](#), [Go::CurveModel](#), and [Go::VolumeModel](#).

```
29.60.3.8 virtual double Go::CompositeModel::curvature (int idx, double * par) const [pure virtual]
```

[Curvature](#) of an entity (Curve or Surface), Only Curve is implemented yet.

**Parameters**

|            |                                               |
|------------|-----------------------------------------------|
| <i>idx</i> | Index of entity                               |
| <i>par</i> | Parameter value at which to compute curvature |

**Returns**

The curvature.

Implemented in [Go::SurfaceModel](#), [Go::CompositeCurve](#), [Go::CurveModel](#), and [Go::VolumeModel](#).

```
29.60.3.9 virtual void Go::CompositeModel::evaluate (int idx, double par[], Point & pnt) const [pure virtual]
```

Evaluate position

## Parameters

|              |                 |
|--------------|-----------------|
| <i>idx</i>   | Index of curve  |
| <i>par[]</i> | Parameter value |

## Return values

|            |        |
|------------|--------|
| <i>pnt</i> | Result |
|------------|--------|

Implemented in [Go::SurfaceModel](#), [Go::CompositeCurve](#), [Go::CurveModel](#), and [Go::VolumeModel](#).

```
29.60.3.10 virtual void Go::CompositeModel::evaluate (int idx, double par[], int nder, std::vector< Point > & der) const
 [pure virtual]
```

Evaluate position and a number of derivatives

## Parameters

|              |                                                   |
|--------------|---------------------------------------------------|
| <i>idx</i>   | Index                                             |
| <i>par[]</i> | Parameter value                                   |
| <i>nder</i>  | Number of derivatives to compute, 0=only position |

## Return values

|            |        |
|------------|--------|
| <i>der</i> | Result |
|------------|--------|

Implemented in [Go::SurfaceModel](#), [Go::CompositeCurve](#), [Go::CurveModel](#), and [Go::VolumeModel](#).

```
29.60.3.11 virtual void Go::CompositeModel::extremalPoint (Point & dir, Point & ext_pnt, int & idx, double ext_par[])
 [pure virtual]
```

Extremal point(s) in a given direction

## Parameters

|                  |                                                             |
|------------------|-------------------------------------------------------------|
| <i>dir</i>       | Direction                                                   |
| <i>ext_pnt</i>   | Found extremal point                                        |
| <i>idx</i>       | Index of curve or surface where the extremal point is found |
| <i>ext_par[]</i> | Parameter value of extremal point                           |

Implemented in [Go::SurfaceModel](#), [Go::CompositeCurve](#), [Go::CurveModel](#), and [Go::VolumeModel](#).

```
29.60.3.12 tpTolerances Go::CompositeModel::getTolerances () [inline]
```

Return topology tolerances.

Definition at line 105 of file CompositeModel.h.

29.60.3.13 `virtual shared_ptr<IntResultsModel> Go::CompositeModel::intersect ( const ftLine & line ) [pure virtual]`

Intersection with a line. Expected output is points, probably one point. Curves can occur in special configurations.

#### Parameters

|             |           |
|-------------|-----------|
| <i>line</i> | The line. |
|-------------|-----------|

#### Returns

Pointer to an [IntResultsModel](#).

Implemented in [Go::SurfaceModel](#), [Go::CompositeCurve](#), [Go::CurveModel](#), and [Go::VolumeModel](#).

29.60.3.14 `virtual shared_ptr<IntResultsModel> Go::CompositeModel::intersect_plane ( const ftPlane & plane ) [pure virtual]`

Intersection with a plane.

#### Parameters

|              |            |
|--------------|------------|
| <i>plane</i> | The plane. |
|--------------|------------|

#### Returns

Pointer to an [IntResultsModel](#).

Implemented in [Go::SurfaceModel](#), [Go::CompositeCurve](#), [Go::CurveModel](#), and [Go::VolumeModel](#).

29.60.3.15 `virtual bool Go::CompositeModel::isDegenerate ( int idx ) const [pure virtual]`

Whether one particular entity is degenerate

#### Parameters

|            |                 |
|------------|-----------------|
| <i>idx</i> | Index of entity |
|------------|-----------------|

#### Returns

Whether the curve is degenerated

Implemented in [Go::SurfaceModel](#), [Go::CompositeCurve](#), [Go::CurveModel](#), and [Go::VolumeModel](#).

29.60.3.16 `virtual int Go::CompositeModel::nmbEntities ( ) const [pure virtual]`

Number of simple entities

**Returns**

Number of simple entities

Implemented in [Go::SurfaceModel](#), [Go::CompositeCurve](#), [Go::CurveModel](#), and [Go::VolumeModel](#).

29.60.3.17 `void Go::CompositeModel::setTolerances ( double gap, double neighbour, double kink, double bend )`

Set or reset topology tolerances

**Parameters**

|                  |                                                                                                                                                                                                                                                                                                       |
|------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>gap</i>       | If the distance between two points are less than 'gap' they are viewed as identical.                                                                                                                                                                                                                  |
| <i>neighbour</i> | Maximum distance between curves or surfaces viewed as adjacent.                                                                                                                                                                                                                                       |
| <i>kink</i>      | If two adjacent curves or surfaces meet with an angle less than 'kink', they are seen as G1 continuous. (Angles in radians)                                                                                                                                                                           |
| <i>bend</i>      | If two surfaces meet along a common boundary and corresponding surface normals form an angle which is larger than 'bend', there is an intentional sharp edge between the surfaces. Similarly if two curves meet with an angle larger than 'bend', there is an intentional corner. (Angles in radians) |

29.60.3.18 `virtual void Go::CompositeModel::tessellate ( std::vector< shared_ptr< GeneralMesh > > & meshes ) const`  
`[pure virtual]`

Tessellate model with respect to a default parameter

**Return values**

|               |                  |
|---------------|------------------|
| <i>meshes</i> | Tesselated model |
|---------------|------------------|

Implemented in [Go::SurfaceModel](#), [Go::CompositeCurve](#), [Go::CurveModel](#), and [Go::VolumeModel](#).

29.60.3.19 `virtual void Go::CompositeModel::tessellate ( int resolution[], std::vector< shared_ptr< GeneralMesh > > & meshes ) const`  
`[pure virtual]`

Tessellate model with respect to a given resolution

**Parameters**

|                      |                  |
|----------------------|------------------|
| <i>resolution</i> [] | Given resolution |
|----------------------|------------------|

**Return values**

|               |                  |
|---------------|------------------|
| <i>meshes</i> | Tesselated model |
|---------------|------------------|

Implemented in [Go::SurfaceModel](#), [Go::CompositeCurve](#), [Go::CurveModel](#), and [Go::VolumeModel](#).



29.60.3.20 `virtual void Go::CompositeModel::tessellate ( double density, std::vector< shared_ptr< GeneralMesh > > & meshes ) const` [pure virtual]

Tessellate model with respect to a given tessellation density

#### Parameters

|                |                      |
|----------------|----------------------|
| <i>density</i> | Tessellation density |
|----------------|----------------------|

#### Return values

|               |                  |
|---------------|------------------|
| <i>meshes</i> | Tesselated model |
|---------------|------------------|

Implemented in [Go::SurfaceModel](#), [Go::CompositeCurve](#), [Go::CurveModel](#), and [Go::VolumeModel](#).

29.60.3.21 `virtual void Go::CompositeModel::tesselatedCtrPolygon ( std::vector< shared_ptr< LineCloud > > & ctr_pol ) const` [pure virtual]

Return a tessellation of the control polygon of this entity

#### Return values

|                |                                           |
|----------------|-------------------------------------------|
| <i>ctr_pol</i> | Tesselated control polygon of this entity |
|----------------|-------------------------------------------|

Implemented in [Go::SurfaceModel](#), [Go::CompositeCurve](#), [Go::CurveModel](#), and [Go::VolumeModel](#).

29.60.3.22 `virtual void Go::CompositeModel::turn ( int idx )` [pure virtual]

Turn parameter directions of one entity

#### Parameters

|            |                 |
|------------|-----------------|
| <i>idx</i> | Index of entity |
|------------|-----------------|

Implemented in [Go::SurfaceModel](#), [Go::CompositeCurve](#), [Go::CurveModel](#), and [Go::VolumeModel](#).

29.60.3.23 `virtual void Go::CompositeModel::turn ( )` [pure virtual]

Turn parameter directions of all entities.

Implemented in [Go::SurfaceModel](#), [Go::CompositeCurve](#), [Go::CurveModel](#), and [Go::VolumeModel](#).

## 29.60.4 Member Data Documentation

29.60.4.1 `int Go::CompositeModel::closest_idx_` [mutable], [protected]

Definition at line 238 of file CompositeModel.h.

#### 29.60.4.2 tpTolerances Go::CompositeModel::toptol\_ [protected]

Definition at line 237 of file CompositeModel.h.

The documentation for this class was generated from the following file:

- [compositemodel/include/GoTools/compositemodel/CompositeModel.h](#)

## 29.61 Go::CompositeModelFactory Class Reference

```
#include <CompositeModelFactory.h>
```

### Public Member Functions

- [CompositeModelFactory](#) ([double](#) approxtol, [double](#) gap, [double](#) neighbour, [double](#) kink, [double](#) bend)  
*Constructor.*
- [~CompositeModelFactory](#) ()  
*Destructor.*
- [SurfaceModel](#) \* [createEmpty](#) ()  
*Create an empty model.*
- [CompositeModel](#) \* [createFromIges](#) (std::istream &is, [bool](#) use\_filetol=[false](#), [bool](#) prefer\_surfacemodel=[true](#))
- std::vector< shared\_ptr< [CompositeModel](#) > > [getModelsFromIges](#) (std::istream &is, [bool](#) use\_filetol=[false](#))  
*Get all connected models in the IGES file.*
- [CompositeModel](#) \* [createFromG2](#) (std::istream &is, [bool](#) prefer\_surfacemodel=[true](#))
- std::vector< shared\_ptr< [CompositeModel](#) > > [getModelsFromG2](#) (std::istream &is, [bool](#) use\_filetol=[false](#))  
*Get all connected models in the g2 file.*
- [SurfaceModel](#) \* [createFromSisl](#) (std::vector< [SISLSurf](#) \* > &surfaces)  
*Read a vector of sisl surfaces.*
- [CompositeCurve](#) \* [createFromSisl](#) (std::vector< [SISLCurve](#) \* > &curves)  
*Read a vector of sisl curves.*
- [SurfaceModel](#) \* [createFromBox](#) ([Point](#) corner, [Point](#) side\_vec, [Point](#) plane\_vec, [double](#) side1\_length, [double](#) side2\_length, [double](#) side3\_length)
- [SurfaceModel](#) \* [createFromSphere](#) ([Point](#) centre, [double](#) radius)
- [SurfaceModel](#) \* [createFromSphere](#) ([Point](#) centre, [Point](#) axis, [Point](#) equator, int latitude, int longitude)
- [SurfaceModel](#) \* [createFromCylinder](#) ([Point](#) bottom\_pos, [Point](#) cylinder\_axis, [Point](#) major\_axis, [Point](#) minor\_axis)
- [SurfaceModel](#) \* [interpolateCurves](#) (const std::vector< shared\_ptr< [SplineCurve](#) > > &curves, std::vector< [double](#) > &parvals, int open, int [degree](#)=3)
- [SurfaceModel](#) \* [interpolateCurves2](#) (const std::vector< shared\_ptr< [SplineCurve](#) > > &curves, std::vector< [double](#) > &param, int open, int [degree](#)=3)
- [SurfaceModel](#) \* [interpolateCurves](#) (const std::vector< shared\_ptr< [SplineCurve](#) > > &curves, std::vector< int > &crv\_type, std::vector< [double](#) > &parvals, int open, int [degree](#)=3)  
*Interpolate a set of positional curves where a tangent curve may be added.*
- [SurfaceModel](#) \* [interpolateCurves2](#) (const std::vector< shared\_ptr< [SplineCurve](#) > > &curves, std::vector< int > &crv\_type, std::vector< [double](#) > &param, int open, int [degree](#)=3)
- [CompositeCurve](#) \* [createCircularArc](#) ([Point](#) centre, [Point](#) start\_pt, [double](#) angle, [Point](#) axis)
- [CompositeCurve](#) \* [createEllipticArc](#) ([Point](#) centre, [Point](#) direction, [double](#) r1, [double](#) r2, [double](#) startpar, [double](#) angle, [Point](#) axis)
- [CompositeCurve](#) \* [createLineSegment](#) ([Point](#) startpt, [Point](#) endpt)  
*Create line segment bewteen the two points startpt and endpt.*

### 29.61.1 Detailed Description

[Factory](#) class for creating children of [CompositeModel](#)

Definition at line 63 of file CompositeModelFactory.h.

### 29.61.2 Constructor & Destructor Documentation

29.61.2.1 **Go::CompositeModelFactory::CompositeModelFactory ( *double approx\_tol*, *double gap*, *double neighbour*, *double kink*, *double bend* )**

Constructor.

29.61.2.2 **Go::CompositeModelFactory::~~CompositeModelFactory ( )**

Destructor.

### 29.61.3 Member Function Documentation

29.61.3.1 **CompositeCurve\* Go::CompositeModelFactory::createCircularArc ( *Point centre*, *Point start\_pt*, *double angle*, *Point axis* )**

Create a circular arc centre : [Circle](#) centre start\_pt : Start point of arc angle : Angle of arc axis : Axis of the plane in which the circle lies

29.61.3.2 **CompositeCurve\* Go::CompositeModelFactory::createEllipticArc ( *Point centre*, *Point direction*, *double r1*, *double r2*, *double startpar*, *double angle*, *Point axis* )**

Create an elliptic arc centre : [Ellipse](#) centre direction : Direction of major ellips axis r1 : Major radius r2 : Minor radius startpar : Start parameter of segment angle : Angle of arc axis : Axis of the plane in which the ellipse lies

29.61.3.3 **SurfaceModel\* Go::CompositeModelFactory::createEmpty ( )**

Create an empty model.

29.61.3.4 **SurfaceModel\* Go::CompositeModelFactory::createFromBox ( *Point corner*, *Point side\_vec*, *Point plane\_vec*, *double side1\_length*, *double side2\_length*, *double side3\_length* )**

Make surface model from a box Input is one box corner, the vector from this corner towards another corner in the same box side, yet another vector in the plane defining this box side and the lengths of the box sides. The first length corresponds to the vector defining the box side, the second to the other side in the defined plane and the third defines the depth of the box. The lengths may be negative.

29.61.3.5 **SurfaceModel\*** Go::CompositeModelFactory::createFromCylinder ( **Point** *bottom\_pos*, **Point** *cylinder\_axis*, **Point** *major\_axis*, **Point** *minor\_axis* )

Make surface model from truncated cylinder *bottom\_pos* : centre of cylinder at bottom *cylinder\_axis* : [Cylinder](#) axis. The length of the vector defines the cylinder height *major\_axis* : Major axis in ellipse at cylinder bottom. The vector length defines the major radius *minor\_axis* : Minor axis in ellipse at cylinder bottom. The vector length defines the minor radius

29.61.3.6 **CompositeModel\*** Go::CompositeModelFactory::createFromG2 ( **std::istream** & *is*, **bool** *prefer\_surfacemodel = true* )

Read G2 file One model is produced

29.61.3.7 **CompositeModel\*** Go::CompositeModelFactory::createFromIges ( **std::istream** & *is*, **bool** *use\_filetol = false*, **bool** *prefer\_surfacemodel = true* )

Read IGES file One model is produced

29.61.3.8 **SurfaceModel\*** Go::CompositeModelFactory::createFromSisl ( **std::vector**< **SISLSurf** \* > & *surfaces* )

Read a vector of sisl surfaces.

29.61.3.9 **CompositeCurve\*** Go::CompositeModelFactory::createFromSisl ( **std::vector**< **SISLCurve** \* > & *curves* )

Read a vector of sisl curves.

29.61.3.10 **SurfaceModel\*** Go::CompositeModelFactory::createFromSphere ( **Point** *centre*, **double** *radius* )

Make surface model from sphere Input is sphere center and radius

29.61.3.11 **SurfaceModel\*** Go::CompositeModelFactory::createFromSphere ( **Point** *centre*, **Point** *axis*, **Point** *equator*, **int** *latitude*, **int** *longitude* )

Make surface model from octants of a sphere Input is sphere center, axis toward north pole, vector from centre to point on equator and the number of octants. The length on the equator vector gives the sphere radius *latitude* = 1 : Octants in northern hemisphere *latitude* = 2 : Octants in both hemispheres *longitude* = 1 : Octants in first quadrant *longitude* = 2 : Octants in first and second quadrant *longitude* = 3 : Octants in first, second and third quadrant *longitude* = 4 : Octants in all quadrants

29.61.3.12 **CompositeCurve\*** Go::CompositeModelFactory::createLineSegment ( **Point** *startpt*, **Point** *endpt* )

Create line segment between the two points *startpt* and *endpt*.

29.61.3.13 `std::vector<shared_ptr<CompositeModel>> Go::CompositeModelFactory::getModelsFromG2 ( std::istream & is, bool use_filetol = false )`

Get all connected models in the g2 file.

29.61.3.14 `std::vector<shared_ptr<CompositeModel>> Go::CompositeModelFactory::getModelsFromIges ( std::istream & is, bool use_filetol = false )`

Get all connected models in the IGES file.

29.61.3.15 `SurfaceModel* Go::CompositeModelFactory::interpolateCurves ( const std::vector< shared_ptr< SplineCurve >> & curves, std::vector< double > & parvals, int open, int degree = 3 )`

Interpolate a set of positional curves. Automatic parameterization. The parameter values corresponding to the curves are given as output

29.61.3.16 `SurfaceModel* Go::CompositeModelFactory::interpolateCurves ( const std::vector< shared_ptr< SplineCurve >> & curves, std::vector< int > & crv_type, std::vector< double > & parvals, int open, int degree = 3 )`

Interpolate a set of positional curves where a tangent curve may be added.

positional curves are given as output NB! [Rational](#) input curves are not handled

29.61.3.17 `SurfaceModel* Go::CompositeModelFactory::interpolateCurves2 ( const std::vector< shared_ptr< SplineCurve >> & curves, std::vector< double > & param, int open, int degree = 3 )`

Interpolate a set of positional curves. Parameterization is given. If a closed surface is specified, one extra parameter value must be specified

29.61.3.18 `SurfaceModel* Go::CompositeModelFactory::interpolateCurves2 ( const std::vector< shared_ptr< SplineCurve >> & curves, std::vector< int > & crv_type, std::vector< double > & param, int open, int degree = 3 )`

Interpolate a set of positional curves where a tangent curve may be added. Parameterization is given and corresponds to the positional curves. NB! [Rational](#) input curves are not handled

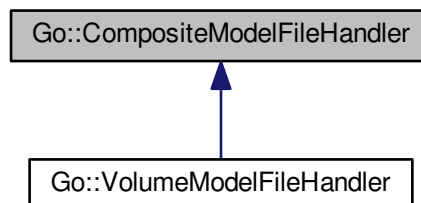
The documentation for this class was generated from the following file:

- `compositemodel/include/GoTools/compositemodel/CompositeModelFactory.h`

## 29.62 Go::CompositeModelFileHandler Class Reference

```
#include <CompositeModelFileHandler.h>
```

Inheritance diagram for Go::CompositeModelFileHandler:



### Classes

- struct [sfcvinfo](#)

### Public Member Functions

- [CompositeModelFileHandler](#) ()
- [~CompositeModelFileHandler](#) ()
- void [clear](#) ()
- void [writeStart](#) (std::ostream &os)
- void [writeEnd](#) (std::ostream &os)
- void [writeHeader](#) (const std::string &file\_content\_info, std::ostream &os)
- void [writeGeomObj](#) (const std::vector< shared\_ptr< [GeomObject](#) > > &geom\_obj, const std::vector< int > &obj\_id, std::ostream &os)
- void [writeSurfModel](#) (Go::SurfaceModel &surf\_model, std::ostream &os, int surfmodel\_id=-1, bool faces=true)
- void [writeBody](#) (shared\_ptr< [Body](#) > &body, std::ostream &os, int body\_id=-1)
- std::vector< shared\_ptr< [GeomObject](#) > > [readGeomObj](#) (const char \*filein)
- [SurfaceModel](#) [readSurfModel](#) (const char \*filein, int id=-1)
- shared\_ptr< [SurfaceModel](#) > [readShell](#) (const char \*filein, int id=-1)
- shared\_ptr< [Body](#) > [readBody](#) (const char \*filein, int id=-1)

### Protected Member Functions

- virtual shared\_ptr< [GeomObject](#) > [createGeomObject](#) (const ObjectHeader &obj\_header) const
- void [writeFaces](#) (std::ostream &os)
- void [readFaces](#) (const char \*filein)

## Protected Attributes

- `const int MAJOR_VERSION_`
- `const int MINOR_VERSION_`
- `const std::string indent_`
- `std::map< shared_ptr< GeomObject >, int > geom_objects_`
- `std::map< shared_ptr< SurfaceModel >, int > shells_`
- `std::map< shared_ptr< ftFaceBase >, int > faces_`
- `std::map< shared_ptr< Loop >, int > loops_`
- `std::map< shared_ptr< ftEdgeBase >, int > edges_`
- `std::map< shared_ptr< Vertex >, int > vertices_`
- `std::map< int, shared_ptr< ftSurface > > faces2_`
- `std::map< int, shared_ptr< GeomObject > > geom_objects2_`

### 29.62.1 Detailed Description

Definition at line 61 of file `CompositeModelFileHandler.h`.

### 29.62.2 Constructor & Destructor Documentation

29.62.2.1 `Go::CompositeModelFileHandler::CompositeModelFileHandler ( )` `[inline]`

Definition at line 65 of file `CompositeModelFileHandler.h`.

29.62.2.2 `Go::CompositeModelFileHandler::~~CompositeModelFileHandler ( )`

### 29.62.3 Member Function Documentation

29.62.3.1 `void Go::CompositeModelFileHandler::clear ( )`

29.62.3.2 `virtual shared_ptr< GeomObject > Go::CompositeModelFileHandler::createGeomObject ( const ObjectHeader & obj_header ) const` `[protected]`, `[virtual]`

29.62.3.3 `shared_ptr< Body > Go::CompositeModelFileHandler::readBody ( const char * filein, int id = -1 )`

29.62.3.4 `void Go::CompositeModelFileHandler::readFaces ( const char * filein )` `[protected]`

29.62.3.5 `std::vector< shared_ptr< GeomObject > > Go::CompositeModelFileHandler::readGeomObj ( const char * filein )`

29.62.3.6 `shared_ptr< SurfaceModel > Go::CompositeModelFileHandler::readShell ( const char * filein, int id = -1 )`

29.62.3.7 `SurfaceModel Go::CompositeModelFileHandler::readSurfModel ( const char * filein, int id = -1 )`

29.62.3.8 `void Go::CompositeModelFileHandler::writeBody ( shared_ptr< Body > & body, std::ostream & os, int body_id = -1 )`

29.62.3.9 void Go::CompositeModelFileHandler::writeEnd ( std::ostream & os )

29.62.3.10 void Go::CompositeModelFileHandler::writeFaces ( std::ostream & os ) [protected]

29.62.3.11 void Go::CompositeModelFileHandler::writeGeomObj ( const std::vector< shared\_ptr< **GeomObject** > > & geom\_obj, const std::vector< int > & obj\_id, std::ostream & os )

29.62.3.12 void Go::CompositeModelFileHandler::writeHeader ( const std::string & file\_content\_info, std::ostream & os )

29.62.3.13 void Go::CompositeModelFileHandler::writeStart ( std::ostream & os )

29.62.3.14 void Go::CompositeModelFileHandler::writeSurfModel ( Go::SurfaceModel & surf\_model, std::ostream & os, int surfmodel\_id = -1, bool faces = true )

## 29.62.4 Member Data Documentation

29.62.4.1 std::map<shared\_ptr<ftEdgeBase>, int> Go::CompositeModelFileHandler::edges\_ [protected]

Definition at line 140 of file CompositeModelFileHandler.h.

29.62.4.2 std::map<int, shared\_ptr<ftSurface>> Go::CompositeModelFileHandler::faces2\_ [protected]

Definition at line 144 of file CompositeModelFileHandler.h.

29.62.4.3 std::map<shared\_ptr<ftFaceBase>, int> Go::CompositeModelFileHandler::faces\_ [protected]

Definition at line 138 of file CompositeModelFileHandler.h.

29.62.4.4 std::map<int, shared\_ptr<GeomObject>> Go::CompositeModelFileHandler::geom\_objects2\_ [protected]

Definition at line 145 of file CompositeModelFileHandler.h.

29.62.4.5 std::map<shared\_ptr<GeomObject>, int> Go::CompositeModelFileHandler::geom\_objects\_ [protected]

Definition at line 135 of file CompositeModelFileHandler.h.

29.62.4.6 const std::string Go::CompositeModelFileHandler::indent\_ [protected]

Definition at line 125 of file CompositeModelFileHandler.h.



29.62.4.7 `std::map<shared_ptr<Loop>, int>` `Go::CompositeModelFileHandler::loops_` [protected]

Definition at line 139 of file `CompositeModelFileHandler.h`.

29.62.4.8 `const int` `Go::CompositeModelFileHandler::MAJOR_VERSION_` [protected]

Definition at line 123 of file `CompositeModelFileHandler.h`.

29.62.4.9 `const int` `Go::CompositeModelFileHandler::MINOR_VERSION_` [protected]

Definition at line 124 of file `CompositeModelFileHandler.h`.

29.62.4.10 `std::map<shared_ptr<SurfaceModel>, int>` `Go::CompositeModelFileHandler::shells_` [protected]

Definition at line 137 of file `CompositeModelFileHandler.h`.

29.62.4.11 `std::map<shared_ptr<Vertex>, int>` `Go::CompositeModelFileHandler::vertices_` [protected]

Definition at line 141 of file `CompositeModelFileHandler.h`.

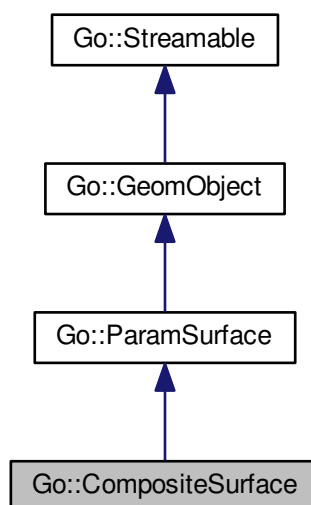
The documentation for this class was generated from the following file:

- `compositemodel/include/GoTools/compositemodel/CompositeModelFileHandler.h`

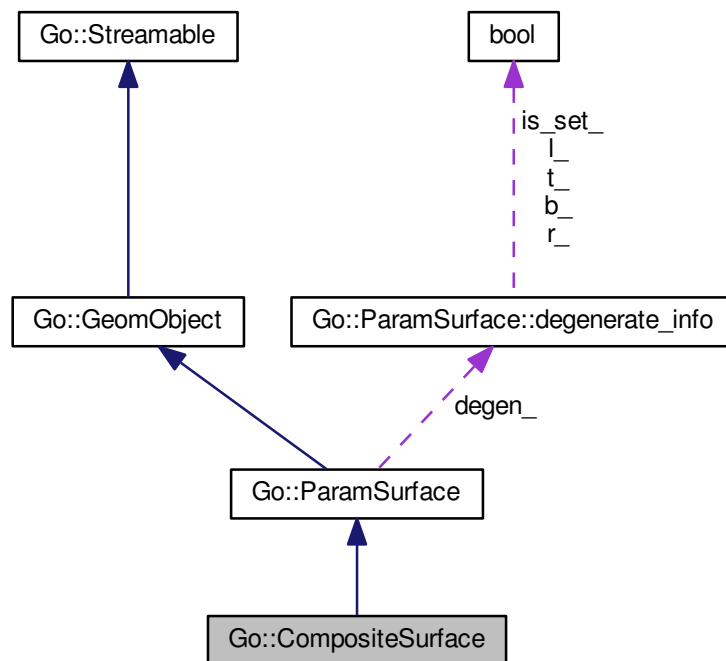
## 29.63 Go::CompositeSurface Class Reference

```
#include <CompositeSurface.h>
```

Inheritance diagram for `Go::CompositeSurface`:



Collaboration diagram for Go::CompositeSurface:



## Public Member Functions

- [CompositeSurface](#) ()
- [CompositeSurface](#) (const std::vector< [SplineSurface](#) > &domain\_maps, const [SplineSurface](#) &surf)
- virtual void [read](#) (std::istream &is)
- virtual void [write](#) (std::ostream &os) const
- virtual [BoundingBox](#) [boundingBox](#) () const
  - Return the object's bounding box.*
- virtual int [dimension](#) () const
  - Return the dimension of the space in which the object lies (usually 2 or 3)*
- virtual [ClassType](#) [instanceType](#) () const
  - Return the class type identifier of a given, derived instance of [GeomObject](#).*
- virtual [~CompositeSurface](#) ()
  - Virtual destructor, enables safe inheritance.*
- virtual [CompositeSurface](#) \* [clone](#) () const
- virtual const [RectDomain](#) & [parameterDomain](#) () const
- virtual [RectDomain](#) [containingDomain](#) () const
- void [setParameterDomain](#) (double u1, double u2, double v1, double v2)
- virtual [CurveLoop](#) [outerBoundaryLoop](#) (double degenerate\_epsilon=DEFAULT\_SPACE\_EPSILON) const
- virtual std::vector< [CurveLoop](#) > [allBoundaryLoops](#) (double degenerate\_epsilon=DEFAULT\_SPACE\_EPSILON) const
- virtual [DirectionCone](#) [normalCone](#) () const
- virtual [DirectionCone](#) [tangentCone](#) (bool padir\_is\_u) const
- virtual [CompositeBox](#) [compositeBox](#) () const

- virtual void [point](#) ([Point](#) &pt, [double](#) upar, [double](#) vpar) [const](#)
- virtual void [point](#) (std::vector< [Point](#) > &pts, [double](#) upar, [double](#) vpar, int derivs, [bool](#) u\_from\_right=true, [bool](#) v\_from\_right=true, [double](#) resolution=1.0e-12) [const](#)
- virtual void [normal](#) ([Point](#) &n, [double](#) upar, [double](#) vpar) [const](#)
- virtual std::vector< shared\_ptr< [ParamCurve](#) > > [constParamCurves](#) ([double](#) parameter, [bool](#) paddir\_is\_u) [const](#)
- virtual std::vector< shared\_ptr< [ParamSurface](#) > > [subSurfaces](#) ([double](#) from\_upar, [double](#) from\_vpar, [double](#) to\_upar, [double](#) to\_vpar, [double](#) fuzzy=DEFAULT\_PARAMETER\_EPSILON) [const](#)
- virtual [double](#) [nextSegmentVal](#) (int dir, [double](#) par, [bool](#) forward, [double](#) tol) [const](#)
- virtual void [closestPoint](#) ([const Point](#) &pt, [double](#) &clo\_u, [double](#) &clo\_v, [Point](#) &clo\_pt, [double](#) &clo\_dist, [double](#) epsilon, [const RectDomain](#) \*domain\_of\_interest=NULL, [double](#) \*seed=0) [const](#)
- virtual void [closestBoundaryPoint](#) ([const Point](#) &pt, [double](#) &clo\_u, [double](#) &clo\_v, [Point](#) &clo\_pt, [double](#) &clo\_dist, [double](#) epsilon, [const RectDomain](#) \*rd=NULL, [double](#) \*seed=0) [const](#)
- virtual void [getCornerPoints](#) (std::vector< std::pair< [Point](#), [Point](#) > > &corners) [const](#)
- virtual void [getBoundaryInfo](#) ([Point](#) &pt1, [Point](#) &pt2, [double](#) epsilon, [SplineCurve](#) \*&cv, [SplineCurve](#) \*&crosscv, [double](#) knot\_tol=1e-05) [const](#)
- virtual void [turnOrientation](#) ()
  - Turns the direction of the normal of the surface.*
- virtual void [reverseParameterDirection](#) ([bool](#) direction\_is\_u)
- virtual void [swapParameterDirection](#) ()
  - Swaps the two parameter directions.*
- virtual [bool](#) [isDegenerate](#) ([bool](#) &b, [bool](#) &r, [bool](#) &t, [bool](#) &l, [double](#) tolerance) [const](#)
- virtual [double](#) [area](#) ([double](#) tol) [const](#)
- virtual [bool](#) [inDomain](#) ([double](#) u, [double](#) v, [double](#) eps=1.0e-4) [const](#)
  - Check if a parameter pair lies inside the domain of this surface.*
- virtual int [inDomain2](#) ([double](#) u, [double](#) v, [double](#) eps=1.0e-4) [const](#)
- virtual [bool](#) [onBoundary](#) ([double](#) u, [double](#) v, [double](#) eps=1.0e-4) [const](#)
  - Check if a parameter pair lies at the boundary of this surface.*
- virtual [Point](#) [closestInDomain](#) ([double](#) u, [double](#) v) [const](#)
- virtual void [getDegenerateCorners](#) (std::vector< [Point](#) > &deg\_corners, [double](#) tol) [const](#)
  - Check for parallel and anti-parallel partial derivatives in surface corners.*

## Static Public Member Functions

- static [ClassType](#) [classType](#) ()
  - Return the class type identifier of a given class derived from [GeomObject](#).*

## Additional Inherited Members

### 29.63.1 Detailed Description

Brief description. Detailed description.

Definition at line 52 of file CompositeSurface.h.

### 29.63.2 Constructor & Destructor Documentation

#### 29.63.2.1 Go::CompositeSurface::CompositeSurface ( ) [inline]

Definition at line 55 of file CompositeSurface.h.

29.63.2.2 `Go::CompositeSurface::CompositeSurface ( const std::vector< SplineSurface > & domain_maps, const SplineSurface & surf ) [inline]`

Definition at line 57 of file CompositeSurface.h.

29.63.2.3 `virtual Go::CompositeSurface::~~CompositeSurface ( ) [inline],[virtual]`

Virtual destructor, enables safe inheritance.

Definition at line 109 of file CompositeSurface.h.

### 29.63.3 Member Function Documentation

29.63.3.1 `virtual std::vector<CurveLoop> Go::CompositeSurface::allBoundaryLoops ( double degenerate_epsilon = DEFAULT_SPACE_EPSILON ) const [inline],[virtual]`

Returns the anticlockwise outer boundary loop of the surface, together with clockwise loops of any interior boundaries, such that the surface always is 'to the left of' the loops.

#### Parameters

|                           |                                                            |
|---------------------------|------------------------------------------------------------|
| <i>degenerate_epsilon</i> | edges whose length is smaller than this value are ignored. |
|---------------------------|------------------------------------------------------------|

#### Returns

a vector containing CurveLoops. The first of these describe the outer boundary of the surface (clockwise), whereas the others describe boundaries of interior holes (clockwise).

Implements [Go::ParamSurface](#).

Definition at line 159 of file CompositeSurface.h.

29.63.3.2 `virtual double Go::CompositeSurface::area ( double tol ) const [inline],[virtual]`

Compute the total area of this surface up to some tolerance

#### Parameters

|            |                                                                                                                            |
|------------|----------------------------------------------------------------------------------------------------------------------------|
| <i>tol</i> | the relative tolerance when approximating the area, i.e. stop iteration when error becomes smaller than tol/(surface area) |
|------------|----------------------------------------------------------------------------------------------------------------------------|

#### Returns

the area calculated

Implements [Go::ParamSurface](#).

Definition at line 418 of file CompositeSurface.h.

29.63.3.3 `virtual BoundingBox Go::CompositeSurface::boundingBox ( ) const [inline],[virtual]`

Return the object's bounding box.

Implements [Go::GeomObject](#).

Definition at line 92 of file CompositeSurface.h.

29.63.3.4 `static ClassType Go::CompositeSurface::classType ( ) [inline],[static]`

Return the class type identifier of a given class derived from [GeomObject](#).

Definition at line 104 of file CompositeSurface.h.

29.63.3.5 `virtual CompositeSurface* Go::CompositeSurface::clone ( ) const [inline],[virtual]`

make a clone of this surface and return a pointer to it (user is responsible for clearing up memory afterwards).

Returns

pointer to cloned object

Implements [Go::ParamSurface](#).

Definition at line 115 of file CompositeSurface.h.

29.63.3.6 `virtual void Go::CompositeSurface::closestBoundaryPoint ( const Point & pt, double & clo_u, double & clo_v, Point & clo_pt, double & clo_dist, double epsilon, const RectDomain * rd = NULL, double * seed = 0 ) const [inline],[virtual]`

Iterates to the closest point to pt on the boundary of the surface.

See also

[closestPoint\(\)](#)

Implements [Go::ParamSurface](#).

Definition at line 346 of file CompositeSurface.h.

29.63.3.7 `virtual Point Go::CompositeSurface::closestInDomain ( double u, double v ) const [inline],[virtual]`

Fetch the parameter value in the parameter domain of the surface closest to the parameter pair (u,v)

Implements [Go::ParamSurface](#).

Definition at line 438 of file CompositeSurface.h.

29.63.3.8 `virtual void Go::CompositeSurface::closestPoint ( const Point & pt, double & clo_u, double & clo_v, Point & clo_pt, double & clo_dist, double epsilon, const RectDomain * domain_of_interest = NULL, double * seed = 0 ) const [inline],[virtual]`

Iterates to the closest point to pt on the surface.

## Parameters

|                           |                                                                                                                              |
|---------------------------|------------------------------------------------------------------------------------------------------------------------------|
| <i>pt</i>                 | the point to find the closest point to                                                                                       |
| <i>clo_u</i>              | u parameter of the closest point                                                                                             |
| <i>clo_v</i>              | v parameter of the closest point                                                                                             |
| <i>clo_pt</i>             | the geometric position of the closest point                                                                                  |
| <i>clo_dist</i>           | the distance between pt and clo_pt                                                                                           |
| <i>epsilon</i>            | parameter tolerance (will in any case not be higher than sqrt(machine_precision) x magnitude of solution)                    |
| <i>domain_of_interest</i> | pointer to parameter domain in which to search for closest point. If a NULL pointer is used, the entire surface is searched. |
| <i>seed</i>               | pointer to parameter values where iteration starts.                                                                          |

Reimplemented from [Go::ParamSurface](#).

Definition at line 333 of file CompositeSurface.h.

**29.63.3.9** virtual **CompositeBox** **Go::CompositeSurface::compositeBox** ( ) const [inline],[virtual]

Creates a composite box enclosing the surface. The composite box consists of an inner and an edge box. The inner box is supposed to be made from the interior of the surface, while the edge box is made from the boundary curves. The default implementation simply makes both boxes identical to the regular bounding box.

## Returns

the [CompositeBox](#) of the surface, as specified above

Reimplemented from [Go::ParamSurface](#).

Definition at line 186 of file CompositeSurface.h.

**29.63.3.10** virtual **std::vector<shared\_ptr<ParamCurve>>** **Go::CompositeSurface::constParamCurves** ( double *parameter*, bool *pardir\_is\_u* ) const [inline],[virtual]

Get the curve(s) obtained by intersecting the surface with one of its constant parameter curves. For surfaces without holes, this will be the parameter curve itself; for surfaces with interior holes this may be a collection of several, disjoint curves.

## Parameters

|                    |                                                                                                                                              |
|--------------------|----------------------------------------------------------------------------------------------------------------------------------------------|
| <i>parameter</i>   | parameter value for the constant parameter (either u or v)                                                                                   |
| <i>pardir_is_u</i> | specify whether the <i>moving</i> parameter (as opposed to the <i>constant</i> parameter) is the first ('true') or the second ('false') one. |

## Returns

a vector containing shared pointers to the obtained, newly constructed constant-parameter curves.

Implements [Go::ParamSurface](#).

Definition at line 278 of file CompositeSurface.h.

29.63.3.11 `virtual RectDomain Go::CompositeSurface::containingDomain ( ) const [inline],[virtual]`

Get a rectangular parameter domain that is guaranteed to contain the surface's [parameterDomain\(\)](#). It may be the same. There is no guarantee that this is the smallest domain containing the actual domain.

Returns

a [RectDomain](#) that is guaranteed to include the surface's total parameter domain.

Implements [Go::ParamSurface](#).

Definition at line 130 of file `CompositeSurface.h`.

29.63.3.12 `virtual int Go::CompositeSurface::dimension ( ) const [inline],[virtual]`

Return the dimension of the space in which the object lies (usually 2 or 3)

Implements [Go::GeomObject](#).

Definition at line 96 of file `CompositeSurface.h`.

29.63.3.13 `virtual void Go::CompositeSurface::getBoundaryInfo ( Point & pt1, Point & pt2, double epsilon, SplineCurve *& cv, SplineCurve *& crosscv, double knot_tol = 1e-05 ) const [inline],[virtual]`

Get the boundary curve segment between two points on the boundary, as well as the cross-tangent curve. If the given points are not positioned on the same boundary (within a certain tolerance), no curves will be created.

Parameters

|                 |                                                                                                                                                                                                                                                                                                                                    |
|-----------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>pt1</i>      | the first point on the boundary, given by the user                                                                                                                                                                                                                                                                                 |
| <i>pt2</i>      | the second point on the boundary, given by the user                                                                                                                                                                                                                                                                                |
| <i>epsilon</i>  | the tolerance used when determining whether the given points are lying on a boundary, and if they do, whether they both lie on the <i>same</i> boundary.                                                                                                                                                                           |
| <i>cv</i>       | upon return, this will point to a newly created curve representing the boundary curve between 'pt1' and 'pt2'. The user assumes ownership of this object and is responsible for its deletion. No curve is created if the given points are not found to lie on the same boundary.                                                   |
| <i>crosscv</i>  | upon return, this will point to a newly created curve representing the cross-boundary curve between 'pt1' and 'pt2'. The user assumes ownership of this object and is responsible for its deletion. The direction is outwards from the surface. No curve is created if the given points are not found to lie on the same boundary. |
| <i>knot_tol</i> | tolerance used when working with the knot-vector, to specify how close a parameter value must be to a knot in order to be considered 'on' the knot.                                                                                                                                                                                |

Implements [Go::ParamSurface](#).

Definition at line 381 of file `CompositeSurface.h`.

29.63.3.14 `virtual void Go::CompositeSurface::getCornerPoints ( std::vector< std::pair< Point, Point > > & corners ) const [inline],[virtual]`

Return surface corners, geometric and parametric points in that sequence

Implements [Go::ParamSurface](#).

Definition at line 357 of file CompositeSurface.h.

```
29.63.3.15 virtual void Go::CompositeSurface::getDegenerateCorners (std::vector< Point > & deg_corners, double tol)
 const [inline],[virtual]
```

Check for parallel and anti-parallel partial derivatives in surface corners.

Implements [Go::ParamSurface](#).

Definition at line 443 of file CompositeSurface.h.

```
29.63.3.16 virtual bool Go::CompositeSurface::inDomain (double u, double v, double eps = 1.0e-4) const
 [inline],[virtual]
```

Check if a parameter pair lies inside the domain of this surface.

Implements [Go::ParamSurface](#).

Definition at line 423 of file CompositeSurface.h.

```
29.63.3.17 virtual int Go::CompositeSurface::inDomain2 (double u, double v, double eps = 1.0e-4) const
 [inline],[virtual]
```

Check if a parameter pair lies inside the domain of this surface return value = 0: outside = 1: internal = 2: at the boundary

Implements [Go::ParamSurface](#).

Definition at line 428 of file CompositeSurface.h.

```
29.63.3.18 virtual ClassType Go::CompositeSurface::instanceType () const [inline],[virtual]
```

Return the class type identifier of a given, derived instance of [GeomObject](#).

Implements [Go::GeomObject](#).

Definition at line 100 of file CompositeSurface.h.

```
29.63.3.19 virtual bool Go::CompositeSurface::isDegenerate (bool & b, bool & r, bool & t, bool & l, double tolerance)
 const [inline],[virtual]
```

The order of the edge indicators (bottom, right, top, left) matches the edge\_number of edgeCurve(). Query whether any of the four boundary curves are degenerate (zero length) within a certain tolerance. In the below, we refer to 'u' as the first parameter and 'v' as the second.



## Parameters

|                  |                                                                                                    |
|------------------|----------------------------------------------------------------------------------------------------|
| <i>b</i>         | 'true' upon return of function if the boundary ( $v = v_{\min}$ ) is degenerate                    |
| <i>r</i>         | 'true' upon return of function if the boundary ( $v = v_{\max}$ ) is degenerate                    |
| <i>t</i>         | 'true' upon return of function if the boundary ( $u = u_{\min}$ ) is degenerate                    |
| <i>l</i>         | 'true' upon return of function if the boundary ( $u = u_{\max}$ ) is degenerate                    |
| <i>tolerance</i> | boundaries are considered degenerate if their length is shorter than this value, given by the user |

## Returns

'true' if at least one boundary curve was found to be degenerate, 'false' otherwise.

Reimplemented from [Go::ParamSurface](#).

Definition at line 414 of file CompositeSurface.h.

```
29.63.3.20 virtual double Go::CompositeSurface::nextSegmentVal (int dir, double par, bool forward, double tol) const
 [inline], [virtual]
```

Determine the parameter value of the start of the 'next segment' from a parameter value, along a given parameter direction. A 'segment' is here defined as a parameter interval in which there will be no discontinuities in derivatives or other artifacts. For spline objects, a segment will typically be the interval between two consecutive, non-coincident knots.

## Parameters

|                |                                                                                                      |
|----------------|------------------------------------------------------------------------------------------------------|
| <i>dir</i>     | the parameter direction in which we search for the next segment (0 or 1)                             |
| <i>par</i>     | the parameter value starting from which we search for the start value of the next segment            |
| <i>forward</i> | define whether we shall move forward ('true') or backwards when searching along this parameter       |
| <i>tol</i>     | tolerance used for determining whether the 'par' is already located <i>on</i> the next segment value |

## Returns

the value of the start value of the next segment (or the end of the previous segment, if we are moving backwards...)

Implements [Go::ParamSurface](#).

Definition at line 318 of file CompositeSurface.h.

```
29.63.3.21 virtual void Go::CompositeSurface::normal (Point & n, double upar, double vpar) const [inline],
 [virtual]
```

Evaluate the surface's position and a certain number of derivatives at a given parameter.

## Parameters

|               |                                 |
|---------------|---------------------------------|
| <i>upar</i>   | the first parameter             |
| <i>vpar</i>   | the second parameter            |
| <i>derivs</i> | number of requested derivatives |

**Returns**

the vector containing the evaluated values. Its size will be equal to  $(\text{derivs}+1) * (\text{derivs}+2) / 2$ . Its first entry is the surface's position at the given parameter pair. Then, if 'derivs' > 0, the two next entries will be the surface tangents along the first and second parameter direction. The next three entries are the second- and cross derivatives, in the order (du2, dudv, dv2), and similar for even higher derivatives. Evaluates the surface normal for a given parameter pair

**Parameters**

|             |                                                      |
|-------------|------------------------------------------------------|
| <i>n</i>    | the computed normal will be written to this variable |
| <i>upar</i> | the first parameter                                  |
| <i>vpar</i> | the second parameter                                 |

Implements [Go::ParamSurface](#).

Definition at line 253 of file CompositeSurface.h.

**29.63.3.22** virtual **DirectionCone** Go::CompositeSurface::normalCone ( ) const [inline],[virtual]

Creates a [DirectionCone](#) covering all normals to this surface.

**Returns**

a [DirectionCone](#) (not necessarily the smallest) containing all normals to this surface.

Implements [Go::ParamSurface](#).

Definition at line 166 of file CompositeSurface.h.

**29.63.3.23** virtual **bool** Go::CompositeSurface::onBoundary ( double *u*, double *v*, double *eps* = 1.0e-4 ) const [inline],[virtual]

Check if a parameter pair lies at the boundary of this surface.

Implements [Go::ParamSurface](#).

Definition at line 433 of file CompositeSurface.h.

**29.63.3.24** virtual **CurveLoop** Go::CompositeSurface::outerBoundaryLoop ( double *degenerate\_epsilon* = DEFAULT\_SPACE\_EPSILON ) const [inline],[virtual]

Returns the anticlockwise, outer boundary loop of the surface.

**Parameters**

|                           |                                                            |
|---------------------------|------------------------------------------------------------|
| <i>degenerate_epsilon</i> | edges whose length is smaller than this value are ignored. |
|---------------------------|------------------------------------------------------------|

**Returns**

a [CurveLoop](#) describing the anticlockwise, outer boundary loop of the surface.

Implements [Go::ParamSurface](#).

Definition at line 147 of file CompositeSurface.h.

**29.63.3.25** `virtual const RectDomain& Go::CompositeSurface::parameterDomain ( ) const` `[inline],[virtual]`

Return the parameter domain of the surface. This may be a simple rectangular domain ([RectDomain](#)) or any other subclass of [Domain](#) (such as [GoCurveBoundedDomain](#), found in the `sisl_dependent` module).

**Returns**

a [Domain](#) object describing the parametric domain of the surface

Implements [Go::ParamSurface](#).

Definition at line 122 of file CompositeSurface.h.

**29.63.3.26** `virtual void Go::CompositeSurface::point ( Point & pt, double upar, double vpar ) const` `[inline],[virtual]`

Evaluates the surface's position for a given parameter pair.

**Parameters**

|             |                                              |
|-------------|----------------------------------------------|
| <i>pt</i>   | the result of the evaluation is written here |
| <i>upar</i> | the first parameter                          |
| <i>vpar</i> | the second parameter                         |

Implements [Go::ParamSurface](#).

Definition at line 193 of file CompositeSurface.h.

**29.63.3.27** `virtual void Go::CompositeSurface::point ( std::vector< Point > & pts, double upar, double vpar, int derivs, bool u_from_right = true, bool v_from_right = true, double resolution = 1.0e-12 ) const` `[inline],[virtual]`

Evaluates the surface's position and a certain number of derivatives for a given parameter pair.

**Parameters**

|            |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>pts</i> | the vector containing the evaluated values. Its size must be set by the user prior to calling this function, and should be equal to $(derivs+1) * (derivs+2) / 2$ . Upon completion of the function, its first entry is the surface's position at the given parameter pair. Then, if 'derivs' > 0, the two next entries will be the surface tangents along the first and second parameter direction. The next three entries are the second- and cross derivatives, in the order (du2, dudv, dv2), and similar for even higher derivatives. |
|------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

## Parameters

|                     |                                                                                                                                                                  |
|---------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>upar</i>         | the first parameter                                                                                                                                              |
| <i>vpar</i>         | the second parameter                                                                                                                                             |
| <i>derivs</i>       | number of requested derivatives                                                                                                                                  |
| <i>u_from_right</i> | specify whether derivatives along the first parameter are to be calculated from the right ('true', default) or from the left ('false')                           |
| <i>v_from_right</i> | specify whether derivatives along the second parameter are to be calculated from the right ('true', default) or from the left ('false')                          |
| <i>resolution</i>   | tolerance used when determining whether parameters are located at special values of the parameter domain (in particular; knot values in case of spline objects). |

Implements [Go::ParamSurface](#).

Definition at line 230 of file CompositeSurface.h.

```
29.63.3.28 virtual void Go::CompositeSurface::read (std::istream & is) [inline],[virtual]
```

read object from stream

## Parameters

|           |                                  |
|-----------|----------------------------------|
| <i>is</i> | stream from which object is read |
|-----------|----------------------------------|

Implements [Go::Streamable](#).

Definition at line 69 of file CompositeSurface.h.

```
29.63.3.29 virtual void Go::CompositeSurface::reverseParameterDirection (bool direction_is_u) [inline],[virtual]
```

Reverses the direction of the basis in input direction.

## Parameters

|                       |                                                                                                                       |
|-----------------------|-----------------------------------------------------------------------------------------------------------------------|
| <i>direction_is_u</i> | if 'true', the first parameter direction will be reversed, otherwise, the second parameter direction will be reversed |
|-----------------------|-----------------------------------------------------------------------------------------------------------------------|

Implements [Go::ParamSurface](#).

Definition at line 393 of file CompositeSurface.h.

```
29.63.3.30 void Go::CompositeSurface::setParameterDomain (double u1, double u2, double v1, double v2) [inline],[virtual]
```

set the parameter domain to a given rectangle

## Parameters

|           |                                         |
|-----------|-----------------------------------------|
| <i>u1</i> | new min. value of first parameter span  |
| <i>u2</i> | new max. value of first parameter span  |
| <i>v1</i> | new min. value of second parameter span |
| <i>v2</i> | new max. value of second parameter span |

Reimplemented from [Go::ParamSurface](#).

Definition at line 138 of file CompositeSurface.h.

```
29.63.3.31 virtual std::vector<shared_ptr<ParamSurface>> Go::CompositeSurface::subSurfaces (double from_upar,
double from_vpar, double to_upar, double to_vpar, double fuzzy = DEFAULT_PARAMETER_EPSILON
) const [inline],[virtual]
```

Get the surface(s) obtained by cropping the parameter domain of this surface between given values for the first and second parameter. In general, for surfaces with no interior holes, the result will be *one* surface; however, for surfaces with interior holes, the result might be *several disjoint* surfaces.

## Parameters

|                  |                                                                       |
|------------------|-----------------------------------------------------------------------|
| <i>from_upar</i> | lower value for the first parameter in the subdomain                  |
| <i>from_vpar</i> | lower value for the second parameter in the subdomain                 |
| <i>to_upar</i>   | upper value for the first parameter in the subdomain                  |
| <i>to_vpar</i>   | upper value for the second parameter in the subdomain                 |
| <i>fuzzy</i>     | tolerance used when determining intersection with interior boundaries |

## Returns

a vector contained shared pointers to the obtained, newly constructed sub-surfaces.

Implements [Go::ParamSurface](#).

Definition at line 295 of file CompositeSurface.h.

```
29.63.3.32 virtual void Go::CompositeSurface::swapParameterDirection () [inline],[virtual]
```

Swaps the two parameter directions.

Implements [Go::ParamSurface](#).

Definition at line 397 of file CompositeSurface.h.

```
29.63.3.33 virtual DirectionCone Go::CompositeSurface::tangentCone (bool pdir_is_u) const [inline],
[virtual]
```

Creates a [DirectionCone](#) covering all tangents to this surface along a given parameter direction.

## Parameters

|                                |                                                                                                                                                                                                  |
|--------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>pardir_is↔<br/>_u</code> | if 'true', then the <a href="#">DirectionCone</a> will be defined on basis of the surface's tangents along the first parameter direction. Otherwise the second parameter direction will be used. |
|--------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

## Returns

a [DirectionCone](#) (not necessarily the smallest) containing all tangents to this surface along the specified parameter direction.

Implements [Go::ParamSurface](#).

Definition at line 176 of file CompositeSurface.h.

**29.63.3.34** `virtual void Go::CompositeSurface::turnOrientation ( ) [inline],[virtual]`

Turns the direction of the normal of the surface.

Implements [Go::ParamSurface](#).

Definition at line 387 of file CompositeSurface.h.

**29.63.3.35** `virtual void Go::CompositeSurface::write ( std::ostream & os ) const [inline],[virtual]`

write object to stream

## Parameters

|                 |                                   |
|-----------------|-----------------------------------|
| <code>os</code> | stream to which object is written |
|-----------------|-----------------------------------|

Implements [Go::Streamable](#).

Definition at line 81 of file CompositeSurface.h.

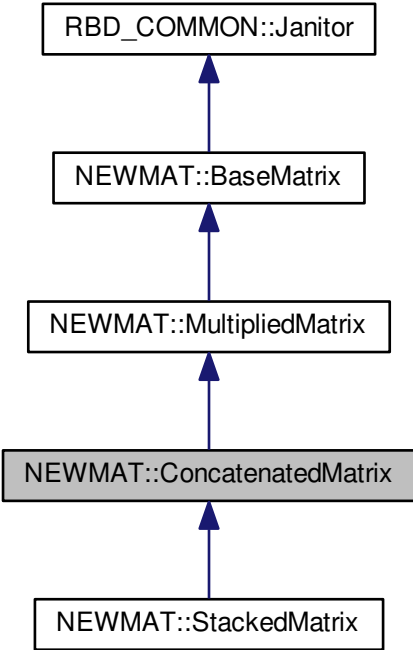
The documentation for this class was generated from the following file:

- [gotools-core/include/GoTools/geometry/CompositeSurface.h](#)

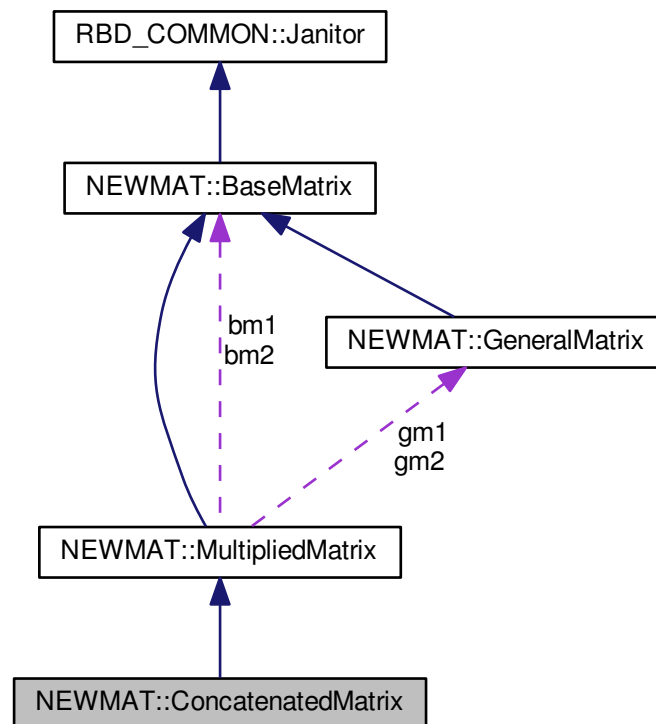
## 29.64 NEWMAT::ConcatenatedMatrix Class Reference

```
#include <newmat.h>
```

Inheritance diagram for NEWMAT::ConcatenatedMatrix:



Collaboration diagram for NEWMAT::ConcatenatedMatrix:



### Public Member Functions

- [~ConcatenatedMatrix \(\)](#)
- [MatrixBandWidth BandWidth \(\) const](#)
- [GeneralMatrix \\* Evaluate \(MatrixType mt=MatrixTypeUnSp\)](#)

### Protected Member Functions

- [ConcatenatedMatrix \(const BaseMatrix \\*bm1x, const BaseMatrix \\*bm2x\)](#)

### Friends

- class [BaseMatrix](#)
- class [GeneralMatrix](#)
- class [GenericMatrix](#)

### Additional Inherited Members

#### 29.64.1 Detailed Description

Definition at line 1254 of file newmat.h.



## 29.64.2 Constructor & Destructor Documentation

29.64.2.1 **NEWMAT::ConcatenatedMatrix::ConcatenatedMatrix** ( **const** **BaseMatrix** \* *bm1x*, **const** **BaseMatrix** \* *bm2x* )  
[inline], [protected]

Definition at line 1257 of file newmat.h.

29.64.2.2 **NEWMAT::ConcatenatedMatrix::~~ConcatenatedMatrix** ( ) [inline]

Definition at line 1264 of file newmat.h.

## 29.64.3 Member Function Documentation

29.64.3.1 **MatrixBandWidth** **ConcatenatedMatrix::BandWidth** ( ) **const** [virtual]

Reimplemented from [NEWMAT::MultipliedMatrix](#).

Definition at line 456 of file newmat4.cpp.

29.64.3.2 **GeneralMatrix** \* **ConcatenatedMatrix::Evaluate** ( **MatrixType** *mt* = **MatrixTypeUnSp** ) [virtual]

Reimplemented from [NEWMAT::MultipliedMatrix](#).

Reimplemented in [NEWMAT::StackedMatrix](#).

Definition at line 796 of file newmat7.cpp.

## 29.64.4 Friends And Related Function Documentation

29.64.4.1 **friend class** **BaseMatrix** [friend]

Definition at line 1260 of file newmat.h.

29.64.4.2 **friend class** **GeneralMatrix** [friend]

Definition at line 1261 of file newmat.h.

29.64.4.3 **friend class** **GenericMatrix** [friend]

Definition at line 1262 of file newmat.h.

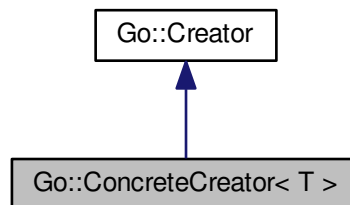
The documentation for this class was generated from the following files:

- [newmat/include/newmat.h](#)
- [newmat/src/newmat4.cpp](#)
- [newmat/src/newmat7.cpp](#)

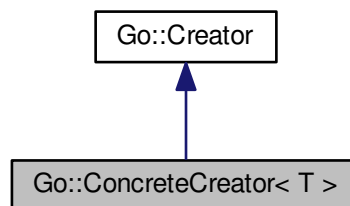
## 29.65 Go::ConcreteCreator< T > Class Template Reference

```
#include <Factory.h>
```

Inheritance diagram for Go::ConcreteCreator< T >:



Collaboration diagram for Go::ConcreteCreator< T >:



### Public Member Functions

- [ConcreteCreator](#) ()
- [GeomObject \\* create](#) ()

#### 29.65.1 Detailed Description

```
template<class T>
class Go::ConcreteCreator< T >
```

This is the concrete [Creator](#) class for generating GeomObject-derived classes. The class is only intended for internal use by the [Factory](#) class. The user should not have to worry about it.

Definition at line 64 of file Factory.h.

## 29.65.2 Constructor & Destructor Documentation

29.65.2.1 `template<class T> Go::ConcreteCreator< T >::ConcreteCreator ( ) [inline]`

Definition at line 67 of file `Factory.h`.

## 29.65.3 Member Function Documentation

29.65.3.1 `template<class T> GeomObject* Go::ConcreteCreator< T >::create ( ) [inline],[virtual]`

Implements [Go::Creator](#).

Definition at line 68 of file `Factory.h`.

The documentation for this class was generated from the following file:

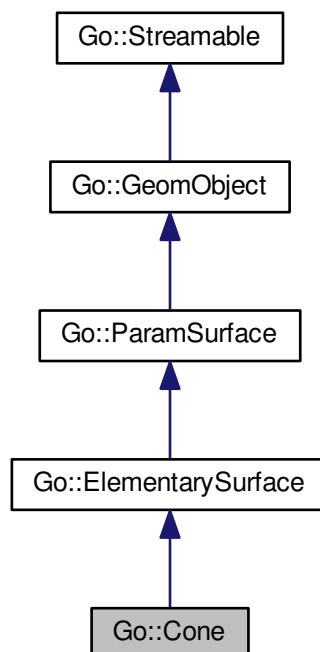
- `gtools-core/include/GoTools/geometry/Factory.h`

## 29.66 Go::Cone Class Reference

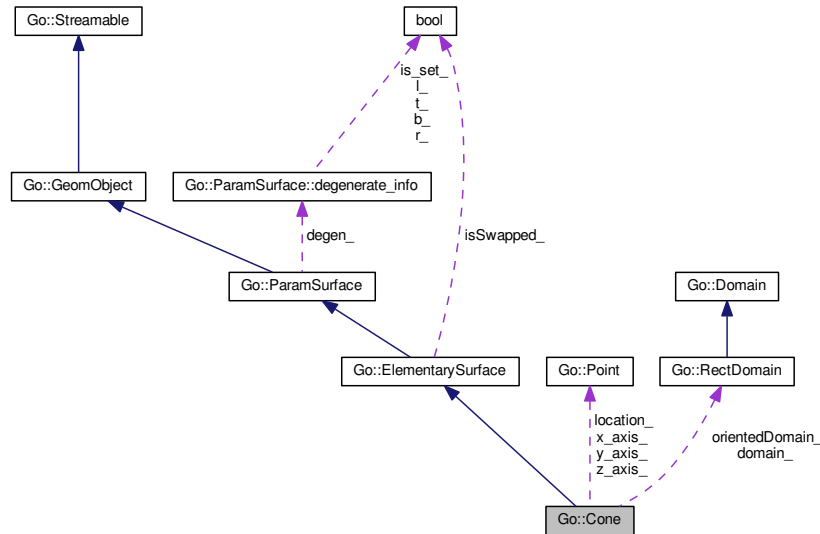
Class that represents a cone. It is a subclass of [ElementarySurface](#), and thus has a parametrization and is non-selfintersecting.

```
#include <Cone.h>
```

Inheritance diagram for `Go::Cone`:



Collaboration diagram for Go::Cone:



## Public Member Functions

- [Cone](#) ()
- [Cone](#) (double radius, [Point](#) location, [Point](#) z\_axis, [Point](#) x\_axis, double cone\_angle, bool isSwapped=false)
- virtual [~Cone](#) ()
  - Virtual destructor - ensures safe inheritance.*
- virtual void [read](#) (std::istream &is)
- virtual void [write](#) (std::ostream &os) const
- virtual int [dimension](#) () const
  - Return the dimension of the space in which the object lies (usually 2 or 3)*
- virtual [ClassType](#) [instanceType](#) () const
  - Return the class type identifier of a given, derived instance of [GeomObject](#).*
- virtual [BoundingBox](#) [boundingBox](#) () const
  - Return empty box if infinite cone.*
- virtual [Cone](#) \* [clone](#) () const
- const [RectDomain](#) & [parameterDomain](#) () const
- std::vector< [CurveLoop](#) > [allBoundaryLoops](#) (double degenerate\_epsilon=DEFAULT\_SPACE\_EPSILON) const
- [DirectionCone](#) [normalCone](#) () const
- [DirectionCone](#) [tangentCone](#) (bool pardir\_is\_u) const
- void [point](#) ([Point](#) &pt, double upar, double vpar) const
- void [point](#) (std::vector< [Point](#) > &pts, double upar, double vpar, int derivs, bool u\_from\_right=true, bool v\_from\_right=true, double resolution=1.0e-12) const
- void [normal](#) ([Point](#) &n, double upar, double vpar) const
- std::vector< shared\_ptr< [ParamCurve](#) > > [constParamCurves](#) (double parameter, bool pardir\_is\_u) const
- shared\_ptr< [ParamCurve](#) > [constParamCurve](#) (double parameter, bool pardir\_is\_u, double from, double to) const
- std::vector< shared\_ptr< [ParamSurface](#) > > [subSurfaces](#) (double from\_upar, double from\_vpar, double to\_upar, double to\_vpar, double fuzzy=DEFAULT\_PARAMETER\_EPSILON) const
- double [nextSegmentVal](#) (int dir, double par, bool forward, double tol) const

- void `closestPoint` (`const Point &pt`, `double &clo_u`, `double &clo_v`, `Point &clo_pt`, `double &clo_dist`, `double epsilon`, `const RectDomain *domain_of_interest=NULL`, `double *seed=0`) `const`
- void `closestBoundaryPoint` (`const Point &pt`, `double &clo_u`, `double &clo_v`, `Point &clo_pt`, `double &clo_dist`, `double epsilon`, `const RectDomain *rd=NULL`, `double *seed=0`) `const`
- void `getBoundaryInfo` (`Point &pt1`, `Point &pt2`, `double epsilon`, `SplineCurve *&cv`, `SplineCurve *&crosscv`, `double knot_tol=1e-05`) `const`
- `bool isDegenerate` (`bool &b`, `bool &r`, `bool &t`, `bool &l`, `double tolerance`) `const`
- virtual void `getDegenerateCorners` (`std::vector< Point > &deg_corners`, `double tol`) `const`  
*Check for paralell and anti paralell partial derivatives in surface corners.*
- `double getRadius` () `const`  
*Radius at the location.*
- `Point getLocation` () `const`  
*Point on cone axis.*
- void `getCoordinateAxes` (`Point &x_axis`, `Point &y_axis`, `Point &z_axis`) `const`  
*Local coordinate axes. The z\_axis corresponds to the cone axis.*
- `double getConeAngle` () `const`  
*The opening angle of the cone.*
- void `setParameterBounds` (`double from_upar`, `double from_vpar`, `double to_upar`, `double to_vpar`)  
*Limit the cone surface by limiting the parameter domain.*
- void `setParamBoundsU` (`double from_upar`, `double to_upar`)
- virtual void `setParamBoundsV` (`double from_vpar`, `double to_vpar`)
- `bool isBounded` () `const`
- `shared_ptr< Circle > getCircle` (`double par`) `const`
- `bool isClosed` (`bool &closed_dir_u`, `bool &closed_dir_v`) `const`  
*Check if the surface is closed.*
- `Cone * subSurface` (`double from_upar`, `double from_vpar`, `double to_upar`, `double to_vpar`, `double fuzzy=D←EFAULT_PARAMETER_EPSILON`) `const`  
*Return the part of the cone surface limited by the parameter bounds.*
- virtual `SplineSurface * geometrySurface` () `const`  
*Create a SplineSurface representation of the cone.*
- virtual `SplineSurface * createSplineSurface` () `const`  
*Create a SplineSurface representation of the cone.*
- `shared_ptr< Line > getLine` (`double upar`) `const`
- `shared_ptr< ElementaryCurve > getElementaryParamCurve` (`ElementaryCurve *space_crv`, `double tol`, `const Point *start_par_pt=NULL`, `const Point *end_par_pt=NULL`) `const`
- virtual `bool isAxisRotational` (`Point &centre`, `Point &axis`, `Point &vec`, `double &angle`)

### Static Public Member Functions

- static `ClassType classType` ()

### Protected Member Functions

- void `setCoordinateAxes` ()

### Protected Attributes

- `double radius_`
- `Point location_`
- `Point z_axis_`
- `Point x_axis_`
- `Point y_axis_`
- `double cone_angle_`
- `RectDomain domain_`
- `RectDomain orientedDomain_`

### 29.66.1 Detailed Description

Class that represents a cone. It is a subclass of [ElementarySurface](#), and thus has a parametrization and is non-selfintersecting.

A [Cone](#) has a natural parametrization in terms of the angle  $u$  and distance  $v$ :  $\mathbf{p}(u, v) = \mathbf{C} + (R + v \tan \alpha)((\cos u) \mathbf{x} + (\sin u) \mathbf{y}) + v \mathbf{z}$ , where  $\mathbf{C}$  is a position vector,  $R$  is the radius,  $\mathbf{x}$ ,  $\mathbf{y}$  and  $\mathbf{z}$  are the (local) axes, and  $\alpha$  is the cone angle. The parametrization is bounded by:  $0 \leq u \leq 2\pi$ ,  $-\infty < v < \infty$ . The dimension is 3.

Definition at line 69 of file Cone.h.

### 29.66.2 Constructor & Destructor Documentation

#### 29.66.2.1 `Go::Cone::Cone( ) [inline]`

Default constructor. Constructs an uninitialized [Cone](#) which can only be assigned to or read into.

Definition at line 74 of file Cone.h.

#### 29.66.2.2 `Go::Cone::Cone( double radius, Point location, Point z_axis, Point x_axis, double cone_angle, bool isSwapped = false )`

Constructor. Input is the radius, the location, the direction of the z-axis and the (possibly approximate) x-axis. The local coordinate axes are normalized even if `z_axis` and/or `x_axis` are not unit vectors.

#### 29.66.2.3 `virtual Go::Cone::~~Cone( ) [virtual]`

Virtual destructor - ensures safe inheritance.

### 29.66.3 Member Function Documentation

#### 29.66.3.1 `std::vector<CurveLoop> Go::Cone::allBoundaryLoops( double degenerate_epsilon = DEFAULT_SPACE_EPSILON ) const [virtual]`

Returns the anticlockwise outer boundary loop of the surface, together with clockwise loops of any interior boundaries, such that the surface always is 'to the left of' the loops.

##### Parameters

|                                 |                                                            |
|---------------------------------|------------------------------------------------------------|
| <code>degenerate_epsilon</code> | edges whose length is smaller than this value are ignored. |
|---------------------------------|------------------------------------------------------------|

##### Returns

a vector containing `CurveLoops`. The first of these describe the outer boundary of the surface (clockwise), whereas the others describe boundaries of interior holes (clockwise).

Implements [Go::ParamSurface](#).

29.66.3.2 `virtual BoundingBox Go::Cone::boundingBox( ) const [virtual]`

Return empty box if infinite cone.

Implements [Go::GeomObject](#).

29.66.3.3 `static ClassType Go::Cone::classType( ) [inline],[static]`

Definition at line 101 of file Cone.h.

29.66.3.4 `virtual Cone* Go::Cone::clone( ) const [virtual]`

make a clone of this surface and return a pointer to it (user is responsible for clearing up memory afterwards).

Returns

pointer to cloned object

Implements [Go::ElementarySurface](#).

29.66.3.5 `void Go::Cone::closestBoundaryPoint( const Point &pt, double &clo_u, double &clo_v, Point &clo_pt, double &clo_dist, double epsilon, const RectDomain *rd=NULL, double *seed=0 ) const [virtual]`

Iterates to the closest point to pt on the boundary of the surface.

See also

[closestPoint\(\)](#)

Implements [Go::ParamSurface](#).

29.66.3.6 `void Go::Cone::closestPoint( const Point &pt, double &clo_u, double &clo_v, Point &clo_pt, double &clo_dist, double epsilon, const RectDomain *domain_of_interest=NULL, double *seed=0 ) const [virtual]`

Iterates to the closest point to pt on the surface.

Parameters

|                             |                                                                                                                              |
|-----------------------------|------------------------------------------------------------------------------------------------------------------------------|
| <i>pt</i>                   | the point to find the closest point to                                                                                       |
| <i>clo_u</i>                | u parameter of the closest point                                                                                             |
| <i>clo_v</i>                | v parameter of the closest point                                                                                             |
| <i>clo_pt</i>               | the geometric position of the closest point                                                                                  |
| <i>clo_dist</i>             | the distance between pt and clo_pt                                                                                           |
| <i>epsilon</i>              | parameter tolerance (will in any case not be higher than sqrt(machine_precision) x magnitude of solution)                    |
| <i>domain_of_interest</i>   | pointer to parameter domain in which to search for closest point. If a NULL pointer is used, the entire surface is searched. |
| <b>Generated by Doxygen</b> |                                                                                                                              |
| <i>seed</i>                 | pointer to parameter values where iteration starts.                                                                          |

Reimplemented from [Go::ParamSurface](#).

29.66.3.7 `shared_ptr<ParamCurve> Go::Cone::constParamCurve ( double parameter, bool pardir_is_u, double from, double to ) const`

29.66.3.8 `std::vector<shared_ptr<ParamCurve> > Go::Cone::constParamCurves ( double parameter, bool pardir_is_u ) const` [virtual]

Get the curve(s) obtained by intersecting the surface with one of its constant parameter curves. For surfaces without holes, this will be the parameter curve itself; for surfaces with interior holes this may be a collection of several, disjoint curves.

#### Parameters

|                    |                                                                                                                                              |
|--------------------|----------------------------------------------------------------------------------------------------------------------------------------------|
| <i>parameter</i>   | parameter value for the constant parameter (either u or v)                                                                                   |
| <i>pardir_is_u</i> | specify whether the <i>moving</i> parameter (as opposed to the <i>constant</i> parameter) is the first ('true') or the second ('false') one. |

#### Returns

a vector containing shared pointers to the obtained, newly constructed constant-parameter curves.

Implements [Go::ParamSurface](#).

29.66.3.9 `virtual SplineSurface* Go::Cone::createSplineSurface ( ) const` [virtual]

Create a [SplineSurface](#) representation of the cone.

Implements [Go::ElementarySurface](#).

29.66.3.10 `virtual int Go::Cone::dimension ( ) const` [virtual]

Return the dimension of the space in which the object lies (usually 2 or 3)

Implements [Go::GeomObject](#).

29.66.3.11 `virtual SplineSurface* Go::Cone::geometrySurface ( ) const` [virtual]

Create a [SplineSurface](#) representation of the cone.

Implements [Go::ElementarySurface](#).

29.66.3.12 `void Go::Cone::getBoundaryInfo ( Point & pt1, Point & pt2, double epsilon, SplineCurve *& cv, SplineCurve *& crosscv, double knot_tol=1e-05 ) const` [virtual]

Get the boundary curve segment between two points on the boundary, as well as the cross-tangent curve. If the given points are not positioned on the same boundary (within a certain tolerance), no curves will be created.



## Parameters

|                 |                                                                                                                                                                                                                                                                                                                                    |
|-----------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>pt1</i>      | the first point on the boundary, given by the user                                                                                                                                                                                                                                                                                 |
| <i>pt2</i>      | the second point on the boundary, given by the user                                                                                                                                                                                                                                                                                |
| <i>epsilon</i>  | the tolerance used when determining whether the given points are lying on a boundary, and if they do, whether they both lie on the <i>same</i> boundary.                                                                                                                                                                           |
| <i>cv</i>       | upon return, this will point to a newly created curve representing the boundary curve between 'pt1' and 'pt2'. The user assumes ownership of this object and is responsible for its deletion. No curve is created if the given points are not found to lie on the same boundary.                                                   |
| <i>crosscv</i>  | upon return, this will point to a newly created curve representing the cross-boundary curve between 'pt1' and 'pt2'. The user assumes ownership of this object and is responsible for its deletion. The direction is outwards from the surface. No curve is created if the given points are not found to lie on the same boundary. |
| <i>knot_tol</i> | tolerance used when working with the knot-vector, to specify how close a parameter value must be to a knot in order to be considered 'on' the knot.                                                                                                                                                                                |

Implements [Go::ParamSurface](#).

29.66.3.13 `shared_ptr<Circle> Go::Cone::getCircle ( double par ) const`

29.66.3.14 `double Go::Cone::getConeAngle ( ) const [inline]`

The opening angle of the cone.

Definition at line 192 of file Cone.h.

29.66.3.15 `void Go::Cone::getCoordinateAxes ( Point & x_axis, Point & y_axis, Point & z_axis ) const [inline]`

Local coordinate axes. The *z\_axis* corresponds to the cone axis.

Definition at line 184 of file Cone.h.

29.66.3.16 `virtual void Go::Cone::getDegenerateCorners ( std::vector< Point > & deg_corners, double tol ) const [virtual]`

Check for parallel and anti parallel partial derivatives in surface corners.

Implements [Go::ParamSurface](#).

29.66.3.17 `shared_ptr<ElementaryCurve> Go::Cone::getElementaryParamCurve ( ElementaryCurve * space_crv, double tol, const Point * start_par_pt = NULL, const Point * end_par_pt = NULL ) const [virtual]`

Fetch the parameter curve in the domain of the elementary surface corresponding to a given elementary curve in geometry space if this curve has a simpler elementary representation. Otherwise, nothing is returned

Reimplemented from [Go::ElementarySurface](#).

29.66.3.18 `shared_ptr<Line> Go::Cone::getLine ( double upar ) const`

29.66.3.19 `Point Go::Cone::getLocation ( ) const [inline]`

[Point](#) on cone axis.

Definition at line 180 of file Cone.h.

29.66.3.20 `double Go::Cone::getRadius ( ) const [inline]`

Radius at the location.

Definition at line 176 of file Cone.h.

29.66.3.21 `virtual ClassType Go::Cone::instanceType ( ) const [virtual]`

Return the class type identifier of a given, derived instance of [GeomObject](#).

Implements [Go::GeomObject](#).

29.66.3.22 `virtual bool Go::Cone::isAxisRotational ( Point & centre, Point & axis, Point & vec, double & angle ) [virtual]`

Check if the surface is axis rotational. Only true if a connection to an axis rotational elementary surface exist The axis and rotational angle is only specified if the surface is actually rotational

Reimplemented from [Go::ParamSurface](#).

29.66.3.23 `bool Go::Cone::isBounded ( ) const [virtual]`

Query if parametrization is bounded. Only the *v* parameter direction is queried. The *u* parameter is always bounded.

Returns

*true* if bounded, *false* otherwise

Reimplemented from [Go::ElementarySurface](#).

29.66.3.24 `bool Go::Cone::isClosed ( bool & closed_dir_u, bool & closed_dir_v ) const [virtual]`

Check if the surface is closed.

Reimplemented from [Go::ElementarySurface](#).

29.66.3.25 `bool Go::Cone::isDegenerate ( bool & b, bool & r, bool & t, bool & l, double tolerance ) const [virtual]`

The order of the edge indicators (bottom, right, top, left) matches the `edge_number` of `edgeCurve()`. Query whether any of the four boundary curves are degenerate (zero length) within a certain tolerance. In the below, we refer to 'u' as the first parameter and 'v' as the second.

## Parameters

|                  |                                                                                                    |
|------------------|----------------------------------------------------------------------------------------------------|
| <i>b</i>         | 'true' upon return of function if the boundary ( $v = v\_min$ ) is degenerate                      |
| <i>r</i>         | 'true' upon return of function if the boundary ( $v = v\_max$ ) is degenerate                      |
| <i>t</i>         | 'true' upon return of function if the boundary ( $u = u\_min$ ) is degenerate                      |
| <i>l</i>         | 'true' upon return of function if the boundary ( $u = u\_max$ ) is degenerate                      |
| <i>tolerance</i> | boundaries are considered degenerate if their length is shorter than this value, given by the user |

## Returns

'true' if at least one boundary curve was found to be degenerate, 'false' otherwise.

Reimplemented from [Go::ParamSurface](#).

**29.66.3.26** `double Go::Cone::nextSegmentVal ( int dir, double par, bool forward, double tol ) const` [virtual]

Determine the parameter value of the start of the 'next segment' from a parameter value, along a given parameter direction. A 'segment' is here defined as a parameter interval in which there will be no discontinuities in derivatives or other artifacts. For spline objects, a segment will typically be the interval between two consecutive, non-coincident knots.

## Parameters

|                |                                                                                                |
|----------------|------------------------------------------------------------------------------------------------|
| <i>dir</i>     | the parameter direction in which we search for the next segment (0 or 1)                       |
| <i>par</i>     | the parameter value starting from which we search for the start value of the next segment      |
| <i>forward</i> | define whether we shall move forward ('true') or backwards when searching along this parameter |
| <i>tol</i>     | tolerance used for determining whether the 'par' is already located on the next segment value  |

## Returns

the value of the start value of the next segment (or the end of the previous segment, if we are moving backwards...)

Implements [Go::ParamSurface](#).

**29.66.3.27** `void Go::Cone::normal ( Point & n, double upar, double vpar ) const` [virtual]

Evaluates the surface normal for a given parameter pair

## Parameters

|             |                                                      |
|-------------|------------------------------------------------------|
| <i>n</i>    | the computed normal will be written to this variable |
| <i>upar</i> | the first parameter                                  |
| <i>vpar</i> | the second parameter                                 |

Implements [Go::ParamSurface](#).

29.66.3.28 **DirectionCone** Go::Cone::normalCone ( ) const [virtual]

Creates a [DirectionCone](#) covering all normals to this surface.

Returns

a [DirectionCone](#) (not necessarily the smallest) containing all normals to this surface.

Implements [Go::ParamSurface](#).

29.66.3.29 **const RectDomain&** Go::Cone::parameterDomain ( ) const [virtual]

Return the parameter domain of the surface. This may be a simple rectangular domain ([RectDomain](#)) or any other subclass of [Domain](#) (such as [GoCurveBoundedDomain](#), found in the `sisl_dependent` module).

Returns

a [Domain](#) object describing the parametric domain of the surface

Implements [Go::ParamSurface](#).

29.66.3.30 **void** Go::Cone::point ( **Point & pt**, **double upar**, **double vpar** ) const [virtual]

Evaluates the surface's position for a given parameter pair.

Parameters

|             |                                              |
|-------------|----------------------------------------------|
| <i>pt</i>   | the result of the evaluation is written here |
| <i>upar</i> | the first parameter                          |
| <i>vpar</i> | the second parameter                         |

Implements [Go::ParamSurface](#).

29.66.3.31 **void** Go::Cone::point ( **std::vector< Point > & pts**, **double upar**, **double vpar**, **int derivs**, **bool u\_from\_right = true**, **bool v\_from\_right = true**, **double resolution = 1.0e-12** ) const [virtual]

Evaluates the surface's position and a certain number of derivatives for a given parameter pair.

Parameters

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|-------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>pts</i>  | the vector containing the evaluated values. Its size must be set by the user prior to calling this function, and should be equal to $(derivs+1) * (derivs+2) / 2$ . Upon completion of the function, its first entry is the surface's position at the given parameter pair. Then, if 'derivs' > 0, the two next entries will be the surface tangents along the first and second parameter direction. The next three entries are the second- and cross derivatives, in the order (du2, dudv, dv2), and similar for even higher derivatives. |
| <i>upar</i> | the first parameter                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| <i>vpar</i> | the second parameter                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |

## Parameters

|                     |                                                                                                                                                                  |
|---------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>derivs</i>       | number of requested derivatives                                                                                                                                  |
| <i>u_from_right</i> | specify whether derivatives along the first parameter are to be calculated from the right ('true', default) or from the left ('false')                           |
| <i>v_from_right</i> | specify whether derivatives along the second parameter are to be calculated from the right ('true', default) or from the left ('false')                          |
| <i>resolution</i>   | tolerance used when determining whether parameters are located at special values of the parameter domain (in particular; knot values in case of spline objects). |

Implements [Go::ParamSurface](#).

**29.66.3.32** virtual void Go::Cone::read ( std::istream & *is* ) [virtual]

read object from stream

## Parameters

|           |                                  |
|-----------|----------------------------------|
| <i>is</i> | stream from which object is read |
|-----------|----------------------------------|

Implements [Go::Streamable](#).

**29.66.3.33** void Go::Cone::setCoordinateAxes ( ) [protected]

**29.66.3.34** void Go::Cone::setParamBoundsU ( double *from\_upar*, double *to\_upar* )

Set parameter bounds in the *u* direction. *u* is the "angular" direction.

**29.66.3.35** virtual void Go::Cone::setParamBoundsV ( double *from\_vpar*, double *to\_vpar* ) [virtual]

Set parameter bounds in the *v* direction. *v* is the "linear" direction.

**29.66.3.36** void Go::Cone::setParameterBounds ( double *from\_upar*, double *from\_vpar*, double *to\_upar*, double *to\_vpar* ) [virtual]

Limit the cone surface by limiting the parameter domain.

Implements [Go::ElementarySurface](#).

**29.66.3.37** Cone\* Go::Cone::subSurface ( double *from\_upar*, double *from\_vpar*, double *to\_upar*, double *to\_vpar*, double *fuzzy* = DEFAULT\_PARAMETER\_EPSILON ) const

Return the part of the cone surface limited by the parameter bounds.

**29.66.3.38** std::vector<shared\_ptr<ParamSurface>> Go::Cone::subSurfaces ( double *from\_upar*, double *from\_vpar*, double *to\_upar*, double *to\_vpar*, double *fuzzy* = DEFAULT\_PARAMETER\_EPSILON ) const [virtual]

Get the surface(s) obtained by cropping the parameter domain of this surface between given values for the first and second parameter. In general, for surfaces with no interior holes, the result will be *one* surface; however, for surfaces with interior holes, the result might be *several disjoint* surfaces.

## Parameters

|                  |                                                                       |
|------------------|-----------------------------------------------------------------------|
| <i>from_upar</i> | lower value for the first parameter in the subdomain                  |
| <i>from_vpar</i> | lower value for the second parameter in the subdomain                 |
| <i>to_upar</i>   | upper value for the first parameter in the subdomain                  |
| <i>to_vpar</i>   | upper value for the second parameter in the subdomain                 |
| <i>fuzzy</i>     | tolerance used when determining intersection with interior boundaries |

## Returns

a vector contained shared pointers to the obtained, newly constructed sub-surfaces.

Implements [Go::ParamSurface](#).

**29.66.3.39** `DirectionCone Go::Cone::tangentCone ( bool pardir_is_u ) const` [virtual]

Creates a [DirectionCone](#) covering all tangents to this surface along a given parameter direction.

## Parameters

|                    |                                                                                                                                                                                                  |
|--------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>pardir_is_u</i> | if 'true', then the <a href="#">DirectionCone</a> will be defined on basis of the surface's tangents along the first parameter direction. Otherwise the second parameter direction will be used. |
|--------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

## Returns

a [DirectionCone](#) (not necessarily the smallest) containing all tangents to this surface along the specified parameter direction.

Implements [Go::ParamSurface](#).

**29.66.3.40** `virtual void Go::Cone::write ( std::ostream & os ) const` [virtual]

write object to stream

## Parameters

|           |                                   |
|-----------|-----------------------------------|
| <i>os</i> | stream to which object is written |
|-----------|-----------------------------------|

Implements [Go::Streamable](#).

**29.66.4 Member Data Documentation**

**29.66.4.1** `double Go::Cone::cone_angle_` [protected]

Definition at line 245 of file Cone.h.

**29.66.4.2** `RectDomain Go::Cone::domain_` [protected]

Definition at line 247 of file Cone.h.

**29.66.4.3** `Point Go::Cone::location_` [protected]

Definition at line 241 of file Cone.h.

**29.66.4.4** `RectDomain Go::Cone::orientedDomain_` [mutable], [protected]

Definition at line 248 of file Cone.h.

**29.66.4.5** `double Go::Cone::radius_` [protected]

Definition at line 240 of file Cone.h.

**29.66.4.6** `Point Go::Cone::x_axis_` [protected]

Definition at line 243 of file Cone.h.

**29.66.4.7** `Point Go::Cone::y_axis_` [protected]

Definition at line 244 of file Cone.h.

**29.66.4.8** `Point Go::Cone::z_axis_` [protected]

Definition at line 242 of file Cone.h.

The documentation for this class was generated from the following file:

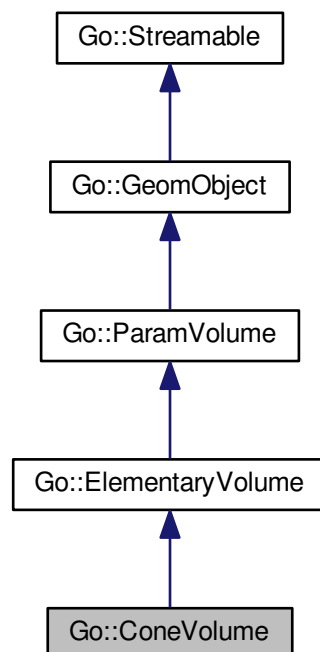
- `gotools-core/include/GoTools/geometry/Cone.h`

## 29.67 Go::ConeVolume Class Reference

Class that represents a solid cone. It is a subclass of [ElementaryVolume](#), and has a natural parametrization in terms of a radius  $u$ , an angle  $v$ , and distance  $w$ :  $\mathbf{p}(u, v, w) = \mathbf{C} + u (R + w \tan \alpha)((\cos v) \mathbf{x} + (\sin v) \mathbf{y}) + w \mathbf{z}$ , where  $\mathbf{C}$  is the cone apex,  $R$  is the radius when  $w = 0$ ,  $\alpha$  is the cone angle, and  $\mathbf{x}$ ,  $\mathbf{y}$  and  $\mathbf{z}$  are the (local) axes. The parametrization is bounded by:  $0 \leq u \leq 1$ ,  $0 \leq v \leq 2\pi$ ,  $-\infty < w < \infty$ . The dimension is 3.

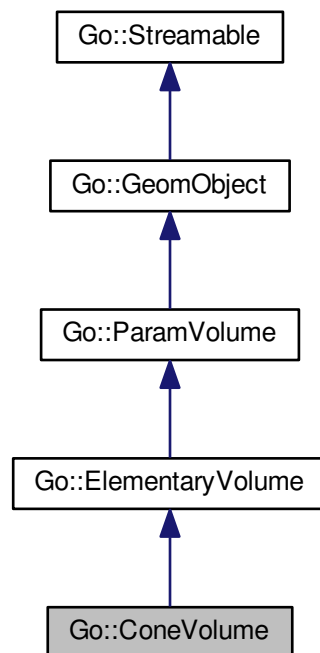
```
#include <ConeVolume.h>
```

Inheritance diagram for Go::ConeVolume:





Collaboration diagram for Go::ConeVolume:



## Public Member Functions

- [ConeVolume](#) ()
- [ConeVolume](#) (double radius, [Point](#) location, [Point](#) z\_axis, [Point](#) x\_axis, double cone\_angle)
- virtual [~ConeVolume](#) ()
  - Virtual destructor - ensures safe inheritance.*
- virtual void [read](#) (std::istream &is)
- virtual void [write](#) (std::ostream &os) const
- virtual int [dimension](#) () const
  - Return the dimension of the space in which the object lies (usually 2 or 3)*
- virtual [ClassType](#) [instanceType](#) () const
  - Return the class type identifier of a given, derived instance of [GeomObject](#).*
- virtual [BoundingBox](#) [boundingBox](#) () const
  - Return the object's bounding box.*
- virtual [ConeVolume](#) \* [clone](#) () const
- [DirectionCone](#) [tangentCone](#) (int paddir) const
- const [Array](#)< double, 6 > [parameterSpan](#) () const
- void [point](#) ([Point](#) &pt, double upar, double vpar, double wpar) const
- void [point](#) (std::vector< [Point](#) > &pts, double upar, double vpar, double wpar, int derivs, bool u\_from\_↔right=true, bool v\_from\_right=true, bool w\_from\_right=true, double resolution=1.0e-12) const
- double [nextSegmentVal](#) (int dir, double par, bool forward, double tol) const
- void [closestPoint](#) (const [Point](#) &pt, double &clo\_u, double &clo\_v, double &clo\_w, [Point](#) &clo\_pt, double &clo\_↔\_dist, double epsilon, double \*seed=0) const
- void [reverseParameterDirection](#) (int paddir)

- void [swapParameterDirection](#) (int paddir1, int paddir2)
- `std::vector< shared_ptr< ParamSurface > > getAllBoundarySurfaces () const`  
*Fetch all boundary surfaces corresponding to the volume.*
- virtual void [translate](#) (const [Point](#) &vec)  
*Translate.*
- [SplineVolume](#) \* [geometryVolume](#) () const  
*Make a NURBS representation of the object.*
- void [setParameters](#) (double from\_par, double to\_par, int paddir)  
*Restrict the size of the torus in one parameter direction.*
- void [useCentreDegen](#) ()  
*swept linearly and shrinked or extended*
- void [useCornerDegen](#) ()  
*occur if the cone tip is included in the volume.*

### Static Public Member Functions

- static [ClassType](#) [classType](#) ()

#### 29.67.1 Detailed Description

Class that represents a solid cone. It is a subclass of [ElementaryVolume](#), and has a natural parametrization in terms of a radius  $u$ , an angle  $v$ , and distance  $w$ :  $\mathbf{p}(u, v, w) = \mathbf{C} + u (R + w \tan \alpha)((\cos v) \mathbf{x} + (\sin v) \mathbf{y}) + w \mathbf{z}$ , where  $\mathbf{C}$  is the cone apex,  $R$  is the radius when  $w = 0$ ,  $\alpha$  is the cone angle, and  $\mathbf{x}$ ,  $\mathbf{y}$  and  $\mathbf{z}$  are the (local) axes. The parametrization is bounded by:  $0 \leq u \leq 1$ ,  $0 \leq v \leq 2\pi$ ,  $-\infty < w < \infty$ . The dimension is 3.

Definition at line 65 of file [ConeVolume.h](#).

#### 29.67.2 Constructor & Destructor Documentation

##### 29.67.2.1 `Go::ConeVolume::ConeVolume ( ) [inline]`

Default constructor. Constructs an uninitialized [ConeVolume](#) which can only be assigned to or read into.

Definition at line 70 of file [ConeVolume.h](#).

##### 29.67.2.2 `Go::ConeVolume::ConeVolume ( double radius, Point location, Point z_axis, Point x_axis, double cone_angle )`

Constructor. Input is the radius, the location, the direction of the z-axis and the (possibly approximate) x-axis.

##### 29.67.2.3 `virtual Go::ConeVolume::~~ConeVolume ( ) [virtual]`

Virtual destructor - ensures safe inheritance.

### 29.67.3 Member Function Documentation

29.67.3.1 virtual **BoundingBox** Go::ConeVolume::boundingBox ( ) const [virtual]

Return the object's bounding box.

Implements [Go::GeomObject](#).

29.67.3.2 static **ClassType** Go::ConeVolume::classType ( ) [inline],[static]

Definition at line 96 of file ConeVolume.h.

29.67.3.3 virtual **ConeVolume\*** Go::ConeVolume::clone ( ) const [virtual]

make a clone of this volume and return a pointer to it (user is responsible for clearing up memory afterwards).

#### Returns

pointer to cloned object

Implements [Go::ElementaryVolume](#).

29.67.3.4 void Go::ConeVolume::closestPoint ( const **Point** & *pt*, double & *clo\_u*, double & *clo\_v*, double & *clo\_w*, **Point** & *clo\_pt*, double & *clo\_dist*, double *epsilon*, double \* *seed* = 0 ) const [virtual]

Iterates to the closest point to *pt* on the volume.

#### Parameters

|                 |                                                                                                                      |
|-----------------|----------------------------------------------------------------------------------------------------------------------|
| <i>pt</i>       | the point to find the closest point to                                                                               |
| <i>clo_u</i>    | u parameter of the closest point                                                                                     |
| <i>clo_v</i>    | v parameter of the closest point                                                                                     |
| <i>clo_w</i>    | w parameter of the closest point                                                                                     |
| <i>clo_pt</i>   | the geometric position of the closest point                                                                          |
| <i>clo_dist</i> | the distance between <i>pt</i> and <i>clo_pt</i>                                                                     |
| <i>epsilon</i>  | parameter tolerance (will in any case not be higher than $\sqrt{\text{machine\_precision}}$ x magnitude of solution) |
| <i>seed</i>     | pointer to parameter values where iteration starts.                                                                  |

Implements [Go::ParamVolume](#).

29.67.3.5 virtual int Go::ConeVolume::dimension ( ) const [virtual]

Return the dimension of the space in which the object lies (usually 2 or 3)

Implements [Go::GeomObject](#).

29.67.3.6 **SplineVolume\*** `Go::ConeVolume::geometryVolume ( ) const` [virtual]

Make a NURBS representation of the object.

Implements [Go::ElementaryVolume](#).

29.67.3.7 `std::vector<shared_ptr<ParamSurface>>` `Go::ConeVolume::getAllBoundarySurfaces ( ) const`  
[virtual]

Fetch all boundary surfaces corresponding to the volume.

Implements [Go::ParamVolume](#).

29.67.3.8 **virtual ClassType** `Go::ConeVolume::instanceType ( ) const` [virtual]

Return the class type identifier of a given, derived instance of [GeomObject](#).

Implements [Go::GeomObject](#).

29.67.3.9 **double** `Go::ConeVolume::nextSegmentVal ( int dir, double par, bool forward, double tol ) const`  
[virtual]

Determine the parameter value of the start of the 'next segment' from a parameter value, along a given parameter direction. A 'segment' is here defined as a parameter interval in which there will be no discontinuities in derivatives or other artifacts. For spline objects, a segment will typically be the interval between two consecutive, non-coincident knots.

#### Parameters

|                |                                                                                                      |
|----------------|------------------------------------------------------------------------------------------------------|
| <i>dir</i>     | the parameter direction in which we search for the next segment (0, 1 or 2)                          |
| <i>par</i>     | the parameter value starting from which we search for the start value of the next segment            |
| <i>forward</i> | define whether we shall move forward ('true') or backwards when searching along this parameter       |
| <i>tol</i>     | tolerance used for determining whether the 'par' is already located <i>on</i> the next segment value |

#### Returns

the value of the start value of the next segment (or the end of the previous segment, if we are moving backwards...)

Implements [Go::ParamVolume](#).

29.67.3.10 **const Array<double,6>** `Go::ConeVolume::parameterSpan ( ) const` [virtual]

Return the parameter domain of the volume. This is an array containing the start and end parameter values. The spline's parameter *i* has its start value at the array position (2*i*) and its end parameter value at the array position (2*i*+1)

#### Returns

An array describing the parametric domain of the volume

Implements [Go::ParamVolume](#).

29.67.3.11 `void Go::ConeVolume::point ( Point & pt, double upar, double vpar, double wpar ) const` `[virtual]`

Evaluates the volume's position for a given parameter triple.

#### Parameters

|             |                                              |
|-------------|----------------------------------------------|
| <i>pt</i>   | the result of the evaluation is written here |
| <i>upar</i> | the first parameter                          |
| <i>vpar</i> | the second parameter                         |
| <i>wpar</i> | the third parameter                          |

Implements [Go::ParamVolume](#).

29.67.3.12 `void Go::ConeVolume::point ( std::vector< Point > & pts, double upar, double vpar, double wpar, int derivs, bool u_from_right = true, bool v_from_right = true, bool w_from_right = true, double resolution = 1.0e-12 ) const` `[virtual]`

Evaluates the volume's position and a certain number of derivatives for a given parameter triple.

#### Parameters

|                     |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|---------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>pts</i>          | the vector containing the evaluated values. Its size must be set by the user prior to calling this function. Upon completion of the function, its first entry is the volume's position at the given parameter pair. Then, if ' <i>derivs</i> ' > 0, the two next entries will be the volume tangents along the first, second and third parameter direction. The next three entries are the second- and cross derivatives, in the order ( <i>du</i> <sup>2</sup> , <i>dudv</i> , <i>dudw</i> , <i>dv</i> <sup>2</sup> , <i>dvdw</i> , <i>dw</i> <sup>2</sup> ), and similar for even higher derivatives. |
| <i>upar</i>         | the first parameter                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| <i>vpar</i>         | the second parameter                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| <i>wpar</i>         | the third parameter                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| <i>derivs</i>       | number of requested derivatives                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| <i>u_from_right</i> | specify whether derivatives along the first parameter are to be calculated from the right ('true', default) or from the left ('false')                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| <i>v_from_right</i> | specify whether derivatives along the second parameter are to be calculated from the right ('true', default) or from the left ('false')                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| <i>w_from_right</i> | specify whether derivatives along the third parameter are to be calculated from the right ('true', default) or from the left ('false')                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| <i>resolution</i>   | tolerance used when determining whether parameters are located at special values of the parameter domain (in particular; knot values in case of spline objects).                                                                                                                                                                                                                                                                                                                                                                                                                                        |

Implements [Go::ParamVolume](#).

29.67.3.13 `virtual void Go::ConeVolume::read ( std::istream & is )` `[virtual]`

read object from stream

#### Parameters

|           |                                  |
|-----------|----------------------------------|
| <i>is</i> | stream from which object is read |
|-----------|----------------------------------|

Implements [Go::Streamable](#).

29.67.3.14 void `Go::ConeVolume::reverseParameterDirection ( int pardir )` [virtual]

Reverses the direction of the basis in input direction.

#### Parameters

|               |                                      |
|---------------|--------------------------------------|
| <i>pardir</i> | which parameter direction to reverse |
|---------------|--------------------------------------|

Implements [Go::ParamVolume](#).

29.67.3.15 void `Go::ConeVolume::setParameters ( double from_par, double to_par, int pardir )`

Restrict the size of the torus in one parameter direction.

29.67.3.16 void `Go::ConeVolume::swapParameterDirection ( int pardir1, int pardir2 )` [virtual]

Swaps two parameter directions

#### Parameters

|                |                                                           |
|----------------|-----------------------------------------------------------|
| <i>pardir1</i> | One of the parameter directions (0, 1 or 2) to be swapped |
| <i>pardir2</i> | The other parameter direction (0, 1 or 2) to be swapped   |

Implements [Go::ParamVolume](#).

29.67.3.17 **DirectionCone** `Go::ConeVolume::tangentCone ( int pardir ) const` [virtual]

Creates a [DirectionCone](#) covering all tangents to this volume along a given parameter direction.

#### Parameters

|               |                                                                                                                                                                                                                                     |
|---------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>pardir</i> | if 1, the <a href="#">DirectionCone</a> will be defined on basis of the volume's tangents along the first parameter direction. If 2, the second parameter direction will be used. If 3, the third parameter direction will be used. |
|---------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

#### Returns

a [DirectionCone](#) (not necessarily the smallest) containing all tangents to this volume along the specified parameter direction.

Implements [Go::ParamVolume](#).

29.67.3.18 virtual void Go::ConeVolume::translate ( const Point & vec ) [virtual]

Translate.

Implements [Go::ParamVolume](#).

29.67.3.19 void Go::ConeVolume::useCentreDegen ( ) [inline]

swept linearly and shrunked or extended

A NURBS representation will be a disc with degeneracy in the centre

Definition at line 154 of file ConeVolume.h.

29.67.3.20 void Go::ConeVolume::useCornerDegen ( ) [inline]

occur if the cone tip is included in the volume.

A NURBS representation will be a disc with degenerate corners swept linearly and shrunked or extended. Note that extra degeneracies

Definition at line 160 of file ConeVolume.h.

29.67.3.21 virtual void Go::ConeVolume::write ( std::ostream & os ) const [virtual]

write object to stream

Parameters

|    |                                   |
|----|-----------------------------------|
| os | stream to which object is written |
|----|-----------------------------------|

Implements [Go::Streamable](#).

The documentation for this class was generated from the following file:

- [trivariate/include/GoTools/trivariate/ConeVolume.h](#)

## 29.68 Go::ConnectionFunctor Class Reference

```
#include <IntersectionPoolUtils.h>
```

### Public Member Functions

- [ConnectionFunctor](#) (const std::vector< shared\_ptr< [IntersectionPoint](#) > > &ipoints)
- [bool operator\(\)](#) (int a, int b)

### 29.68.1 Detailed Description

Definition at line 407 of file IntersectionPoolUtils.h.

### 29.68.2 Constructor & Destructor Documentation

29.68.2.1 `Go::ConnectionFuncutor::ConnectionFuncutor ( const std::vector< shared_ptr< IntersectionPoint > > & ipoints ) [inline]`

Definition at line 411 of file IntersectionPoolUtils.h.

### 29.68.3 Member Function Documentation

29.68.3.1 `bool Go::ConnectionFuncutor::operator() ( int a, int b ) [inline]`

Definition at line 414 of file IntersectionPoolUtils.h.

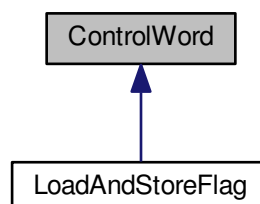
The documentation for this class was generated from the following file:

- intersections/include/GoTools/intersections/[IntersectionPoolUtils.h](#)

## 29.69 ControlWord Class Reference

```
#include <controlw.h>
```

Inheritance diagram for ControlWord:





## Public Member Functions

- [ControlWord \(\)](#)
- [ControlWord \(int i\)](#)
- [ControlWord operator\\* \(ControlWord i\) const](#)
- [void operator\\*= \(ControlWord i\)](#)
- [ControlWord operator+ \(ControlWord i\) const](#)
- [void operator+= \(ControlWord i\)](#)
- [ControlWord operator- \(ControlWord i\) const](#)
- [void operator-= \(ControlWord i\)](#)
- [bool operator>= \(ControlWord i\) const](#)
- [bool operator<= \(ControlWord i\) const](#)
- [ControlWord operator^ \(ControlWord i\) const](#)
- [ControlWord operator~ \(\) const](#)
- [int operator+ \(\) const](#)
- [int operator! \(\) const](#)

## Protected Attributes

- [int cw](#)

### 29.69.1 Detailed Description

Definition at line 9 of file controlw.h.

### 29.69.2 Constructor & Destructor Documentation

29.69.2.1 [ControlWord::ControlWord \( \)](#) `[inline]`

Definition at line 14 of file controlw.h.

29.69.2.2 [ControlWord::ControlWord \( int i \)](#) `[inline]`

Definition at line 15 of file controlw.h.

### 29.69.3 Member Function Documentation

29.69.3.1 [int ControlWord::operator! \( \) const](#) `[inline]`

Definition at line 43 of file controlw.h.

29.69.3.2 [ControlWord ControlWord::operator\\* \( ControlWord i \) const](#) `[inline]`

Definition at line 18 of file controlw.h.

29.69.3.3 `void ControlWord::operator*=( ControlWord i ) [inline]`

Definition at line 20 of file controlw.h.

29.69.3.4 `ControlWord ControlWord::operator+( ControlWord i ) const [inline]`

Definition at line 23 of file controlw.h.

29.69.3.5 `int ControlWord::operator+( ) const [inline]`

Definition at line 42 of file controlw.h.

29.69.3.6 `void ControlWord::operator+=( ControlWord i ) [inline]`

Definition at line 25 of file controlw.h.

29.69.3.7 `ControlWord ControlWord::operator-( ControlWord i ) const [inline]`

Definition at line 28 of file controlw.h.

29.69.3.8 `void ControlWord::operator-=( ControlWord i ) [inline]`

Definition at line 30 of file controlw.h.

29.69.3.9 `bool ControlWord::operator<=( ControlWord i ) const [inline]`

Definition at line 34 of file controlw.h.

29.69.3.10 `bool ControlWord::operator>=( ControlWord i ) const [inline]`

Definition at line 33 of file controlw.h.

29.69.3.11 `ControlWord ControlWord::operator^( ControlWord i ) const [inline]`

Definition at line 37 of file controlw.h.

29.69.3.12 `ControlWord ControlWord::operator~( ) const [inline]`

Definition at line 39 of file controlw.h.

### 29.69.4 Member Data Documentation

29.69.4.1 `int ControlWord::cw` [protected]

Definition at line 12 of file `controlw.h`.

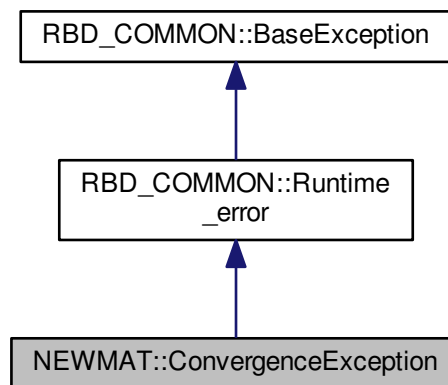
The documentation for this class was generated from the following file:

- `newmat/include/controlw.h`

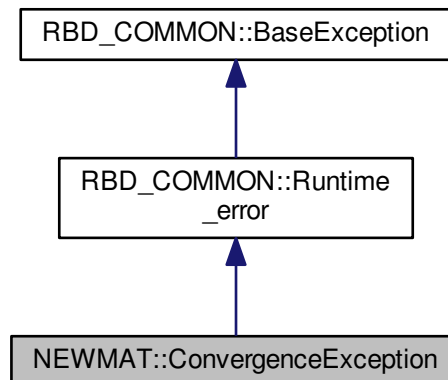
## 29.70 NEWMAT::ConvergenceException Class Reference

```
#include <newmat.h>
```

Inheritance diagram for NEWMAT::ConvergenceException:



Collaboration diagram for NEWMAT::ConvergenceException:



## Public Member Functions

- [ConvergenceException](#) (const [GeneralMatrix](#) &A)
- [ConvergenceException](#) (const char \*c)

## Static Public Attributes

- static unsigned long [Select](#)

## Additional Inherited Members

### 29.70.1 Detailed Description

Definition at line 1602 of file newmat.h.

### 29.70.2 Constructor & Destructor Documentation

#### 29.70.2.1 [ConvergenceException::ConvergenceException](#) ( const [GeneralMatrix](#) & A )

Definition at line 63 of file newmatex.cpp.

#### 29.70.2.2 [ConvergenceException::ConvergenceException](#) ( const char \* c )

Definition at line 72 of file newmatex.cpp.

### 29.70.3 Member Data Documentation

#### 29.70.3.1 unsigned long [ConvergenceException::Select](#) [static]

Definition at line 1605 of file newmat.h.

The documentation for this class was generated from the following files:

- newmat/include/[newmat.h](#)
- newmat/src/[newmatex.cpp](#)

## 29.71 [Go::CoordinateSystem](#)< Dim > Class Template Reference

Defines a Cartesian coordinate system.

```
#include <CoordinateSystem.h>
```

## Public Types

- typedef [MatrixXD](#)< double, Dim > [Matrix](#)
- typedef [Array](#)< double, Dim > [Vector](#)

## Public Member Functions

- [CoordinateSystem](#) ()
- [CoordinateSystem](#) (const [Matrix](#) &rot, const [Vector](#) &trans)
- [Vector](#) operator\* (const [Vector](#) &v)
- [CoordinateSystem](#) operator\* (const [CoordinateSystem](#) &c)
- [Matrix](#) & rot ()  
*Get the rotation matrix.*
- const [Matrix](#) & rot () const  
*Get the rotation matrix. Const version.*
- [Vector](#) & tr ()  
*Get the translation vector.*
- const [Vector](#) & tr () const  
*Get the translation vector. Const version.*

### 29.71.1 Detailed Description

```
template<int Dim>
class Go::CoordinateSystem< Dim >
```

Defines a Cartesian coordinate system.

Definition at line 52 of file [CoordinateSystem.h](#).

### 29.71.2 Member Typedef Documentation

29.71.2.1 `template<int Dim> typedef MatrixXD<double, Dim> Go::CoordinateSystem< Dim >::Matrix`

Definition at line 55 of file [CoordinateSystem.h](#).

29.71.2.2 `template<int Dim> typedef Array<double, Dim> Go::CoordinateSystem< Dim >::Vector`

Definition at line 56 of file [CoordinateSystem.h](#).

### 29.71.3 Constructor & Destructor Documentation

29.71.3.1 `template<int Dim> Go::CoordinateSystem< Dim >::CoordinateSystem ( ) [inline]`

The default constructor creates an object with an identity rotation matrix and zero translation vector.

Definition at line 60 of file [CoordinateSystem.h](#).

29.71.3.2 `template<int Dim> Go::CoordinateSystem< Dim >::CoordinateSystem ( const Matrix & rot, const Vector & trans ) [inline]`

Creates a [CoordinateSystem](#) with the specified rotation matrix and translation vector.

Definition at line 68 of file [CoordinateSystem.h](#).

#### 29.71.4 Member Function Documentation

29.71.4.1 `template<int Dim> Vector Go::CoordinateSystem< Dim >::operator* ( const Vector & v ) [inline]`

Get the 'global' coordinates of a vector 'v' expressed in the [CoordinateSystem](#).

Definition at line 74 of file [CoordinateSystem.h](#).

29.71.4.2 `template<int Dim> CoordinateSystem Go::CoordinateSystem< Dim >::operator* ( const CoordinateSystem< Dim > & c ) [inline]`

Create a [CoordinateSystem](#) that is a composition of 'this' [CoordinateSystem](#) and 'c'.

Definition at line 81 of file [CoordinateSystem.h](#).

29.71.4.3 `template<int Dim> Matrix& Go::CoordinateSystem< Dim >::rot ( ) [inline]`

Get the rotation matrix.

Definition at line 89 of file [CoordinateSystem.h](#).

29.71.4.4 `template<int Dim> const Matrix& Go::CoordinateSystem< Dim >::rot ( ) const [inline]`

Get the rotation matrix. Const version.

Definition at line 92 of file [CoordinateSystem.h](#).

29.71.4.5 `template<int Dim> Vector& Go::CoordinateSystem< Dim >::tr ( ) [inline]`

Get the translation vector.

Definition at line 95 of file [CoordinateSystem.h](#).

29.71.4.6 `template<int Dim> const Vector& Go::CoordinateSystem< Dim >::tr ( ) const [inline]`

Get the translation vector. Const version.

Definition at line 98 of file [CoordinateSystem.h](#).

The documentation for this class was generated from the following file:

- [gotools-core/include/GoTools/utils/CoordinateSystem.h](#)

## 29.72 Go::CPUclock Class Reference

A class for measuring CPU time in programs.

```
#include <CPUclock.h>
```

### Public Member Functions

- [CPUclock \(\)](#)  
*The constructor initializes the time of last call.*
- [double getInterval \(\)](#)
- [void initTime \(\)](#)
- [double getTime \(\)](#)  
*Returns time in seconds since January 1, 1970.*

### 29.72.1 Detailed Description

A class for measuring CPU time in programs.

Definition at line 49 of file CPUclock.h.

### 29.72.2 Constructor & Destructor Documentation

#### 29.72.2.1 Go::CPUclock::CPUclock ( )

The constructor initializes the time of last call.

### 29.72.3 Member Function Documentation

#### 29.72.3.1 double Go::CPUclock::getInterval ( )

The increase in time (seconds) since last call to [getInterval\(\)](#) or to [initTime \(\)](#).

#### 29.72.3.2 double Go::CPUclock::getTime ( )

Returns time in seconds since January 1, 1970.

#### 29.72.3.3 void Go::CPUclock::initTime ( ) `[inline]`

Start measuring the length of an interval (typically called prior to the computational job to be measured).

Definition at line 69 of file CPUclock.h.

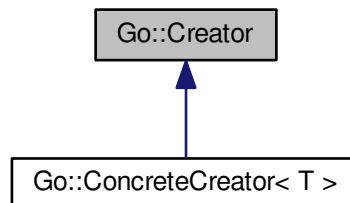
The documentation for this class was generated from the following file:

- [gotools-core/include/GoTools/utills/CPUclock.h](#)

## 29.73 Go::Creator Class Reference

```
#include <Factory.h>
```

Inheritance diagram for Go::Creator:



### Public Member Functions

- virtual [GeomObject](#) \* `create` ()=0
- virtual `~Creator` ()

#### 29.73.1 Detailed Description

Abstract base class for Creators. These are policy classes for generating objects derived from [GeomObject](#). This class is only intended for internal use by the [Factory](#) class. The user should not have to worry about it.

Definition at line 53 of file [Factory.h](#).

#### 29.73.2 Constructor & Destructor Documentation

29.73.2.1 `virtual Go::Creator::~~Creator ( )` [[inline](#)], [[virtual](#)]

Definition at line 57 of file [Factory.h](#).

#### 29.73.3 Member Function Documentation

29.73.3.1 `virtual GeomObject* Go::Creator::create ( )` [[pure virtual](#)]

Implemented in [Go::ConcreteCreator< T >](#).

The documentation for this class was generated from the following file:

- [gotools-core/include/GoTools/geometry/Factory.h](#)



## 29.74 Go::CrossesValue Class Reference

IntersectionPoolUtils. predicate for STL function.

```
#include <IntersectionPoolUtils.h>
```

### Public Types

- typedef `const shared_ptr< IntersectionLink > argument_type`
- typedef `bool result_type`

### Public Member Functions

- `CrossesValue` (int *par\_dir*, double *value*, double *eps*)
- `bool operator()` (`const shared_ptr< IntersectionLink > &l`) `const`

#### 29.74.1 Detailed Description

IntersectionPoolUtils. predicate for STL function.

Definition at line 128 of file IntersectionPoolUtils.h.

#### 29.74.2 Member Typedef Documentation

29.74.2.1 `typedef const shared_ptr<IntersectionLink> Go::CrossesValue::argument_type`

Definition at line 157 of file IntersectionPoolUtils.h.

29.74.2.2 `typedef bool Go::CrossesValue::result_type`

Definition at line 158 of file IntersectionPoolUtils.h.

#### 29.74.3 Constructor & Destructor Documentation

29.74.3.1 `Go::CrossesValue::CrossesValue ( int par_dir, double value, double eps )` `[inline]`

Definition at line 132 of file IntersectionPoolUtils.h.

#### 29.74.4 Member Function Documentation

29.74.4.1 `bool Go::CrossesValue::operator() ( const shared_ptr< IntersectionLink > &l ) const` `[inline]`

Definition at line 134 of file IntersectionPoolUtils.h.

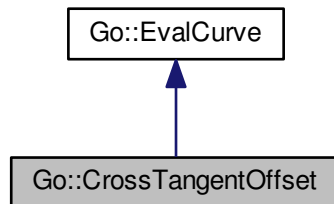
The documentation for this class was generated from the following file:

- intersections/include/GoTools/intersections/[IntersectionPoolUtils.h](#)

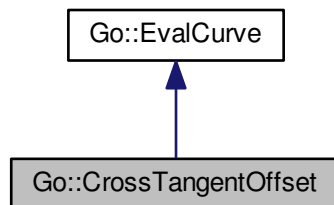
## 29.75 Go::CrossTangentOffset Class Reference

```
#include <CrossTangentOffset.h>
```

Inheritance diagram for Go::CrossTangentOffset:



Collaboration diagram for Go::CrossTangentOffset:



### Public Member Functions

- [CrossTangentOffset](#) (shared\_ptr< [SplineCurve](#) > &poscurve, shared\_ptr< [SplineCurve](#) > &tangcv1, shared\_ptr< [SplineCurve](#) > &tangcv2, shared\_ptr< [SplineCurve](#) > &blend1, shared\_ptr< [SplineCurve](#) > &blend2)
- virtual [~CrossTangentOffset](#) ()  
*virtual destructor enables safe inheritance*
- virtual [Point](#) [eval](#) (double t) const
- virtual void [eval](#) (double t, int n, [Point](#) der[]) const
- virtual [double](#) [start](#) () const
- virtual [double](#) [end](#) () const
- virtual [int](#) [dim](#) () const
- virtual [bool](#) [approximationOK](#) (double par, [Point](#) approxpos, [double](#) tol1, [double](#) tol2) const

### 29.75.1 Detailed Description

This curve represent an offset curve from a given space curve, along a direction obtained by blending two 'cross-tangent curves', and with an offset distance which is a linear function interpolating the cross-tangent length at the start and end of the curve.

Definition at line 57 of file CrossTangentOffset.h.

### 29.75.2 Constructor & Destructor Documentation

**29.75.2.1** `Go::CrossTangentOffset::CrossTangentOffset ( shared_ptr< SplineCurve > & poscurve, shared_ptr< SplineCurve > & tangcv1, shared_ptr< SplineCurve > & tangcv2, shared_ptr< SplineCurve > & blend1, shared_ptr< SplineCurve > & blend2 )`

Constructor, taking a curve from which we take the offset, and four other curves used to calculate the offset direction and magnitude. Two of these curves are the cross-tangent curves and the two other are blending functions (dimension 1). To find the offset direction at a given point, the two cross-tangent curves are evaluated at the specified parameter, multiplied by their respective blending functions and added together. The offset length is computed by linearly interpolating the length of this blended cross-tangent at the start and end parameter of the curve.

#### Parameters

|                 |                                                          |
|-----------------|----------------------------------------------------------|
| <i>poscurve</i> | the curve from which we take the offset                  |
| <i>tangcv1</i>  | the first cross-tangent curve                            |
| <i>tangcv2</i>  | the second cross-tangent curve                           |
| <i>blend1</i>   | the blending function for the first cross-tangent curve  |
| <i>blend2</i>   | the blending function for the second cross-tangent curve |

**29.75.2.2** `virtual Go::CrossTangentOffset::~~CrossTangentOffset ( ) [virtual]`

virtual destructor enables safe inheritance

### 29.75.3 Member Function Documentation

**29.75.3.1** `virtual bool Go::CrossTangentOffset::approximationOK ( double par, Point approxpos, double tol1, double tol2 ) const [virtual]`

Inherited from [EvalCurve::approximationOK\(\)](#). For this class, both tolerances are used.

#### Parameters

|                  |                                                                                                                                                                                                                                                                                                                   |
|------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>par</i>       | the parameter at which to check the curve                                                                                                                                                                                                                                                                         |
| <i>approxpos</i> | the position we want to check whether or not the curve approximates for parameter 'par'.                                                                                                                                                                                                                          |
| <i>tol1</i>      | spatial approximation tolerance. If the evaluated position is outside this tolerance, 'false' is returned. If it inside this tolerance "by far", 'true' is returned. Otherwise, 'tol2' is taken into account.                                                                                                     |
| <i>tol2</i>      | This tolerance is taken into account when the evaluated point is within 'tol1' from 'approxpos', but not convincingly so. In that case this tolerance is used to check whether the evaluated cross tangent lies in the plane spanned by the tangent curves at this point. (It is thus an <i>angle</i> tolerance). |

**Returns**

'true' if the curve approximates the point at the parameter, 'false' otherwise.

Implements [Go::EvalCurve](#).

**29.75.3.2** `virtual int Go::CrossTangentOffset::dim ( ) const [virtual]`

Get the dimension of the space in which the curve lies.

**Returns**

the space dimension of the curve.

Implements [Go::EvalCurve](#).

**29.75.3.3** `virtual double Go::CrossTangentOffset::end ( ) const [virtual]`

Get the end parameter of the curve.

**Returns**

the end parameter of the curve.

Implements [Go::EvalCurve](#).

**29.75.3.4** `virtual Point Go::CrossTangentOffset::eval ( double t ) const [virtual]`

Evaluate a point on the curve for a given parameter

**Parameters**

|          |                                                |
|----------|------------------------------------------------|
| <i>t</i> | the parameter for which to evaluate the curve. |
|----------|------------------------------------------------|

**Returns**

the evaluated point

Implements [Go::EvalCurve](#).

**29.75.3.5** `virtual void Go::CrossTangentOffset::eval ( double t, int n, Point der[] ) const [virtual]`

Evaluate a point and a certain number of derivatives on the curve for a given parameter.

**Parameters**

|          |                                                |
|----------|------------------------------------------------|
| <i>t</i> | the parameter for which to evaluate the curve. |
| <i>n</i> | the number of derivatives (0 or more)          |

## Return values

|            |                                                                                                                                                                                                                                                                                                                                                                      |
|------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>der</i> | pointer to an array of Points where the result will be written. The position will be stored first, then the first derivative (tangent), then the second, etc.. <b>NB:</b> For most (all) derived classes of 'EvalCurve', the implementation actually only supports the computation of one derivative, i.e. if $n > 1$ , only one derivative will be computed anyway. |
|------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

Implements [Go::EvalCurve](#).

29.75.3.6 `virtual double Go::CrossTangentOffset::start ( ) const [virtual]`

Get the start parameter of the curve.

## Returns

the start parameter of the curve.

Implements [Go::EvalCurve](#).

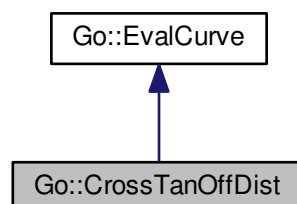
The documentation for this class was generated from the following file:

- [gotools-core/include/GoTools/creators/CrossTangentOffset.h](#)

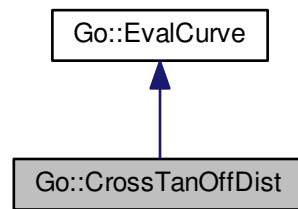
## 29.76 Go::CrossTanOffDist Class Reference

```
#include <CrossTanOffDist.h>
```

Inheritance diagram for Go::CrossTanOffDist:



Collaboration diagram for Go::CrossTanOffDist:



## Public Member Functions

- `CrossTanOffDist` (`shared_ptr< SplineCurve > &poscurve`, `shared_ptr< SplineCurve > &tangcv1`, `shared_ptr< SplineCurve > &tangcv2`, `shared_ptr< SplineCurve > &blend1`, `shared_ptr< SplineCurve > &blend2`, `shared_ptr< SplineCurve > &opposite1`, `shared_ptr< SplineCurve > &opposite2`, `double factor`)
- virtual `~CrossTanOffDist` ()
  - Destructor.*
- virtual `Point eval` (`double t`) `const`
- virtual void `eval` (`double t`, `int n`, `Point der[ ]`) `const`
- virtual `double start` () `const`
- virtual `double end` () `const`
- virtual `int dim` () `const`
- virtual `bool approximationOK` (`double par`, `Point approxpos`, `double tol1`, `double tol2`) `const`

### 29.76.1 Detailed Description

This class defines an evaluator-based offset-curve. We're blending between a set of curves seen as an evaluator based curve.

Definition at line 55 of file CrossTanOffDist.h.

### 29.76.2 Constructor & Destructor Documentation

- 29.76.2.1 `Go::CrossTanOffDist::CrossTanOffDist` (`shared_ptr< SplineCurve > &poscurve`, `shared_ptr< SplineCurve > &tangcv1`, `shared_ptr< SplineCurve > &tangcv2`, `shared_ptr< SplineCurve > &blend1`, `shared_ptr< SplineCurve > &blend2`, `shared_ptr< SplineCurve > &opposite1`, `shared_ptr< SplineCurve > &opposite2`, `double factor`)

Constructor

Parameters

|                 |                                     |
|-----------------|-------------------------------------|
| <i>poscurve</i> | the curve to offset from.           |
| <i>tangcv1</i>  | tangent curve along poscurve.       |
| <i>tangcv2</i>  | cross tangent curve along poscurve. |

## Parameters

|                  |                                                                                                                                                         |
|------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>blend1</i>    | 1-dimensional blending function for tangcv1.                                                                                                            |
| <i>blend2</i>    | 1-dimensional blending function for tangcv2.                                                                                                            |
| <i>opposite1</i> | space curve corresponding to poscurve, parametrized in the opposite direction.                                                                          |
| <i>opposite2</i> | space curve corresponding to poscurve, parametrized in the opposite direction. May be equal to opposite1.                                               |
| <i>factor</i>    | if != 1.0 the poscurve, opposite1 & opposite2 will be used to define the opposite curve, so as to minimize differences in parametrization along the cv. |

29.76.2.2 virtual Go::CrossTanOffDist::~~CrossTanOffDist ( ) [virtual]

Destructor.

### 29.76.3 Member Function Documentation

29.76.3.1 virtual bool Go::CrossTanOffDist::approximationOK ( double *par*, Point *approxpos*, double *tol1*, double *tol2* ) const [virtual]

Check if the curve, evaluated at a given parameter, approximates a given position within a given tolerance.

## Parameters

|                  |                                                                                          |
|------------------|------------------------------------------------------------------------------------------|
| <i>par</i>       | the parameter at which to check the curve                                                |
| <i>approxpos</i> | the position we want to check whether or not the curve approximates for parameter 'par'. |
| <i>tol1</i>      | approximation tolerance,                                                                 |
| <i>tol2</i>      | another approximation tolerance (its use is defined by some of the derived classes.      |

## Returns

'true' if the curve approximates the point at the parameter, 'false' otherwise.

Implements [Go::EvalCurve](#).

29.76.3.2 virtual int Go::CrossTanOffDist::dim ( ) const [virtual]

Get the dimension of the space in which the curve lies.

## Returns

the space dimension of the curve.

Implements [Go::EvalCurve](#).

29.76.3.3 `virtual double Go::CrossTanOffDist::end ( ) const [virtual]`

Get the end parameter of the curve.

#### Returns

the end parameter of the curve.

Implements [Go::EvalCurve](#).

29.76.3.4 `virtual Point Go::CrossTanOffDist::eval ( double t ) const [virtual]`

Evaluate a point on the curve for a given parameter

#### Parameters

|          |                                                |
|----------|------------------------------------------------|
| <i>t</i> | the parameter for which to evaluate the curve. |
|----------|------------------------------------------------|

#### Returns

the evaluated point

Implements [Go::EvalCurve](#).

29.76.3.5 `virtual void Go::CrossTanOffDist::eval ( double t, int n, Point der[] ) const [virtual]`

Evaluate a point and a certain number of derivatives on the curve for a given parameter.

#### Parameters

|          |                                                |
|----------|------------------------------------------------|
| <i>t</i> | the parameter for which to evaluate the curve. |
| <i>n</i> | the number of derivatives (0 or more)          |

#### Return values

|            |                                                                                                                                                                                                                                                                                                                                                                                        |
|------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>der</i> | pointer to an array of Points where the result will be written. The position will be stored first, then the first derivative (tangent), then the second, etc.. <b>NB:</b> For most (all) derived classes of ' <a href="#">EvalCurve</a> ', the implementation actually only supports the computation of one derivative, i.e. if $n > 1$ , only one derivative will be computed anyway. |
|------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

Implements [Go::EvalCurve](#).

29.76.3.6 `virtual double Go::CrossTanOffDist::start ( ) const [virtual]`

Get the start parameter of the curve.



**Returns**

the start parameter of the curve.

Implements [Go::EvalCurve](#).

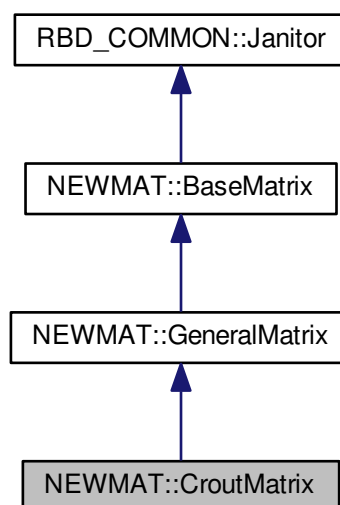
The documentation for this class was generated from the following file:

- [gotools-core/include/GoTools/creators/CrossTanOffDist.h](#)

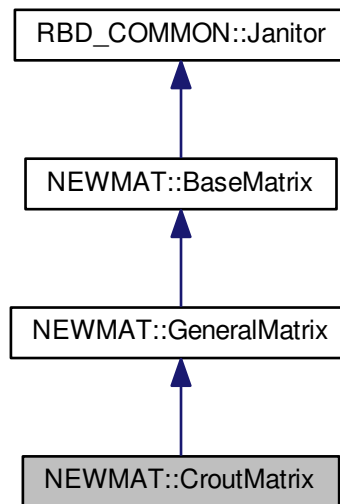
## 29.77 NEWMAT::CroutMatrix Class Reference

```
#include <newmat.h>
```

Inheritance diagram for NEWMAT::CroutMatrix:



Collaboration diagram for NEWMAT::CroutMatrix:



## Public Member Functions

- [CroutMatrix](#) ([const BaseMatrix &](#))
- [MatrixType](#) [Type](#) () [const](#)
- [void lubksb](#) ([Real \\*](#), [int=0](#))
- [~CroutMatrix](#) ()
- [GeneralMatrix \\* MakeSolver](#) ()
- [LogAndSign](#) [LogDeterminant](#) () [const](#)
- [void Solver](#) ([MatrixColX &](#), [const MatrixColX &](#))
- [void GetRow](#) ([MatrixRowCol &](#))
- [void GetCol](#) ([MatrixRowCol &](#))
- [void GetCol](#) ([MatrixColX &c](#))
- [void operator=](#) ([const BaseMatrix &](#))
- [void operator=](#) ([const CroutMatrix &m](#))
- [void CleanUp](#) ()
- [bool IsEqual](#) ([const GeneralMatrix &](#)) [const](#)
- [bool IsSingular](#) () [const](#)

## Additional Inherited Members

### 29.77.1 Detailed Description

Definition at line 862 of file newmat.h.

## 29.77.2 Constructor & Destructor Documentation

### 29.77.2.1 CroutMatrix::CroutMatrix ( const BaseMatrix & m )

Definition at line 141 of file newmat4.cpp.

### 29.77.2.2 CroutMatrix::~CroutMatrix ( )

Definition at line 155 of file newmat4.cpp.

## 29.77.3 Member Function Documentation

### 29.77.3.1 void CroutMatrix::CleanUp ( ) [virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 838 of file newmat4.cpp.

### 29.77.3.2 void CroutMatrix::GetCol ( MatrixRowCol & ) [virtual]

Implements [NEWMAT::GeneralMatrix](#).

Definition at line 286 of file newmatex.cpp.

### 29.77.3.3 void NEWMAT::CroutMatrix::GetCol ( MatrixCoIX & c ) [inline],[virtual]

Implements [NEWMAT::GeneralMatrix](#).

Definition at line 878 of file newmat.h.

### 29.77.3.4 void CroutMatrix::GetRow ( MatrixRowCol & ) [virtual]

Implements [NEWMAT::GeneralMatrix](#).

Definition at line 284 of file newmatex.cpp.

### 29.77.3.5 bool CroutMatrix::IsEqual ( const GeneralMatrix & A ) const [virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 1011 of file newmat7.cpp.

### 29.77.3.6 bool NEWMAT::CroutMatrix::IsSingular ( ) const [inline]

Definition at line 883 of file newmat.h.

29.77.3.7 **LogAndSign** `CroutMatrix::LogDeterminant ( ) const` [virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 694 of file `newmat8.cpp`.

29.77.3.8 `void CroutMatrix::lubksb ( Real * B, int mini = 0 )`

Definition at line 97 of file `newmat8.cpp`.

29.77.3.9 **GeneralMatrix\*** `NEWMAT::CroutMatrix::MakeSolver ( )` [inline],[virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 873 of file `newmat.h`.

29.77.3.10 `void CroutMatrix::operator= ( const BaseMatrix & )`

Definition at line 288 of file `newmatex.cpp`.

29.77.3.11 `void NEWMAT::CroutMatrix::operator= ( const CroutMatrix & m )` [inline]

Definition at line 880 of file `newmat.h`.

29.77.3.12 `void CroutMatrix::Solver ( MatrixColX & mcout, const MatrixColX & mcin )` [virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 38 of file `newmat7.cpp`.

29.77.3.13 **MatrixType** `CroutMatrix::Type ( ) const` [virtual]

Implements [NEWMAT::GeneralMatrix](#).

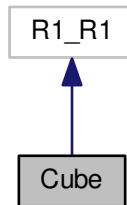
Definition at line 392 of file `newmat4.cpp`.

The documentation for this class was generated from the following files:

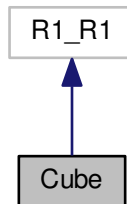
- `newmat/include/newmat.h`
- `newmat/src/newmat4.cpp`
- `newmat/src/newmat7.cpp`
- `newmat/src/newmat8.cpp`
- `newmat/src/newmatex.cpp`

## 29.78 Cube Class Reference

Inheritance diagram for Cube:



Collaboration diagram for Cube:



### 29.78.1 Detailed Description

Definition at line 19 of file `sl_ex.cpp`.

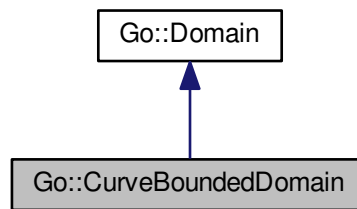
The documentation for this class was generated from the following file:

- [newmat/app/sl\\_ex.cpp](#)

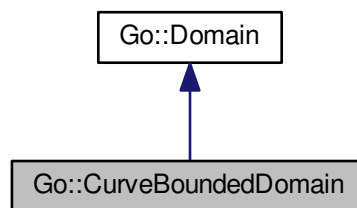
## 29.79 Go::CurveBoundedDomain Class Reference

```
#include <CurveBoundedDomain.h>
```

Inheritance diagram for Go::CurveBoundedDomain:



Collaboration diagram for Go::CurveBoundedDomain:



## Public Member Functions

- [CurveBoundedDomain](#) ()  
*Constructor generating an empty domain.*
- [CurveBoundedDomain](#) (std::vector< shared\_ptr< [CurveLoop](#) > > loops)
- [CurveBoundedDomain](#) (shared\_ptr< [CurveLoop](#) > ccw\_loop)
- virtual [~CurveBoundedDomain](#) ()  
*Virtual destructor, enables safe inheritance.*
- virtual [bool isInDomain](#) (const Array< double, 2 > &point, double tolerance) const
- virtual [int isInDomain2](#) (const Array< double, 2 > &point, double tolerance) const
- virtual [bool isOnBoundary](#) (const Array< double, 2 > &point, double tolerance) const
- [bool isOnCorner](#) (const Array< double, 2 > &point, double tolerance) const
- virtual [void closestInDomain](#) (const Array< double, 2 > &point, Array< double, 2 > &clo\_pt, double tolerance) const
- virtual [void closestOnBoundary](#) (const Array< double, 2 > &point, Array< double, 2 > &clo\_bd\_pt, double tolerance) const
- [void getInternalPoint](#) (double &upar, double &vpar) const
- [RectDomain containingDomain](#) () const
- [void clipWithDomain](#) (int pdir, double parval, double tolerance, shared\_ptr< [ParamSurface](#) > srf, std::vector< shared\_ptr< [CurveOnSurface](#) > > &trim\_pieces) const

- void [findPcurveInsideSegments](#) (const [SplineCurve](#) &curve, double tolerance, std::vector< double > &params\_start\_end\_interval) const
- void [findPcurveInsideSegments](#) (const [SplineCurve](#) &curve, double tolerance, std::vector< double > &params\_start\_end\_interval, std::vector< double > &boundary\_params, std::vector< int > &boundary\_loops, std::vector< int > &boundary\_curves) const
- void [getInsideIntervals](#) (int pdir, double parval1, double parval2, double tolerance, std::vector< std::pair< double, double > > &insideInts) const
- int [positionPointInDomain](#) (int pdir, double parval1, double parval2, double tolerance) const

### 29.79.1 Detailed Description

A 2D parameter domain represented by its boundaries. The domain is not necessarily connected.

Definition at line 56 of file [CurveBoundedDomain.h](#).

### 29.79.2 Constructor & Destructor Documentation

#### 29.79.2.1 Go::CurveBoundedDomain::CurveBoundedDomain ( ) [inline]

Constructor generating an empty domain.

Definition at line 60 of file [CurveBoundedDomain.h](#).

#### 29.79.2.2 Go::CurveBoundedDomain::CurveBoundedDomain ( std::vector< shared\_ptr< [CurveLoop](#) > > loops )

The curve loop must contain either 2D [ParamCurve](#) objects or ?D [CurveOnSurface](#) objects. One of the loops (the first, preferably) should be ccw, the rest cw. Constructor generating a domain specified by one or more 2D [CurveLoops](#).

##### Parameters

|              |                                                                                                                                                                                                                                                      |
|--------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>loops</i> | The loops defining the domain to be created. The domain is defined as the union of the interior of these loops. The <a href="#">CurveLoop</a> s must contain either 2D <a href="#">ParamCurve</a> objects XD <a href="#">CurveOnSurface</a> objects. |
|--------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

#### 29.79.2.3 Go::CurveBoundedDomain::CurveBoundedDomain ( shared\_ptr< [CurveLoop](#) > ccw\_loop )

Constructor generating a (connected) domain specified by a [CurveLoop](#)

##### Parameters

|                 |                                                                                                                          |
|-----------------|--------------------------------------------------------------------------------------------------------------------------|
| <i>ccw_loop</i> | the curve loop specifying the domain. The domain will represent the interior of this (supposedly counterclockwise loop). |
|-----------------|--------------------------------------------------------------------------------------------------------------------------|

#### 29.79.2.4 virtual Go::CurveBoundedDomain::~~CurveBoundedDomain ( ) [virtual]

Virtual destructor, enables safe inheritance.

### 29.79.3 Member Function Documentation

29.79.3.1 `void Go::CurveBoundedDomain::clipWithDomain ( int pardir, double parval, double tolerance, shared_ptr< ParamSurface > srf, std::vector< shared_ptr< CurveOnSurface > > & trim_pieces ) const`

On the 2D parameter plane, consider the (iso)curve defined by fixing one of the parameters at a certain value. Now, pick those parts of this curve that are covered by 'this' [CurveBoundedDomain](#). Then, lift these (parameter) curves into 3D space by means of a [SplineSurface](#). This function will return the [CurveOnSurface](#) s defined by this procedure.

#### Parameters

|                  |                                                                                                                                                                                                                                                                                                                     |
|------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>pardir</i>    | this parameter specifies which of the two parameter directions that should be 'free' (the other will be fixed). 'pardir' should take the value of '1' or '2'. A value of '1' means that the <i>first</i> parameter direction will be free, while '2' means that the <i>second</i> parameter direction will be free. |
| <i>parval</i>    | the parameter value of the fixed parameter                                                                                                                                                                                                                                                                          |
| <i>tolerance</i> | the tolerance when determining which parts of the isoparameter curve are located inside 'this' <a href="#">CurveBoundedDomain</a> .                                                                                                                                                                                 |
| <i>srf</i>       | The surface used to 'lift' the resulting isoparameter curve intervals                                                                                                                                                                                                                                               |

#### Return values

|                    |                                                                   |
|--------------------|-------------------------------------------------------------------|
| <i>trim_pieces</i> | vector containing the resulting <a href="#">CurveOnSurface</a> s. |
|--------------------|-------------------------------------------------------------------|

29.79.3.2 `virtual void Go::CurveBoundedDomain::closestInDomain ( const Array< double, 2 > & point, Array< double, 2 > & clo_pt, double tolerance ) const` [virtual]

Find the parameter pair contained in the domain that is closest (using Euclidean distance in  $R^2$ ) to a given parameter pair. If the given parameter pair is already in the domain, then the answer is obviously that parameter pair

#### Parameters

|              |                                                                                                                                    |
|--------------|------------------------------------------------------------------------------------------------------------------------------------|
| <i>point</i> | the (u,v) parameter pair that we want to find the closest parameter pair to <i>inside</i> the <a href="#">CurveBoundedDomain</a> . |
|--------------|------------------------------------------------------------------------------------------------------------------------------------|

#### Return values

|               |                                        |
|---------------|----------------------------------------|
| <i>clo_pt</i> | the resulting closest parameter point. |
|---------------|----------------------------------------|

#### Parameters

|                  |                                                                                      |
|------------------|--------------------------------------------------------------------------------------|
| <i>tolerance</i> | the tolerance used in defining whether the given point is already inside the domain. |
|------------------|--------------------------------------------------------------------------------------|

Implements [Go::Domain](#).



29.79.3.3 `virtual void Go::CurveBoundedDomain::closestOnBoundary ( const Array< double, 2 > & point, Array< double, 2 > & clo_bd_pt, double tolerance ) const [virtual]`

Find the parameter pair on the boundary of the domain that is closest (using Euclidean distance in  $R^2$ ) to a given parameter pair. If the point is already considered *on* the boundary, then the answer is obviously the same point.

#### Parameters

|              |                                                                                                                                     |
|--------------|-------------------------------------------------------------------------------------------------------------------------------------|
| <i>point</i> | the (u,v) parameter pair that we want to find to closest parameter pair to <i>on</i> the <a href="#">CurveBoundedDomain</a> border. |
|--------------|-------------------------------------------------------------------------------------------------------------------------------------|

#### Return values

|                  |                                  |
|------------------|----------------------------------|
| <i>clo_bd_pt</i> | the closest point on the border. |
|------------------|----------------------------------|

#### Parameters

|                  |                                                           |
|------------------|-----------------------------------------------------------|
| <i>tolerance</i> | the tolerance used in defining whether the given point is |
|------------------|-----------------------------------------------------------|

Implements [Go::Domain](#).

29.79.3.4 `RectDomain Go::CurveBoundedDomain::containingDomain ( ) const`

Get a rectangular domain that is guaranteed to contain the [CurveBoundedDomain](#).

#### Returns

a [RectDomain](#) that contains this [CurveBoundedDomain](#). If the loops defining the [CurveBoundedDomain](#) are specified by 2D parameter curves, the [RectDomain](#) will be the smallest rectangular domain containing all the control points of the boundary curves. In the opposite case, the loops are assumed to be [CurveOnSurface](#)s, and the returned [RectDomain](#) will be that of the underlying surface.

29.79.3.5 `void Go::CurveBoundedDomain::findPcurveInsideSegments ( const SplineCurve & curve, double tolerance, std::vector< double > & params_start_end_interval ) const`

Given a curve in the 2D parameter plane, determine those parts of the curve that are contained inside 'this' [CurveBoundedDomain](#).

#### Parameters

|                  |                                                                                                                              |
|------------------|------------------------------------------------------------------------------------------------------------------------------|
| <i>curve</i>     | this is the 2D curve that we want to examine                                                                                 |
| <i>tolerance</i> | this is the tolerance used when determining which parts of of 'curve' are inside 'this' <a href="#">CurveBoundedDomain</a> . |

## Return values

|                                  |                                                                                                                                                                                                                                                                                 |
|----------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>params_start_end_interval</i> | a vector containing the start- and end parameters of the curve segments that were found to be inside this domain. An even indexed entry marks the start parameter of a curve segment, while the following, odd indexed entry marks the end parameter of the same curve segment. |
|----------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

29.79.3.6 void Go::CurveBoundedDomain::findPcurveInsideSegments ( const SplineCurve & *curve*, double *tolerance*, std::vector< double > & *params\_start\_end\_interval*, std::vector< double > & *boundary\_params*, std::vector< int > & *boundary\_loops*, std::vector< int > & *boundary\_curves* ) const

Given a curve in the 2D parameter plane, determine those parts of the curve that are contained inside 'this' [CurveBoundedDomain](#). Also store positional information about the intersections with the boundary curves

## Parameters

|                  |                                                                                                                              |
|------------------|------------------------------------------------------------------------------------------------------------------------------|
| <i>curve</i>     | this is the 2D curve that we want to examine                                                                                 |
| <i>tolerance</i> | this is the tolerance used when determining which parts of of 'curve' are inside 'this' <a href="#">CurveBoundedDomain</a> . |

## Return values

|                                  |                                                                                                                                                                                                                                                                                 |
|----------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>params_start_end_interval</i> | a vector containing the start- and end parameters of the curve segments that were found to be inside this domain. An even indexed entry marks the start parameter of a curve segment, while the following, odd indexed entry marks the end parameter of the same curve segment. |
| <i>boundary_params</i>           | a vector containing the parameters on the boundary curves in the same order as in <i>params_start_end_interval</i>                                                                                                                                                              |
| <i>boundary_loops</i>            | a vector containing the loop index of the boundary curves in the same order as in <i>params_start_end_interval</i>                                                                                                                                                              |
| <i>boundary_curves</i>           | a vector containing the boundary curve position among the curves in the same loop, in the same order as in <i>params_start_end_interval</i>                                                                                                                                     |

29.79.3.7 void Go::CurveBoundedDomain::getInsideIntervals ( int *pardir*, double *parval1*, double *parval2*, double *tolerance*, std::vector< std::pair< double, double > > & *insideInts* ) const

Fetch all intervals in one parameter direction going through a specific point lying inside the bounded domain.

29.79.3.8 void Go::CurveBoundedDomain::getInternalPoint ( double & *upar*, double & *vpar* ) const

Return an arbitrary internal parameter point in the domain (not on the boundary)

29.79.3.9 virtual bool Go::CurveBoundedDomain::isInDomain ( const Array< double, 2 > & *point*, double *tolerance* ) const [virtual]

Query whether a given parameter pair is inside the domain or not.

## Parameters

|                  |                                                                                                                                                                                                                                                         |
|------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>point</i>     | array containing the parameter pair                                                                                                                                                                                                                     |
| <i>tolerance</i> | the tolerance to be used. In order to be considered 'inside', the point must be located inside one of the defining <a href="#">CurveLoop</a> s, as well as being at a distance more than 'tolerance' from any point on that <a href="#">CurveLoop</a> . |

## Returns

'true' if the point is found to be inside the domain, 'false' otherwise.

Implements [Go::Domain](#).

```
29.79.3.10 virtual int Go::CurveBoundedDomain::isInDomain2 (const Array< double, 2 > & point, double tolerance)
 const [virtual]
```

Query whether a given parameter pair is inside the domain or not.

## Parameters

|                  |                                                                                                                                                                                                                                                         |
|------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>point</i>     | array containing the parameter pair                                                                                                                                                                                                                     |
| <i>tolerance</i> | the tolerance to be used. In order to be considered 'inside', the point must be located inside one of the defining <a href="#">CurveLoop</a> s, as well as being at a distance more than 'tolerance' from any point on that <a href="#">CurveLoop</a> . |

## Returns

'1' if the point is found to be inside the domain, '2' if it is found to be on the boundary '0' otherwise.

Implements [Go::Domain](#).

```
29.79.3.11 virtual bool Go::CurveBoundedDomain::isOnBoundary (const Array< double, 2 > & point, double tolerance
) const [virtual]
```

check whether a given parameter pair is located *on* the domain boundary

## Parameters

|                  |                                                                                                                       |
|------------------|-----------------------------------------------------------------------------------------------------------------------|
| <i>point</i>     | array containing the parameter pair                                                                                   |
| <i>tolerance</i> | the tolerance used. (how 'far' from the boundary our parameter pair can be and still be considered 'on' the boundary. |

## Returns

'true' if the parameter pair is considered to be on the boundary, 'false' otherwise.

Implements [Go::Domain](#).

29.79.3.12 `bool Go::CurveBoundedDomain::isOnCorner ( const Array< double, 2 > &point, double tolerance ) const`

Check if the given parameter pair is located on the endpoint of some curve in the curve loop

29.79.3.13 `int Go::CurveBoundedDomain::positionPointInDomain ( int pdir, double parval1, double parval2, double tolerance ) const`

Check the position of a parameter point in the domain. Return value: -1 : Outside of outer loop 0 : Inside domain  $j > 0$  : Inside hole number  $j$ , i.e. inside loop number  $j$

The documentation for this class was generated from the following file:

- [gotools-core/include/GoTools/geometry/CurveBoundedDomain.h](#)

## 29.80 Go::CurveLoop Class Reference

```
#include <CurveLoop.h>
```

### Public Member Functions

- [CurveLoop \(\)](#)  
*Create an empty loop.*
- [CurveLoop \(const std::vector< shared\\_ptr< ParamCurve > > &curves, double space\\_epsilon\)](#)
- [virtual ~CurveLoop \(\)](#)  
*Virtual destructor allows safe inheritance.*
- [void swap \(CurveLoop &other\)](#)  
*Quick swap of one CurveLoop with another.*
- [void setCurves \(const std::vector< shared\\_ptr< ParamCurve > > &curves\)](#)
- [void turnOrientation \(\)](#)  
*Reverse the direction of all curves and their mutual order.*
- [void setSpaceEpsilon \(const double space\\_epsilon\)](#)
- [double getSpaceEpsilon \(\) const](#)
- [int size \(\) const](#)
- [std::vector< shared\\_ptr< ParamCurve > >::const\\_iterator begin \(\) const](#)
- [std::vector< shared\\_ptr< ParamCurve > >::const\\_iterator end \(\) const](#)
- [std::vector< shared\\_ptr< ParamCurve > >::iterator begin \(\)](#)
- [std::vector< shared\\_ptr< ParamCurve > >::iterator end \(\)](#)
- [shared\\_ptr< ParamCurve > operator\[\] \(int index\) const](#)
- [void closestPoint \(const Point &pt, int &clo\\_ind, double &clo\\_par, Point &clo\\_pt, double &clo\\_dist\) const](#)
- [void closestParPoint \(const Point &pt, int &clo\\_ind, double &clo\\_par, Point &clo\\_pt, double &clo\\_dist\) const](#)
- [std::vector< shared\\_ptr< ParamCurve > > getCurves \(\)](#)  
*Return the curves in the loop as a vector.*
- [std::vector< Point > getCorners \(\) const](#)  
*Return joint points between curves.*
- [void getSmoothCurves \(std::vector< std::vector< shared\\_ptr< ParamCurve > > > &curves, double ang\\_tol\)](#)  
*Collect curves with smooth connections.*
- [bool isValid \(\) const](#)
- [bool fixInvalidLoop \(double &max\\_gap\)](#)
- [bool simplify \(double tol, double ang\\_tol, double &max\\_dist\)](#)  
*Simplify this curve loop if possible by reducing the number of curves.*
- [double getMaxCurveDist \(\) const](#)  
*Maximum distance between subsequent curves.*

### 29.80.1 Detailed Description

[CurveLoop](#) represents a closed loop defined by the composition of a set of curves. The start point of curve (i+1) should coincide (within a certain tolerance) with the end point of curve (i), and the end point of the last curve should coincide with the start point of the first curve. CurveLoops are useful in, for example, describing the boundary of a surface.

Definition at line 107 of file CurveLoop.h.

### 29.80.2 Constructor & Destructor Documentation

#### 29.80.2.1 Go::CurveLoop::CurveLoop ( )

Create an empty loop.

#### 29.80.2.2 Go::CurveLoop::CurveLoop ( const std::vector< shared\_ptr< ParamCurve > > & curves, double space\_epsilon )

Create a loop based on a given set of curves.

##### Parameters

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>curves</i>        | a vector with shared pointers to the curves defining the <a href="#">CurveLoop</a> . The start point of each curve should match the end point of the previous curve within the given tolerance. The same is the same for the start point on the first curve and the end point on the last curve). Moreover, it is expected that all curves are of the same type, and that they all lie in the same space (ie. have the same dimension). |
| <i>space_epsilon</i> | the given tolerance for defining coincidence between start/end points on curves.                                                                                                                                                                                                                                                                                                                                                        |

#### 29.80.2.3 virtual Go::CurveLoop::~CurveLoop ( ) [virtual]

Virtual destructor allows safe inheritance.

### 29.80.3 Member Function Documentation

#### 29.80.3.1 std::vector< shared\_ptr< ParamCurve > >::const\_iterator Go::CurveLoop::begin ( ) const [inline]

Get a const iterator to the first curve in the [CurveLoop](#)

##### Returns

a const iterator to the first curve in the [CurveLoop](#)

Definition at line 161 of file CurveLoop.h.

29.80.3.2 `std::vector< shared_ptr<ParamCurve> >::iterator Go::CurveLoop::begin ( )` `[inline]`

Get a iterator to the first curve in the [CurveLoop](#)

Returns

a iterator to the first curve in the [CurveLoop](#)

Definition at line 171 of file CurveLoop.h.

29.80.3.3 `void Go::CurveLoop::closestParPoint ( const Point & pt, int & clo_ind, double & clo_par, Point & clo_pt, double & clo_dist ) const`

View the loop as a curve in the parameter space and compute the closest point in the loop. This function is only interesting if the curves constituting the [CurveLoop](#) are of type "CurveOnSurface". In that case, the curves have a 2D representation in the parametric domain of the surface, as well as a 3D representation in geometric space. This function will search for the closest point in *parametric* space to a *parametric* (2D) point specified by the user.

Parameters

|                 |                                                                                                                                                                                         |
|-----------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>pt</i>       | The point given by the user. It should be a 2D point referring to the parametric domain of a surface. We want to determine the closest point to this on the <a href="#">CurveLoop</a> . |
| <i>clo_ind</i>  | Upon return: the index of the curve segment on which the closest point was found.                                                                                                       |
| <i>clo_par</i>  | Upon return: the parameter of the detected closest point on the curve segment containing it (a scalar value)                                                                            |
| <i>clo_pt</i>   | Upon return: the curve segment's closest point represented as a 2D parameter pair in the domain of the underlying surface.                                                              |
| <i>clo_dist</i> | Upon return: the distance between the point given by the user and the found closest point, measured in the parametric domain of the underlying surface.                                 |

29.80.3.4 `void Go::CurveLoop::closestPoint ( const Point & pt, int & clo_ind, double & clo_par, Point & clo_pt, double & clo_dist ) const`

Find the closest point on the curve loop to a point specified by the user.

Parameters

|                 |                                                                                                                |
|-----------------|----------------------------------------------------------------------------------------------------------------|
| <i>pt</i>       | The point given by the user. We want to determine the closest point to this on the <a href="#">CurveLoop</a> . |
| <i>clo_ind</i>  | Upon return: the index of the curve segment on which the closest point was found.                              |
| <i>clo_par</i>  | Upon return: the parameter of the detected closest point on the curve containing it.                           |
| <i>clo_pt</i>   | Upon return: the geometric position of the detected closest point                                              |
| <i>clo_dist</i> | Upon return: the distance to the detected closest point.                                                       |

29.80.3.5 `std::vector< shared_ptr<ParamCurve> >::const_iterator Go::CurveLoop::end ( ) const` `[inline]`

Get a const iterator to one-past-the-last curve in the [CurveLoop](#)

**Returns**

a const iterator to the one-past-the-last curve in the [CurveLoop](#)

Definition at line 166 of file CurveLoop.h.

**29.80.3.6** `std::vector< shared_ptr<ParamCurve> >::iterator Go::CurveLoop::end ( )` `[inline]`

Get a iterator to one-past-the-last curve in the [CurveLoop](#)

**Returns**

a iterator to the one-past-the-last curve in the [CurveLoop](#)

Definition at line 176 of file CurveLoop.h.

**29.80.3.7** `bool Go::CurveLoop::fixInvalidLoop ( double & max_gap )`

If loop is invalid, we try to fix loop. If we fail we alter nothing. Return value is true if loop is valid. `max_gap` is set to the largest gap in the loop.

**29.80.3.8** `std::vector<Point> Go::CurveLoop::getCorners ( ) const`

Return joint points between curves.

**29.80.3.9** `std::vector<shared_ptr<ParamCurve> > Go::CurveLoop::getCurves ( )` `[inline]`

Return the curves in the loop as a vector.

Definition at line 222 of file CurveLoop.h.

**29.80.3.10** `double Go::CurveLoop::getMaxCurveDist ( ) const` `[inline]`

Maximum distance between subsequent curves.

Definition at line 249 of file CurveLoop.h.

**29.80.3.11** `void Go::CurveLoop::getSmoothCurves ( std::vector< std::vector< shared_ptr< ParamCurve > > > & curves, double angtol )`

Collect curves with smooth connections.

**29.80.3.12** `double Go::CurveLoop::getSpaceEpsilon ( ) const`

Get the tolerance value (used to determine whether the start/end points on curves are coincident).

**29.80.3.13** `bool Go::CurveLoop::isValid ( ) const`

If the curves do not form a simple loop within the input tolerance, the object is invalid.

**29.80.3.14** `shared_ptr<ParamCurve> Go::CurveLoop::operator[] ( int index ) const`

Get a shared pointer to the i'th curve in the [CurveLoop](#)

## Parameters

|              |                                  |
|--------------|----------------------------------|
| <i>index</i> | the index of the requested curve |
|--------------|----------------------------------|

## Returns

a shared pointer to the requested curve.

29.80.3.15 void Go::CurveLoop::setCurves ( const std::vector< shared\_ptr< ParamCurve > > & curves )

Reset the [CurveLoop](#) based on a given vector of curves. (But keep the previously set tolerance value.)

## Parameters

|               |                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|---------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>curves</i> | a vector with shared pointers to the curves defining the <a href="#">CurveLoop</a> . The start point of each curve should match the end point of the previous curve within the given tolerance. The same is the same for the start point on the first curve and the end point on the last curve). Moreover, it is expected that all curves are of the same type, and that they all lie in the same space (ie. have the same dimension). |
|---------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

29.80.3.16 void Go::CurveLoop::setSpaceEpsilon ( const double space\_epsilon )

Set the tolerance (used to determine whether the start/end points on curves are coincident) to a given value

## Parameters

|                      |                                 |
|----------------------|---------------------------------|
| <i>space_epsilon</i> | set the tolerance to this value |
|----------------------|---------------------------------|

29.80.3.17 bool Go::CurveLoop::simplify ( double tol, double ang\_tol, double & max\_dist )

Simplify this curve loop if possible by reducing the number of curves.

29.80.3.18 int Go::CurveLoop::size ( ) const [inline]

Query the number of curves constituting the [CurveLoop](#)

## Returns

the number of curves constituting the [CurveLoop](#)

Definition at line 157 of file CurveLoop.h.

29.80.3.19 void Go::CurveLoop::swap ( CurveLoop & other )

Quick swap of one [CurveLoop](#) with another.



29.80.3.20 void Go::CurveLoop::turnOrientation ( )

Reverse the direction of all curves and their mutual order.

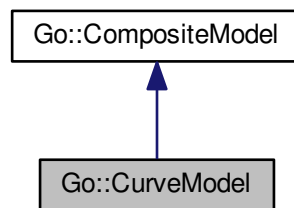
The documentation for this class was generated from the following file:

- [gotools-core/include/GoTools/geometry/CurveLoop.h](#)

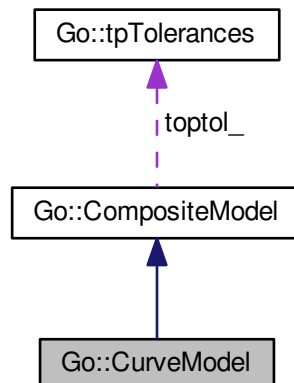
## 29.81 Go::CurveModel Class Reference

```
#include <CurveModel.h>
```

Inheritance diagram for Go::CurveModel:



Collaboration diagram for Go::CurveModel:



## Public Member Functions

- [CurveModel](#) (double gap, double neighbour, double kink, double bend, std::vector< shared\_ptr< [ParamCurve](#) > > &curves)
- [~CurveModel](#) ()
  - Destructor.*
- virtual [CurveModel](#) \* clone () const
- virtual int nmbEntities () const
- shared\_ptr< [ParamCurve](#) > getCurve (int idx) const
- int getIndex ([ParamCurve](#) \*curve) const
- virtual void evaluate (int idx, double par[], [Point](#) &pnt) const
- virtual void evaluate (int idx, double par[], int nder, std::vector< [Point](#) > &der) const
- virtual void closestPoint ([Point](#) &pnt, [Point](#) &clo\_pnt, int &idx, double clo\_par[], double &dist)
- virtual shared\_ptr< [IntResultsModel](#) > intersect (const [ftLine](#) &line)
- virtual shared\_ptr< [IntResultsModel](#) > intersect\_plane (const [ftPlane](#) &plane)
- virtual void extremalPoint ([Point](#) &dir, [Point](#) &ext\_pnt, int &idx, double ext\_par[])
- Extremal point(s) in a given direction.*
- virtual [BoundingBox](#) boundingBox ()
- virtual [BoundingBox](#) boundingBox (int idx) const
- virtual bool isDegenerate (int idx) const
- virtual double curvature (int idx, double \*par) const
- virtual void turn (int idx)
- virtual void turn ()
- virtual void tessellate (std::vector< shared\_ptr< [GeneralMesh](#) > > &meshes) const
- virtual void tessellate (int resolution[], std::vector< shared\_ptr< [GeneralMesh](#) > > &meshes) const
- virtual void tessellate (double density, std::vector< shared\_ptr< [GeneralMesh](#) > > &meshes) const
- virtual void tessellatedCtrPolygon (std::vector< shared\_ptr< [LineCloud](#) > > &ctr\_pol) const
- std::vector< shared\_ptr< [CompositeCurve](#) > > fetchCompositeCurves () const

## Additional Inherited Members

### 29.81.1 Detailed Description

A curve model including topological information

Definition at line 65 of file [CurveModel.h](#).

### 29.81.2 Constructor & Destructor Documentation

**29.81.2.1** `Go::CurveModel::CurveModel ( double gap, double neighbour, double kink, double bend, std::vector< shared_ptr< ParamCurve > > & curves )`

Constructor The sequence of the input curves will be kept, but a permutation array will sort the curves according to continuity

#### Parameters

|                  |                                                                                                                |
|------------------|----------------------------------------------------------------------------------------------------------------|
| <i>gap</i>       | If the distance between two points is less than 'gap' they are viewed as identical.                            |
| <i>neighbour</i> | Maximum distance between curves viewed as adjacent.                                                            |
| <i>kink</i>      | If two adjacent curves meet with an angle less than 'kink', they are seen as G1 continous. (angles in radians) |
| <i>bend</i>      | If two curves meet with an angle larger than 'bend', there is an intentional corner. (angles in radians)       |
| <i>curves</i>    | A vector of curves.                                                                                            |

### 29.81.2.2 Go::CurveModel::~~CurveModel ( )

Destructor.

## 29.81.3 Member Function Documentation

### 29.81.3.1 virtual BoundingBox Go::CurveModel::boundingBox ( ) [virtual]

Bounding box of the entire composite curve

#### Returns

Bounding box

Implements [Go::CompositeModel](#).

### 29.81.3.2 virtual BoundingBox Go::CurveModel::boundingBox ( int *idx* ) const [virtual]

Bounding box corresponding to one curve

#### Parameters

|            |                |
|------------|----------------|
| <i>idx</i> | Index of curve |
|------------|----------------|

#### Returns

Bounding box

Implements [Go::CompositeModel](#).

### 29.81.3.3 virtual CurveModel\* Go::CurveModel::clone ( ) const [virtual]

Make a copy of the current model

#### Returns

Pointer to a copy of this [CurveModel](#)

Reimplemented from [Go::CompositeModel](#).

### 29.81.3.4 virtual void Go::CurveModel::closestPoint ( Point & *pnt*, Point & *clo\_pnt*, int & *idx*, double *clo\_par*[], double & *dist* ) [virtual]

Closest point between a given point and this composite curve Returns one point We could think of specifying a function that returns all global closest points, only for global search. It is also possible to return closest points within a given range from the best closest point, but only one point for each curve in the model.

## Parameters

|            |             |
|------------|-------------|
| <i>pnt</i> | Input point |
|------------|-------------|

## Return values

|                  |                                                      |
|------------------|------------------------------------------------------|
| <i>clo_pnt</i>   | Found closest point                                  |
| <i>idx</i>       | Index of curve where the closest point is found      |
| <i>clo_par[]</i> | Parameter value corresponding to the closest point   |
| <i>dist</i>      | Distance between input point and found closest point |

Implements [Go::CompositeModel](#).

29.81.3.5 `virtual double Go::CurveModel::curvature ( int idx, double * par ) const` [virtual]

Curvature of a curve

## Parameters

|            |                                               |
|------------|-----------------------------------------------|
| <i>idx</i> | Index of curve                                |
| <i>par</i> | Parameter value at which to compute curvature |

## Returns

The curvature.

Implements [Go::CompositeModel](#).

29.81.3.6 `virtual void Go::CurveModel::evaluate ( int idx, double par[], Point & pnt ) const` [virtual]

Evaluate position

## Parameters

|              |                 |
|--------------|-----------------|
| <i>idx</i>   | Index of curve  |
| <i>par[]</i> | Parameter value |

## Return values

|            |        |
|------------|--------|
| <i>pnt</i> | Result |
|------------|--------|

Implements [Go::CompositeModel](#).

29.81.3.7 `virtual void Go::CurveModel::evaluate ( int idx, double par[], int nder, std::vector< Point > & der ) const`  
 [virtual]

Evaluate position and a number of derivatives The sequence is position, first derivative, second derivative, etc.

#### Parameters

|               |                                                   |
|---------------|---------------------------------------------------|
| <i>idx</i>    | Index                                             |
| <i>par</i> [] | Parameter value                                   |
| <i>nder</i>   | Number of derivatives to compute, 0=only position |

#### Return values

|            |        |
|------------|--------|
| <i>der</i> | Result |
|------------|--------|

Implements [Go::CompositeModel](#).

29.81.3.8 `virtual void Go::CurveModel::extremalPoint ( Point & dir, Point & ext_pnt, int & idx, double ext_par[])`  
 [virtual]

Extremal point(s) in a given direction.

#### Parameters

|            |           |
|------------|-----------|
| <i>dir</i> | Direction |
|------------|-----------|

#### Return values

|                   |                                                  |
|-------------------|--------------------------------------------------|
| <i>ext_pnt</i>    | Found extremal point                             |
| <i>idx</i>        | Index of curve where the extremal point is found |
| <i>ext_par</i> [] | Parameter value of extremal point                |

Implements [Go::CompositeModel](#).

29.81.3.9 `std::vector<shared_ptr<CompositeCurve>> Go::CurveModel::fetchCompositeCurves ( ) const`

Fetch all uniquely connected composite curves

#### Returns

Vector of pointers to the composite curves

29.81.3.10 `shared_ptr<ParamCurve> Go::CurveModel::getCurve ( int idx ) const`

Return one curve Note that the index corresponds to the sequence of which the curves are added to the [CurveModel](#), not the position in the composite curve

## Parameters

|            |                |
|------------|----------------|
| <i>idx</i> | Index of curve |
|------------|----------------|

## Returns

Pointer to curve

29.81.3.11 `int Go::CurveModel::getIndex ( ParamCurve * curve ) const`

Given a curve in the composite curve, return the index of this curve

## Parameters

|              |                  |
|--------------|------------------|
| <i>curve</i> | Pointer to curve |
|--------------|------------------|

## Returns

Index to curve

29.81.3.12 `virtual shared_ptr<IntResultsModel> Go::CurveModel::intersect ( const ftLine & line ) [virtual]`

Intersection with a line. Expected output is points, probably one point. Curves can occur in special configurations.

## Parameters

|             |           |
|-------------|-----------|
| <i>line</i> | The line. |
|-------------|-----------|

## Returns

Pointer to an [IntResultsModel](#).

Implements [Go::CompositeModel](#).

29.81.3.13 `virtual shared_ptr<IntResultsModel> Go::CurveModel::intersect_plane ( const ftPlane & plane ) [virtual]`

Intersection with a plane.

## Parameters

|              |            |
|--------------|------------|
| <i>plane</i> | The plane. |
|--------------|------------|

**Returns**

Pointer to an [IntResultsModel](#).

Implements [Go::CompositeModel](#).

**29.81.3.14** `virtual bool Go::CurveModel::isDegenerate ( int idx ) const [virtual]`

Whether one particular curve is degenerated

**Parameters**

|            |                |
|------------|----------------|
| <i>idx</i> | Index of curve |
|------------|----------------|

**Returns**

Whether the curve is degenerated

Implements [Go::CompositeModel](#).

**29.81.3.15** `virtual int Go::CurveModel::nmbEntities ( ) const [virtual]`

Number of simple entities

**Returns**

Number of simple entities

Implements [Go::CompositeModel](#).

**29.81.3.16** `virtual void Go::CurveModel::tessellate ( std::vector< shared_ptr< GeneralMesh > > & meshes ) const [virtual]`

Tessellate all curves with respect to a default resolution

**Return values**

|               |                  |
|---------------|------------------|
| <i>meshes</i> | Tesselated model |
|---------------|------------------|

Implements [Go::CompositeModel](#).

**29.81.3.17** `virtual void Go::CurveModel::tessellate ( int resolution[], std::vector< shared_ptr< GeneralMesh > > & meshes ) const [virtual]`

Tessellate all curves with respect to a given resolution

## Parameters

|                      |                                               |
|----------------------|-----------------------------------------------|
| <i>resolution</i> [] | All curves are tessellated with resolution[0] |
|----------------------|-----------------------------------------------|

## Return values

|               |                   |
|---------------|-------------------|
| <i>meshes</i> | Tessellated model |
|---------------|-------------------|

Implements [Go::CompositeModel](#).

29.81.3.18 `virtual void Go::CurveModel::tessellate ( double density, std::vector< shared_ptr< GeneralMesh > > & meshes ) const` [virtual]

Tessellate all curves with respect to a given tessellation density

## Parameters

|                |                      |
|----------------|----------------------|
| <i>density</i> | Tessellation density |
|----------------|----------------------|

## Return values

|               |                   |
|---------------|-------------------|
| <i>meshes</i> | Tessellated model |
|---------------|-------------------|

Implements [Go::CompositeModel](#).

29.81.3.19 `virtual void Go::CurveModel::tessellatedCtrPolygon ( std::vector< shared_ptr< LineCloud > > & ctr_pol ) const` [virtual]

Tessellate the control polygon of all curves.

## Return values

|                |                                                    |
|----------------|----------------------------------------------------|
| <i>ctr_pol</i> | Tessellation of the control polygon of all curves. |
|----------------|----------------------------------------------------|

Implements [Go::CompositeModel](#).

29.81.3.20 `virtual void Go::CurveModel::turn ( int idx )` [virtual]

Turn parameter direction of one curve. An update in the topology structures is required.

## Parameters

|            |                |
|------------|----------------|
| <i>idx</i> | Index of curve |
|------------|----------------|

Implements [Go::CompositeModel](#).



29.81.3.21 virtual void Go::CurveModel::turn ( ) [virtual]

Turn parameter directions of all curves. An update in the topology structures is required.

Implements [Go::CompositeModel](#).

The documentation for this class was generated from the following file:

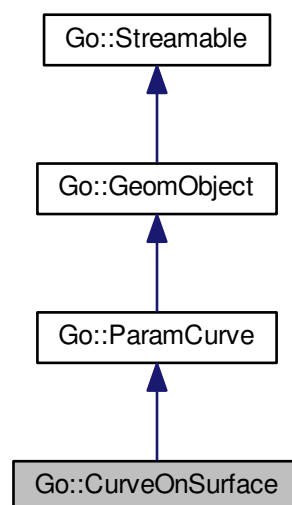
- [compositemodel/include/GoTools/compositemodel/CurveModel.h](#)

## 29.82 Go::CurveOnSurface Class Reference

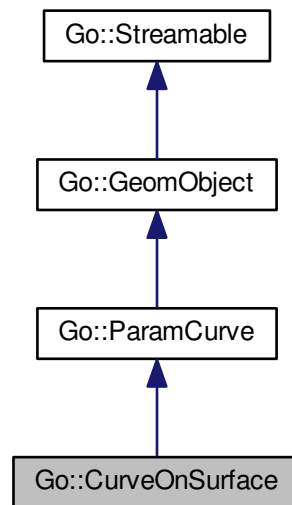
A curve living on a parametric surface. It either has got information about the curve in geometry space and in the parameter domain of the surface or the ability to compute the other representation given one. The curve may have information on whether it is a constant parameter or boundary curve on the surface.

```
#include <CurveOnSurface.h>
```

Inheritance diagram for Go::CurveOnSurface:



Collaboration diagram for `Go::CurveOnSurface`:



## Public Member Functions

- [CurveOnSurface](#) ()
- [CurveOnSurface](#) (shared\_ptr< [ParamSurface](#) > surf, shared\_ptr< [ParamCurve](#) > curve, bool preferparameter)
- [CurveOnSurface](#) (shared\_ptr< [ParamSurface](#) > surf, shared\_ptr< [ParamCurve](#) > curve, int constdir, double constpar, int boundary)
  - Constructor for constant parameter curves.*
- [CurveOnSurface](#) (shared\_ptr< [ParamSurface](#) > surf, int constdir, double constpar, double par1, double par2, int boundary)
- [CurveOnSurface](#) (shared\_ptr< [ParamSurface](#) > surf, shared\_ptr< [ParamCurve](#) > pcurve, shared\_ptr< [ParamCurve](#) > spacecurve, bool preferparameter, int ccm, int constdir, double constpar, int boundary, bool same\_orientation)
  - Constructor with all information.*
- [CurveOnSurface](#) (shared\_ptr< [ParamSurface](#) > surf, shared\_ptr< [ParamCurve](#) > pcurve, shared\_ptr< [ParamCurve](#) > spacecurve, bool preferparameter, int ccm=0)
- [CurveOnSurface](#) (const [CurveOnSurface](#) &surface\_curve)
- [CurveOnSurface](#) & operator= (const [CurveOnSurface](#) &other)
- virtual [~CurveOnSurface](#) ()
- virtual void [read](#) (std::istream &is)
- virtual void [write](#) (std::ostream &os) const
- virtual [BoundingBox](#) boundingBox () const
- virtual [DirectionCone](#) directionCone () const
- virtual int [dimension](#) () const
  - Dimension of geometry space.*
- virtual [ClassType](#) instanceType () const
  - Type of current object.*
- virtual [CurveOnSurface](#) \* [clone](#) () const

- virtual void `point` (`Point` &pt, `double` tpar) `const`
- virtual void `point` (`std::vector`< `Point` > &pts, `double` tpar, `int` derivs, `bool` from\_right=true) `const`
- virtual `double` `startparam` () `const`
- virtual `double` `endparam` () `const`
- virtual void `reverseParameterDirection` (`bool` switchparam=false)
- virtual void `setParameterInterval` (`double` t1, `double` t2)
  - Linear reparametrization. The meaning is changed for elementary curves.*
- virtual `SplineCurve` \* `geometryCurve` ()
- virtual `bool` `isDegenerate` (`double` degenerate\_epsilon)
- virtual `CurveOnSurface` \* `subCurve` (`double` from\_par, `double` to\_par, `double` fuzzy=DEFAULT\_PARAMETER\_EPSILON) `const`
- virtual `std::vector`< `shared_ptr`< `ParamCurve` > > `split` (`double` param, `double` fuzzy=DEFAULT\_PARAMETER\_EPSILON) `const`
  - Split curve in a specified parameter value.*
- virtual void `closestPoint` (`const Point` &pt, `double` tmin, `double` tmax, `double` &clo\_t, `Point` &clo\_pt, `double` &clo\_dist, `double` const \*seed=0) `const`
- virtual `double` `length` (`double` tol)
- virtual void `appendCurve` (`ParamCurve` \*cv, `bool` reparam=true)
- virtual void `appendCurve` (`ParamCurve` \*cv, `int` continuity, `double` &dist, `bool` reparam=true)
- void `setUnderlyingSurface` (`shared_ptr`< `ParamSurface` > surface)
- void `setCurves` (`shared_ptr`< `ParamCurve` > spacecurve, `shared_ptr`< `ParamCurve` > parametercurve)
- void `setSpaceCurve` (`shared_ptr`< `ParamCurve` > spacecurve)
- void `setParameterCurve` (`shared_ptr`< `ParamCurve` > parametercurve)
- void `unsetParameterCurve` ()
  - Remove parameter curve information.*
- `bool` `ensureParCrvExistence` (`double` epsgeo, `const RectDomain` \*domain\_of\_interest=NULL, `const Point` \*start\_par\_pt=NULL, `const Point` \*end\_par\_pt=NULL)
- `bool` `makeParameterCurve` (`double` tol, `const Point` &par1, `const Point` &par2)
  - Make parameter curve between given end parameters.*
- `bool` `translateParameterCurve` (`const Point` &dir)
- `bool` `translateSwapParameterCurve` (`const Point` &dir, `double` sgn, `int` pdir)
- `bool` `setDomainParCrv` (`double` umin, `double` umax, `double` vmin, `double` vmax, `double` uminprev, `double` umaxprev, `double` vminprev, `double` vmaxprev)
- void `unsetSpaceCurve` ()
  - Remove space curve information.*
- `bool` `ensureSpaceCrvExistence` (`double` tol)
  - Generate space curve if it doesn't exist.*
- void `setCurveTypeInfo` (`int` ccm)
  - Set curve information.*
- `int` `getCurveTypeInfo` ()
- `bool` `updateIsoCurves` ()
  - Update constant parameter curves.*
- `bool` `updateIsoCurves` (`int` constdir, `double` constpar, `int` boundary)
- `bool` `updateCurves` (`double` epsge)
  - Update curves. Reapproximate the dependent curve within the tolerance epsge.*
- `bool` `updateCurves` (`Point` vx1, `Point` vx2, `double` epsge)
  - Update curves.*
- void `enableSameOrientation` ()
  - Make sure that the curves have the same orientation.*
- virtual `double` `nextSegmentVal` (`double` par, `bool` forward, `double` tol) `const`
- `shared_ptr`< `ParamSurface` > `underlyingSurface` ()
- `shared_ptr`< `ParamCurve` > `parameterCurve` ()
- `shared_ptr`< `ParamCurve` > `spaceCurve` ()

- `shared_ptr< const ParamSurface > underlyingSurface () const`
- `shared_ptr< const ParamCurve > parameterCurve () const`
- `shared_ptr< const ParamCurve > spaceCurve () const`
- `bool parPref () const`
- `void setParPref (bool prefer)`
- `RectDomain containingDomain () const`
- `Point faceParameter (double crv_par, const RectDomain *domain_of_interest=NULL) const`  
*Fetch the surface parameter corresponding to a curve parameter.*
- `int whichBoundary (double tol, bool &same_orientation) const`
- `bool isConstantCurve () const`  
*Check if the curve is a constant parameter curve and marked as such.*
- `bool isConstantCurve (double tol, int &parDir, double &parVal) const`
- `void getConstantCurveInfo (int &constDir, double &constVal, int &bd, bool &same_orientation) const`  
*Fetch recorded constant curve information.*
- `bool sameParameterDomain () const`
- `bool sameOrientation () const`
- `bool sameTrace (double tol, int nmb_sample=5) const`
- `bool sameCurve (double tol, int nmb_sample=5) const`
- `void makeCurvesConsistent (bool prefer_parameter)`
- `int curveCreationMethod () const`
- `double maxTraceDiff (int nmb_sample=5) const`
- `virtual bool isAxisRotational (Point &centre, Point &axis, Point &vec, double &angle)`
- `virtual bool isLinear (Point &dir, double tol)`  
*Check if the curve is linear.*
- `virtual bool isInPlane (const Point &loc, const Point &axis, double eps, Point &normal) const`  
*Check if the curve lies in a plane passing through a given axis.*
- `virtual bool isInPlane (const Point &norm, double eps, Point &pos) const`  
*Check if the curve lies in a plane with a given normal.*

### Static Public Member Functions

- `static ClassType classType ()`

### Additional Inherited Members

#### 29.82.1 Detailed Description

A curve living on a parametric surface. It either has got information about the curve in geometry space and in the parameter domain of the surface or the ability to compute the other representation given one. The curve may have information on whether it is a constant parameter or boundary curve on the surface.

Definition at line 62 of file CurveOnSurface.h.

#### 29.82.2 Constructor & Destructor Documentation

##### 29.82.2.1 Go::CurveOnSurface::CurveOnSurface ( )

Define an empty `CurveOnSurface` that can be assigned or `read()` into

##### 29.82.2.2 Go::CurveOnSurface::CurveOnSurface ( shared\_ptr< ParamSurface > surf, shared\_ptr< ParamCurve > curve, bool preferparameter )

Construct a `CurveOnSurface` by specifying the surface and either a space curve or a curve in the parameter plane. Does not clone any of the input, just sets (smart) pointers.

## Parameters

|                        |                                                                                                                                                                                                                                                                                          |
|------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>surf</i>            | pointer to the underlying surface                                                                                                                                                                                                                                                        |
| <i>curve</i>           | pointer to the curve specifying the <a href="#">CurveOnSurface</a> . This curve may either be in the parametric domain of the surface (2D curve), or a space curve that the user has assured to be coincident with the surface. 'preferparameter' specifies which kind of curve this is. |
| <i>preferparameter</i> | if this is set to 'true', then 'curve' is assumed to be a curve in the parametric domain of the surface. Otherwise, it is assumed to be a space (3D) curve.                                                                                                                              |

29.82.2.3 `Go::CurveOnSurface::CurveOnSurface ( shared_ptr< ParamSurface > surf, shared_ptr< ParamCurve > curve, int constdir, double constpar, int boundary )`

Constructor for constant parameter curves.

29.82.2.4 `Go::CurveOnSurface::CurveOnSurface ( shared_ptr< ParamSurface > surf, int constdir, double constpar, double par1, double par2, int boundary )`

Constructor for constant parameter curves. The curve is to be restricted to the parameter interval [par1, par2].

29.82.2.5 `Go::CurveOnSurface::CurveOnSurface ( shared_ptr< ParamSurface > surf, shared_ptr< ParamCurve > pcurve, shared_ptr< ParamCurve > spacecurve, bool preferparameter, int ccm, int constdir, double constpar, int boundary, bool same_orientation )`

Constructor with all information.

29.82.2.6 `Go::CurveOnSurface::CurveOnSurface ( shared_ptr< ParamSurface > surf, shared_ptr< ParamCurve > pcurve, shared_ptr< ParamCurve > spacecurve, bool preferparameter, int ccm = 0 )`

Construct a [CurveOnSurface](#) by specifying the surface, the space curve and the curve in the parameter plane. The arguments are checked for consistency, and if they appear incoherent, an exception will be thrown. Does not clone any of the input, just sets (smart) pointers.

## Parameters

|                        |                                                                                                                                                                                                   |
|------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>surf</i>            | pointer to the underlying surface                                                                                                                                                                 |
| <i>pcurve</i>          | pointer to the curve representing the <a href="#">CurveOnSurface</a> in the parametric domain of the surface.                                                                                     |
| <i>spacecurve</i>      | pointer to the curve that is the spatial (3D) representation of the <a href="#">CurveOnSurface</a> .                                                                                              |
| <i>preferparameter</i> | specify whether the parametric curve or the space curve are preferred for internal computations.                                                                                                  |
| <i>ccm</i>             | curve creation method. Specify how curve was created. 0 = undefined, 1 = projection onto surface, 2 = intersection of two surfaces, 3 = isoparametric curve (i.e. either a u-curve or a v-curve). |

29.82.2.7 `Go::CurveOnSurface::CurveOnSurface ( const CurveOnSurface & surface_curve )`

Copy constructor. The copy constructor will not `clone()` the underlying surface, but it will `clone()` both the parametric and the spatial curve.

Parameters

|                      |                                                                                         |
|----------------------|-----------------------------------------------------------------------------------------|
| <i>surface_curve</i> | the <a href="#">CurveOnSurface</a> to copy into 'this' <a href="#">CurveOnSurface</a> . |
|----------------------|-----------------------------------------------------------------------------------------|

29.82.2.8 `virtual Go::CurveOnSurface::~~CurveOnSurface ( ) [virtual]`

Destructor. Trivial because memory is managed by `shared_ptr`.

### 29.82.3 Member Function Documentation

29.82.3.1 `virtual void Go::CurveOnSurface::appendCurve ( ParamCurve * cv, bool reparam = true ) [virtual]`

append a curve to this curve, with eventual reparametrization NB: This virtual member function currently only works for `SplineCurves` and `CurveOnSurfaces`. Moreover, 'this' curve and the 'cv' curve must be of the same type.

Parameters

|                |                                                          |
|----------------|----------------------------------------------------------|
| <i>cv</i>      | the curve to append to 'this' curve.                     |
| <i>reparam</i> | specify whether or not there should be reparametrization |

Implements [Go::ParamCurve](#).

29.82.3.2 `virtual void Go::CurveOnSurface::appendCurve ( ParamCurve * cv, int continuity, double & dist, bool reparam = true ) [virtual]`

append a curve to this curve, with eventual reparametrization

Parameters

|                   |                                                                                                       |
|-------------------|-------------------------------------------------------------------------------------------------------|
| <i>cv</i>         | the curve to append to 'this' curve.                                                                  |
| <i>continuity</i> | the required continuity at the transition. Can be $G^{(-1)}$ and upwards.                             |
| <i>dist</i>       | a measure of the local distorsion around the transition in order to achieve the specified continuity. |
| <i>reparam</i>    | specify whether or not there should be reparametrization                                              |

Implements [Go::ParamCurve](#).

29.82.3.3 `virtual BoundingBox Go::CurveOnSurface::boundingBox ( ) const [virtual]`

Axis align box surrounding this object Computed with respect to the space curve if this one exists, otherwise the underlying surface

Implements [Go::GeomObject](#).

29.82.3.4 `static ClassType Go::CurveOnSurface::classType ( ) [inline],[static]`

Definition at line 159 of file CurveOnSurface.h.

29.82.3.5 `virtual CurveOnSurface* Go::CurveOnSurface::clone ( ) const [inline],[virtual]`

The clone-function is inherited from [GeomObject](#), but overridden here to get a covariant return type (for those compilers that allow this).

Implements [Go::ParamCurve](#).

Definition at line 174 of file CurveOnSurface.h.

29.82.3.6 `virtual void Go::CurveOnSurface::closestPoint ( const Point & pt, double tmin, double tmax, double & clo_t, Point & clo_pt, double & clo_dist, double const * seed = 0 ) const [virtual]`

Compute the closest point from an interval of this curve to a specified point.

Parameters

|                 |                                                                                                                                          |
|-----------------|------------------------------------------------------------------------------------------------------------------------------------------|
| <i>pt</i>       | point we want to find the closest point to                                                                                               |
| <i>tmin</i>     | start parameter of search interval                                                                                                       |
| <i>tmax</i>     | end parameter of search interval                                                                                                         |
| <i>clo_t</i>    | upon function return, 'clo_t' will contain the parameter value of the closest point found.                                               |
| <i>clo_pt</i>   | upon function return, 'clo_pt' will contain the position of the closest point found.                                                     |
| <i>clo_dist</i> | upon function return, 'clo_dist' will contain the distance between 'pt' and the closest point found.                                     |
| <i>seed</i>     | pointer to initial guess value, provided by the user (can be 0, for which the algorithm will determine a (hopefully) reasonable choice). |

Implements [Go::ParamCurve](#).

29.82.3.7 `RectDomain Go::CurveOnSurface::containingDomain ( ) const`

Get the rectangle enclosing the underlying surface's parametric domain.

Returns

the [RectDomain](#) for the underlying surface.

29.82.3.8 `int Go::CurveOnSurface::curveCreationMethod ( ) const`

Return curve creation method 0 = undefined, 1 = projection onto surface, 2 = intersection of two surfaces, 3 = isoparametric curve (i.e. either a u-curve or a v-curve).

29.82.3.9 `virtual int Go::CurveOnSurface::dimension ( ) const [virtual]`

Dimension of geometry space.

Implements [Go::GeomObject](#).

29.82.3.10 `virtual DirectionCone Go::CurveOnSurface::directionCone ( ) const [virtual]`

[Cone](#) surrounding the set of tangent directions corresponding to the surface. Only computed if the space curve exists

Implements [Go::ParamCurve](#).

29.82.3.11 `void Go::CurveOnSurface::enableSameOrientation ( )`

Make sure that the curves have the same orientation.

29.82.3.12 `virtual double Go::CurveOnSurface::endparam ( ) const [virtual]`

Query the end parameter of the curve

Returns

the curve's end parameter

Implements [Go::ParamCurve](#).

29.82.3.13 `bool Go::CurveOnSurface::ensureParCrvExistence ( double epsgeo, const RectDomain * domain_of_interest = NULL, const Point * start_par_pt = NULL, const Point * end_par_pt = NULL )`

Generate parameter curve if it doesn't exist Optionally include start and/or end parameter point.

29.82.3.14 `bool Go::CurveOnSurface::ensureSpaceCrvExistence ( double tol )`

Generate space curve if it doesn't exist.

29.82.3.15 `Point Go::CurveOnSurface::faceParameter ( double crv_par, const RectDomain * domain_of_interest = NULL ) const`

Fetch the surface parameter corresponding to a curve parameter.



29.82.3.16 virtual **SplineCurve\*** Go::CurveOnSurface::geometryCurve ( ) [virtual]

If the definition of this [ParamCurve](#) contains a [SplineCurve](#) describing its spatial shape, then this function will return a pointer to this [SplineCurve](#). Otherwise it will return a null pointer. The returned curve is NEWed, so the user is responsible for deleting it. This function may have side-effects.

#### Returns

a pointer to a [SplineCurve](#) representation of the [ParamCurve](#), if it exists. Null pointer otherwise.

Implements [Go::ParamCurve](#).

29.82.3.17 void Go::CurveOnSurface::getConstantCurveInfo ( int & *constdir*, double & *constval*, int & *bd*, bool & *same\_orientation* ) const [inline]

Fetch recorded constant curve information.

Definition at line 417 of file CurveOnSurface.h.

29.82.3.18 int Go::CurveOnSurface::getCurveTypeInfo ( ) [inline]

Definition at line 323 of file CurveOnSurface.h.

29.82.3.19 virtual **ClassType** Go::CurveOnSurface::instanceType ( ) const [virtual]

Type of current object.

Implements [Go::GeomObject](#).

29.82.3.20 virtual **bool** Go::CurveOnSurface::isAxisRotational ( **Point** & *centre*, **Point** & *axis*, **Point** & *vec*, double & *angle* ) [virtual]

Check if the curve is axis rotational. Only true if a connection to an axis rotational elementary curve exist The axis and rotational angle is only specified if the curve is actually rotational

Reimplemented from [Go::ParamCurve](#).

29.82.3.21 **bool** Go::CurveOnSurface::isConstantCurve ( ) const [inline]

Check if the curve is a constant parameter curve and marked as such.

Definition at line 407 of file CurveOnSurface.h.

29.82.3.22 **bool** Go::CurveOnSurface::isConstantCurve ( double *tol*, int & *pardir*, double & *parval* ) const

Check if the curve is a constant parameter curve with regard to the parameter tol

29.82.3.23 virtual **bool** Go::CurveOnSurface::isDegenerate ( double *degenerate\_epsilon* ) [virtual]

Query whether the curve is degenerate (collapsed into a single point).

## Parameters

|                           |                                                                                                                                                   |
|---------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>degenerate_epsilon</i> | the tolerance used in determine whether the curve is degenerate. A curve is considered degenerate if its total length is shorter than this value. |
|---------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------|

## Returns

`true` if the curve is degenerate, `false` otherwise.

Implements [Go::ParamCurve](#).

**29.82.3.24** `virtual bool Go::CurveOnSurface::isInPlane ( const Point & loc, const Point & axis, double eps, Point & normal ) const` `[virtual]`

Check if the curve lies in a plane passing through a given axis.

Reimplemented from [Go::ParamCurve](#).

**29.82.3.25** `virtual bool Go::CurveOnSurface::isInPlane ( const Point & norm, double eps, Point & pos ) const` `[virtual]`

Check if the curve lies in a plane with a given normal.

Reimplemented from [Go::ParamCurve](#).

**29.82.3.26** `virtual bool Go::CurveOnSurface::isLinear ( Point & dir, double tol )` `[virtual]`

Check if the curve is linear.

Reimplemented from [Go::ParamCurve](#).

**29.82.3.27** `virtual double Go::CurveOnSurface::length ( double tol )` `[virtual]`

Compute the total length of this curve up to some tolerance

## Parameters

|            |                                                                                                                                           |
|------------|-------------------------------------------------------------------------------------------------------------------------------------------|
| <i>tol</i> | the relative tolerance when approximating the length, i.e. stop iteration when error becomes smaller than <code>tol/(curve length)</code> |
|------------|-------------------------------------------------------------------------------------------------------------------------------------------|

## Returns

the length calculated

Implements [Go::ParamCurve](#).

29.82.3.28 void Go::CurveOnSurface::makeCurvesConsistent ( bool *prefer\_parameter* )

29.82.3.29 bool Go::CurveOnSurface::makeParameterCurve ( double *tol*, const Point & *par1*, const Point & *par2* )

Make parameter curve between given end parameters.

29.82.3.30 double Go::CurveOnSurface::maxTraceDiff ( int *nmb\_sample* = 5 ) const

Maximum distance between curve represented by parameter curve and space curve in a number of sampling points

29.82.3.31 virtual double Go::CurveOnSurface::nextSegmentVal ( double *par*, bool *forward*, double *tol* ) const  
[virtual]

Inherited from [ParamCurve](#). If the parametric curve is set to be the 'preferred' one, this function will return the next segment value for the parametric curve; otherwise it will return the next segment value for the spatial 3D curve. See also [ParamCurve::nextSegmentVal\(\)](#)

Reimplemented from [Go::ParamCurve](#).

29.82.3.32 CurveOnSurface& Go::CurveOnSurface::operator= ( const CurveOnSurface & *other* )

Assignment operator. Like the copy constructor, the assignment operator [clone\(\)](#)s the curves, and not the surface.

Parameters

|              |                                                                                         |
|--------------|-----------------------------------------------------------------------------------------|
| <i>other</i> | the <a href="#">CurveOnSurface</a> to copy into 'this' <a href="#">CurveOnSurface</a> . |
|--------------|-----------------------------------------------------------------------------------------|

29.82.3.33 shared\_ptr<ParamCurve> Go::CurveOnSurface::parameterCurve ( ) [inline]

Get a shared pointer to the curve in the parameter domain.

Returns

a shared pointer to the curve in the parameter domain

Definition at line 360 of file CurveOnSurface.h.

29.82.3.34 shared\_ptr<const ParamCurve> Go::CurveOnSurface::parameterCurve ( ) const [inline]

Get a constant, shared pointer to the curve in the parameter domain.

Returns

a const-pointer to the curve in the parameter domain.

Definition at line 375 of file CurveOnSurface.h.

29.82.3.35 `bool Go::CurveOnSurface::parPref ( ) const [inline]`

Query whether the parameter curve or the space curve is preferred for computation in this object.

Definition at line 385 of file CurveOnSurface.h.

29.82.3.36 `virtual void Go::CurveOnSurface::point ( Point & pt, double tpar ) const [virtual]`

Evaluate the curve's position at a given parameter

Parameters

|             |                                                                      |
|-------------|----------------------------------------------------------------------|
| <i>pt</i>   | the evaluated position will be written to this <a href="#">Point</a> |
| <i>tpar</i> | the parameter for which we wish to evaluate the curve                |

Implements [Go::ParamCurve](#).

29.82.3.37 `virtual void Go::CurveOnSurface::point ( std::vector< Point > & pts, double tpar, int derivs, bool from_right = true ) const [virtual]`

Inherited from [ParamCurve](#). Only works for 'derivs' = 0 or 1.

See also

[ParamCurve::point\(\)](#)

Implements [Go::ParamCurve](#).

29.82.3.38 `virtual void Go::CurveOnSurface::read ( std::istream & is ) [virtual]`

read object from stream

Parameters

|           |                                  |
|-----------|----------------------------------|
| <i>is</i> | stream from which object is read |
|-----------|----------------------------------|

Implements [Go::Streamable](#).

29.82.3.39 `virtual void Go::CurveOnSurface::reverseParameterDirection ( bool switchparam = false ) [virtual]`

Set the parameter direction of the curve. The curve's parameter interval will always remain constant, but by flipping the parameter direction, the curve will be traced the opposite way when moving a parameter over the parameter interval.

## Parameters

|                    |                                                                                                                                                   |
|--------------------|---------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>switchparam</i> | if true, and the curve is 2D, the x and y coordinates should be swapped. This is used when turning the orientation of bounded (trimmed) surfaces. |
|--------------------|---------------------------------------------------------------------------------------------------------------------------------------------------|

Implements [Go::ParamCurve](#).

**29.82.3.40** `bool Go::CurveOnSurface::sameCurve ( double tol, int nmb_sample = 5 ) const`

Legality test regarding the consistence of geometry and parameter curve. Check if the two curves represent identical curves with respect to the tolerance tol. Pointwise check.

**29.82.3.41** `bool Go::CurveOnSurface::sameOrientation ( ) const`

Legality test regarding the consistence of geometry and parameter curve. Check if the two curves have the same orientation

**29.82.3.42** `bool Go::CurveOnSurface::sameParameterDomain ( ) const`

Legality test regarding the consistence of geometry and parameter curve. Check if the curves have got the same parameter domain

**29.82.3.43** `bool Go::CurveOnSurface::sameTrace ( double tol, int nmb_sample = 5 ) const`

Legality test regarding the consistence of geometry and parameter curve. Check if the two curves describe the same trace with respect to the tolerance tol. Pointwise check.

**29.82.3.44** `void Go::CurveOnSurface::setCurves ( shared_ptr< ParamCurve > spacecurve, shared_ptr< ParamCurve > parametercurve ) [inline]`

Replace the curves describing the curve on surface. The curve preference is not changed

Definition at line 253 of file CurveOnSurface.h.

**29.82.3.45** `void Go::CurveOnSurface::setCurveTypeInfo ( int ccm ) [inline]`

Set curve information.

Definition at line 318 of file CurveOnSurface.h.

**29.82.3.46** `bool Go::CurveOnSurface::setDomainParCrv ( double umin, double umax, double vmin, double vmax, double uminprev, double umaxprev, double vminprev, double vmaxprev )`

Represent the parameter curve associated with this surface curve on a different domain

29.82.3.47 `void Go::CurveOnSurface::setParameterCurve ( shared_ptr< ParamCurve > parametercurve ) [inline]`

Replace the parameter curve corresponding to this curve on surface curve. Used for instance in relation to reparameterizations of the related surface. Use with care!

Definition at line 270 of file CurveOnSurface.h.

29.82.3.48 `virtual void Go::CurveOnSurface::setParameterInterval ( double t1, double t2 ) [virtual]`

Linear reparametrization. The meaning is changed for elementary curves.

Implements [Go::ParamCurve](#).

29.82.3.49 `void Go::CurveOnSurface::setParPref ( bool prefer ) [inline]`

Definition at line 388 of file CurveOnSurface.h.

29.82.3.50 `void Go::CurveOnSurface::setSpaceCurve ( shared_ptr< ParamCurve > spacecurve ) [inline]`

Replace the space curve corresponding to this curve on surface curve. Use with care!

Definition at line 262 of file CurveOnSurface.h.

29.82.3.51 `void Go::CurveOnSurface::setUnderlyingSurface ( shared_ptr< ParamSurface > surface ) [inline]`

Set the underlying surface to the one pointed to by the argument

#### Parameters

|                |                                                                                                |
|----------------|------------------------------------------------------------------------------------------------|
| <i>surface</i> | the pointer to the surface we will set as underlying for this <a href="#">CurveOnSurface</a> . |
|----------------|------------------------------------------------------------------------------------------------|

Definition at line 248 of file CurveOnSurface.h.

29.82.3.52 `shared_ptr<ParamCurve> Go::CurveOnSurface::spaceCurve ( ) [inline]`

Get a shared pointer to the space curve

#### Returns

a shared pointer to the space curve.

Definition at line 365 of file CurveOnSurface.h.

29.82.3.53 `shared_ptr<const ParamCurve> Go::CurveOnSurface::spaceCurve ( ) const` `[inline]`

Get a constant, shared pointer to the space curve

#### Returns

a const-shared pointer to the space curve.

Definition at line 380 of file CurveOnSurface.h.

29.82.3.54 `virtual std::vector<shared_ptr<ParamCurve> > Go::CurveOnSurface::split ( double param, double fuzzy = DEFAULT_PARAMETER_EPSILON ) const` `[virtual]`

Split curve in a specified parameter value.

Reimplemented from [Go::ParamCurve](#).

29.82.3.55 `virtual double Go::CurveOnSurface::startparam ( ) const` `[virtual]`

Query the start parameter of the curve

#### Returns

the curve's start parameter

Implements [Go::ParamCurve](#).

29.82.3.56 `virtual CurveOnSurface* Go::CurveOnSurface::subCurve ( double from_par, double to_par, double fuzzy = DEFAULT_PARAMETER_EPSILON ) const` `[virtual]`

Returns a curve which is a part of this curve. The result is NEWed, so the user is responsible for deleting it. NB: It is not guaranteed that the [ParamCurve](#) that is returned is of the same type as the curve itself. Thus, the returned curve might be a [SplineCurve](#).

#### Parameters

|                 |                                                                                                                                                                                                   |
|-----------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>from_par</i> | start value of parameter interval that will define the subcurve                                                                                                                                   |
| <i>to_par</i>   | end value of parameter interval that will define the subcurve                                                                                                                                     |
| <i>fuzzy</i>    | since subCurve works on those curves who are spline-based, this tolerance defines how close the start and end parameter must be to an existing knot in order to be considered <i>on</i> the knot. |

#### Returns

a pointer to a new subcurve which represents the part of the curve between 'from\_par' and 'to\_par'. It will be spline-based and have a k-regular knotvector. The user is responsible for deleting this subcurve when it is no longer needed.

Implements [Go::ParamCurve](#).

29.82.3.57 **bool** Go::CurveOnSurface::translateParameterCurve ( **const Point & dir** )

Translate an existing parameter curve in a given direction Return value: Whether or not the translation was possible

29.82.3.58 **bool** Go::CurveOnSurface::translateSwapParameterCurve ( **const Point & dir, double sgn, int pdir** )

Translate an existing parameter curve in a given direction and swap sign is specified Return value: Whether or not the translation was possible

29.82.3.59 **shared\_ptr<ParamSurface>** Go::CurveOnSurface::underlyingSurface ( ) `[inline]`

Get a shared pointer to the underlying surface

#### Returns

a shared pointer to the underlying surface

Definition at line 355 of file CurveOnSurface.h.

29.82.3.60 **shared\_ptr<const ParamSurface>** Go::CurveOnSurface::underlyingSurface ( ) **const** `[inline]`

Get a constant, shared pointer to the underlying surface

#### Returns

a const-pointer to the underlying surface

Definition at line 370 of file CurveOnSurface.h.

29.82.3.61 **void** Go::CurveOnSurface::unsetParameterCurve ( ) `[inline]`

Remove parameter curve information.

Definition at line 276 of file CurveOnSurface.h.

29.82.3.62 **void** Go::CurveOnSurface::unsetSpaceCurve ( ) `[inline]`

Remove space curve information.

Definition at line 308 of file CurveOnSurface.h.

29.82.3.63 **bool** Go::CurveOnSurface::updateCurves ( **double epsge** )

Update curves. Reapproximate the dependent curve within the tolerance epsge.



29.82.3.64 `bool Go::CurveOnSurface::updateCurves ( Point vx1, Point vx2, double epsge )`

Update curves.

29.82.3.65 `bool Go::CurveOnSurface::updateIsoCurves ( )`

Update constant parameter curves.

29.82.3.66 `bool Go::CurveOnSurface::updateIsoCurves ( int constdir, double constpar, int boundary )`

Update constant parameter curves. New constant parameter curve information is provided. Used in for instance change of parameter directions in the associated surface

29.82.3.67 `int Go::CurveOnSurface::whichBoundary ( double tol, bool & same_orientation ) const`

Check if this curve lies at a surface boundary, in that case which one -1 = none, 0 = umin, 1 = umax, 2 = vmin, 3 = vmax NB! same\_orientation is set only if the curve is defined to lie at constant parameter curve. Set to 'false' if direction is opposite that of the surface.

29.82.3.68 `virtual void Go::CurveOnSurface::write ( std::ostream & os ) const [virtual]`

write object to stream

Parameters

|                 |                                   |
|-----------------|-----------------------------------|
| <code>os</code> | stream to which object is written |
|-----------------|-----------------------------------|

Implements [Go::Streamable](#).

The documentation for this class was generated from the following file:

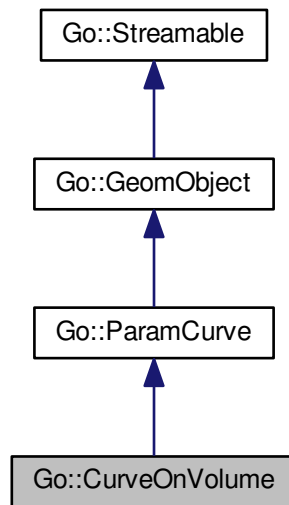
- `gotools-core/include/GoTools/geometry/CurveOnSurface.h`

## 29.83 Go::CurveOnVolume Class Reference

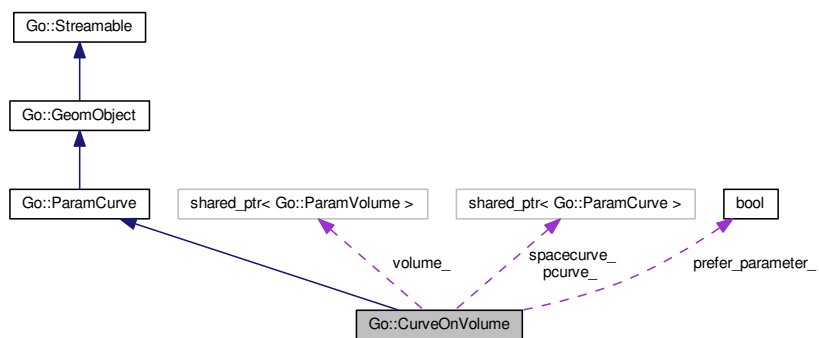
A curve living on a parametric volume. It either has got information about the curve in geometry space and in the parameter domain of the volume or the ability to compute the other representation given one.

```
#include <CurveOnVolume.h>
```

Inheritance diagram for Go::CurveOnVolume:



Collaboration diagram for Go::CurveOnVolume:



## Public Member Functions

- [CurveOnVolume](#) ()
- [CurveOnVolume](#) (shared\_ptr< [ParamVolume](#) > vol, shared\_ptr< [ParamCurve](#) > curve, bool preferparameter)
- [CurveOnVolume](#) (shared\_ptr< [ParamVolume](#) > vol, shared\_ptr< [ParamCurve](#) > pcurve, shared\_ptr< [ParamCurve](#) > spacecurve, bool preferparameter)
- [CurveOnVolume](#) (const [CurveOnVolume](#) &volume\_curve)
- [CurveOnVolume](#) & operator= (const [CurveOnVolume](#) &other)
- virtual [~CurveOnVolume](#) ()
- virtual void [read](#) (std::istream &is)

- virtual void [write](#) (std::ostream &os) [const](#)
- virtual [BoundingBox](#) [boundingBox](#) () [const](#)
- virtual [DirectionCone](#) [directionCone](#) () [const](#)
- virtual int [dimension](#) () [const](#)  
*Dimension of geometry space.*
- virtual [ClassType](#) [instanceType](#) () [const](#)  
*Type of current object.*
- virtual [CurveOnVolume](#) \* [clone](#) () [const](#)
- virtual void [point](#) ([Point](#) &pt, double tpar) [const](#)
- virtual void [point](#) (std::vector< [Point](#) > &pts, double tpar, int derivs, bool from\_right=true) [const](#)
- virtual double [startparam](#) () [const](#)
- virtual double [endparam](#) () [const](#)
- virtual void [reverseParameterDirection](#) (bool switchparam=false)
- virtual void [setParameterInterval](#) (double t1, double t2)  
*Linear reparametrization. The meaning is changed for elementary curves.*
- virtual [SplineCurve](#) \* [geometryCurve](#) ()
- virtual bool [isDegenerate](#) (double degenerate\_epsilon)
- virtual [CurveOnVolume](#) \* [subCurve](#) (double from\_par, double to\_par, double fuzzy=DEFAULT\_PARAMETER\_EPSILON) [const](#)
- virtual std::vector< shared\_ptr< [ParamCurve](#) > > [split](#) (double param, double fuzzy=DEFAULT\_PARAMETER\_EPSILON) [const](#)  
*Split curve in a specified parameter value.*
- virtual void [closestPoint](#) (const [Point](#) &pt, double tmin, double tmax, double &clo\_t, [Point](#) &clo\_pt, double &clo\_dist, double const \*seed=0) [const](#)
- virtual double [length](#) (double tol)
- virtual void [appendCurve](#) ([ParamCurve](#) \*cv, bool reparam=true)
- virtual void [appendCurve](#) ([ParamCurve](#) \*cv, int continuity, double &dist, bool reparam=true)
- void [setUnderlyingVolume](#) (shared\_ptr< [ParamVolume](#) > volume)
- void [setCurves](#) (shared\_ptr< [ParamCurve](#) > spacecurve, shared\_ptr< [ParamCurve](#) > parametercurve)
- void [setSpaceCurve](#) (shared\_ptr< [ParamCurve](#) > spacecurve)
- void [setParameterCurve](#) (shared\_ptr< [ParamCurve](#) > parametercurve)
- void [unsetParameterCurve](#) ()  
*Remove parameter curve information.*
- void [unsetSpaceCurve](#) ()  
*Remove space curve information.*
- virtual double [nextSegmentVal](#) (double par, bool forward, double tol) [const](#)
- shared\_ptr< [ParamVolume](#) > [underlyingVolume](#) ()
- shared\_ptr< [ParamCurve](#) > [parameterCurve](#) ()
- shared\_ptr< [ParamCurve](#) > [spaceCurve](#) ()
- shared\_ptr< const [ParamVolume](#) > [underlyingVolume](#) () [const](#)
- shared\_ptr< const [ParamCurve](#) > [parameterCurve](#) () [const](#)
- shared\_ptr< const [ParamCurve](#) > [spaceCurve](#) () [const](#)
- bool [parPref](#) () [const](#)
- void [setParPref](#) (bool prefer)
- [RectDomain](#) [containingDomain](#) () [const](#)
- [Point](#) [volumeParameter](#) (double crv\_par, const [RectDomain](#) \*domain\_of\_interest=NULL) [const](#)  
*Fetch the volume parameter corresponding to a curve parameter.*

## Static Public Member Functions

- static [ClassType](#) [classType](#) ()

## Protected Attributes

- `shared_ptr< ParamVolume > volume_`  
*The underlying volume.*
- `shared_ptr< ParamCurve > pcurve_`
- `shared_ptr< ParamCurve > spacecurve_`  
*An instance of the curve in the volume. May point to null.*
- `bool prefer_parameter_`  
*Which representation to prefer if both exist.*

## Additional Inherited Members

### 29.83.1 Detailed Description

A curve living on a parametric volume. It either has got information about the curve in geometry space and in the parameter domain of the volume or the ability to compute the other representation given one.

Definition at line 60 of file CurveOnVolume.h.

### 29.83.2 Constructor & Destructor Documentation

#### 29.83.2.1 Go::CurveOnVolume::CurveOnVolume ( )

Define an empty [CurveOnVolume](#) that can be assigned or [read\(\)](#) into

#### 29.83.2.2 Go::CurveOnVolume::CurveOnVolume ( shared\_ptr< ParamVolume > vol, shared\_ptr< ParamCurve > curve, bool preferparameter )

Construct a [CurveOnVolume](#) by specifying the volume and either a space curve or a curve in the parameter plane. Does not clone any of the input, just sets (smart) pointers.

#### Parameters

|                        |                                                                                                                                                                                                                                                                                       |
|------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>vol</i>             | pointer to the underlying volume                                                                                                                                                                                                                                                      |
| <i>curve</i>           | pointer to the curve specifying the <a href="#">CurveOnVolume</a> . This curve may either be in the parametric domain of the volume (3D curve), or a space curve that the user has assured to be coincident with the volume. 'preferparameter' specifies which kind of curve this is. |
| <i>preferparameter</i> | if this is set to 'true', then 'curve' is assumed to be a curve in the parametric domain of the surface. Otherwise, it is assumed to be a space (3D) curve.                                                                                                                           |

#### 29.83.2.3 Go::CurveOnVolume::CurveOnVolume ( shared\_ptr< ParamVolume > vol, shared\_ptr< ParamCurve > pcurve, shared\_ptr< ParamCurve > spacecurve, bool preferparameter )

#### 29.83.2.4 Go::CurveOnVolume::CurveOnVolume ( const CurveOnVolume & volume\_curve )

Copy constructor. The copy constructor will not [clone\(\)](#) the underlying volume, but it will [clone\(\)](#) both the parametric and the spatial curve.

## Parameters

|                     |                                                                                       |
|---------------------|---------------------------------------------------------------------------------------|
| <i>volume_curve</i> | the <a href="#">CurveOnVolume</a> to copy into 'this' <a href="#">CurveOnVolume</a> . |
|---------------------|---------------------------------------------------------------------------------------|

29.83.2.5 `virtual Go::CurveOnVolume::~~CurveOnVolume ( ) [virtual]`

Destructor. Trivial because memory is managed by `shared_ptr`.

### 29.83.3 Member Function Documentation

29.83.3.1 `virtual void Go::CurveOnVolume::appendCurve ( ParamCurve * cv, bool reparam = true ) [virtual]`

append a curve to this curve, with eventual reparametrization NB: This virtual member function currently only works for `SplineCurves` and `CurveOnSurfaces`. Moreover, 'this' curve and the 'cv' curve must be of the same type.

## Parameters

|                |                                                          |
|----------------|----------------------------------------------------------|
| <i>cv</i>      | the curve to append to 'this' curve.                     |
| <i>reparam</i> | specify whether or not there should be reparametrization |

Implements [Go::ParamCurve](#).

29.83.3.2 `virtual void Go::CurveOnVolume::appendCurve ( ParamCurve * cv, int continuity, double & dist, bool reparam = true ) [virtual]`

append a curve to this curve, with eventual reparametrization

## Parameters

|                   |                                                                                                       |
|-------------------|-------------------------------------------------------------------------------------------------------|
| <i>cv</i>         | the curve to append to 'this' curve.                                                                  |
| <i>continuity</i> | the required continuity at the transition. Can be $G^{(-1)}$ and upwards.                             |
| <i>dist</i>       | a measure of the local distortion around the transition in order to achieve the specified continuity. |
| <i>reparam</i>    | specify whether or not there should be reparametrization                                              |

Implements [Go::ParamCurve](#).

29.83.3.3 `virtual BoundingBox Go::CurveOnVolume::boundingBox ( ) const [virtual]`

Axis align box surrounding this object Computed with respect to the space curve if this one exists, otherwise the underlying surface

Implements [Go::GeomObject](#).

29.83.3.4 **static ClassType** Go::CurveOnVolume::classType ( ) [inline],[static]

Definition at line 124 of file CurveOnVolume.h.

29.83.3.5 **virtual CurveOnVolume\*** Go::CurveOnVolume::clone ( ) const [inline],[virtual]

The clone-function is inherited from [GeomObject](#), but overridden here to get a covariant return type (for those compilers that allow this).

Implements [Go::ParamCurve](#).

Definition at line 130 of file CurveOnVolume.h.

29.83.3.6 **virtual void** Go::CurveOnVolume::closestPoint ( const Point & pt, double tmin, double tmax, double & clo\_t, Point & clo\_pt, double & clo\_dist, double const \* seed = 0 ) const [virtual]

Compute the closest point from an interval of this curve to a specified point.

Parameters

|                 |                                                                                                                                          |
|-----------------|------------------------------------------------------------------------------------------------------------------------------------------|
| <i>pt</i>       | point we want to find the closest point to                                                                                               |
| <i>tmin</i>     | start parameter of search interval                                                                                                       |
| <i>tmax</i>     | end parameter of search interval                                                                                                         |
| <i>clo_t</i>    | upon function return, 'clo_t' will contain the parameter value of the closest point found.                                               |
| <i>clo_pt</i>   | upon function return, 'clo_pt' will contain the position of the closest point found.                                                     |
| <i>clo_dist</i> | upon function return, 'clo_dist' will contain the distance between 'pt' and the closest point found.                                     |
| <i>seed</i>     | pointer to initial guess value, provided by the user (can be 0, for which the algorithm will determine a (hopefully) reasonable choice). |

Implements [Go::ParamCurve](#).

29.83.3.7 **RectDomain** Go::CurveOnVolume::containingDomain ( ) const

Get the rectangle enclosing the underlying volume's parametric domain.

Returns

the [RectDomain](#) for the underlying volume.

29.83.3.8 **virtual int** Go::CurveOnVolume::dimension ( ) const [virtual]

Dimension of geometry space.

Implements [Go::GeomObject](#).

29.83.3.9 virtual **DirectionCone** Go::CurveOnVolume::directionCone ( ) const [virtual]

**Cone** surrounding the set of tangent directions corresponding to the surface. Only computed if the space curve exists

Implements [Go::ParamCurve](#).

29.83.3.10 virtual **double** Go::CurveOnVolume::endparam ( ) const [virtual]

Query the end parameter of the curve

Returns

the curve's end parameter

Implements [Go::ParamCurve](#).

29.83.3.11 virtual **SplineCurve\*** Go::CurveOnVolume::geometryCurve ( ) [virtual]

If the definition of this [ParamCurve](#) contains a [SplineCurve](#) describing its spatial shape, then this function will return a pointer to this [SplineCurve](#). Otherwise it will return a null pointer. The returned curve is NEWed, so the user is responsible for deleting it. This function may have side-effects.

Returns

a pointer to a [SplineCurve](#) representation of the [ParamCurve](#), if it exists. Null pointer otherwise.

Implements [Go::ParamCurve](#).

29.83.3.12 virtual **ClassType** Go::CurveOnVolume::instanceType ( ) const [virtual]

Type of current object.

Implements [Go::GeomObject](#).

29.83.3.13 virtual **bool** Go::CurveOnVolume::isDegenerate ( **double** *degenerate\_epsilon* ) [virtual]

Query whether the curve is degenerate (collapsed into a single point).

Parameters

|                           |                                                                                                                                                   |
|---------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>degenerate_epsilon</i> | the tolerance used in determine whether the curve is degenerate. A curve is considered degenerate if its total length is shorter than this value. |
|---------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------|

**Returns**

`true` if the curve is degenerate, `false` otherwise.

Implements [Go::ParamCurve](#).

**29.83.3.14** `virtual double Go::CurveOnVolume::length ( double tol ) [virtual]`

Compute the total length of this curve up to some tolerance

**Parameters**

|            |                                                                                                                                           |
|------------|-------------------------------------------------------------------------------------------------------------------------------------------|
| <i>tol</i> | the relative tolerance when approximating the length, i.e. stop iteration when error becomes smaller than <code>tol/(curve length)</code> |
|------------|-------------------------------------------------------------------------------------------------------------------------------------------|

**Returns**

the length calculated

Implements [Go::ParamCurve](#).

**29.83.3.15** `virtual double Go::CurveOnVolume::nextSegmentVal ( double par, bool forward, double tol ) const [virtual]`

Inherited from [ParamCurve](#). If the parametric curve is set to be the 'preferred' one, this function will return the next segment value for the parametric curve; otherwise it will return the next segment value for the spatial 3D curve. See also [ParamCurve::nextSegmentVal\(\)](#)

Reimplemented from [Go::ParamCurve](#).

**29.83.3.16** `CurveOnVolume& Go::CurveOnVolume::operator= ( const CurveOnVolume & other )`

Assignment operator. Like the copy constructor, the assignment operator [clone\(\)](#)s the curves, and not the volume.

**Parameters**

|              |                                                                                       |
|--------------|---------------------------------------------------------------------------------------|
| <i>other</i> | the <a href="#">CurveOnVolume</a> to copy into 'this' <a href="#">CurveOnVolume</a> . |
|--------------|---------------------------------------------------------------------------------------|

**29.83.3.17** `shared_ptr<ParamCurve> Go::CurveOnVolume::parameterCurve ( ) [inline]`

Get a shared pointer to the curve in the parameter domain.

**Returns**

a shared pointer to the curve in the parameter domain

Definition at line 252 of file [CurveOnVolume.h](#).



29.83.3.18 `shared_ptr<const ParamCurve> Go::CurveOnVolume::parameterCurve ( ) const [inline]`

Get a constant, shared pointer to the curve in the parameter domain.

#### Returns

a const-pointer to the curve in the parameter domain.

Definition at line 267 of file CurveOnVolume.h.

29.83.3.19 `bool Go::CurveOnVolume::parPref ( ) const [inline]`

Query whether the parameter curve or the space curve is preferred for computation in this object.

Definition at line 277 of file CurveOnVolume.h.

29.83.3.20 `virtual void Go::CurveOnVolume::point ( Point & pt, double tpar ) const [virtual]`

Evaluate the curve's position at a given parameter

#### Parameters

|             |                                                                      |
|-------------|----------------------------------------------------------------------|
| <i>pt</i>   | the evaluated position will be written to this <a href="#">Point</a> |
| <i>tpar</i> | the parameter for which we wish to evaluate the curve                |

Implements [Go::ParamCurve](#).

29.83.3.21 `virtual void Go::CurveOnVolume::point ( std::vector< Point > & pts, double tpar, int derivs, bool from_right = true ) const [virtual]`

Inherited from [ParamCurve](#). Only works for 'derivs' = 0 or 1.

#### See also

[ParamCurve::point\(\)](#)

Implements [Go::ParamCurve](#).

29.83.3.22 `virtual void Go::CurveOnVolume::read ( std::istream & is ) [virtual]`

read object from stream

#### Parameters

|           |                                  |
|-----------|----------------------------------|
| <i>is</i> | stream from which object is read |
|-----------|----------------------------------|

Implements [Go::Streamable](#).

29.83.3.23 `virtual void Go::CurveOnVolume::reverseParameterDirection ( bool switchparam = false ) [virtual]`

Set the parameter direction of the curve. The curve's parameter interval will always remain constant, but by flipping the parameter direction, the curve will be traced the opposite way when moving a parameter over the parameter interval.

#### Parameters

|                    |                                                                                                                                                   |
|--------------------|---------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>switchparam</i> | if true, and the curve is 2D, the x and y coordinates should be swapped. This is used when turning the orientation of bounded (trimmed) surfaces. |
|--------------------|---------------------------------------------------------------------------------------------------------------------------------------------------|

Implements [Go::ParamCurve](#).

29.83.3.24 `void Go::CurveOnVolume::setCurves ( shared_ptr< ParamCurve > spacecurve, shared_ptr< ParamCurve > parametercurve ) [inline]`

Replace the curves describing the curve on volume. The curve preference is not changed

Definition at line 201 of file CurveOnVolume.h.

29.83.3.25 `void Go::CurveOnVolume::setParameterCurve ( shared_ptr< ParamCurve > parametercurve ) [inline]`

Replace the parameter curve corresponding to this curve on volume curve. Used for instance in relation to reparameterizations of the related volume. Use with care!

Definition at line 218 of file CurveOnVolume.h.

29.83.3.26 `virtual void Go::CurveOnVolume::setParameterInterval ( double t1, double t2 ) [virtual]`

Linear reparametrization. The meaning is changed for elementary curves.

Implements [Go::ParamCurve](#).

29.83.3.27 `void Go::CurveOnVolume::setParPref ( bool prefer ) [inline]`

Definition at line 280 of file CurveOnVolume.h.

29.83.3.28 `void Go::CurveOnVolume::setSpaceCurve ( shared_ptr< ParamCurve > spacecurve ) [inline]`

Replace the space curve corresponding to this curve on volume curve. Use with care!

Definition at line 210 of file CurveOnVolume.h.

29.83.3.29 `void Go::CurveOnVolume::setUnderlyingVolume ( shared_ptr< ParamVolume > volume ) [inline]`

Set the underlying surface to the one pointed to by the argument

## Parameters

|                |                                                                                               |
|----------------|-----------------------------------------------------------------------------------------------|
| <i>surface</i> | the pointer to the surface we will set as underlying for this <a href="#">CurveOnVolume</a> . |
|----------------|-----------------------------------------------------------------------------------------------|

Definition at line 196 of file CurveOnVolume.h.

**29.83.3.30** `shared_ptr<ParamCurve> Go::CurveOnVolume::spaceCurve ( ) [inline]`

Get a shared pointer to the space curve

## Returns

a shared pointer to the space curve.

Definition at line 257 of file CurveOnVolume.h.

**29.83.3.31** `shared_ptr<const ParamCurve> Go::CurveOnVolume::spaceCurve ( ) const [inline]`

Get a constant, shared pointer to the space curve

## Returns

a const-shared pointer to the space curve.

Definition at line 272 of file CurveOnVolume.h.

**29.83.3.32** `virtual std::vector<shared_ptr<ParamCurve>> Go::CurveOnVolume::split ( double param, double fuzzy = DEFAULT_PARAMETER_EPSILON ) const [virtual]`

Split curve in a specified parameter value.

Reimplemented from [Go::ParamCurve](#).

**29.83.3.33** `virtual double Go::CurveOnVolume::startparam ( ) const [virtual]`

Query the start parameter of the curve

## Returns

the curve's start parameter

Implements [Go::ParamCurve](#).

**29.83.3.34** `virtual CurveOnVolume* Go::CurveOnVolume::subCurve ( double from_par, double to_par, double fuzzy = DEFAULT_PARAMETER_EPSILON ) const [virtual]`

Returns a curve which is a part of this curve. The result is NEWed, so the user is responsible for deleting it. NB: It is not guaranteed that the [ParamCurve](#) that is returned is of the same type as the curve itself. Thus, the returned curve might be a [SplineCurve](#).

## Parameters

|                 |                                                                                                                                                                                                   |
|-----------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>from_par</i> | start value of parameter interval that will define the subcurve                                                                                                                                   |
| <i>to_par</i>   | end value of parameter interval that will define the subcurve                                                                                                                                     |
| <i>fuzzy</i>    | since subCurve works on those curves who are spline-based, this tolerance defines how close the start and end parameter must be to an existing knot in order to be considered <i>on</i> the knot. |

## Returns

a pointer to a new subcurve which represents the part of the curve between 'from\_par' and 'to\_par'. It will be spline-based and have a k-regular knotvector. The user is responsible for deleting this subcurve when it is no longer needed.

Implements [Go::ParamCurve](#).

**29.83.3.35** `shared_ptr<ParamVolume> Go::CurveOnVolume::underlyingVolume ( ) [inline]`

Get a shared pointer to the underlying volume

## Returns

a shared pointer to the underlying volume

Definition at line 247 of file CurveOnVolume.h.

**29.83.3.36** `shared_ptr<const ParamVolume> Go::CurveOnVolume::underlyingVolume ( ) const [inline]`

Get a constant, shared pointer to the underlying volume

## Returns

a const-pointer to the underlying volume

Definition at line 262 of file CurveOnVolume.h.

**29.83.3.37** `void Go::CurveOnVolume::unsetParameterCurve ( ) [inline]`

Remove parameter curve information.

Definition at line 224 of file CurveOnVolume.h.

**29.83.3.38** `void Go::CurveOnVolume::unsetSpaceCurve ( ) [inline]`

Remove space curve information.

Definition at line 231 of file CurveOnVolume.h.

**29.83.3.39** `Point Go::CurveOnVolume::volumeParameter ( double crv_par, const RectDomain * domain_of_interest = NULL ) const`

Fetch the volume parameter corresponding to a curve parameter.

**29.83.3.40** `virtual void Go::CurveOnVolume::write ( std::ostream & os ) const [virtual]`

write object to stream

## Parameters

|           |                                   |
|-----------|-----------------------------------|
| <i>os</i> | stream to which object is written |
|-----------|-----------------------------------|

Implements [Go::Streamable](#).

## 29.83.4 Member Data Documentation

**29.83.4.1** `shared_ptr<ParamCurve> Go::CurveOnVolume::pcurve_` [protected]

The 2D curve in the parameter domain of the volume. May point to null.

Definition at line 297 of file CurveOnVolume.h.

**29.83.4.2** `bool Go::CurveOnVolume::prefer_parameter_` [protected]

Which representation to prefer if both exist.

Definition at line 301 of file CurveOnVolume.h.

**29.83.4.3** `shared_ptr<ParamCurve> Go::CurveOnVolume::spacecurve_` [protected]

An instance of the curve in the volume. May point to null.

Definition at line 299 of file CurveOnVolume.h.

**29.83.4.4** `shared_ptr<ParamVolume> Go::CurveOnVolume::volume_` [protected]

The underlying volume.

Definition at line 294 of file CurveOnVolume.h.

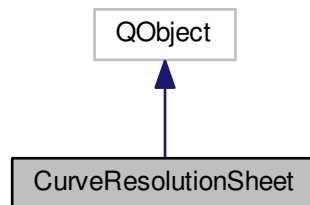
The documentation for this class was generated from the following file:

- [trivariate/include/GoTools/trivariate/CurveOnVolume.h](#)

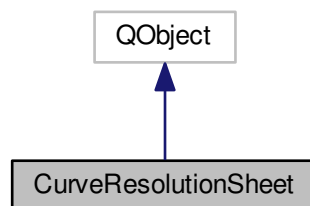
## 29.84 CurveResolutionSheet Class Reference

```
#include <CurveResolutionSheet.h>
```

Inheritance diagram for CurveResolutionSheet:



Collaboration diagram for CurveResolutionSheet:



### Public Slots

- void [ok](#) ()

### Signals

- void [return\\_value](#) (int)

### Public Member Functions

- [CurveResolutionSheet](#) (int res=500)
- virtual void [createSheet](#) (QWidget \*parent, [gvObserver](#) \*obs)

### 29.84.1 Detailed Description

Ui\_CurveResolutionSheet:

Definition at line 52 of file CurveResolutionSheet.h.

### 29.84.2 Constructor & Destructor Documentation

29.84.2.1 `CurveResolutionSheet::CurveResolutionSheet ( int res = 500 )` `[inline]`

Definition at line 58 of file CurveResolutionSheet.h.

### 29.84.3 Member Function Documentation

29.84.3.1 `virtual void CurveResolutionSheet::createSheet ( QWidget * parent, gvObserver * obs )` `[virtual]`

29.84.3.2 `void CurveResolutionSheet::ok ( )` `[slot]`

29.84.3.3 `void CurveResolutionSheet::return_value ( int )` `[signal]`

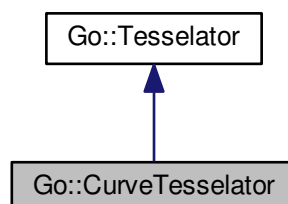
The documentation for this class was generated from the following file:

- [viewlib/include/GoTools/viewlib/CurveResolutionSheet.h](#)

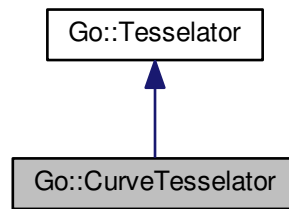
## 29.85 Go::CurveTesselator Class Reference

```
#include <CurveTesselator.h>
```

Inheritance diagram for Go::CurveTesselator:



Collaboration diagram for Go::CurveTesselator:



## Public Member Functions

- [CurveTesselator](#) (`const ParamCurve &curve`)  
*Constructor.*
- virtual [~CurveTesselator](#) ()  
*Destructor.*
- virtual void [tesselate](#) ()  
*Perform tessellation. The density depends on the resolution.*
- `shared_ptr< LineStrip >` [getMesh](#) ()  
*Fetch the linear approximation represented as a [LineStrip](#).*
- void [changeRes](#) (int n)  
*Set tessellation resolution (number of points to evaluate)*
- void [getRes](#) (int &n)  
*Fetch info about resolution.*

### 29.85.1 Detailed Description

Tesselate curve to produce a linear approximation of the curve for visualization purposed

Definition at line 53 of file CurveTesselator.h.

### 29.85.2 Constructor & Destructor Documentation

29.85.2.1 `Go::CurveTesselator::CurveTesselator ( const ParamCurve & curve )` [`inline`]

Constructor.

Definition at line 57 of file CurveTesselator.h.

29.85.2.2 `virtual Go::CurveTesselator::~~CurveTesselator ( )` [`virtual`]

Destructor.



### 29.85.3 Member Function Documentation

29.85.3.1 `void Go::CurveTesselator::changeRes ( int n )` `[inline]`

Set tessellation resolution (number of points to evaluate)

Definition at line 84 of file CurveTesselator.h.

29.85.3.2 `shared_ptr<LineStrip> Go::CurveTesselator::getMesh ( )` `[inline]`

Fetch the linear approximation represented as a [LineStrip](#).

Definition at line 70 of file CurveTesselator.h.

29.85.3.3 `void Go::CurveTesselator::getRes ( int & n )` `[inline]`

Fetch info about resolution.

Definition at line 90 of file CurveTesselator.h.

29.85.3.4 `virtual void Go::CurveTesselator::tessellate ( )` `[virtual]`

Perform tessellation. The density depends on the resolution.

Implements [Go::Tesselator](#).

The documentation for this class was generated from the following file:

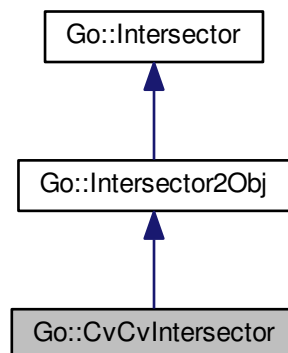
- `gtools-core/include/GoTools/tesselator/CurveTesselator.h`

## 29.86 Go::CvCvIntersector Class Reference

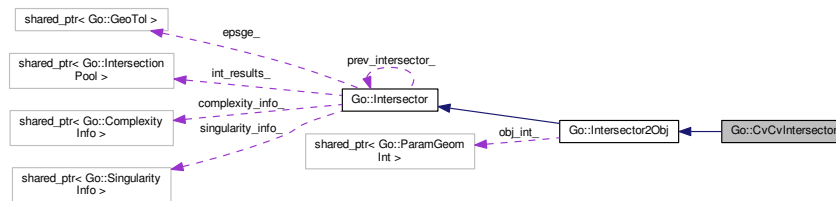
Class that performs intersection between two parametric curves.

```
#include <CvCvIntersector.h>
```

Inheritance diagram for Go::CvCvIntersector:



Collaboration diagram for `Go::CvCvIntersector`:



## Public Member Functions

- `CvCvIntersector` (`shared_ptr< ParamGeomInt > curve1`, `shared_ptr< ParamGeomInt > curve2`, `double epsge`, `Intersector *prev=0`, `int eliminated_parameter=-1`, `double eliminated_value=0`)
- `CvCvIntersector` (`shared_ptr< ParamGeomInt > curve1`, `shared_ptr< ParamGeomInt > curve2`, `shared_ptr< GeoTol > epsge`, `Intersector *prev=0`, `int eliminated_parameter=-1`, `double eliminated_value=0`)
- virtual `~CvCvIntersector` ()  
*Destructor.*
- virtual `int numParams` () `const`

## Protected Member Functions

- virtual `shared_ptr< Intersector > lowerOrderIntersector` (`shared_ptr< ParamGeomInt > obj1`, `shared_ptr< ParamGeomInt > obj2`, `Intersector *prev=0`, `int eliminated_parameter=-1`, `double eliminated_value=0`)
- virtual `int performRotatedBoxTest` (`double eps1`, `double eps2`)
- virtual `bool foundIntersectionNearBoundary` ()
- virtual `int simpleCase2` (`Point &axis1`, `Point &axis2`)
- virtual `int checkCoincidence` ()
- virtual `void microCase` ()
- virtual `int updateIntersections` ()
- virtual `int repairIntersections` ()
- virtual `int linearCase` ()
- virtual `int doSubdivide` ()
- virtual `void writeOut` ()

## Additional Inherited Members

### 29.86.1 Detailed Description

Class that performs intersection between two parametric curves.

Definition at line 52 of file `CvCvIntersector.h`.

### 29.86.2 Constructor & Destructor Documentation

29.86.2.1 `Go::CvCvIntersector::CvCvIntersector` (`shared_ptr< ParamGeomInt > curve1`, `shared_ptr< ParamGeomInt > curve2`, `double epsge`, `Intersector * prev = 0`, `int eliminated_parameter = -1`, `double eliminated_value = 0` )

Constructor. The last two variables pertains only if the parent has one more parameter than the `Intersector` to be constructed.

## Parameters

|                             |                                                                           |
|-----------------------------|---------------------------------------------------------------------------|
| <i>curve1</i>               | of type <a href="#">ParamCurveInt</a> .                                   |
| <i>curve2</i>               | of type <a href="#">ParamCurveInt</a> .                                   |
| <i>epsge</i>                | the associated tolerance.                                                 |
| <i>prev</i>                 | the "parent" <a href="#">Intersector</a> (0 if there is no parent).       |
| <i>eliminated_parameter</i> | the index of the parameter that was removed from the parent <i>prev</i> . |
| <i>eliminated_value</i>     | the value of the parameter that was removed from the parent <i>prev</i> . |

**29.86.2.2** `Go::CvCvIntersector::CvCvIntersector ( shared_ptr< ParamGeomInt > curve1, shared_ptr< ParamGeomInt > curve2, shared_ptr< GeoTol > epsge, Intersector * prev = 0, int eliminated_parameter = -1, double eliminated_value = 0 )`

Constructor. The last two variables pertains only if the parent has one more parameter than the [Intersector](#) to be constructed.

## Parameters

|                             |                                                                           |
|-----------------------------|---------------------------------------------------------------------------|
| <i>curve1</i>               | of type <a href="#">ParamCurveInt</a> .                                   |
| <i>curve2</i>               | of type <a href="#">ParamCurveInt</a> .                                   |
| <i>epsge</i>                | the associated tolerance.                                                 |
| <i>prev</i>                 | the "parent" <a href="#">Intersector</a> (0 if there is no parent).       |
| <i>eliminated_parameter</i> | the index of the parameter that was removed from the parent <i>prev</i> . |
| <i>eliminated_value</i>     | the value of the parameter that was removed from the parent <i>prev</i> . |

**29.86.2.3** `virtual Go::CvCvIntersector::~~CvCvIntersector ( ) [virtual]`

Destructor.

**29.86.3 Member Function Documentation**

**29.86.3.1** `virtual int Go::CvCvIntersector::checkCoincidence ( ) [protected],[virtual]`

Implements [Go::Intersector2Obj](#).

**29.86.3.2** `virtual int Go::CvCvIntersector::doSubdivide ( ) [protected],[virtual]`

Implements [Go::Intersector2Obj](#).

**29.86.3.3** `virtual bool Go::CvCvIntersector::foundIntersectionNearBoundary ( ) [protected],[virtual]`

Reimplemented from [Go::Intersector2Obj](#).

29.86.3.4 `virtual int Go::CvCvIntersector::linearCase ( ) [protected],[virtual]`

Implements [Go::Intersector2Obj](#).

29.86.3.5 `virtual shared_ptr<Intersector> Go::CvCvIntersector::lowerOrderIntersector ( shared_ptr< ParamGeomInt > obj1, shared_ptr< ParamGeomInt > obj2, Intersector * prev = 0, int eliminated_parameter = -1, double eliminated_value = 0 ) [protected],[virtual]`

Implements [Go::Intersector2Obj](#).

29.86.3.6 `virtual void Go::CvCvIntersector::microCase ( ) [protected],[virtual]`

Implements [Go::Intersector2Obj](#).

29.86.3.7 `virtual int Go::CvCvIntersector::numParams ( ) const [inline],[virtual]`

Return the number of parameter directions for the intersection.

#### Returns

The number of parameter directions for the intersection.

Implements [Go::Intersector](#).

Definition at line 103 of file CvCvIntersector.h.

29.86.3.8 `virtual int Go::CvCvIntersector::performRotatedBoxTest ( double eps1, double eps2 ) [protected],[virtual]`

Reimplemented from [Go::Intersector2Obj](#).

29.86.3.9 `virtual int Go::CvCvIntersector::repairIntersections ( ) [inline],[protected],[virtual]`

Implements [Go::Intersector](#).

Definition at line 128 of file CvCvIntersector.h.

29.86.3.10 `virtual int Go::CvCvIntersector::simpleCase2 ( Point & axis1, Point & axis2 ) [protected],[virtual]`

Reimplemented from [Go::Intersector2Obj](#).

29.86.3.11 `virtual int Go::CvCvIntersector::updateIntersections ( ) [protected],[virtual]`

Implements [Go::Intersector2Obj](#).

29.86.3.12 virtual void Go::CvCvIntersector::writeOut ( ) [protected],[virtual]

Reimplemented from [Go::Intersector2Obj](#).

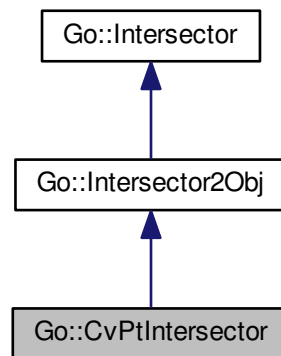
The documentation for this class was generated from the following file:

- intersections/include/GoTools/intersections/[CvCvIntersector.h](#)

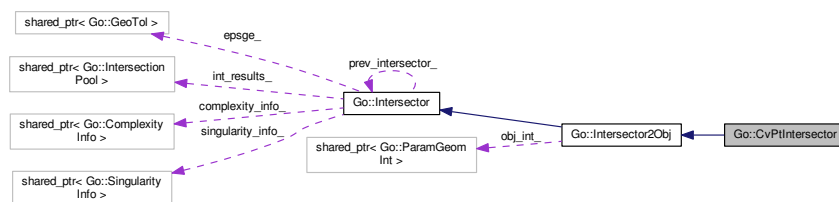
## 29.87 Go::CvPtIntersector Class Reference

```
#include <CvPtIntersector.h>
```

Inheritance diagram for Go::CvPtIntersector:



Collaboration diagram for Go::CvPtIntersector:



### Public Member Functions

- [CvPtIntersector](#) (shared\_ptr< [ParamGeomInt](#) > obj1, shared\_ptr< [ParamGeomInt](#) > obj2, shared\_ptr< [GeoTol](#) > epsge, [Intersector](#) \*prev=0, int eliminated\_parameter=-1, double eliminated\_value=0)
- [CvPtIntersector](#) (shared\_ptr< [ParamGeomInt](#) > obj1, shared\_ptr< [ParamGeomInt](#) > obj2, double epsge, [Intersector](#) \*prev=0, int eliminated\_parameter=-1, double eliminated\_value=0)
- virtual [~CvPtIntersector](#) ()  
*Destructor.*
- virtual int numParams () const

## Protected Member Functions

- virtual `shared_ptr< Intersector > lowerOrderIntersector` (`shared_ptr< ParamGeomInt > obj1`, `shared_ptr< ParamGeomInt > obj2`, `Intersector *prev=0`, `int eliminated_parameter=-1`, `double eliminated_value=0`)
- virtual `int checkCoincidence` ()
- virtual `void microCase` ()
- virtual `int updateIntersections` ()
- virtual `int repairIntersections` ()
- virtual `int linearCase` ()
- virtual `int doSubdivide` ()

## Additional Inherited Members

### 29.87.1 Detailed Description

This class performs intersection between a parametric curve and a point.

Definition at line 53 of file `CvPtIntersector.h`.

### 29.87.2 Constructor & Destructor Documentation

**29.87.2.1** `Go::CvPtIntersector::CvPtIntersector ( shared_ptr< ParamGeomInt > obj1, shared_ptr< ParamGeomInt > obj2, shared_ptr< GeoTol > epsge, Intersector * prev = 0, int eliminated_parameter = -1, double eliminated_value = 0 )`

Constructor. One of the objects should refer to a curve, the other a point (this is not checked compile-time, so we rely on the user to obey this rule). The last two variables are relevant only if the parent has one more parameter than the [Intersector](#) to be constructed.

#### Parameters

|                             |                                                                                                                    |
|-----------------------------|--------------------------------------------------------------------------------------------------------------------|
| <i>obj1</i>                 | either of type <a href="#">ParamCurveInt</a> or <a href="#">ParamPointInt</a> .                                    |
| <i>obj2</i>                 | either of type <a href="#">ParamPointInt</a> or <a href="#">ParamCurveInt</a> (not the same type as <i>obj1</i> ). |
| <i>epsg</i>                 | the associated tolerance.                                                                                          |
| <i>prev</i>                 | the "parent" <a href="#">Intersector</a> (0 if there is no parent).                                                |
| <i>eliminated_parameter</i> | the index (0) of the parameter that was removed from the parent <i>prev</i> .                                      |
| <i>eliminated_value</i>     | the value of the parameter that was removed from the parent <i>prev</i> .                                          |

**29.87.2.2** `Go::CvPtIntersector::CvPtIntersector ( shared_ptr< ParamGeomInt > obj1, shared_ptr< ParamGeomInt > obj2, double epsge, Intersector * prev = 0, int eliminated_parameter = -1, double eliminated_value = 0 )`

Constructor. One of the objects should refer a curve, the other a point (this is not checked compile-time, so we rely on the user to obey this rule). The last two variables are relevant only if the parent has one more parameter than the [Intersector](#) to be constructed.

#### Parameters

|             |                                                                                 |
|-------------|---------------------------------------------------------------------------------|
| <i>obj1</i> | either of type <a href="#">ParamCurveInt</a> or <a href="#">ParamPointInt</a> . |
|-------------|---------------------------------------------------------------------------------|

## Parameters

|                             |                                                                                                                    |
|-----------------------------|--------------------------------------------------------------------------------------------------------------------|
| <i>obj2</i>                 | either of type <a href="#">ParamPointInt</a> or <a href="#">ParamCurveInt</a> (not the same type as <i>obj1</i> ). |
| <i>epsge</i>                | the associated tolerance.                                                                                          |
| <i>prev</i>                 | the "parent" <a href="#">Intersector</a> (0 if there is no parent).                                                |
| <i>eliminated_parameter</i> | the index (0) of the parameter that was removed from the parent <i>prev</i> .                                      |
| <i>eliminated_value</i>     | the value of the parameter that was removed from the parent <i>prev</i> .                                          |

29.87.2.3 `virtual Go::CvPtIntersector::~~CvPtIntersector ( ) [virtual]`

Destructor.

### 29.87.3 Member Function Documentation

29.87.3.1 `virtual int Go::CvPtIntersector::checkCoincidence ( ) [protected],[virtual]`

Implements [Go::Intersector2Obj](#).

29.87.3.2 `virtual int Go::CvPtIntersector::doSubdivide ( ) [protected],[virtual]`

Implements [Go::Intersector2Obj](#).

29.87.3.3 `virtual int Go::CvPtIntersector::linearCase ( ) [protected],[virtual]`

Implements [Go::Intersector2Obj](#).

29.87.3.4 `virtual shared_ptr<Intersector> Go::CvPtIntersector::lowerOrderIntersector ( shared_ptr< ParamGeomInt > obj1, shared_ptr< ParamGeomInt > obj2, Intersector * prev = 0, int eliminated_parameter = -1, double eliminated_value = 0 ) [protected],[virtual]`

Implements [Go::Intersector2Obj](#).

29.87.3.5 `virtual void Go::CvPtIntersector::microCase ( ) [protected],[virtual]`

Implements [Go::Intersector2Obj](#).

29.87.3.6 `virtual int Go::CvPtIntersector::numParams ( ) const [inline],[virtual]`

Return the number of parameter directions for the object.

#### Returns

the number of parameter directions

Implements [Go::Intersector](#).

Definition at line 110 of file `CvPtIntersector.h`.

29.87.3.7 `virtual int Go::CvPtIntersector::repairIntersections ( ) [inline],[protected],[virtual]`

Implements [Go::Intersector](#).

Definition at line 129 of file CvPtIntersector.h.

29.87.3.8 `virtual int Go::CvPtIntersector::updateIntersections ( ) [protected],[virtual]`

Implements [Go::Intersector2Obj](#).

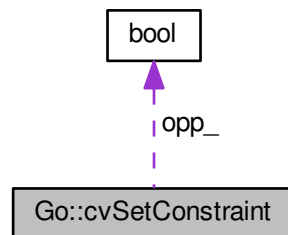
The documentation for this class was generated from the following file:

- intersections/include/GoTools/intersections/CvPtIntersector.h

## 29.88 Go::cvSetConstraint Struct Reference

```
#include <SmoothCurveSet.h>
```

Collaboration diagram for Go::cvSetConstraint:



### Public Member Functions

- `cvSetConstraint` (int cv1\_id, double cv1\_par, int cv1\_der, int cv2\_id, double cv2\_par, int cv2\_der, bool opp)

### Public Attributes

- int `cv1_id_`  
*Index of first curve.*
- double `cv1_par_`  
*Parameter in first curve.*
- int `cv1_der_`  
*The derivative of first curve involved in the constraint.*
- int `cv2_id_`  
*Index of second curve.*
- double `cv2_par_`  
*Parameter in second curve.*
- int `cv2_der_`  
*The derivative of second curve involved in the constraint.*
- bool `opp_`  
*If true the evaluation of the second curve should be negated.*



### 29.88.1 Detailed Description

Side constraint on modification of curve set. Defines a specific relations between two curves in given parameter values

Definition at line 51 of file SmoothCurveSet.h.

### 29.88.2 Constructor & Destructor Documentation

29.88.2.1 `Go::cvSetConstraint::cvSetConstraint ( int cv1_id, double cv1_par, int cv1_der, int cv2_id, double cv2_par, int cv2_der, bool opp ) [inline]`

Definition at line 58 of file SmoothCurveSet.h.

### 29.88.3 Member Data Documentation

29.88.3.1 `int Go::cvSetConstraint::cv1_der_`

The derivative of first curve involved in the constraint.

Definition at line 69 of file SmoothCurveSet.h.

29.88.3.2 `int Go::cvSetConstraint::cv1_id_`

Index of first curve.

Definition at line 65 of file SmoothCurveSet.h.

29.88.3.3 `double Go::cvSetConstraint::cv1_par_`

Parameter in first curve.

Definition at line 67 of file SmoothCurveSet.h.

29.88.3.4 `int Go::cvSetConstraint::cv2_der_`

The derivative of second curve involved in the constraint.

Definition at line 75 of file SmoothCurveSet.h.

29.88.3.5 `int Go::cvSetConstraint::cv2_id_`

Index of second curve.

Definition at line 71 of file SmoothCurveSet.h.

### 29.88.3.6 `double Go::cvSetConstraint::cv2_par_`

Parameter in second curve.

Definition at line 73 of file `SmoothCurveSet.h`.

### 29.88.3.7 `bool Go::cvSetConstraint::opp_`

If true the evaluation of the second curve should be negated.

Definition at line 77 of file `SmoothCurveSet.h`.

The documentation for this struct was generated from the following file:

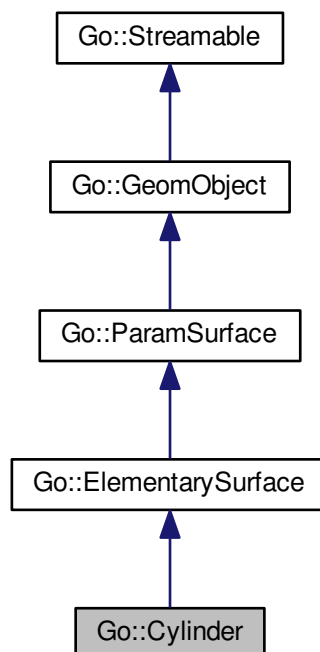
- [gotools-core/include/GoTools/creators/SmoothCurveSet.h](#)

## 29.89 `Go::Cylinder` Class Reference

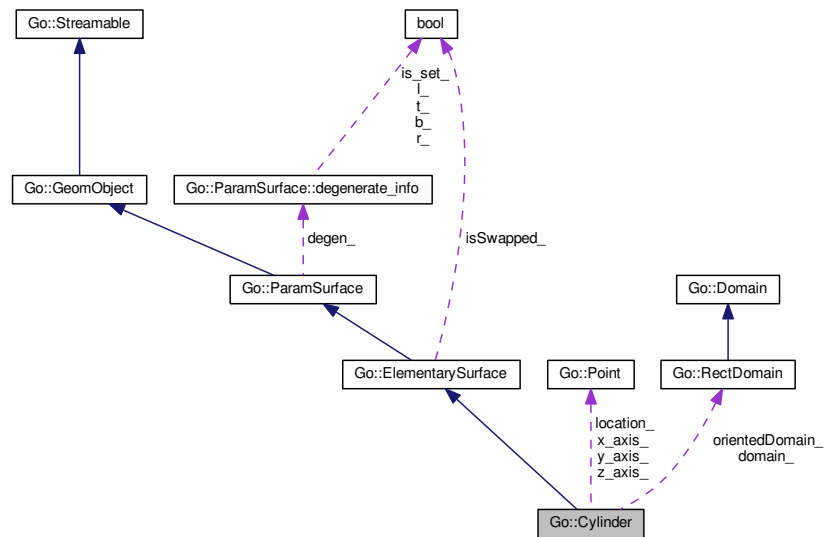
Class that represents a cylinder. It is a subclass of [ElementarySurface](#), and thus has a parametrization and is non-selfintersecting.

```
#include <Cylinder.h>
```

Inheritance diagram for `Go::Cylinder`:



Collaboration diagram for Go::Cylinder:



## Public Member Functions

- [Cylinder](#) ()
- [Cylinder](#) (double radius, [Point](#) location, [Point](#) z\_axis, [Point](#) x\_axis, bool isSwapped=false)
- virtual [~Cylinder](#) ()  
*Virtual destructor - ensures safe inheritance.*
- virtual void [read](#) (std::istream &is)
- virtual void [write](#) (std::ostream &os) const
- virtual int [dimension](#) () const  
*Return the dimension of the space in which the object lies (usually 2 or 3)*
- virtual [ClassType](#) [instanceType](#) () const  
*Return the class type identifier of a given, derived instance of [GeomObject](#).*
- virtual [BoundingBox](#) [boundingBox](#) () const  
*Return empty box if infinite cylinder.*
- virtual [Cylinder](#) \* [clone](#) () const
- const [RectDomain](#) & [parameterDomain](#) () const
- std::vector< [CurveLoop](#) > [allBoundaryLoops](#) (double degenerate\_epsilon=DEFAULT\_SPACE\_EPSILON) const
- [DirectionCone](#) [normalCone](#) () const
- [DirectionCone](#) [tangentCone](#) (bool paddir\_is\_u) const
- void [point](#) ([Point](#) &pt, double upar, double vpar) const
- void [point](#) (std::vector< [Point](#) > &pts, double upar, double vpar, int derivs, bool u\_from\_right=true, bool v\_from\_right=true, double resolution=1.0e-12) const
- void [normal](#) ([Point](#) &n, double upar, double vpar) const
- std::vector< shared\_ptr< [ParamCurve](#) > > [constParamCurves](#) (double parameter, bool paddir\_is\_u) const
- shared\_ptr< [ParamCurve](#) > [constParamCurve](#) (double parameter, bool paddir\_is\_u, double from, double to) const
- std::vector< shared\_ptr< [ParamSurface](#) > > [subSurfaces](#) (double from\_upar, double from\_vpar, double to\_upar, double to\_vpar, double fuzzy=DEFAULT\_PARAMETER\_EPSILON) const
- double [nextSegmentVal](#) (int dir, double par, bool forward, double tol) const

- void `closestPoint` (`const Point` &pt, `double` &clo\_u, `double` &clo\_v, `Point` &clo\_pt, `double` &clo\_dist, `double` epsilon, `const RectDomain` \*domain\_of\_interest=NULL, `double` \*seed=0) `const`
- void `closestBoundaryPoint` (`const Point` &pt, `double` &clo\_u, `double` &clo\_v, `Point` &clo\_pt, `double` &clo\_dist, `double` epsilon, `const RectDomain` \*rd=NULL, `double` \*seed=0) `const`
- void `getBoundaryInfo` (`Point` &pt1, `Point` &pt2, `double` epsilon, `SplineCurve` \*&cv, `SplineCurve` \*&crosscv, `double` knot\_tol=1e-05) `const`
- `bool` `isDegenerate` (`bool` &b, `bool` &r, `bool` &t, `bool` &l, `double` tolerance) `const`
- virtual void `getDegenerateCorners` (`std::vector`< `Point` > &deg\_corners, `double` tol) `const`  
*Check for paralell and anti paralell partial derivatives in surface corners.*
- virtual `shared_ptr`< `ElementaryCurve` > `getElementaryParamCurve` (`ElementaryCurve` \*space\_crv, `double` tol, `const Point` \*start\_par\_pt=NULL, `const Point` \*end\_par\_pt=NULL) `const`
- `double` `getRadius` () `const`  
*Cylinder radius.*
- `Point` `getLocation` () `const`  
*Point on cylinder axis.*
- void `getCoordinateAxes` (`Point` &x\_axis, `Point` &y\_axis, `Point` &z\_axis) `const`  
*Local coordinate axes. The z\_axis corresponds to the cylinder axis.*
- virtual void `setParameterBounds` (`double` from\_upar, `double` from\_vpar, `double` to\_upar, `double` to\_vpar)  
*Limit the cylinder surface by limiting the parameter domain.*
- void `setParamBoundsU` (`double` from\_upar, `double` to\_upar)
- void `setParamBoundsV` (`double` from\_vpar, `double` to\_vpar)
- `bool` `isBounded` () `const`
- `bool` `isClosed` (`bool` &closed\_dir\_u, `bool` &closed\_dir\_v) `const`  
*Check if the surface is closed.*
- `Cylinder` \* `subSurface` (`double` from\_upar, `double` from\_vpar, `double` to\_upar, `double` to\_vpar, `double` fuzzy=DEFAULT\_PARAMETER\_EPSILON) `const`
- virtual `SplineSurface` \* `geometrySurface` () `const`  
*Create a SplineSurface representation of the cylinder.*
- virtual `SplineSurface` \* `createSplineSurface` () `const`  
*Create a SplineSurface representation of the cylinder.*
- `shared_ptr`< `Circle` > `getCircle` (`double` par) `const`
- virtual `bool` `isAxisRotational` (`Point` &centre, `Point` &axis, `Point` &vec, `double` &angle)  
*Confirm that this surface is axis rotational.*
- virtual `bool` `isLinear` (`Point` &dir1, `Point` &dir2, `double` tol)  
*The surface is linear in one direction. Fetch it.*
- void `rotate` (`double` rot\_ang\_rad)  
*Rotate the cylinder (moving the seam given by the parametrization).*

### Static Public Member Functions

- static `ClassType` `classType` ()

### Protected Member Functions

- void `setCoordinateAxes` ()

## Protected Attributes

- [double radius\\_](#)
- [Point location\\_](#)
- [Point z\\_axis\\_](#)
- [Point x\\_axis\\_](#)
- [Point y\\_axis\\_](#)
- [RectDomain domain\\_](#)
- [RectDomain orientedDomain\\_](#)

### 29.89.1 Detailed Description

Class that represents a cylinder. It is a subclass of [ElementarySurface](#), and thus has a parametrization and is non-selfintersecting.

A [Cylinder](#) has a natural parametrization in terms of the angle  $u$  and distance  $v$ :  $\mathbf{p}(u, v) = \mathbf{C} + R(\cos(u) \mathbf{x} + \sin(u) \mathbf{y}) + v \mathbf{z}$ , where  $\mathbf{C}$  is a position vector,  $R$  is the radius, and  $\mathbf{x}$ ,  $\mathbf{y}$  and  $\mathbf{z}$  are the (local) axes. The parametrization is bounded by:  $0 \leq u \leq 2\pi$ ,  $-\infty < v < \infty$ . The dimension is 3.

Definition at line 69 of file Cylinder.h.

### 29.89.2 Constructor & Destructor Documentation

#### 29.89.2.1 `Go::Cylinder::Cylinder ( ) [inline]`

Default constructor. Constructs an uninitialized [Cylinder](#) which can only be assigned to or read into.

Definition at line 74 of file Cylinder.h.

#### 29.89.2.2 `Go::Cylinder::Cylinder ( double radius, Point location, Point z_axis, Point x_axis, bool isSwapped = false )`

Constructor. Input is the radius, the location, the direction of the z-axis and the (possibly approximate) x-axis. The local coordinate axes are normalized even if `z_axis` and/or `x_axis` are not unit vectors.

#### 29.89.2.3 `virtual Go::Cylinder::~~Cylinder ( ) [virtual]`

Virtual destructor - ensures safe inheritance.

### 29.89.3 Member Function Documentation

#### 29.89.3.1 `std::vector<CurveLoop> Go::Cylinder::allBoundaryLoops ( double degenerate_epsilon = DEFAULT_SPACE_EPSILON ) const [virtual]`

Returns the anticlockwise outer boundary loop of the surface, together with clockwise loops of any interior boundaries, such that the surface always is 'to the left of' the loops.

## Parameters

|                           |                                                            |
|---------------------------|------------------------------------------------------------|
| <i>degenerate_epsilon</i> | edges whose length is smaller than this value are ignored. |
|---------------------------|------------------------------------------------------------|

## Returns

a vector containing CurveLoops. The first of these describe the outer boundary of the surface (clockwise), whereas the others describe boundaries of interior holes (clockwise).

Implements [Go::ParamSurface](#).

**29.89.3.2** virtual **BoundingBox** Go::Cylinder::boundingBox ( ) const [virtual]

Return empty box if infinite cylinder.

Implements [Go::GeomObject](#).

**29.89.3.3** static **ClassType** Go::Cylinder::classType ( ) [inline],[static]

Definition at line 101 of file Cylinder.h.

**29.89.3.4** virtual **Cylinder\*** Go::Cylinder::clone ( ) const [virtual]

make a clone of this surface and return a pointer to it (user is responsible for clearing up memory afterwards).

## Returns

pointer to cloned object

Implements [Go::ElementarySurface](#).

**29.89.3.5** void Go::Cylinder::closestBoundaryPoint ( const Point & *pt*, double & *clo\_u*, double & *clo\_v*, Point & *clo\_pt*, double & *clo\_dist*, double *epsilon*, const RectDomain \* *rd* = NULL, double \* *seed* = 0 ) const [virtual]

Iterates to the closest point to *pt* on the boundary of the surface.

## See also

[closestPoint\(\)](#)

Implements [Go::ParamSurface](#).

**29.89.3.6** void Go::Cylinder::closestPoint ( const Point & *pt*, double & *clo\_u*, double & *clo\_v*, Point & *clo\_pt*, double & *clo\_dist*, double *epsilon*, const RectDomain \* *domain\_of\_interest* = NULL, double \* *seed* = 0 ) const [virtual]

Iterates to the closest point to *pt* on the surface.

## Parameters

|                           |                                                                                                                              |
|---------------------------|------------------------------------------------------------------------------------------------------------------------------|
| <i>pt</i>                 | the point to find the closest point to                                                                                       |
| <i>clo_u</i>              | u parameter of the closest point                                                                                             |
| <i>clo_v</i>              | v parameter of the closest point                                                                                             |
| <i>clo_pt</i>             | the geometric position of the closest point                                                                                  |
| <i>clo_dist</i>           | the distance between pt and clo_pt                                                                                           |
| <i>epsilon</i>            | parameter tolerance (will in any case not be higher than sqrt(machine_precision) x magnitude of solution)                    |
| <i>domain_of_interest</i> | pointer to parameter domain in which to search for closest point. If a NULL pointer is used, the entire surface is searched. |
| <i>seed</i>               | pointer to parameter values where iteration starts.                                                                          |

Reimplemented from [Go::ParamSurface](#).

29.89.3.7 `shared_ptr<ParamCurve> Go::Cylinder::constParamCurve ( double parameter, bool pdir_is_u, double from, double to ) const`

29.89.3.8 `std::vector<shared_ptr<ParamCurve> > Go::Cylinder::constParamCurves ( double parameter, bool pdir_is_u ) const [virtual]`

Get the curve(s) obtained by intersecting the surface with one of its constant parameter curves. For surfaces without holes, this will be the parameter curve itself; for surfaces with interior holes this may be a collection of several, disjoint curves.

## Parameters

|                  |                                                                                                                                              |
|------------------|----------------------------------------------------------------------------------------------------------------------------------------------|
| <i>parameter</i> | parameter value for the constant parameter (either u or v)                                                                                   |
| <i>pdir_is_u</i> | specify whether the <i>moving</i> parameter (as opposed to the <i>constant</i> parameter) is the first ('true') or the second ('false') one. |

## Returns

a vector containing shared pointers to the obtained, newly constructed constant-parameter curves.

Implements [Go::ParamSurface](#).

29.89.3.9 `virtual SplineSurface* Go::Cylinder::createSplineSurface ( ) const [virtual]`

Create a [SplineSurface](#) representation of the cylinder.

Implements [Go::ElementarySurface](#).

29.89.3.10 `virtual int Go::Cylinder::dimension ( ) const [virtual]`

Return the dimension of the space in which the object lies (usually 2 or 3)

Implements [Go::GeomObject](#).

29.89.3.11 `virtual SplineSurface* Go::Cylinder::geometrySurface ( ) const` [virtual]

Create a [SplineSurface](#) representation of the cylinder.

Implements [Go::ElementarySurface](#).

29.89.3.12 `void Go::Cylinder::getBoundaryInfo ( Point & pt1, Point & pt2, double epsilon, SplineCurve *& cv, SplineCurve *& crosscv, double knot_tol = 1e-05 ) const` [virtual]

Get the boundary curve segment between two points on the boundary, as well as the cross-tangent curve. If the given points are not positioned on the same boundary (within a certain tolerance), no curves will be created.

#### Parameters

|                 |                                                                                                                                                                                                                                                                                                                                    |
|-----------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>pt1</i>      | the first point on the boundary, given by the user                                                                                                                                                                                                                                                                                 |
| <i>pt2</i>      | the second point on the boundary, given by the user                                                                                                                                                                                                                                                                                |
| <i>epsilon</i>  | the tolerance used when determining whether the given points are lying on a boundary, and if they do, whether they both lie on the <i>same</i> boundary.                                                                                                                                                                           |
| <i>cv</i>       | upon return, this will point to a newly created curve representing the boundary curve between 'pt1' and 'pt2'. The user assumes ownership of this object and is responsible for its deletion. No curve is created if the given points are not found to lie on the same boundary.                                                   |
| <i>crosscv</i>  | upon return, this will point to a newly created curve representing the cross-boundary curve between 'pt1' and 'pt2'. The user assumes ownership of this object and is responsible for its deletion. The direction is outwards from the surface. No curve is created if the given points are not found to lie on the same boundary. |
| <i>knot_tol</i> | tolerance used when working with the knot-vector, to specify how close a parameter value must be to a knot in order to be considered 'on' the knot.                                                                                                                                                                                |

Implements [Go::ParamSurface](#).

29.89.3.13 `shared_ptr<Circle> Go::Cylinder::getCircle ( double par ) const`

Get the circle that is given by fixing  $v$  ( $u$  if swapped) at the value  $par$ . Bounds in the  $u$ -direction ( $v$ -direction if swapped) will be preserved - thus the "circle" might be a circular arc.

#### Parameters

|            |                                                |
|------------|------------------------------------------------|
| <i>par</i> | angular parameter where the circle is computed |
|------------|------------------------------------------------|

#### Returns

Pointer to circle or circular arc

29.89.3.14 `void Go::Cylinder::getCoordinateAxes ( Point & x_axis, Point & y_axis, Point & z_axis ) const` [inline]

Local coordinate axes. The  $z\_axis$  corresponds to the cylinder axis.

Definition at line 189 of file Cylinder.h.



```
29.89.3.15 virtual void Go::Cylinder::getDegenerateCorners (std::vector< Point > & deg_corners, double tol) const
[virtual]
```

Check for parallel and anti parallel partial derivatives in surface corners.

Implements [Go::ParamSurface](#).

```
29.89.3.16 virtual shared_ptr<ElementaryCurve> Go::Cylinder::getElementaryParamCurve (ElementaryCurve *
space_crv, double tol, const Point * start_par_pt = NULL, const Point * end_par_pt = NULL) const
[virtual]
```

Fetch the parameter curve in the domain of the elementary surface corresponding to a given elementary curve in geometry space if this curve has a simpler elementary representation. Otherwise, nothing is returned

Reimplemented from [Go::ElementarySurface](#).

```
29.89.3.17 Point Go::Cylinder::getLocation () const [inline]
```

[Point](#) on cylinder axis.

Definition at line 185 of file Cylinder.h.

```
29.89.3.18 double Go::Cylinder::getRadius () const [inline]
```

[Cylinder](#) radius.

Definition at line 181 of file Cylinder.h.

```
29.89.3.19 virtual ClassType Go::Cylinder::instanceType () const [virtual]
```

Return the class type identifier of a given, derived instance of [GeomObject](#).

Implements [Go::GeomObject](#).

```
29.89.3.20 virtual bool Go::Cylinder::isAxisRotational (Point & centre, Point & axis, Point & vec, double & angle)
[virtual]
```

Confirm that this surface is axis rotational.

Reimplemented from [Go::ParamSurface](#).

```
29.89.3.21 bool Go::Cylinder::isBounded () const [virtual]
```

Query if parametrization is bounded. All four parameter bounds must be finite for this to be true.

Returns

*true* if bounded, *false* otherwise

Reimplemented from [Go::ElementarySurface](#).

29.89.3.22 `bool Go::Cylinder::isClosed ( bool & closed_dir_u, bool & closed_dir_v ) const` [virtual]

Check if the surface is closed.

Reimplemented from [Go::ElementarySurface](#).

29.89.3.23 `bool Go::Cylinder::isDegenerate ( bool & b, bool & r, bool & t, bool & l, double tolerance ) const`  
[virtual]

The order of the edge indicators (bottom, right, top, left) matches the `edge_number` of `edgeCurve()`. Query whether any of the four boundary curves are degenerate (zero length) within a certain tolerance. In the below, we refer to 'u' as the first parameter and 'v' as the second.

#### Parameters

|                  |                                                                                                    |
|------------------|----------------------------------------------------------------------------------------------------|
| <i>b</i>         | 'true' upon return of function if the boundary ( $v = v_{\min}$ ) is degenerate                    |
| <i>r</i>         | 'true' upon return of function if the boundary ( $v = v_{\max}$ ) is degenerate                    |
| <i>t</i>         | 'true' upon return of function if the boundary ( $u = u_{\min}$ ) is degenerate                    |
| <i>l</i>         | 'true' upon return of function if the boundary ( $u = u_{\max}$ ) is degenerate                    |
| <i>tolerance</i> | boundaries are considered degenerate if their length is shorter than this value, given by the user |

#### Returns

'true' if at least one boundary curve was found to be degenerate, 'false' otherwise.

Reimplemented from [Go::ParamSurface](#).

29.89.3.24 `virtual bool Go::Cylinder::isLinear ( Point & dir1, Point & dir2, double tol )` [virtual]

The surface is linear in one direction. Fetch it.

Reimplemented from [Go::ParamSurface](#).

29.89.3.25 `double Go::Cylinder::nextSegmentVal ( int dir, double par, bool forward, double tol ) const` [virtual]

Determine the parameter value of the start of the 'next segment' from a parameter value, along a given parameter direction. A 'segment' is here defined as a parameter interval in which there will be no discontinuities in derivatives or other artifacts. For spline objects, a segment will typically be the interval between two consecutive, non-coincident knots.

#### Parameters

|                |                                                                                                      |
|----------------|------------------------------------------------------------------------------------------------------|
| <i>dir</i>     | the parameter direction in which we search for the next segment (0 or 1)                             |
| <i>par</i>     | the parameter value starting from which we search for the start value of the next segment            |
| <i>forward</i> | define whether we shall move forward ('true') or backwards when searching along this parameter       |
| <i>tol</i>     | tolerance used for determining whether the 'par' is already located <i>on</i> the next segment value |

**Returns**

the value of the start value of the next segment (or the end of the previous segment, if we are moving backwards...)

Implements [Go::ParamSurface](#).

**29.89.3.26** `void Go::Cylinder::normal ( Point & n, double upar, double vpar ) const` [virtual]

Evaluates the surface normal for a given parameter pair

**Parameters**

|             |                                                      |
|-------------|------------------------------------------------------|
| <i>n</i>    | the computed normal will be written to this variable |
| <i>upar</i> | the first parameter                                  |
| <i>vpar</i> | the second parameter                                 |

Implements [Go::ParamSurface](#).

**29.89.3.27** `DirectionCone Go::Cylinder::normalCone ( ) const` [virtual]

Creates a [DirectionCone](#) covering all normals to this surface.

**Returns**

a [DirectionCone](#) (not necessarily the smallest) containing all normals to this surface.

Implements [Go::ParamSurface](#).

**29.89.3.28** `const RectDomain& Go::Cylinder::parameterDomain ( ) const` [virtual]

Return the parameter domain of the surface. This may be a simple rectangular domain ([RectDomain](#)) or any other subclass of [Domain](#) (such as [GoCurveBoundedDomain](#), found in the `sisl_dependent` module).

**Returns**

a [Domain](#) object describing the parametric domain of the surface

Implements [Go::ParamSurface](#).

**29.89.3.29** `void Go::Cylinder::point ( Point & pt, double upar, double vpar ) const` [virtual]

Evaluates the surface's position for a given parameter pair.

**Parameters**

|             |                                              |
|-------------|----------------------------------------------|
| <i>pt</i>   | the result of the evaluation is written here |
| <i>upar</i> | the first parameter                          |
| <i>vpar</i> | the second parameter                         |

Implements [Go::ParamSurface](#).

```
29.89.3.30 void Go::Cylinder::point (std::vector< Point > & pts, double upar, double vpar, int derivs, bool u_from_right
= true, bool v_from_right = true, double resolution = 1.0e-12) const [virtual]
```

Evaluates the surface's position and a certain number of derivatives for a given parameter pair.

#### Parameters

|                     |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|---------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>pts</i>          | the vector containing the evaluated values. Its size must be set by the user prior to calling this function, and should be equal to $(derivs+1) * (derivs+2) / 2$ . Upon completion of the function, its first entry is the surface's position at the given parameter pair. Then, if 'derivs' > 0, the two next entries will be the surface tangents along the first and second parameter direction. The next three entries are the second- and cross derivatives, in the order (du2, dudv, dv2), and similar for even higher derivatives. |
| <i>upar</i>         | the first parameter                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| <i>vpar</i>         | the second parameter                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| <i>derivs</i>       | number of requested derivatives                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| <i>u_from_right</i> | specify whether derivatives along the first parameter are to be calculated from the right ('true', default) or from the left ('false')                                                                                                                                                                                                                                                                                                                                                                                                     |
| <i>v_from_right</i> | specify whether derivatives along the second parameter are to be calculated from the right ('true', default) or from the left ('false')                                                                                                                                                                                                                                                                                                                                                                                                    |
| <i>resolution</i>   | tolerance used when determining whether parameters are located at special values of the parameter domain (in particular; knot values in case of spline objects).                                                                                                                                                                                                                                                                                                                                                                           |

Implements [Go::ParamSurface](#).

```
29.89.3.31 virtual void Go::Cylinder::read (std::istream & is) [virtual]
```

read object from stream

#### Parameters

|           |                                  |
|-----------|----------------------------------|
| <i>is</i> | stream from which object is read |
|-----------|----------------------------------|

Implements [Go::Streamable](#).

```
29.89.3.32 void Go::Cylinder::rotate (double rot_ang_rad)
```

Rotate the cylinder (moving the seam given by the parametrization).

```
29.89.3.33 void Go::Cylinder::setCoordinateAxes () [protected]
```

```
29.89.3.34 void Go::Cylinder::setParamBoundsU (double from_upar, double to_upar)
```

Set parameter bounds in the *u* direction. *u* is the "angular" direction.

29.89.3.35 void Go::Cylinder::setParamBoundsV ( double *from\_vpar*, double *to\_vpar* )

Set parameter bounds in the *v* direction. *v* is the "linear" direction.

29.89.3.36 virtual void Go::Cylinder::setParameterBounds ( double *from\_upar*, double *from\_vpar*, double *to\_upar*, double *to\_vpar* ) [virtual]

Limit the cylinder surface by limiting the parameter domain.

Implements [Go::ElementarySurface](#).

29.89.3.37 Cylinder\* Go::Cylinder::subSurface ( double *from\_upar*, double *from\_vpar*, double *to\_upar*, double *to\_vpar*, double *fuzzy* = DEFAULT\_PARAMETER\_EPSILON ) const

29.89.3.38 std::vector<shared\_ptr<ParamSurface>> Go::Cylinder::subSurfaces ( double *from\_upar*, double *from\_vpar*, double *to\_upar*, double *to\_vpar*, double *fuzzy* = DEFAULT\_PARAMETER\_EPSILON ) const [virtual]

Get the surface(s) obtained by cropping the parameter domain of this surface between given values for the first and second parameter. In general, for surfaces with no interior holes, the result will be *one* surface; however, for surfaces with interior holes, the result might be *several disjoint* surfaces.

#### Parameters

|                  |                                                                       |
|------------------|-----------------------------------------------------------------------|
| <i>from_upar</i> | lower value for the first parameter in the subdomain                  |
| <i>from_vpar</i> | lower value for the second parameter in the subdomain                 |
| <i>to_upar</i>   | upper value for the first parameter in the subdomain                  |
| <i>to_vpar</i>   | upper value for the second parameter in the subdomain                 |
| <i>fuzzy</i>     | tolerance used when determining intersection with interior boundaries |

#### Returns

a vector contained shared pointers to the obtained, newly constructed sub-surfaces.

Implements [Go::ParamSurface](#).

29.89.3.39 DirectionCone Go::Cylinder::tangentCone ( bool *pardir\_is\_u* ) const [virtual]

Creates a [DirectionCone](#) covering all tangents to this surface along a given parameter direction.

#### Parameters

|                    |                                                                                                                                                                                                  |
|--------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>pardir_is_u</i> | if 'true', then the <a href="#">DirectionCone</a> will be defined on basis of the surface's tangents along the first parameter direction. Otherwise the second parameter direction will be used. |
|--------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

**Returns**

a [DirectionCone](#) (not necessarily the smallest) containing all tangents to this surface along the specified parameter direction.

Implements [Go::ParamSurface](#).

**29.89.3.40** `virtual void Go::Cylinder::write ( std::ostream & os ) const` [virtual]

write object to stream

**Parameters**

|                 |                                   |
|-----------------|-----------------------------------|
| <code>os</code> | stream to which object is written |
|-----------------|-----------------------------------|

Implements [Go::Streamable](#).

**29.89.4 Member Data Documentation**

**29.89.4.1** `RectDomain Go::Cylinder::domain_` [protected]

Definition at line 252 of file Cylinder.h.

**29.89.4.2** `Point Go::Cylinder::location_` [protected]

Definition at line 247 of file Cylinder.h.

**29.89.4.3** `RectDomain Go::Cylinder::orientedDomain_` [mutable],[protected]

Definition at line 253 of file Cylinder.h.

**29.89.4.4** `double Go::Cylinder::radius_` [protected]

Definition at line 246 of file Cylinder.h.

**29.89.4.5** `Point Go::Cylinder::x_axis_` [protected]

Definition at line 249 of file Cylinder.h.

**29.89.4.6** `Point Go::Cylinder::y_axis_` [protected]

Definition at line 250 of file Cylinder.h.

## 29.89.4.7 Point Go::Cylinder::z\_axis\_ [protected]

Definition at line 248 of file Cylinder.h.

The documentation for this class was generated from the following file:

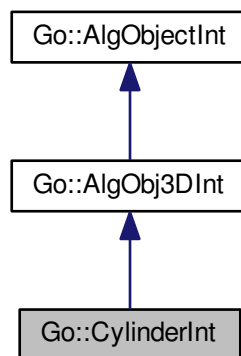
- [gotools-core/include/GoTools/geometry/Cylinder.h](#)

## 29.90 Go::CylinderInt Class Reference

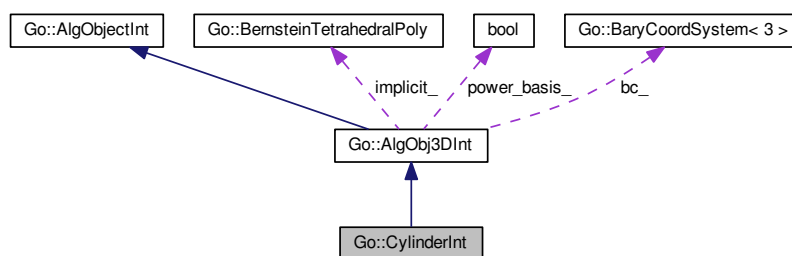
Class for cylindrical algebraic intersection objects.

```
#include <CylinderInt.h>
```

Inheritance diagram for Go::CylinderInt:



Collaboration diagram for Go::CylinderInt:



## Public Member Functions

- [CylinderInt](#) ()
  - [CylinderInt](#) ([Point ax\\_pt](#), [Point ax\\_dir](#), [double radius](#))
  - virtual [~CylinderInt](#) ()
- Destructor.*
- void [read](#) ([std::istream &is](#))
  - [Point ax\\_pt](#) () const
  - [Point ax\\_dir](#) () const
  - [double radius](#) () const
  - [SplineSurface](#) \* [surface](#) ([Point](#) bottom\_pos, [double](#) height) const

## Additional Inherited Members

### 29.90.1 Detailed Description

Class for cylindrical algebraic intersection objects.

Definition at line 56 of file `CylinderInt.h`.

### 29.90.2 Constructor & Destructor Documentation

#### 29.90.2.1 `Go::CylinderInt::CylinderInt ( )`

Constructor. Used when reading from file.

#### 29.90.2.2 `Go::CylinderInt::CylinderInt ( Point ax\_pt, Point ax\_dir, double radius )`

Constructor.

#### Parameters

|               |                                                      |
|---------------|------------------------------------------------------|
| <i>ax_pt</i>  | reference point on the center axis of the cylinder.  |
| <i>ax_dir</i> | direction vector of the center axis of the cylinder. |
| <i>radius</i> | the radius of the cylinder.                          |

#### 29.90.2.3 `virtual Go::CylinderInt::~~CylinderInt ( )` [virtual]

Destructor.

### 29.90.3 Member Function Documentation

#### 29.90.3.1 `Point Go::CylinderInt::ax_dir ( )` const

Get the axis direction vector..



**Returns**

the axis direction vector.

**29.90.3.2 Point Go::CylinderInt::ax\_pt ( ) const**

Get the axis reference point.

**Returns**

the axis reference point.

**29.90.3.3 double Go::CylinderInt::radius ( ) const**

Get the radius of the cylinder.

**Returns**

the radius of the cylinder.

**29.90.3.4 void Go::CylinderInt::read ( std::istream & is )**

Read a cylinder description from file.

**Parameters**

|           |                                                 |
|-----------|-------------------------------------------------|
| <i>is</i> | the stream containing the cylinder description. |
|-----------|-------------------------------------------------|

**29.90.3.5 SplineSurface\* Go::CylinderInt::surface ( Point *bottom\_pos*, double *height* ) const**

Get a [SplineSurface](#) representing the cylinder. This can be used to visualize the object. Since the object is to be tessellated we make sure it is bounded.

**Parameters**

|                   |                                         |
|-------------------|-----------------------------------------|
| <i>bottom_pos</i> | the lowest point on the direction axis. |
| <i>height</i>     | the height of the cylinder.             |

**Returns**

The spline surface of the cylinder.

The documentation for this class was generated from the following file:

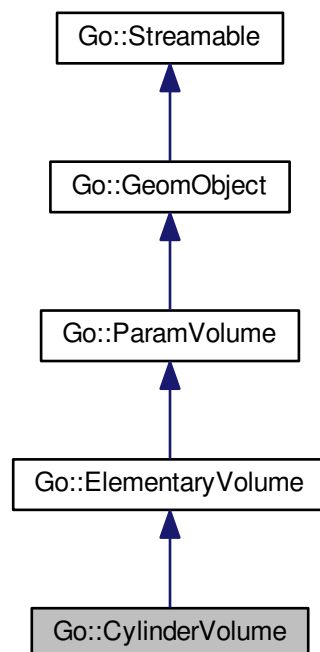
- intersections/include/GoTools/intersections/[CylinderInt.h](#)

## 29.91 Go::CylinderVolume Class Reference

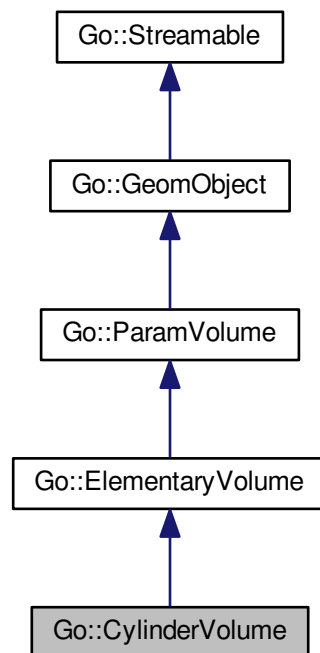
Class that represents a solid cylinder, maybe with empty interior like a straight tube. It is a subclass of [ElementaryVolume](#), and has a natural parametrization in terms of a radius  $u$ , an angle  $v$ , and distance  $w$ :  $\mathbf{p}(u, v, w) = \mathbf{C} + u(\cos(v) \mathbf{x} + \sin(v) \mathbf{y}) + w \mathbf{z}$ , where  $\mathbf{C}$  is a position vector, and  $\mathbf{x}$ ,  $\mathbf{y}$  and  $\mathbf{z}$  are the (local) axes. The parametrization is bounded by:  $R \leq u \leq S$ ,  $0 \leq v \leq 2\pi$ ,  $-\infty < w < \infty$ , where  $R$  and  $S$  are the inner and outer radius. The dimension is 3.

```
#include <CylinderVolume.h>
```

Inheritance diagram for Go::CylinderVolume:



Collaboration diagram for Go::CylinderVolume:



## Public Member Functions

- [CylinderVolume](#) ()
- [CylinderVolume](#) ([Point](#) centre, [double](#) radius, [Point](#) normal, [Point](#) x\_axis)
- virtual [~CylinderVolume](#) ()
  - Virtual destructor - ensures safe inheritance.*
- virtual void [read](#) ([std::istream](#) &is)
- virtual void [write](#) ([std::ostream](#) &os) **const**
- virtual int [dimension](#) () **const**
  - Return the dimension of the space in which the object lies (usually 2 or 3)*
- virtual [ClassType](#) [instanceType](#) () **const**
  - Return the class type identifier of a given, derived instance of [GeomObject](#).*
- virtual [BoundingBox](#) [boundingBox](#) () **const**
  - Return the object's bounding box.*
- virtual [CylinderVolume](#) \* [clone](#) () **const**
- [DirectionCone](#) [tangentCone](#) (int paddir) **const**
- **const** [Array](#) < [double](#), 6 > [parameterSpan](#) () **const**
- void [point](#) ([Point](#) &pt, [double](#) upar, [double](#) vpar, [double](#) wpar) **const**
- void [point](#) ([std::vector](#) < [Point](#) > &pts, [double](#) upar, [double](#) vpar, [double](#) wpar, int derivs, [bool](#) u\_from\_↔\_right=true, [bool](#) v\_from\_right=true, [bool](#) w\_from\_right=true, [double](#) resolution=1.0e-12) **const**
- [double](#) [nextSegmentVal](#) (int dir, [double](#) par, [bool](#) forward, [double](#) tol) **const**
- virtual void [closestPoint](#) (**const** [Point](#) &pt, [double](#) &clo\_u, [double](#) &clo\_v, [double](#) &clo\_w, [Point](#) &clo\_pt, [double](#) &clo\_dist, [double](#) epsilon, [double](#) \*seed=0) **const**
- void [reverseParameterDirection](#) (int paddir)

- void [swapParameterDirection](#) (int pardir1, int pardir2)
- virtual std::vector< shared\_ptr< [ParamSurface](#) > > [getAllBoundarySurfaces](#) () const  
*Fetch all boundary surfaces corresponding to the volume.*
- virtual void [translate](#) (const [Point](#) &vec)  
*Translate.*
- [SplineVolume](#) \* [geometryVolume](#) () const  
*Make a NURBS representation of the object.*
- void [setParameters](#) (double from\_par, double to\_par, int pardir)  
*Restrict the size of the torus in one parameter direction.*
- void [useCentreDegen](#) ()
- void [useCornerDegen](#) ()

### Static Public Member Functions

- static [ClassType](#) [classType](#) ()

#### 29.91.1 Detailed Description

Class that represents a solid cylinder, maybe with empty interior like a straight tube. It is a subclass of [ElementaryVolume](#), and has a natural parametrization in terms of a radius  $u$ , an angle  $v$ , and distance  $w$ :  $\mathbf{p}(u, v, w) = \mathbf{C} + u(\cos(v) \mathbf{x} + \sin(v) \mathbf{y}) + w \mathbf{z}$ , where  $\mathbf{C}$  is a position vector, and  $\mathbf{x}$ ,  $\mathbf{y}$  and  $\mathbf{z}$  are the (local) axes. The parametrization is bounded by:  $R \leq u \leq S$ ,  $0 \leq v \leq 2\pi$ ,  $-\infty < w < \infty$ , where  $R$  and  $S$  are the inner and outer radius. The dimension is 3.

Definition at line 66 of file [CylinderVolume.h](#).

#### 29.91.2 Constructor & Destructor Documentation

##### 29.91.2.1 [Go::CylinderVolume::CylinderVolume](#) ( ) [[inline](#)]

Default constructor. Constructs an uninitialized [CylinderVolume](#) which can only be assigned to or read into.

Definition at line 71 of file [CylinderVolume.h](#).

##### 29.91.2.2 [Go::CylinderVolume::CylinderVolume](#) ( [Point](#) *centre*, [double](#) *radius*, [Point](#) *normal*, [Point](#) *x\_axis* )

Constructor. Input is the centre, outer radius, the normal and the `x_axis` (possibly approximate). The inner radius is set to 0, and the cylinder is set to have infinite length in both directions

##### 29.91.2.3 [virtual Go::CylinderVolume::~~CylinderVolume](#) ( ) [[virtual](#)]

Virtual destructor - ensures safe inheritance.

### 29.91.3 Member Function Documentation

29.91.3.1 virtual **BoundingBox** Go::CylinderVolume::boundingBox ( ) const [virtual]

Return the object's bounding box.

Implements [Go::GeomObject](#).

29.91.3.2 static **ClassType** Go::CylinderVolume::classType ( ) [inline],[static]

Definition at line 96 of file CylinderVolume.h.

29.91.3.3 virtual **CylinderVolume\*** Go::CylinderVolume::clone ( ) const [virtual]

make a clone of this volume and return a pointer to it (user is responsible for clearing up memory afterwards).

#### Returns

pointer to cloned object

Implements [Go::ElementaryVolume](#).

29.91.3.4 virtual void Go::CylinderVolume::closestPoint ( const **Point** & *pt*, double & *clo\_u*, double & *clo\_v*, double & *clo\_w*, **Point** & *clo\_pt*, double & *clo\_dist*, double *epsilon*, double \* *seed* = 0 ) const [virtual]

Iterates to the closest point to *pt* on the volume.

#### Parameters

|                 |                                                                                                                      |
|-----------------|----------------------------------------------------------------------------------------------------------------------|
| <i>pt</i>       | the point to find the closest point to                                                                               |
| <i>clo_u</i>    | u parameter of the closest point                                                                                     |
| <i>clo_v</i>    | v parameter of the closest point                                                                                     |
| <i>clo_w</i>    | w parameter of the closest point                                                                                     |
| <i>clo_pt</i>   | the geometric position of the closest point                                                                          |
| <i>clo_dist</i> | the distance between <i>pt</i> and <i>clo_pt</i>                                                                     |
| <i>epsilon</i>  | parameter tolerance (will in any case not be higher than $\sqrt{\text{machine\_precision}}$ x magnitude of solution) |
| <i>seed</i>     | pointer to parameter values where iteration starts.                                                                  |

Implements [Go::ParamVolume](#).

29.91.3.5 virtual int Go::CylinderVolume::dimension ( ) const [virtual]

Return the dimension of the space in which the object lies (usually 2 or 3)

Implements [Go::GeomObject](#).

29.91.3.6 **SplineVolume\*** `Go::CylinderVolume::geometryVolume ( ) const` [virtual]

Make a NURBS representation of the object.

Implements [Go::ElementaryVolume](#).

29.91.3.7 `virtual std::vector<shared_ptr<ParamSurface> > Go::CylinderVolume::getAllBoundarySurfaces ( ) const`  
[virtual]

Fetch all boundary surfaces corresponding to the volume.

Implements [Go::ParamVolume](#).

29.91.3.8 `virtual ClassType Go::CylinderVolume::instanceType ( ) const` [virtual]

Return the class type identifier of a given, derived instance of [GeomObject](#).

Implements [Go::GeomObject](#).

29.91.3.9 `double Go::CylinderVolume::nextSegmentVal ( int dir, double par, bool forward, double tol ) const`  
[virtual]

Determine the parameter value of the start of the 'next segment' from a parameter value, along a given parameter direction. A 'segment' is here defined as a parameter interval in which there will be no discontinuities in derivatives or other artifacts. For spline objects, a segment will typically be the interval between two consecutive, non-coincident knots.

#### Parameters

|                |                                                                                                      |
|----------------|------------------------------------------------------------------------------------------------------|
| <i>dir</i>     | the parameter direction in which we search for the next segment (0, 1 or 2)                          |
| <i>par</i>     | the parameter value starting from which we search for the start value of the next segment            |
| <i>forward</i> | define whether we shall move forward ('true') or backwards when searching along this parameter       |
| <i>tol</i>     | tolerance used for determining whether the 'par' is already located <i>on</i> the next segment value |

#### Returns

the value of the start value of the next segment (or the end of the previous segment, if we are moving backwards...)

Implements [Go::ParamVolume](#).

29.91.3.10 `const Array<double,6> Go::CylinderVolume::parameterSpan ( ) const` [virtual]

Return the parameter domain of the volume. This is an array containing the start and end parameter values. The spline's parameter *i* has its start value at the array position (2*i*) and its end parameter value at the array position (2*i*+1)

#### Returns

An array describing the parametric domain of the volume

Implements [Go::ParamVolume](#).

29.91.3.11 `void Go::CylinderVolume::point ( Point & pt, double upar, double vpar, double wpar ) const` [virtual]

Evaluates the volume's position for a given parameter triple.

#### Parameters

|             |                                              |
|-------------|----------------------------------------------|
| <i>pt</i>   | the result of the evaluation is written here |
| <i>upar</i> | the first parameter                          |
| <i>vpar</i> | the second parameter                         |
| <i>wpar</i> | the third parameter                          |

Implements [Go::ParamVolume](#).

29.91.3.12 `void Go::CylinderVolume::point ( std::vector< Point > & pts, double upar, double vpar, double wpar, int derivs, bool u_from_right = true, bool v_from_right = true, bool w_from_right = true, double resolution = 1.0e-12 ) const` [virtual]

Evaluates the volume's position and a certain number of derivatives for a given parameter triple.

#### Parameters

|                     |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|---------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>pts</i>          | the vector containing the evaluated values. Its size must be set by the user prior to calling this function. Upon completion of the function, its first entry is the volume's position at the given parameter pair. Then, if ' <i>derivs</i> ' > 0, the two next entries will be the volume tangents along the first, second and third parameter direction. The next three entries are the second- and cross derivatives, in the order ( <i>du</i> <sup>2</sup> , <i>dudv</i> , <i>dudw</i> , <i>dv</i> <sup>2</sup> , <i>dvdw</i> , <i>dw</i> <sup>2</sup> ), and similar for even higher derivatives. |
| <i>upar</i>         | the first parameter                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| <i>vpar</i>         | the second parameter                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| <i>wpar</i>         | the third parameter                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| <i>derivs</i>       | number of requested derivatives                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| <i>u_from_right</i> | specify whether derivatives along the first parameter are to be calculated from the right ('true', default) or from the left ('false')                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| <i>v_from_right</i> | specify whether derivatives along the second parameter are to be calculated from the right ('true', default) or from the left ('false')                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| <i>w_from_right</i> | specify whether derivatives along the third parameter are to be calculated from the right ('true', default) or from the left ('false')                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| <i>resolution</i>   | tolerance used when determining whether parameters are located at special values of the parameter domain (in particular; knot values in case of spline objects).                                                                                                                                                                                                                                                                                                                                                                                                                                        |

Implements [Go::ParamVolume](#).

29.91.3.13 `virtual void Go::CylinderVolume::read ( std::istream & is )` [virtual]

read object from stream

#### Parameters

|           |                                  |
|-----------|----------------------------------|
| <i>is</i> | stream from which object is read |
|-----------|----------------------------------|

Implements [Go::Streamable](#).

29.91.3.14 void `Go::CylinderVolume::reverseParameterDirection ( int pardir )` [virtual]

Reverses the direction of the basis in input direction.

#### Parameters

|               |                                      |
|---------------|--------------------------------------|
| <i>pardir</i> | which parameter direction to reverse |
|---------------|--------------------------------------|

Implements [Go::ParamVolume](#).

29.91.3.15 void `Go::CylinderVolume::setParameters ( double from_par, double to_par, int pardir )`

Restrict the size of the torus in one parameter direction.

29.91.3.16 void `Go::CylinderVolume::swapParameterDirection ( int pardir1, int pardir2 )` [virtual]

Swaps two parameter directions

#### Parameters

|                |                                                           |
|----------------|-----------------------------------------------------------|
| <i>pardir1</i> | One of the parameter directions (0, 1 or 2) to be swapped |
| <i>pardir2</i> | The other parameter direction (0, 1 or 2) to be swapped   |

Implements [Go::ParamVolume](#).

29.91.3.17 **DirectionCone** `Go::CylinderVolume::tangentCone ( int pardir ) const` [virtual]

Creates a [DirectionCone](#) covering all tangents to this volume along a given parameter direction.

#### Parameters

|               |                                                                                                                                                                                                                                     |
|---------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>pardir</i> | if 1, the <a href="#">DirectionCone</a> will be defined on basis of the volume's tangents along the first parameter direction. If 2, the second parameter direction will be used. If 3, the third parameter direction will be used. |
|---------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

#### Returns

a [DirectionCone](#) (not necessarily the smallest) containing all tangents to this volume along the specified parameter direction.

Implements [Go::ParamVolume](#).



29.91.3.18 virtual void Go::CylinderVolume::translate ( const Point & vec ) [virtual]

Translate.

Implements [Go::ParamVolume](#).

29.91.3.19 void Go::CylinderVolume::useCentreDegen ( ) [inline]

A NURBS representation will be a disc with degeneracy in the centre swept linearly

Definition at line 154 of file CylinderVolume.h.

29.91.3.20 void Go::CylinderVolume::useCornerDegen ( ) [inline]

A NURBS representation will be a disc with degenerate corners swept linearly

Definition at line 159 of file CylinderVolume.h.

29.91.3.21 virtual void Go::CylinderVolume::write ( std::ostream & os ) const [virtual]

write object to stream

Parameters

|    |                                   |
|----|-----------------------------------|
| os | stream to which object is written |
|----|-----------------------------------|

Implements [Go::Streamable](#).

The documentation for this class was generated from the following file:

- [trivariate/include/GoTools/trivariate/CylinderVolume.h](#)

## 29.92 hed::Dart Class Reference

**Dart** class for the half-edge data structure.

```
#include <HeDart.h>
```

### Public Member Functions

- [Dart](#) ()  
*Default constructor.*
- [Dart](#) (Edge \*edge, bool dir=true)  
*Constructor.*
- [Dart](#) (const Dart &dart)

- *Copy constructor.*
- `~Dart ()`
- *Destructor.*
- `Dart & operator= (const Dart &dart)`
- *Assignment operator.*
- `bool operator== (const Dart &dart) const`
- *Comparing dart objects.*
- `bool operator!= (const Dart &dart) const`
- *Comparing dart objects.*
- `Dart & alpha0 ()`
- *Maps the dart to a different node.*
- `Dart & alpha1 ()`
- *Maps the dart to a different edge.*
- `Dart & alpha2 ()`
- *Maps the dart to a different triangle. **Note:** the dart is not changed if it is at the boundary!*

#### Utilities not required by TTL

- `void init (Edge *edge, bool dir=true)`
- `double x () const`
- `double y () const`
- `bool isCounterClockWise () const`
- `Node * getNode () const`
- `Node * getOppositeNode () const`
- `Edge * getEdge () const`

### 29.92.1 Detailed Description

**Dart** class for the half-edge data structure.

See [Application Programming Interface to TTL \(API\)](#) for a detailed description of how the member functions should be implemented.

Definition at line 61 of file HeDart.h.

### 29.92.2 Constructor & Destructor Documentation

#### 29.92.2.1 `hed::Dart::Dart ( )` `[inline]`

Default constructor.

Definition at line 68 of file HeDart.h.

#### 29.92.2.2 `hed::Dart::Dart ( Edge * edge, bool dir = true )` `[inline]`

Constructor.

Definition at line 71 of file HeDart.h.

29.92.2.3 `hed::Dart::Dart ( const Dart & dart ) [inline]`

Copy constructor.

Definition at line 74 of file HeDart.h.

29.92.2.4 `hed::Dart::~~Dart ( ) [inline]`

Destructor.

Definition at line 77 of file HeDart.h.

### 29.92.3 Member Function Documentation

29.92.3.1 `Dart& hed::Dart::alpha0 ( ) [inline]`

Maps the dart to a different node.

Definition at line 101 of file HeDart.h.

29.92.3.2 `Dart& hed::Dart::alpha1 ( ) [inline]`

Maps the dart to a different edge.

Definition at line 104 of file HeDart.h.

29.92.3.3 `Dart& hed::Dart::alpha2 ( ) [inline]`

Maps the dart to a different triangle. **Note:** the dart is not changed if it is at the boundary!

Definition at line 117 of file HeDart.h.

29.92.3.4 `Edge* hed::Dart::getEdge ( ) const [inline]`

Definition at line 142 of file HeDart.h.

29.92.3.5 `Node* hed::Dart::getNode ( ) const [inline]`

Definition at line 140 of file HeDart.h.

29.92.3.6 `Node* hed::Dart::getOppositeNode ( ) const [inline]`

Definition at line 141 of file HeDart.h.

29.92.3.7 `void hed::Dart::init ( Edge * edge, bool dir = true ) [inline]`

Definition at line 133 of file HeDart.h.

29.92.3.8 `bool hed::Dart::isCounterClockWise ( ) const [inline]`

Definition at line 138 of file HeDart.h.

29.92.3.9 `bool hed::Dart::operator!= ( const Dart & dart ) const [inline]`

Comparing dart objects.

Definition at line 96 of file HeDart.h.

29.92.3.10 `Dart& hed::Dart::operator= ( const Dart & dart ) [inline]`

Assignment operator.

Definition at line 80 of file HeDart.h.

29.92.3.11 `bool hed::Dart::operator== ( const Dart & dart ) const [inline]`

Comparing dart objects.

Definition at line 89 of file HeDart.h.

29.92.3.12 `double hed::Dart::x ( void ) const [inline]`

Definition at line 135 of file HeDart.h.

29.92.3.13 `double hed::Dart::y ( void ) const [inline]`

Definition at line 136 of file HeDart.h.

The documentation for this class was generated from the following file:

- [ttl/include/ttl/halfedge/HeDart.h](#)

## 29.93 hetriang::Dart Class Reference

**Dart** class for the half-edge data structure.

```
#include <ttlDart.h>
```

## Public Member Functions

- [Dart \(\)](#)  
*Default constructor.*
- [Dart \(Edge \\*edge, bool dir=true\)](#)  
*Constructor.*
- [Dart \(const Dart &dart\)](#)  
*Copy constructor.*
- [~Dart \(\)](#)  
*Destructor.*
- [Dart & operator= \(const Dart &dart\)](#)  
*Assignment operator.*
- [bool operator== \(const Dart &dart\) const](#)  
*Comparing dart objects.*
- [bool operator!= \(const Dart &dart\) const](#)  
*Comparing dart objects.*
- [Dart & alpha0 \(\)](#)  
*Maps the dart to a different node.*
- [Dart & alpha1 \(\)](#)  
*Maps the dart to a different edge.*
- [Dart & alpha2 \(\)](#)  
*Maps the dart to a different triangle. **Note:** the dart is not changed if it is at the boundary!*

## Utilities not required by TTL

- void [init \(Edge \\*edge, bool dir=true\)](#)
- double [x \(\) const](#)
- double [y \(\) const](#)
- bool [isCounterClockWise \(\) const](#)
- shared\_ptr< Node > [getNode \(\) const](#)
- shared\_ptr< Node > [getOppositeNode \(\) const](#)
- Edge \* [getEdge \(\) const](#)
- void [controllConstDart \(Dart &dart\)](#)
- Dart [makeCopy \(\)](#)
- void [printDart \(\)](#)

### 29.93.1 Detailed Description

**Dart** class for the half-edge data structure.

See [Application Programming Interface to TTL \(API\)](#) for a detailed description of how the member functions should be implemented.

This is how the dart class is implemented for the half-edge data structure:

Definition at line 58 of file `ttlDart.h`.

## 29.93.2 Constructor & Destructor Documentation

### 29.93.2.1 `hetriang::Dart::Dart( )` `[inline]`

Default constructor.

Definition at line 63 of file `ttlDart.h`.

### 29.93.2.2 `hetriang::Dart::Dart( Edge * edge, bool dir = true )` `[inline]`

Constructor.

Definition at line 65 of file `ttlDart.h`.

### 29.93.2.3 `hetriang::Dart::Dart( const Dart & dart )` `[inline]`

Copy constructor.

Definition at line 67 of file `ttlDart.h`.

### 29.93.2.4 `hetriang::Dart::~Dart( )` `[inline]`

Destructor.

Definition at line 69 of file `ttlDart.h`.

## 29.93.3 Member Function Documentation

### 29.93.3.1 `Dart& hetriang::Dart::alpha0( )` `[inline]`

Maps the dart to a different node.

Definition at line 90 of file `ttlDart.h`.

### 29.93.3.2 `Dart& hetriang::Dart::alpha1( )` `[inline]`

Maps the dart to a different edge.

Definition at line 92 of file `ttlDart.h`.

### 29.93.3.3 `Dart& hetriang::Dart::alpha2( )` `[inline]`

Maps the dart to a different triangle. **Note:** the dart is not changed if it is at the boundary!

Definition at line 104 of file `ttlDart.h`.

29.93.3.4 `void hetriang::Dart::controllConstDart ( Dart & dart ) [inline]`

controllConstDart is a function who are supposed to check if the dart representing dstart or dend has been changed during a swap. It's implemented so that this is compared to a specified copy *dart*. In Half-Edge datastructure this is an impossible situation, and the function is therefore empty

Definition at line 130 of file ttIDart.h.

29.93.3.5 `Edge* hetriang::Dart::getEdge ( ) const [inline]`

Definition at line 123 of file ttIDart.h.

29.93.3.6 `shared_ptr<Node> hetriang::Dart::getNode ( ) const [inline]`

Definition at line 121 of file ttIDart.h.

29.93.3.7 `shared_ptr<Node> hetriang::Dart::getOppositeNode ( ) const [inline]`

Definition at line 122 of file ttIDart.h.

29.93.3.8 `void hetriang::Dart::init ( Edge * edge, bool dir = true ) [inline]`

Definition at line 115 of file ttIDart.h.

29.93.3.9 `bool hetriang::Dart::isCounterClockWise ( ) const [inline]`

Definition at line 119 of file ttIDart.h.

29.93.3.10 `Dart hetriang::Dart::makeCopy ( ) [inline]`

Definition at line 132 of file ttIDart.h.

29.93.3.11 `bool hetriang::Dart::operator!= ( const Dart & dart ) const [inline]`

Comparing dart objects.

Definition at line 86 of file ttIDart.h.

29.93.3.12 `Dart& hetriang::Dart::operator= ( const Dart & dart ) [inline]`

Assignment operator.

Definition at line 72 of file ttIDart.h.

29.93.3.13 `bool hetriang::Dart::operator==( const Dart & dart ) const` `[inline]`

Comparing dart objects.

Definition at line 80 of file `ttlDart.h`.

29.93.3.14 `void hetriang::Dart::printDart ( )` `[inline]`

Definition at line 138 of file `ttlDart.h`.

29.93.3.15 `double hetriang::Dart::x ( ) const` `[inline]`

Definition at line 117 of file `ttlDart.h`.

29.93.3.16 `double hetriang::Dart::y ( ) const` `[inline]`

Definition at line 118 of file `ttlDart.h`.

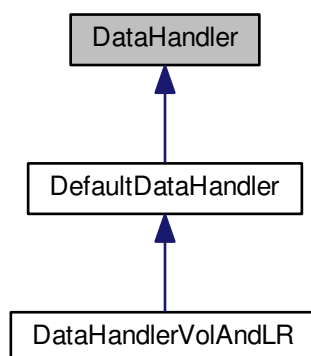
The documentation for this class was generated from the following file:

- `compositemodel/include/GoTools/compositemodel/ttlDart.h`

## 29.94 DataHandler Class Reference

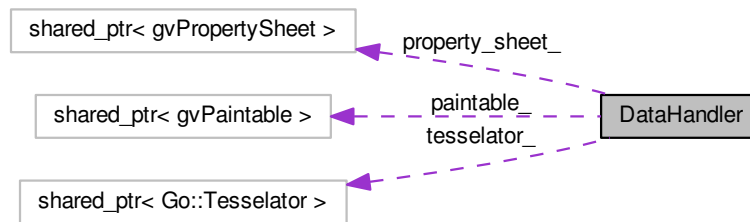
```
#include <DataHandler.h>
```

Inheritance diagram for DataHandler:





Collaboration diagram for DataHandler:



## Public Member Functions

- [DataHandler \(\)](#)
- virtual [~DataHandler \(\)](#)
- virtual void [create](#) (shared\_ptr< [Go::GeomObject](#) > obj, const [gvColor](#) &col, int id)=0
- shared\_ptr< [Go::Tesselator](#) > [tesselator](#) ()
- shared\_ptr< [gvPaintable](#) > [paintable](#) ()
- shared\_ptr< [gvPropertySheet](#) > [propertySheet](#) ()
- void [clear](#) ()

## Protected Attributes

- shared\_ptr< [Go::Tesselator](#) > [tesselator\\_](#)
- shared\_ptr< [gvPaintable](#) > [paintable\\_](#)
- shared\_ptr< [gvPropertySheet](#) > [property\\_sheet\\_](#)

### 29.94.1 Detailed Description

This class connects the four inheritance trees descending from [gvTesselator](#), [gvPaintable](#), [gvPropertySheet](#) and [GeomObject](#).

Definition at line 62 of file [DataHandler.h](#).

### 29.94.2 Constructor & Destructor Documentation

#### 29.94.2.1 DataHandler::DataHandler ( )

Default constructor. In inherited classes, it should Initialize factory (by registering classes).

29.94.2.2 virtual DataHandler::~DataHandler ( ) [virtual]

### 29.94.3 Member Function Documentation

29.94.3.1 void DataHandler::clear ( ) [inline]

Definition at line 87 of file DataHandler.h.

29.94.3.2 virtual void DataHandler::create ( shared\_ptr< Go::GeomObject > obj, const gvColor & col, int id )  
[pure virtual]

Implemented in [DefaultDataHandler](#), and [DataHandlerVolAndLR](#).

29.94.3.3 shared\_ptr<gvPaintable> DataHandler::paintable ( ) [inline]

Definition at line 78 of file DataHandler.h.

29.94.3.4 shared\_ptr<gvPropertySheet> DataHandler::propertySheet ( ) [inline]

Definition at line 82 of file DataHandler.h.

29.94.3.5 shared\_ptr<Go::Tesselator> DataHandler::tesselator ( ) [inline]

Definition at line 74 of file DataHandler.h.

### 29.94.4 Member Data Documentation

29.94.4.1 shared\_ptr<gvPaintable> DataHandler::paintable\_ [protected]

Definition at line 96 of file DataHandler.h.

29.94.4.2 shared\_ptr<gvPropertySheet> DataHandler::property\_sheet\_ [protected]

Definition at line 97 of file DataHandler.h.

29.94.4.3 shared\_ptr<Go::Tesselator> DataHandler::tesselator\_ [protected]

Definition at line 95 of file DataHandler.h.

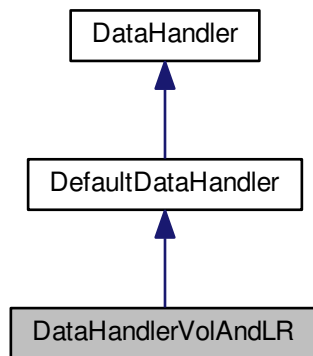
The documentation for this class was generated from the following file:

- [viewlib/include/GoTools/viewlib/DataHandler.h](#)

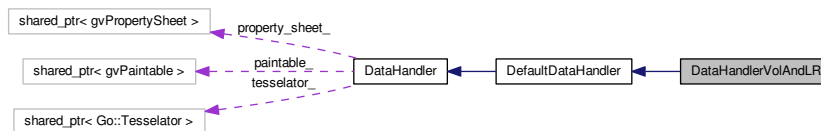
## 29.95 DataHandlerVolAndLR Class Reference

```
#include <DataHandlerVolAndLR.h>
```

Inheritance diagram for DataHandlerVolAndLR:



Collaboration diagram for DataHandlerVolAndLR:



### Public Member Functions

- [DataHandlerVolAndLR \(\)](#)  
*Default constructor. Initializes factory (by registering classes).*
- virtual [~DataHandlerVolAndLR \(\)](#)
- virtual void [create](#) (shared\_ptr< [Go::GeomObject](#) > obj, const [gvColor](#) &col, int id)

### Additional Inherited Members

#### 29.95.1 Detailed Description

Definition at line 45 of file DataHandlerVolAndLR.h.

## 29.95.2 Constructor & Destructor Documentation

### 29.95.2.1 DataHandlerVolAndLR::DataHandlerVolAndLR ( )

Default constructor. Initializes factory (by registering classes).

### 29.95.2.2 virtual DataHandlerVolAndLR::~DataHandlerVolAndLR ( ) [virtual]

## 29.95.3 Member Function Documentation

### 29.95.3.1 virtual void DataHandlerVolAndLR::create ( shared\_ptr< Go::GeomObject > obj, const gvColor & col, int id ) [virtual]

Reimplemented from [DefaultDataHandler](#).

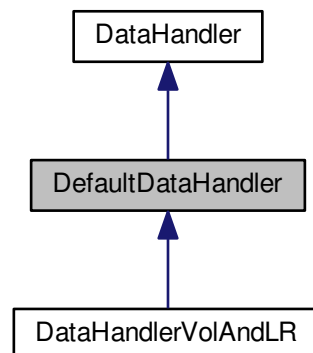
The documentation for this class was generated from the following file:

- [viewlib/include/GoTools/viewlib/vol\\_and\\_lr/DataHandlerVolAndLR.h](#)

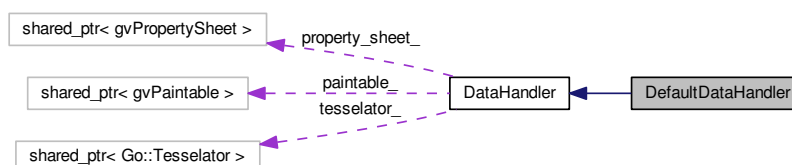
## 29.96 DefaultDataHandler Class Reference

```
#include <DefaultDataHandler.h>
```

Inheritance diagram for DefaultDataHandler:



Collaboration diagram for DefaultDataHandler:



## Public Member Functions

- [DefaultDataHandler](#) ()  
*Default constructor. Initializes factory (by registering classes).*
- virtual [~DefaultDataHandler](#) ()
- virtual void [create](#) (shared\_ptr< [Go::GeomObject](#) > obj, const [gvColor](#) &col, int id)

## Additional Inherited Members

### 29.96.1 Detailed Description

[DefaultDataHandler](#): etc

Definition at line 49 of file [DefaultDataHandler.h](#).

### 29.96.2 Constructor & Destructor Documentation

#### 29.96.2.1 [DefaultDataHandler::DefaultDataHandler](#) ( )

Default constructor. Initializes factory (by registering classes).

#### 29.96.2.2 virtual [DefaultDataHandler::~~DefaultDataHandler](#) ( ) [virtual]

### 29.96.3 Member Function Documentation

#### 29.96.3.1 virtual void [DefaultDataHandler::create](#) ( shared\_ptr< [Go::GeomObject](#) > obj, const [gvColor](#) & col, int id ) [virtual]

Implements [DataHandler](#).

Reimplemented in [DataHandlerVolAndLR](#).

The documentation for this class was generated from the following file:

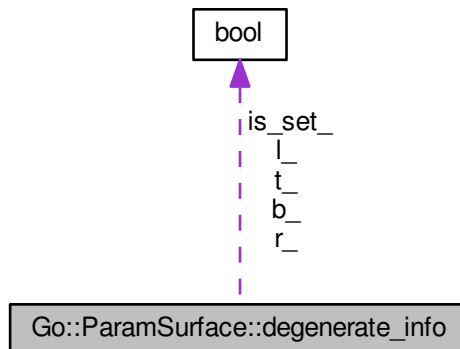
- [viewlib/include/GoTools/viewlib/DefaultDataHandler.h](#)

## 29.97 Go::ParamSurface::degenerate\_info Struct Reference

Degeneracy information regarding one boundary surface of the current surface.

```
#include <ParamSurface.h>
```

Collaboration diagram for Go::ParamSurface::degenerate\_info:



### Public Member Functions

- [degenerate\\_info](#) ()  
*Default constructor. The information is not computed.*

### Public Attributes

- [bool is\\_set\\_](#)  
*Whether or not the degeneracy information is computed.*
- [double tol\\_](#)  
*Tolerance used in computations.*
- [bool b\\_](#)
- [bool r\\_](#)
- [bool t\\_](#)
- [bool l\\_](#)

### 29.97.1 Detailed Description

Degeneracy information regarding one boundary surface of the current surface.

Definition at line 500 of file ParamSurface.h.

## 29.97.2 Constructor & Destructor Documentation

### 29.97.2.1 Go::ParamSurface::degenerate\_info ( ) [inline]

Default constructor. The information is not computed.

Definition at line 511 of file ParamSurface.h.

## 29.97.3 Member Data Documentation

### 29.97.3.1 bool Go::ParamSurface::degenerate\_info::b\_

Indicates if the surface has degenerate boundaries, b\_ = bottom (vmin), r\_ = right (umax), t\_ = top (vmax), l\_ = left (umin)

Definition at line 508 of file ParamSurface.h.

### 29.97.3.2 bool Go::ParamSurface::degenerate\_info::is\_set\_

Whether or not the degeneracy information is computed.

Definition at line 503 of file ParamSurface.h.

### 29.97.3.3 bool Go::ParamSurface::degenerate\_info::l\_

Definition at line 508 of file ParamSurface.h.

### 29.97.3.4 bool Go::ParamSurface::degenerate\_info::r\_

Definition at line 508 of file ParamSurface.h.

### 29.97.3.5 bool Go::ParamSurface::degenerate\_info::t\_

Definition at line 508 of file ParamSurface.h.

### 29.97.3.6 double Go::ParamSurface::degenerate\_info::tol\_

Tolerance used in computations.

Definition at line 505 of file ParamSurface.h.

The documentation for this struct was generated from the following file:

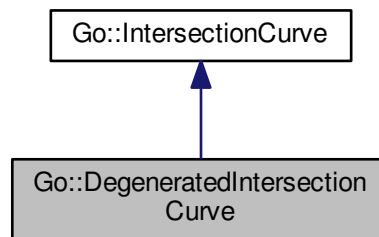
- [gotools-core/include/GoTools/geometry/ParamSurface.h](#)

## 29.98 Go::DegeneratedIntersectionCurve Class Reference

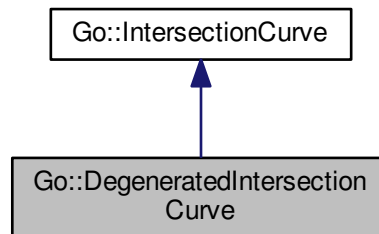
[IntersectionCurve](#) that is degenerated into a single point.

```
#include <IntersectionCurve.h>
```

Inheritance diagram for Go::DegeneratedIntersectionCurve:



Collaboration diagram for Go::DegeneratedIntersectionCurve:



### Public Member Functions

- virtual [~DegeneratedIntersectionCurve](#) ()
- virtual [shared\\_ptr< ParamCurve > getCurve](#) () const
- virtual [shared\\_ptr< ParamCurve > getParamCurve](#) (int obj\_nmb) const
- virtual [bool isIsocurve](#) () const
- virtual [bool isDegenerated](#) () const
- virtual void [getParamSpan](#) (double &start, double &end) const
- virtual void [refine](#) (const double &pos\_tol, const double &angle\_tol)
- virtual void [evaluateAt](#) (double pval, [Point](#) &pos, [Point](#) &tan)



## Friends

- `template<class iterator >`  
`shared_ptr< IntersectionCurve > constructIntersectionCurve (const iterator begin, const iterator end)`

## Additional Inherited Members

### 29.98.1 Detailed Description

[IntersectionCurve](#) that is degenerated into a single point.

Definition at line 221 of file `IntersectionCurve.h`.

### 29.98.2 Constructor & Destructor Documentation

29.98.2.1 `virtual Go::DegeneratedIntersectionCurve::~~DegeneratedIntersectionCurve ( ) [virtual]`

### 29.98.3 Member Function Documentation

29.98.3.1 `virtual void Go::DegeneratedIntersectionCurve::evaluateAt ( double pval, Point & pos, Point & tan ) [inline], [virtual]`

Evaluate the [IntersectionCurve](#) at parameter value *pval*. Its position at this parameter value will be returned in *pos* and its tangent direction will be returned in *tan*.

#### Parameters

|             |                                                                                                                                                                                      |
|-------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>pval</i> | the parameter value for which to evaluate the <a href="#">IntersectionCurve</a> . It should be within the parametric span (which can be obtained from <code>GetParamSpan()</code> ). |
|-------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

#### Return values

|            |                                                                              |
|------------|------------------------------------------------------------------------------|
| <i>pos</i> | the calculated position of the point on the curve at parameter <i>pval</i> . |
|------------|------------------------------------------------------------------------------|

#### Parameters

|            |                                                                |
|------------|----------------------------------------------------------------|
| <i>tan</i> | the calculated tangent of the curve at parameter <i>pval</i> . |
|------------|----------------------------------------------------------------|

Implements [Go::IntersectionCurve](#).

Definition at line 247 of file `IntersectionCurve.h`.

29.98.3.2 `virtual shared_ptr<ParamCurve> Go::DegeneratedIntersectionCurve::getCurve ( ) const [virtual]`

Get out a (shared) pointer to a parametric curve that approximates this [IntersectionCurve](#).

Implements [Go::IntersectionCurve](#).

29.98.3.3 `virtual shared_ptr<ParamCurve> Go::DegeneratedIntersectionCurve::getParamCurve ( int obj_nmb ) const`  
`[virtual]`

Get out a (shared) pointer to the curve in the parametric plane of the first or second object. Should only be called when the concerned object is 2-parametric.

#### Parameters

|                |                                                                                                             |
|----------------|-------------------------------------------------------------------------------------------------------------|
| <i>obj_nmb</i> | should be either 1 or 2, depending on whether you want to get the parametric curve in object 1 or object 2. |
|----------------|-------------------------------------------------------------------------------------------------------------|

#### Returns

a shared pointer to a curve that approximates the intersection curve in the parametric domain of the specified object.

Implements [Go::IntersectionCurve](#).

29.98.3.4 `virtual void Go::DegeneratedIntersectionCurve::getParamSpan ( double & start, double & end ) const`  
`[inline],[virtual]`

Get the start and end value for the parametric span of the [IntersectionCurve](#).

#### Return values

|              |                                                                                         |
|--------------|-----------------------------------------------------------------------------------------|
| <i>start</i> | upon function return this variable will contain the start value of the parametric span. |
| <i>end</i>   | upon function return this variable will contain the end value of the parametric span.   |

Implements [Go::IntersectionCurve](#).

Definition at line 236 of file `IntersectionCurve.h`.

29.98.3.5 `virtual bool Go::DegeneratedIntersectionCurve::isDegenerated ( ) const` `[inline],[virtual]`

Determine if the [IntersectionCurve](#) is in fact a degenerated curve (a single point in space).

#### Returns

'true' if the [IntersectionCurve](#) is degenerated. 'false' otherwise.

Implements [Go::IntersectionCurve](#).

Definition at line 233 of file `IntersectionCurve.h`.

29.98.3.6 `virtual bool Go::DegeneratedIntersectionCurve::isIsocurve ( ) const` `[inline],[virtual]`

Determine if the [IntersectionCurve](#) is in fact representing an isoparametric intersection.

#### Returns

'true' if the [IntersectionCurve](#) is part of an isocurve for one of the objects, 'false' otherwise.

Implements [Go::IntersectionCurve](#).

Definition at line 230 of file `IntersectionCurve.h`.

29.98.3.7 `virtual void Go::DegeneratedIntersectionCurve::refine ( const double & pos_tol, const double & angle_tol )`  
`[inline],[virtual]`

Refine the curve (which means to increase the number of [IntersectionPoint](#) s defining it), so that the curve matches a specific positional and angle tolerance.

#### Parameters

|                        |                                                                                    |
|------------------------|------------------------------------------------------------------------------------|
| <code>pos_tol</code>   | the positional tolerance that the curve should match after refinement.             |
| <code>angle_tol</code> | the angular tolerance that the tangent of the curve should match after refinement. |

Implements [Go::IntersectionCurve](#).

Definition at line 242 of file `IntersectionCurve.h`.

## 29.98.4 Friends And Related Function Documentation

29.98.4.1 `template<class iterator > shared_ptr<IntersectionCurve> constructIntersectionCurve ( const iterator begin, const iterator end )` `[friend]`

Definition at line 601 of file `IntersectionCurve.h`.

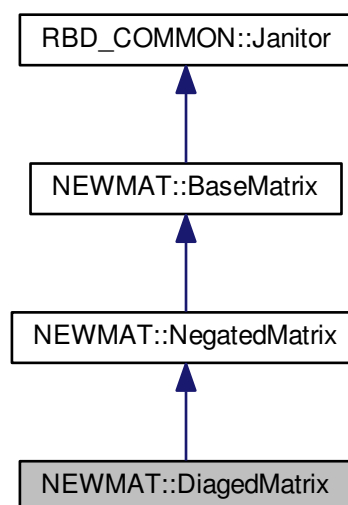
The documentation for this class was generated from the following file:

- `intersections/include/GoTools/intersections/IntersectionCurve.h`

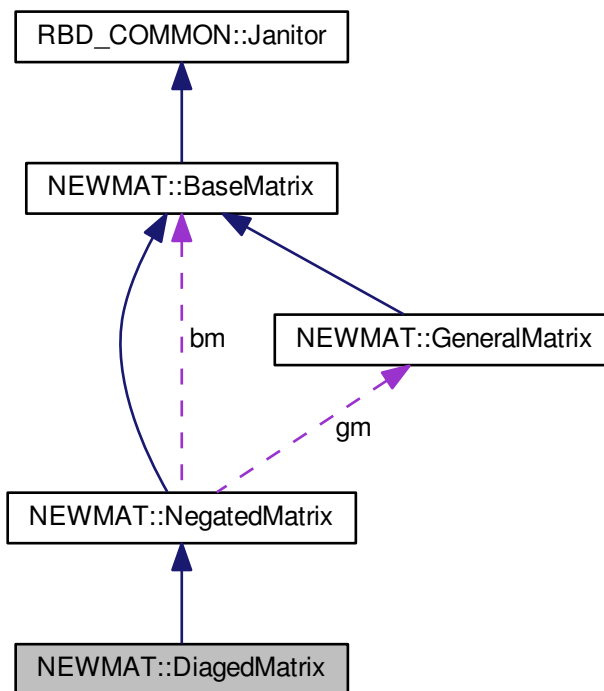
## 29.99 NEWMAT::DiagedMatrix Class Reference

```
#include <newmat.h>
```

Inheritance diagram for `NEWMAT::DiagedMatrix`:



Collaboration diagram for NEWMAT::DiagedMatrix:



### Public Member Functions

- [~DiagedMatrix \(\)](#)
- [GeneralMatrix \\* Evaluate \(MatrixType mt=MatrixTypeUnSp\)](#)
- [MatrixBandWidth BandWidth \(\) const](#)

### Friends

- class [BaseMatrix](#)

### Additional Inherited Members

#### 29.99.1 Detailed Description

Definition at line 1442 of file newmat.h.

#### 29.99.2 Constructor & Destructor Documentation

##### 29.99.2.1 NEWMAT::DiagedMatrix::~~DiagedMatrix ( ) [inline]

Definition at line 1447 of file newmat.h.

### 29.99.3 Member Function Documentation

#### 29.99.3.1 MatrixBandWidth DiagedMatrix::BandWidth ( ) const [virtual]

Reimplemented from [NEWMAT::NegatedMatrix](#).

Definition at line 483 of file newmat4.cpp.

#### 29.99.3.2 GeneralMatrix \* DiagedMatrix::Evaluate ( MatrixType *mt* = MatrixTypeUnSp ) [virtual]

Reimplemented from [NEWMAT::NegatedMatrix](#).

Definition at line 314 of file newmat5.cpp.

### 29.99.4 Friends And Related Function Documentation

#### 29.99.4.1 friend class BaseMatrix [friend]

Definition at line 1445 of file newmat.h.

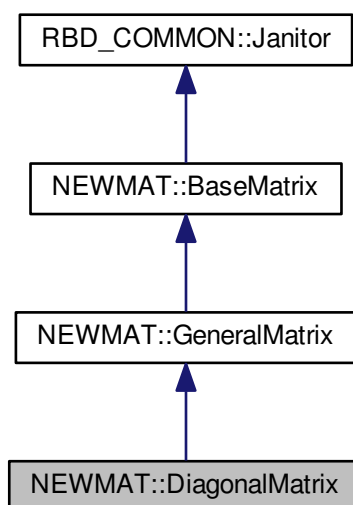
The documentation for this class was generated from the following files:

- [newmat/include/newmat.h](#)
- [newmat/src/newmat4.cpp](#)
- [newmat/src/newmat5.cpp](#)

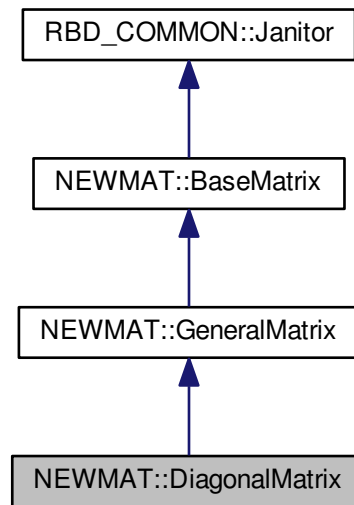
## 29.100 NEWMAT::DiagonalMatrix Class Reference

```
#include <newmat.h>
```

Inheritance diagram for NEWMAT::DiagonalMatrix:



Collaboration diagram for NEWMAT::DiagonalMatrix:



## Public Member Functions

- [DiagonalMatrix \(\)](#)
- [~DiagonalMatrix \(\)](#)
- [DiagonalMatrix \(ArrayLengthSpecifier\)](#)
- [DiagonalMatrix \(const BaseMatrix &\)](#)
- [DiagonalMatrix \(const DiagonalMatrix &gm\)](#)
- [void operator= \(const BaseMatrix &\)](#)
- [void operator= \(Real f\)](#)
- [void operator= \(const DiagonalMatrix &m\)](#)
- [Real & operator\(\) \(int, int\)](#)
- [Real & operator\(\) \(int\)](#)
- [Real operator\(\) \(int, int\) const](#)
- [Real operator\(\) \(int\) const](#)
- [Real & element \(int, int\)](#)
- [Real & element \(int\)](#)
- [Real element \(int, int\) const](#)
- [Real element \(int\) const](#)
- [MatrixType Type \(\) const](#)
- [LogAndSign LogDeterminant \(\) const](#)
- [Real Trace \(\) const](#)
- [void GetRow \(MatrixRowCol &\)](#)
- [void GetCol \(MatrixRowCol &\)](#)
- [void GetCol \(MatrixColX &\)](#)
- [void NextRow \(MatrixRowCol &\)](#)
- [void NextCol \(MatrixRowCol &\)](#)
- [void NextCol \(MatrixColX &\)](#)
- [GeneralMatrix \\* MakeSolver \(\)](#)
- [void Solver \(MatrixColX &, const MatrixColX &\)](#)

- [GeneralMatrix \\* Transpose \(TransposedMatrix \\*, MatrixType\)](#)
- void [ReSize](#) (int)
- void [ReSize](#) (const [GeneralMatrix](#) &A)
- [Real \\* nric](#) () const
- [MatrixBandWidth BandWidth](#) () const

## Additional Inherited Members

### 29.100.1 Detailed Description

Definition at line 746 of file newmat.h.

### 29.100.2 Constructor & Destructor Documentation

29.100.2.1 [NEWMAT::DiagonalMatrix::DiagonalMatrix \( \)](#) [inline]

Definition at line 750 of file newmat.h.

29.100.2.2 [NEWMAT::DiagonalMatrix::~~DiagonalMatrix \( \)](#) [inline]

Definition at line 751 of file newmat.h.

29.100.2.3 [NEWMAT::DiagonalMatrix::DiagonalMatrix \( \[ArrayLengthSpecifier\]\(#\) \)](#)

29.100.2.4 [NEWMAT::DiagonalMatrix::DiagonalMatrix \( const \[BaseMatrix\]\(#\) & \)](#)

29.100.2.5 [NEWMAT::DiagonalMatrix::DiagonalMatrix \( const \[DiagonalMatrix\]\(#\) & gm \)](#) [inline]

Definition at line 754 of file newmat.h.

### 29.100.3 Member Function Documentation

29.100.3.1 [MatrixBandWidth](#) [DiagonalMatrix::BandWidth \( \)](#) const [virtual]

Reimplemented from [NEWMAT::BaseMatrix](#).

Definition at line 403 of file newmat4.cpp.

29.100.3.2 `Real& NEWMAT::DiagonalMatrix::element ( int , int )`

29.100.3.3 `Real& NEWMAT::DiagonalMatrix::element ( int )`

29.100.3.4 `Real NEWMAT::DiagonalMatrix::element ( int , int ) const`

29.100.3.5 `Real NEWMAT::DiagonalMatrix::element ( int ) const`

29.100.3.6 `void NEWMAT::DiagonalMatrix::GetCol ( MatrixRowCol & ) [virtual]`

Implements [NEWMAT::GeneralMatrix](#).

29.100.3.7 `void NEWMAT::DiagonalMatrix::GetCol ( MatrixColIX & ) [virtual]`

Implements [NEWMAT::GeneralMatrix](#).

29.100.3.8 `void DiagonalMatrix::GetRow ( MatrixRowCol & mrc ) [virtual]`

Implements [NEWMAT::GeneralMatrix](#).

Definition at line 228 of file newmat3.cpp.

29.100.3.9 `LogAndSign DiagonalMatrix::LogDeterminant ( ) const [virtual]`

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 647 of file newmat8.cpp.

29.100.3.10 `GeneralMatrix* NEWMAT::DiagonalMatrix::MakeSolver ( ) [inline],[virtual]`

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 780 of file newmat.h.

29.100.3.11 `void NEWMAT::DiagonalMatrix::NextCol ( MatrixRowCol & ) [virtual]`

Reimplemented from [NEWMAT::GeneralMatrix](#).

29.100.3.12 `void NEWMAT::DiagonalMatrix::NextCol ( MatrixColIX & ) [virtual]`

Reimplemented from [NEWMAT::GeneralMatrix](#).



29.100.3.13 void DiagonalMatrix::NextRow ( MatrixRowCol & *mrc* ) [virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 253 of file newmat3.cpp.

29.100.3.14 Real\* NEWMAT::DiagonalMatrix::nric ( ) const [inline]

Definition at line 785 of file newmat.h.

29.100.3.15 Real& NEWMAT::DiagonalMatrix::operator() ( int , int )

29.100.3.16 Real& NEWMAT::DiagonalMatrix::operator() ( int )

29.100.3.17 Real NEWMAT::DiagonalMatrix::operator() ( int , int ) const

29.100.3.18 Real NEWMAT::DiagonalMatrix::operator() ( int ) const

29.100.3.19 void NEWMAT::DiagonalMatrix::operator= ( const BaseMatrix & )

29.100.3.20 void NEWMAT::DiagonalMatrix::operator= ( Real *f* ) [inline]

Definition at line 756 of file newmat.h.

29.100.3.21 void NEWMAT::DiagonalMatrix::operator= ( const DiagonalMatrix & *m* ) [inline]

Definition at line 757 of file newmat.h.

29.100.3.22 void NEWMAT::DiagonalMatrix::ReSize ( int )

29.100.3.23 void DiagonalMatrix::ReSize ( const GeneralMatrix & *A* ) [virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 291 of file newmat4.cpp.

29.100.3.24 void DiagonalMatrix::Solver ( MatrixCoIX & *mrc*, const MatrixCoIX & *mrc1* ) [virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 495 of file newmat2.cpp.

29.100.3.25 **Real DiagonalMatrix::Trace ( ) const** [virtual]

Reimplemented from [NEWMAT::BaseMatrix](#).

Definition at line 546 of file newmat8.cpp.

29.100.3.26 **GeneralMatrix \* DiagonalMatrix::Transpose ( TransposedMatrix \*, MatrixType *mt* )** [virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 57 of file newmat5.cpp.

29.100.3.27 **MatrixType DiagonalMatrix::Type ( ) const** [virtual]

Implements [NEWMAT::GeneralMatrix](#).

Definition at line 389 of file newmat4.cpp.

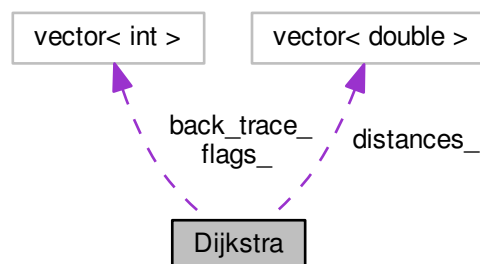
The documentation for this class was generated from the following files:

- [newmat/include/newmat.h](#)
- [newmat/src/newmat2.cpp](#)
- [newmat/src/newmat3.cpp](#)
- [newmat/src/newmat4.cpp](#)
- [newmat/src/newmat5.cpp](#)
- [newmat/src/newmat8.cpp](#)

## 29.101 Dijkstra Class Reference

```
#include <PrDijkstra.h>
```

Collaboration diagram for Dijkstra:



## Public Member Functions

- [Dijkstra](#) ()  
*Constructor.*
- [~Dijkstra](#) ()  
*Destructor.*
- void [setGraph](#) (const [GraphType](#) \*tri)  
*Set the graph to run the algorithm on. Must be run before 'initialize()'.*
- void [initialize](#) ()
- void [setSource](#) (int node\_idx, double dist=0)
- void [run](#) ()
- void [run](#) (double radius)
- void [run](#) (int target)
- double [getDistance](#) (int node\_idx)
- double [getDistance](#) (int node\_idx, int neighbour)
- int [closestNeighbour](#) (int node\_idx)

## Protected Member Functions

- void [setFinished](#) (int node\_idx)
- int [isFinished](#) (int node\_idx)
- void [insertInCandidates](#) (int node\_idx)

## Protected Attributes

- const [GraphType](#) \* [graph\\_](#)
- [HeapType](#) \* [queue\\_](#)
- vector< double > \* [distances\\_](#)
- vector< int > \* [back\\_trace\\_](#)
- vector< int > \* [flags\\_](#)
- double [large\\_distance\\_](#)

### 29.101.1 Detailed Description

[Dijkstra](#) Class implementing the [Dijkstra](#) algorithm

Definition at line 76 of file [PrDijkstra.h](#).

### 29.101.2 Constructor & Destructor Documentation

#### 29.101.2.1 [Dijkstra::Dijkstra](#) ( )

Constructor.

#### 29.101.2.2 [Dijkstra::~~Dijkstra](#) ( )

Destructor.

### 29.101.3 Member Function Documentation

#### 29.101.3.1 `int Dijkstra::closestNeighbour ( int node_idx )`

Get the index of the closest neighbour to the node '*node\_idx*'. Does not give a valid answer until the [Dijkstra](#) algorithm has been [run\(\)](#).

#### 29.101.3.2 `double Dijkstra::getDistance ( int node_idx ) [inline]`

Get the distance associated with the node '*node\_idx*'. Does not give a valid answer until the [Dijkstra](#) algorithm has been [run\(\)](#).

Definition at line 120 of file PrDijkstra.h.

#### 29.101.3.3 `double Dijkstra::getDistance ( int node_idx, int neighbour ) [inline]`

Compute the euclidean distance between two nodes, supposedly neighbours. If they are boundary nodes, their mutual distance will be set to a very large number.

Definition at line 156 of file PrDijkstra.h.

#### 29.101.3.4 `void Dijkstra::initialize ( )`

Initialize internal data structures. Must be done after '[setGraph\(\)](#)' has been called.

#### 29.101.3.5 `void Dijkstra::insertInCandidates ( int node_idx ) [inline],[protected]`

Definition at line 148 of file PrDijkstra.h.

#### 29.101.3.6 `int Dijkstra::isFinished ( int node_idx ) [inline],[protected]`

Definition at line 140 of file PrDijkstra.h.

#### 29.101.3.7 `void Dijkstra::run ( )`

Run the algorithm. Compute the associated distance for all nodes in the graph.

#### 29.101.3.8 `void Dijkstra::run ( double radius )`

Run the algorithm. Compute the associated distance for all nodes in the graph. Edges longer than '*radius*' will be ignored.

29.101.3.9 void Dijkstra::run ( int *target* )

Run the algorithm. Compute the associated distance for nodes in the graph until the distance for the 'target' node has been found. Then skip further computations.

29.101.3.10 void Dijkstra::setFinished ( int *node\_idx* ) [inline],[protected]

Definition at line 133 of file PrDijkstra.h.

29.101.3.11 void Dijkstra::setGraph ( const GraphType \* *tri* ) [inline]

Set the graph to run the algorithm on. Must be run before 'initialize()'.  
[Initialize\(\)](#)

Definition at line 98 of file PrDijkstra.h.

29.101.3.12 void Dijkstra::setSource ( int *node\_idx*, double *dist* = 0 )

Set a 'source' for the algorithm. This is a node for which the associated minimum distance is known. At least one source must be set for the algorithm to work properly.

#### Parameters

|                 |                                                    |
|-----------------|----------------------------------------------------|
| <i>node_idx</i> | the index of the node to be designated as a source |
| <i>dist</i>     | the distance at this node (typically 0).           |

## 29.101.4 Member Data Documentation

29.101.4.1 vector<int>\* Dijkstra::back\_trace\_ [protected]

Definition at line 83 of file PrDijkstra.h.

29.101.4.2 vector<double>\* Dijkstra::distances\_ [protected]

Definition at line 82 of file PrDijkstra.h.

29.101.4.3 vector<int>\* Dijkstra::flags\_ [protected]

Definition at line 84 of file PrDijkstra.h.

29.101.4.4 const GraphType\* Dijkstra::graph\_ [protected]

Definition at line 80 of file PrDijkstra.h.

29.101.4.5 `double Dijkstra::large_distance_` [protected]

Definition at line 85 of file PrDijkstra.h.

29.101.4.6 `HeapType* Dijkstra::queue_` [protected]

Definition at line 81 of file PrDijkstra.h.

The documentation for this class was generated from the following file:

- parametrization/include/GoTools/parametrization/PrDijkstra.h

## 29.102 Go::DirectionCone Class Reference

```
#include <DirectionCone.h>
```

### Public Member Functions

- [DirectionCone](#) ()  
*Constructor making an undefined direction cone.*
- [DirectionCone](#) (const [Point](#) &pt)
- [DirectionCone](#) (const [Point](#) &centre, double angle)
- [~DirectionCone](#) ()  
*Do not inherit from this class – nonvirtual destructor.*
- void [setFromArray](#) (const double \*start, const double \*end, int dim)
- int [dimension](#) () const
- const [Point](#) & [centre](#) () const  
*Return the central direction of the [DirectionCone](#).*
- double [angle](#) () const  
*Return the angle of the [DirectionCone](#).*
- int [greaterThanPi](#) () const
- bool [overlaps](#) (const [DirectionCone](#) &cone) const
- bool [perpendicularOverlaps](#) (const [DirectionCone](#) &cone) const
- bool [containsDirection](#) (const [Point](#) &pt, double tol=0.0) const
- void [addUnionWith](#) (const [Point](#) &pt)
- void [addUnionWith](#) (const [DirectionCone](#) &cone)
- void [read](#) (std::istream &is)  
*Read [DirectionCone](#) from stream.*
- void [write](#) (std::ostream &os) const  
*Write [DirectionCone](#) to stream.*

### 29.102.1 Detailed Description

Class representing a direction cone in Euclidian N-space The direction cone is characterised by a central direction and an angle 'alpha'. All directions whose angle with the central direction is less than 'alpha' is considered to be covered by the cone. If ever 'alpha' equals or exceeds 180 degrees, any possible direction will end up being covered by the cone.

Definition at line 57 of file DirectionCone.h.

## 29.102.2 Constructor & Destructor Documentation

### 29.102.2.1 Go::DirectionCone::DirectionCone ( ) [inline]

Constructor making an undefined direction cone.

Definition at line 60 of file DirectionCone.h.

### 29.102.2.2 Go::DirectionCone::DirectionCone ( const Point & *pt* ) [inline]

Constructor making a direction cone whose central direction is given by 'pt', and whose angle is zero. The length of 'pt' should be assured to be greater than zero, if not, then the [DirectionCone](#) will not be well defined.

Definition at line 66 of file DirectionCone.h.

### 29.102.2.3 Go::DirectionCone::DirectionCone ( const Point & *centre*, double *angle* ) [inline]

Constructor making a direction cone whose central direction is given by 'centre', and whose angle (in radians) is 'angle'. The length of 'centre' should be assured to be greater than zero; if not, then the [DirectionCone](#) will not be well defined.

Definition at line 77 of file DirectionCone.h.

### 29.102.2.4 Go::DirectionCone::~~DirectionCone ( ) [inline]

Do not inherit from this class – nonvirtual destructor.

Definition at line 86 of file DirectionCone.h.

## 29.102.3 Member Function Documentation

### 29.102.3.1 void Go::DirectionCone::addUnionWith ( const Point & *pt* )

If necessary, increase the angle of the [DirectionCone](#) so that it covers the direction represented by 'pt'.

### 29.102.3.2 void Go::DirectionCone::addUnionWith ( const DirectionCone & *cone* )

If necessary, increase the angle of the [DirectionCone](#) so that it covers all directions covered by 'cone'.

### 29.102.3.3 double Go::DirectionCone::angle ( ) const [inline]

Return the angle of the [DirectionCone](#).

Definition at line 105 of file DirectionCone.h.

29.102.3.4 `const Point& Go::DirectionCone::centre ( ) const [inline]`

Return the central direction of the [DirectionCone](#).

Definition at line 102 of file `DirectionCone.h`.

29.102.3.5 `bool Go::DirectionCone::containsDirection ( const Point & pt, double tol = 0.0 ) const`

Return 'true' if the direction specified by 'pt' is contained within the [DirectionCone](#).

29.102.3.6 `int Go::DirectionCone::dimension ( ) const [inline]`

Return the dimension of the Euclidean space in which this [DirectionCone](#) is defined

Definition at line 99 of file `DirectionCone.h`.

29.102.3.7 `int Go::DirectionCone::greaterThanPi ( ) const [inline]`

Inform if the [DirectionCone](#)'s angle is greater than PI. If this is the case, then naturally ANY spatial direction is included in the cone.

Definition at line 110 of file `DirectionCone.h`.

29.102.3.8 `bool Go::DirectionCone::overlaps ( const DirectionCone & cone ) const`

Return 'true' if there exists at least one direction that is covered by both 'this' and the 'cone' [DirectionCone](#)s.

29.102.3.9 `bool Go::DirectionCone::perpendicularOverlaps ( const DirectionCone & cone ) const`

Return 'true' if there exist a direction in one cone that is perpendicular to a direction in the other cone.

29.102.3.10 `void Go::DirectionCone::read ( std::istream & is )`

Read [DirectionCone](#) from stream.

29.102.3.11 `void Go::DirectionCone::setFromArray ( const double * start, const double * end, int dim )`

Re-define an already-existing [DirectionCone](#) from an array of 'dim'-dimensional vectors stored in the memory area between 'start' and 'end'. The first vector in this range will define the central direction of the [DirectionCone](#), and cone's angle will be set wide enough to contain all the other directions given. NB: the length of the range (end - start) must be an exact multiplum of 'dim'.



29.102.3.12 void Go::DirectionCone::write ( std::ostream & os ) const

Write [DirectionCone](#) to stream.

The documentation for this class was generated from the following file:

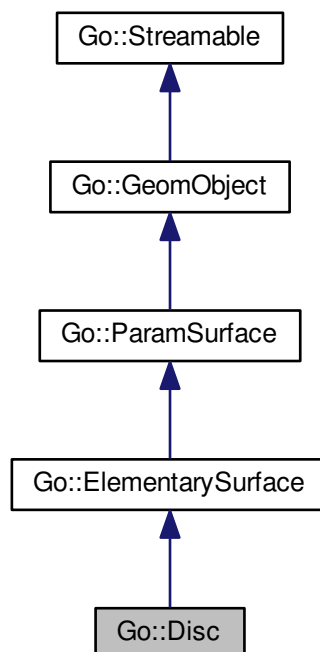
- [gotools-core/include/GoTools/utis/DirectionCone.h](#)

## 29.103 Go::Disc Class Reference

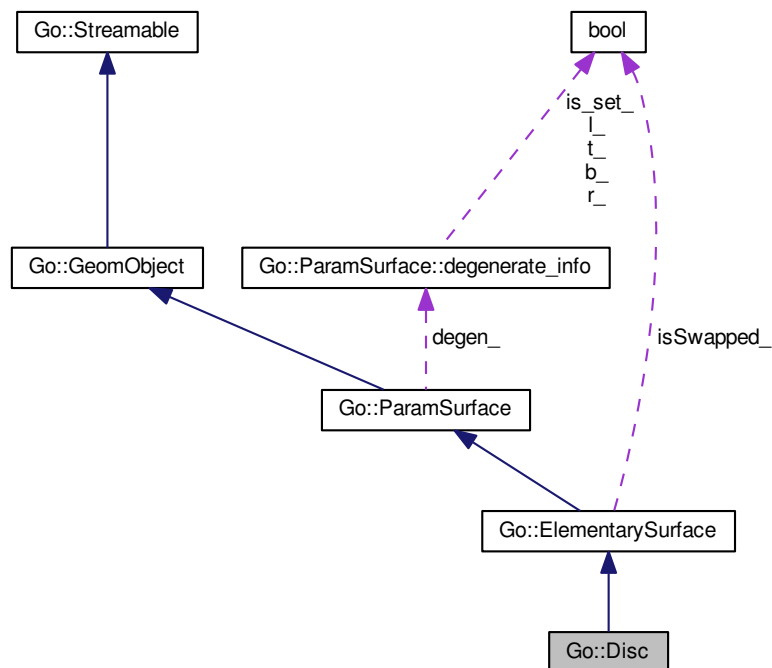
Class that represents a circular disc. It is a subclass of [ElementarySurface](#), and has a natural parametrization by polar coordinates in terms of a radius  $r$  and angle  $v$ :  $\mathbf{p}(r, v) = \mathbf{C} + r(\cos v) \mathbf{x} + (\sin v) \mathbf{y}$ , where  $\mathbf{C}$  is the centre position vector and  $\mathbf{x}$  and  $\mathbf{y}$  are the (local) axes. The parametrization is bounded by:  $0 \leq r \leq R$  and  $0 \leq v \leq 2\pi$ , where  $R$  is the disc radius. The dimension is 2 or 3.

```
#include <Disc.h>
```

Inheritance diagram for Go::Disc:



Collaboration diagram for Go::Disc:



## Public Member Functions

- [Disc](#) ()
- [Disc](#) ([Point](#) centre, [double](#) radius, [Point](#) x\_axis, [Point](#) normal, [bool](#) isSwapped=false)
- virtual [~Disc](#) ()
  - Virtual destructor - ensures safe inheritance.*
- virtual void [read](#) (std::istream &is)
- virtual void [write](#) (std::ostream &os) [const](#)
- virtual int [dimension](#) () [const](#)
  - Return the dimension of the space in which the object lies (usually 2 or 3)*
- virtual [ClassType](#) [instanceType](#) () [const](#)
  - Return the class type identifier of a given, derived instance of [GeomObject](#).*
- virtual [BoundingBox](#) [boundingBox](#) () [const](#)
  - Return the object's bounding box.*
- virtual [Disc](#) \* [clone](#) () [const](#)
- [const](#) [RectDomain](#) & [parameterDomain](#) () [const](#)
- std::vector< [CurveLoop](#) > [allBoundaryLoops](#) ([double](#) degenerate\_epsilon=DEFAULT\_SPACE\_EPSILON) [const](#)
- [DirectionCone](#) [normalCone](#) () [const](#)
- [DirectionCone](#) [tangentCone](#) ([bool](#) paddir\_is\_u) [const](#)
- void [point](#) ([Point](#) &pt, [double](#) upar, [double](#) vpar) [const](#)
- void [point](#) (std::vector< [Point](#) > &pts, [double](#) upar, [double](#) vpar, int derivs, [bool](#) u\_from\_right=true, [bool](#) v\_from\_right=true, [double](#) resolution=1.0e-12) [const](#)
- void [normal](#) ([Point](#) &n, [double](#) upar, [double](#) vpar) [const](#)
- std::vector< shared\_ptr< [ParamCurve](#) > > [constParamCurves](#) ([double](#) parameter, [bool](#) paddir\_is\_u) [const](#)

- `std::vector< shared_ptr< ParamSurface > > subSurfaces (double from_upar, double from_vpar, double to_upar, double to_vpar, double fuzzy=DEFAULT_PARAMETER_EPSILON) const`
- `double nextSegmentVal (int dir, double par, bool forward, double tol) const`
- `void closestPoint (const Point &pt, double &clo_u, double &clo_v, Point &clo_pt, double &clo_dist, double epsilon, const RectDomain *domain_of_interest=NULL, double *seed=0) const`
- `void closestBoundaryPoint (const Point &pt, double &clo_u, double &clo_v, Point &clo_pt, double &clo_dist, double epsilon, const RectDomain *rd=NULL, double *seed=0) const`
- `void getBoundaryInfo (Point &pt1, Point &pt2, double epsilon, SplineCurve *&cv, SplineCurve *&crosscv, double knot_tol=1e-05) const`
- `bool isDegenerate (bool &b, bool &r, bool &t, bool &l, double tolerance) const`
- `virtual void getDegenerateCorners (std::vector< Point > &deg_corners, double tol) const`  
*Check for paralell and anti paralell partial derivatives in surface corners.*
- `bool isBounded () const`
- `bool isClosed (bool &closed_dir_u, bool &closed_dir_v) const`  
*Check if the surface is closed.*
- `Disc * subSurface (double from_upar, double from_vpar, double to_upar, double to_vpar, double fuzzy=DEFAULT_PARAMETER_EPSILON) const`  
*Return the part of the surface limited by the given parameter bounds.*
- `virtual SplineSurface * geometrySurface () const`  
*Create a SplineSurface representation of the Disc.*
- `virtual SplineSurface * createSplineSurface () const`  
*Create a SplineSurface representation of the Disc.*
- `virtual void setParameterBounds (double from_upar, double from_vpar, double to_upar, double to_vpar)`  
*Limit the surface by limiting the parameter domain.*
- `void useCentreDegen ()`  
*The NURBS representation of the disc will have degeneracy in the centre.*
- `void useCornerDegen ()`  
*The NURBS representation of the disc will have degenerate corners.*

### Static Public Member Functions

- `static ClassType classType ()`

### Additional Inherited Members

#### 29.103.1 Detailed Description

Class that represents a circular disc. It is a subclass of [ElementarySurface](#), and has a natural parametrization by polar coordinates in terms of a radius  $r$  and angle  $v$ :  $\mathbf{p}(r, v) = \mathbf{C} + r((\cos v) \mathbf{x} + (\sin v) \mathbf{y})$ , where  $\mathbf{C}$  is the centre position vector and  $\mathbf{x}$  and  $\mathbf{y}$  are the (local) axes. The parametrization is bounded by:  $0 \leq r \leq R$  and  $0 \leq v \leq 2\pi$ , where  $R$  is the disc radius. The dimension is 2 or 3.

A disc also holds degeneracy information for representing it as a [SplineSurface](#) object. There are two ways to parametrize:

1. **Polar coordinates:** The parametrization of the [SplineSurface](#) object almost coincides with the one for the [Disc](#) object itself, i.e. `this->point()` gives almost the same as `geometrySurface()->point()`. The only degeneracy point then is the centre. Note that the corresponding [SplineSurface](#) representation is not an identity mapping of the [Disc](#) because of a reparametrization in the angular direction. This is the default [SplineSurface](#) representation.
2. **Rounded square:** Done by splitting the boundary into four curves that become the boundary curves of the spline surface. Then the four meeting points of the curves become degeneracy points.

Definition at line 83 of file `Disc.h`.

## 29.103.2 Constructor & Destructor Documentation

### 29.103.2.1 `Go::Disc::Disc ( ) [inline]`

Default constructor. Constructs an uninitialized `Disc` which can only be assigned to or read into.

Definition at line 90 of file `Disc.h`.

### 29.103.2.2 `Go::Disc::Disc ( Point centre, double radius, Point x_axis, Point normal, bool isSwapped = false )`

Constructor. Input is the disc centre, disc radius, the `x_axis` and the normal (possibly approximate, only dummy in two-dimensional space). The local coordinate axes are normalized even if `x_axis` and/or `normal` are not unit vectors.

### 29.103.2.3 `virtual Go::Disc::~~Disc ( ) [inline],[virtual]`

Virtual destructor - ensures safe inheritance.

Definition at line 101 of file `Disc.h`.

## 29.103.3 Member Function Documentation

### 29.103.3.1 `std::vector<CurveLoop> Go::Disc::allBoundaryLoops ( double degenerate_epsilon = DEFAULT_SPACE_EPSILON ) const [virtual]`

Returns the anticlockwise outer boundary loop of the surface, together with clockwise loops of any interior boundaries, such that the surface always is 'to the left of' the loops.

#### Parameters

|                                 |                                                            |
|---------------------------------|------------------------------------------------------------|
| <code>degenerate_epsilon</code> | edges whose length is smaller than this value are ignored. |
|---------------------------------|------------------------------------------------------------|

#### Returns

a vector containing `CurveLoops`. The first of these describe the outer boundary of the surface (clockwise), whereas the others describe boundaries of interior holes (clockwise).

Implements [Go::ParamSurface](#).

### 29.103.3.2 `virtual BoundingBox Go::Disc::boundingBox ( ) const [virtual]`

Return the object's bounding box.

Implements [Go::GeomObject](#).

29.103.3.3 `static ClassType Go::Disc::classType ( ) [inline],[static]`

Definition at line 121 of file Disc.h.

29.103.3.4 `virtual Disc* Go::Disc::clone ( ) const [virtual]`

make a clone of this surface and return a pointer to it (user is responsible for clearing up memory afterwards).

Returns

pointer to cloned object

Implements [Go::ElementarySurface](#).

29.103.3.5 `void Go::Disc::closestBoundaryPoint ( const Point & pt, double & clo_u, double & clo_v, Point & clo_pt, double & clo_dist, double epsilon, const RectDomain * rd = NULL, double * seed = 0 ) const [virtual]`

Iterates to the closest point to pt on the boundary of the surface.

See also

[closestPoint\(\)](#)

Implements [Go::ParamSurface](#).

29.103.3.6 `void Go::Disc::closestPoint ( const Point & pt, double & clo_u, double & clo_v, Point & clo_pt, double & clo_dist, double epsilon, const RectDomain * domain_of_interest = NULL, double * seed = 0 ) const [virtual]`

Iterates to the closest point to pt on the surface.

Parameters

|                           |                                                                                                                              |
|---------------------------|------------------------------------------------------------------------------------------------------------------------------|
| <i>pt</i>                 | the point to find the closest point to                                                                                       |
| <i>clo_u</i>              | u parameter of the closest point                                                                                             |
| <i>clo_v</i>              | v parameter of the closest point                                                                                             |
| <i>clo_pt</i>             | the geometric position of the closest point                                                                                  |
| <i>clo_dist</i>           | the distance between pt and clo_pt                                                                                           |
| <i>epsilon</i>            | parameter tolerance (will in any case not be higher than sqrt(machine_precision) x magnitude of solution)                    |
| <i>domain_of_interest</i> | pointer to parameter domain in which to search for closest point. If a NULL pointer is used, the entire surface is searched. |
| <i>seed</i>               | pointer to parameter values where iteration starts.                                                                          |

Reimplemented from [Go::ParamSurface](#).

29.103.3.7 `std::vector<shared_ptr<ParamCurve>> Go::Disc::constParamCurves ( double parameter, bool paddir_is_u ) const [virtual]`

Get the curve(s) obtained by intersecting the surface with one of its constant parameter curves. For surfaces without holes, this will be the parameter curve itself; for surfaces with interior holes this may be a collection of several, disjoint curves.

#### Parameters

|                    |                                                                                                                                              |
|--------------------|----------------------------------------------------------------------------------------------------------------------------------------------|
| <i>parameter</i>   | parameter value for the constant parameter (either u or v)                                                                                   |
| <i>paddir_is_u</i> | specify whether the <i>moving</i> parameter (as opposed to the <i>constant</i> parameter) is the first ('true') or the second ('false') one. |

#### Returns

a vector containing shared pointers to the obtained, newly constructed constant-parameter curves.

Implements [Go::ParamSurface](#).

29.103.3.8 `virtual SplineSurface* Go::Disc::createSplineSurface ( ) const [virtual]`

Create a [SplineSurface](#) representation of the [Disc](#).

Implements [Go::ElementarySurface](#).

29.103.3.9 `virtual int Go::Disc::dimension ( ) const [virtual]`

Return the dimension of the space in which the object lies (usually 2 or 3)

Implements [Go::GeomObject](#).

29.103.3.10 `virtual SplineSurface* Go::Disc::geometrySurface ( ) const [virtual]`

Create a [SplineSurface](#) representation of the [Disc](#).

Implements [Go::ElementarySurface](#).

## Parameters

---

```
29.103.3.11 void Go::Disc::getBoundaryInfo (Point & pt1, Point & pt2, double epsilon, SplineCurve *& cv,
 SplineCurve *& crosscv, double knot_tol = 1e-05) const [virtual]
```

Get the boundary curve segment between two points on the boundary, as well as the cross-tangent curve. If the given points are not positioned on the same boundary (within a certain tolerance), no curves will be created.

## Parameters

|                 |                                                                                                                                                                                                                                                                                                                                    |
|-----------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>pt1</i>      | the first point on the boundary, given by the user                                                                                                                                                                                                                                                                                 |
| <i>pt2</i>      | the second point on the boundary, given by the user                                                                                                                                                                                                                                                                                |
| <i>epsilon</i>  | the tolerance used when determining whether the given points are lying on a boundary, and if they do, whether they both lie on the <i>same</i> boundary.                                                                                                                                                                           |
| <i>cv</i>       | upon return, this will point to a newly created curve representing the boundary curve between 'pt1' and 'pt2'. The user assumes ownership of this object and is responsible for its deletion. No curve is created if the given points are not found to lie on the same boundary.                                                   |
| <i>crosscv</i>  | upon return, this will point to a newly created curve representing the cross-boundary curve between 'pt1' and 'pt2'. The user assumes ownership of this object and is responsible for its deletion. The direction is outwards from the surface. No curve is created if the given points are not found to lie on the same boundary. |
| <i>knot_tol</i> | tolerance used when working with the knot-vector, to specify how close a parameter value must be to a knot in order to be considered 'on' the knot.                                                                                                                                                                                |

Implements [Go::ParamSurface](#).

```
29.103.3.12 virtual void Go::Disc::getDegenerateCorners (std::vector< Point > & deg_corners, double tol) const
 [virtual]
```

Check for parallel and anti parallel partial derivatives in surface corners.

Implements [Go::ParamSurface](#).

```
29.103.3.13 virtual ClassType Go::Disc::instanceType () const [virtual]
```

Return the class type identifier of a given, derived instance of [GeomObject](#).

Implements [Go::GeomObject](#).

```
29.103.3.14 bool Go::Disc::isBounded () const [virtual]
```

Query if parametrization is bounded. All four parameter bounds must be finite for this to be true.

## Returns

*true* if bounded, *false* otherwise

Reimplemented from [Go::ElementarySurface](#).

---

29.103.3.15 **bool** Go::Disc::isClosed ( **bool** & *closed\_dir\_u*, **bool** & *closed\_dir\_v* ) **const** [virtual]

Check if the surface is closed.

Reimplemented from [Go::ElementarySurface](#).

29.103.3.16 **bool** Go::Disc::isDegenerate ( **bool** & *b*, **bool** & *r*, **bool** & *t*, **bool** & *l*, **double** *tolerance* ) **const** [virtual]

The order of the edge indicators (bottom, right, top, left) matches the *edge\_number* of *edgeCurve()*. Query whether any of the four boundary curves are degenerate (zero length) within a certain tolerance. In the below, we refer to 'u' as the first parameter and 'v' as the second.

#### Parameters

|                  |                                                                                                    |
|------------------|----------------------------------------------------------------------------------------------------|
| <i>b</i>         | 'true' upon return of function if the boundary ( $v = v_{\min}$ ) is degenerate                    |
| <i>r</i>         | 'true' upon return of function if the boundary ( $v = v_{\max}$ ) is degenerate                    |
| <i>t</i>         | 'true' upon return of function if the boundary ( $u = u_{\min}$ ) is degenerate                    |
| <i>l</i>         | 'true' upon return of function if the boundary ( $u = u_{\max}$ ) is degenerate                    |
| <i>tolerance</i> | boundaries are considered degenerate if their length is shorter than this value, given by the user |

#### Returns

'true' if at least one boundary curve was found to be degenerate, 'false' otherwise.

Reimplemented from [Go::ParamSurface](#).

29.103.3.17 **double** Go::Disc::nextSegmentVal ( **int** *dir*, **double** *par*, **bool** *forward*, **double** *tol* ) **const** [virtual]

Determine the parameter value of the start of the 'next segment' from a parameter value, along a given parameter direction. A 'segment' is here defined as a parameter interval in which there will be no discontinuities in derivatives or other artifacts. For spline objects, a segment will typically be the interval between two consecutive, non-coincident knots.

#### Parameters

|                |                                                                                                      |
|----------------|------------------------------------------------------------------------------------------------------|
| <i>dir</i>     | the parameter direction in which we search for the next segment (0 or 1)                             |
| <i>par</i>     | the parameter value starting from which we search for the start value of the next segment            |
| <i>forward</i> | define whether we shall move forward ('true') or backwards when searching along this parameter       |
| <i>tol</i>     | tolerance used for determining whether the 'par' is already located <i>on</i> the next segment value |

#### Returns

the value of the start value of the next segment (or the end of the previous segment, if we are moving backwards...)

Implements [Go::ParamSurface](#).



29.103.3.18 `void Go::Disc::normal ( Point & n, double upar, double vpar ) const [virtual]`

Evaluates the surface normal for a given parameter pair

#### Parameters

|             |                                                      |
|-------------|------------------------------------------------------|
| <i>n</i>    | the computed normal will be written to this variable |
| <i>upar</i> | the first parameter                                  |
| <i>vpar</i> | the second parameter                                 |

Implements [Go::ParamSurface](#).

29.103.3.19 `DirectionCone Go::Disc::normalCone ( ) const [virtual]`

Creates a [DirectionCone](#) covering all normals to this surface.

#### Returns

a [DirectionCone](#) (not necessarily the smallest) containing all normals to this surface.

Implements [Go::ParamSurface](#).

29.103.3.20 `const RectDomain& Go::Disc::parameterDomain ( ) const [virtual]`

Return the parameter domain of the surface. This may be a simple rectangular domain ([RectDomain](#)) or any other subclass of [Domain](#) (such as [GoCurveBoundedDomain](#), found in the `sisl_dependent` module).

#### Returns

a [Domain](#) object describing the parametric domain of the surface

Implements [Go::ParamSurface](#).

29.103.3.21 `void Go::Disc::point ( Point & pt, double upar, double vpar ) const [virtual]`

Evaluates the surface's position for a given parameter pair.

#### Parameters

|             |                                              |
|-------------|----------------------------------------------|
| <i>pt</i>   | the result of the evaluation is written here |
| <i>upar</i> | the first parameter                          |
| <i>vpar</i> | the second parameter                         |

Implements [Go::ParamSurface](#).

```
29.103.3.22 void Go::Disc::point (std::vector< Point > & pts, double upar, double vpar, int derivs, bool u_from_right = true, bool v_from_right = true, double resolution = 1.0e-12) const [virtual]
```

Evaluates the surface's position and a certain number of derivatives for a given parameter pair.

#### Parameters

|                     |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|---------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>pts</i>          | the vector containing the evaluated values. Its size must be set by the user prior to calling this function, and should be equal to $(derivs+1) * (derivs+2) / 2$ . Upon completion of the function, its first entry is the surface's position at the given parameter pair. Then, if 'derivs' > 0, the two next entries will be the surface tangents along the first and second parameter direction. The next three entries are the second- and cross derivatives, in the order (du2, dudv, dv2), and similar for even higher derivatives. |
| <i>upar</i>         | the first parameter                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| <i>vpar</i>         | the second parameter                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| <i>derivs</i>       | number of requested derivatives                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| <i>u_from_right</i> | specify whether derivatives along the first parameter are to be calculated from the right ('true', default) or from the left ('false')                                                                                                                                                                                                                                                                                                                                                                                                     |
| <i>v_from_right</i> | specify whether derivatives along the second parameter are to be calculated from the right ('true', default) or from the left ('false')                                                                                                                                                                                                                                                                                                                                                                                                    |
| <i>resolution</i>   | tolerance used when determining whether parameters are located at special values of the parameter domain (in particular; knot values in case of spline objects).                                                                                                                                                                                                                                                                                                                                                                           |

Implements [Go::ParamSurface](#).

```
29.103.3.23 virtual void Go::Disc::read (std::istream & is) [virtual]
```

read object from stream

#### Parameters

|           |                                  |
|-----------|----------------------------------|
| <i>is</i> | stream from which object is read |
|-----------|----------------------------------|

Implements [Go::Streamable](#).

```
29.103.3.24 virtual void Go::Disc::setParameterBounds (double from_upar, double from_vpar, double to_upar, double to_vpar) [virtual]
```

Limit the surface by limiting the parameter domain.

Implements [Go::ElementarySurface](#).

```
29.103.3.25 Disc* Go::Disc::subSurface (double from_upar, double from_vpar, double to_upar, double to_vpar, double fuzzy = DEFAULT_PARAMETER_EPSILON) const
```

Return the part of the surface limited by the given parameter bounds.

```
29.103.3.26 std::vector<shared_ptr<ParamSurface> > Go::Disc::subSurfaces (double from_upar, double from_vpar,
double to_upar, double to_vpar, double fuzzy = DEFAULT_PARAMETER_EPSILON) const
 [virtual]
```

Get the surface(s) obtained by cropping the parameter domain of this surface between given values for the first and second parameter. In general, for surfaces with no interior holes, the result will be *one* surface; however, for surfaces with interior holes, the result might be *several disjoint* surfaces.

#### Parameters

|                  |                                                                       |
|------------------|-----------------------------------------------------------------------|
| <i>from_upar</i> | lower value for the first parameter in the subdomain                  |
| <i>from_vpar</i> | lower value for the second parameter in the subdomain                 |
| <i>to_upar</i>   | upper value for the first parameter in the subdomain                  |
| <i>to_vpar</i>   | upper value for the second parameter in the subdomain                 |
| <i>fuzzy</i>     | tolerance used when determining intersection with interior boundaries |

#### Returns

a vector contained shared pointers to the obtained, newly constructed sub-surfaces.

Implements [Go::ParamSurface](#).

```
29.103.3.27 DirectionCone Go::Disc::tangentCone (bool padir_is_u) const [virtual]
```

Creates a [DirectionCone](#) covering all tangents to this surface along a given parameter direction.

#### Parameters

|                   |                                                                                                                                                                                                  |
|-------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>padir_is_u</i> | if 'true', then the <a href="#">DirectionCone</a> will be defined on basis of the surface's tangents along the first parameter direction. Otherwise the second parameter direction will be used. |
|-------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

#### Returns

a [DirectionCone](#) (not necessarily the smallest) containing all tangents to this surface along the specified parameter direction.

Implements [Go::ParamSurface](#).

```
29.103.3.28 void Go::Disc::useCentreDegen () [inline]
```

The NURBS representation of the disc will have degeneracy in the centre.

Definition at line 212 of file Disc.h.

```
29.103.3.29 void Go::Disc::useCornerDegen () [inline]
```

The NURBS representation of the disc will have degenerate corners.

Definition at line 216 of file Disc.h.

29.103.3.30 `virtual void Go::Disc::write ( std::ostream & os ) const [virtual]`

write object to stream

#### Parameters

|                 |                                   |
|-----------------|-----------------------------------|
| <code>os</code> | stream to which object is written |
|-----------------|-----------------------------------|

Implements [Go::Streamable](#).

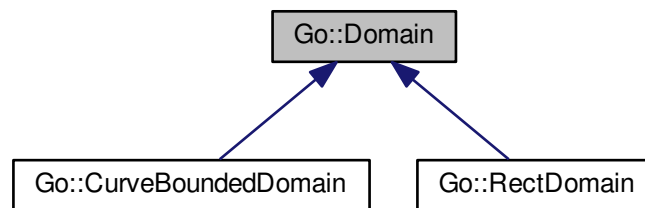
The documentation for this class was generated from the following file:

- [gtools-core/include/GoTools/geometry/Disc.h](#)

## 29.104 Go::Domain Class Reference

```
#include <Domain.h>
```

Inheritance diagram for Go::Domain:



### Public Member Functions

- `virtual ~Domain ()`  
*virtual destructor ensures safe inheritance*
- `virtual bool isInDomain (const Array< double, 2 > &point, double tolerance) const =0`
- `virtual int isInDomain2 (const Array< double, 2 > &point, double tolerance) const =0`
- `virtual bool isOnBoundary (const Array< double, 2 > &point, double tolerance) const =0`
- `virtual void closestInDomain (const Array< double, 2 > &point, Array< double, 2 > &clo_pt, double tolerance) const =0`
- `virtual void closestOnBoundary (const Array< double, 2 > &point, Array< double, 2 > &clo_bd_pt, double tolerance) const =0`

#### 29.104.1 Detailed Description

Abstract base class representing a 2D parameter domain.

Definition at line 54 of file Domain.h.

## 29.104.2 Constructor & Destructor Documentation

29.104.2.1 `virtual Go::Domain::~~Domain ( ) [inline], [virtual]`

virtual destructor ensures safe inheritance

Definition at line 58 of file Domain.h.

## 29.104.3 Member Function Documentation

29.104.3.1 `virtual void Go::Domain::closestInDomain ( const Array< double, 2 > & point, Array< double, 2 > & clo_pt, double tolerance ) const [pure virtual]`

Find the (u, v) point in the [Domain](#) that is closest (using Euclidean distance in  $R^2$ ) to a given (u, v) point. If the given point is in the domain, then the answer is obviously the same point.

### Parameters

|                  |                                                                                                                    |
|------------------|--------------------------------------------------------------------------------------------------------------------|
| <i>point</i>     | the (u,v) parameter pair that we want to find the closest parameter pair to <i>inside</i> <a href="#">Domain</a> . |
| <i>clo_pt</i>    | the resulting closest parameter point.                                                                             |
| <i>tolerance</i> | the tolerance used in defining whether the given point is already inside the domain.                               |

Implemented in [Go::CurveBoundedDomain](#), and [Go::RectDomain](#).

29.104.3.2 `virtual void Go::Domain::closestOnBoundary ( const Array< double, 2 > & point, Array< double, 2 > & clo_bd_pt, double tolerance ) const [pure virtual]`

Find the (u, v) point on the boundary of the [Domain](#) that is closest (using Euclidean distance in  $R^2$ ) to a given (u, v) point. If the point is already considered *on* the boundary, then the answer is obviously the same point.

### Parameters

|                  |                                                                                                                         |
|------------------|-------------------------------------------------------------------------------------------------------------------------|
| <i>point</i>     | the (u,v) parameter pair that we want to find to closest parameter pair to <i>on</i> the <a href="#">Domain</a> border. |
| <i>clo_bd_pt</i> | the resulting closest border point.                                                                                     |
| <i>tolerance</i> | the tolerance used in defining whether the given point is                                                               |

Implemented in [Go::CurveBoundedDomain](#), and [Go::RectDomain](#).

29.104.3.3 `virtual bool Go::Domain::isInDomain ( const Array< double, 2 > & point, double tolerance ) const [pure virtual]`

check whether a given parameter pair is located inside the domain

### Parameters

|                  |                                                                                                  |
|------------------|--------------------------------------------------------------------------------------------------|
| <i>point</i>     | the (u,v)-pair that we want to test.                                                             |
| <i>tolerance</i> | the tolerance used (ruling what to do when 'point' is located very near the edge of the domain). |

**Returns**

'true' if 'point' is inside the domain, or within 'tolerance' from being inside the domain, 'false' otherwise'.

Implemented in [Go::CurveBoundedDomain](#), and [Go::RectDomain](#).

```
29.104.3.4 virtual int Go::Domain::isInDomain2 (const Array< double, 2 > & point, double tolerance) const [pure virtual]
```

Query whether a given parameter pair is inside the domain or not.

**Parameters**

|                  |                                                                                                                                                                                                                                                         |
|------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>point</i>     | array containing the parameter pair                                                                                                                                                                                                                     |
| <i>tolerance</i> | the tolerance to be used. In order to be considered 'inside', the point must be located inside one of the defining <a href="#">CurveLoop</a> s, as well as being at a distance more than 'tolerance' from any point on that <a href="#">CurveLoop</a> . |

**Returns**

'1' if the point is found to be inside the domain, '2' if it is found to be on the boundary '0' otherwise.

Implemented in [Go::CurveBoundedDomain](#), and [Go::RectDomain](#).

```
29.104.3.5 virtual bool Go::Domain::isOnBoundary (const Array< double, 2 > & point, double tolerance) const [pure virtual]
```

check whether a given parameter pair is located on the domain boundary

**Parameters**

|                  |                                                                                                                   |
|------------------|-------------------------------------------------------------------------------------------------------------------|
| <i>point</i>     | the (u,v)-pair that we want to test                                                                               |
| <i>tolerance</i> | the tolerance used (how 'far' from the boundary our (u,v) pair can be and still be considered 'on' the boundary). |

**Returns**

'true' if the point is considered to be on the boundary (within 'tolerance', 'false' otherwise).

Implemented in [Go::CurveBoundedDomain](#), and [Go::RectDomain](#).

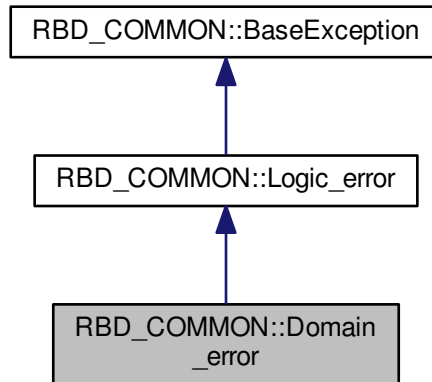
The documentation for this class was generated from the following file:

- [gotools-core/include/GoTools/geometry/Domain.h](#)

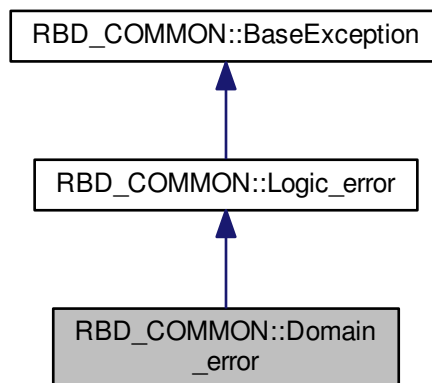
## 29.105 RBD\_COMMON::Domain\_error Class Reference

```
#include <myexcept.h>
```

Inheritance diagram for RBD\_COMMON::Domain\_error:



Collaboration diagram for RBD\_COMMON::Domain\_error:



- static unsigned long [Select](#)
- [Domain\\_error](#) (`const char *a_what=0`)

### Additional Inherited Members

#### 29.105.1 Detailed Description

Definition at line 371 of file `myexcept.h`.

## 29.105.2 Constructor & Destructor Documentation

### 29.105.2.1 Domain\_error::Domain\_error ( const char \* a\_what = 0 )

Definition at line 412 of file myexcept.cpp.

## 29.105.3 Member Data Documentation

### 29.105.3.1 unsigned long Domain\_error::Select [static]

Definition at line 374 of file myexcept.h.

The documentation for this class was generated from the following files:

- [newmat/include/myexcept.h](#)
- [newmat/src/myexcept.cpp](#)

## 29.106 Go::LRSplineSurface::double\_pair\_hash Struct Reference

```
#include <LRSplineSurface.h>
```

### Public Member Functions

- [size\\_t operator\(\)](#) (const std::pair< double, double > &dp) const

### 29.106.1 Detailed Description

Definition at line 117 of file LRSplineSurface.h.

### 29.106.2 Member Function Documentation

#### 29.106.2.1 [size\\_t Go::LRSplineSurface::double\\_pair\\_hash::operator\(\)](#) ( const std::pair< double, double > & dp ) const [inline]

Definition at line 118 of file LRSplineSurface.h.

The documentation for this struct was generated from the following file:

- [Irsplines2D/include/GoTools/Irsplines2D/LRSplineSurface.h](#)

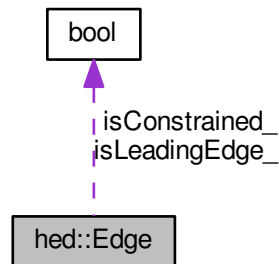


## 29.107 hed::Edge Class Reference

**Edge** class in the in the half-edge data structure.

```
#include <HeTriang.h>
```

Collaboration diagram for hed::Edge:



### Public Member Functions

- [Edge](#) ()  
*Constructor.*
- [~Edge](#) ()  
*Destructor.*
- void [setSourceNode](#) (Node \*node)  
*Sets the source node.*
- void [setNextEdgeInFace](#) (Edge \*edge)  
*Sets the next edge in face.*
- void [setTwinEdge](#) (Edge \*edge)  
*Sets the twin edge.*
- void [setAsLeadingEdge](#) (bool val=true)  
*Sets the edge as a leading edge.*
- [bool isLeadingEdge](#) () const  
*Checks if an edge is a leading edge.*
- void [setConstrained](#) (bool val=true)  
*Sets the edge as a constrained edge.*
- [bool isConstrained](#) () const  
*Checks if an edge is constrained.*
- [Edge \\*](#) [getTwinEdge](#) () const  
*Returns the twin edge.*
- [Edge \\*](#) [getNextEdgeInFace](#) () const  
*Returns the next edge in face.*
- [Node \\*](#) [getSourceNode](#) ()  
*Returns the source node.*
- [Node \\*](#) [getTargetNode](#) ()  
*Returns the target node.*

### 29.107.1 Detailed Description

**Edge** class in the in the half-edge data structure.

Definition at line 152 of file HeTriang.h.

### 29.107.2 Constructor & Destructor Documentation

#### 29.107.2.1 `hed::Edge::Edge( )` [inline]

Constructor.

Definition at line 165 of file HeTriang.h.

#### 29.107.2.2 `hed::Edge::~~Edge( )` [inline]

Destructor.

Definition at line 169 of file HeTriang.h.

### 29.107.3 Member Function Documentation

#### 29.107.3.1 `Edge* hed::Edge::getNextEdgeInFace( ) const` [inline]

Returns the next edge in face.

Definition at line 197 of file HeTriang.h.

#### 29.107.3.2 `Node* hed::Edge::getSourceNode( )` [inline]

Returns the source node.

Definition at line 200 of file HeTriang.h.

#### 29.107.3.3 `Node* hed::Edge::getTargetNode( )` [inline]

Returns the target node.

Definition at line 203 of file HeTriang.h.

#### 29.107.3.4 `Edge* hed::Edge::getTwinEdge( ) const` [inline]

Returns the twin edge.

Definition at line 194 of file HeTriang.h.

29.107.3.5 `bool hed::Edge::isConstrained ( ) const [inline]`

Checks if an edge is constrained.

Definition at line 191 of file HeTriang.h.

29.107.3.6 `bool hed::Edge::isLeadingEdge ( ) const [inline]`

Checks if an edge is a leading edge.

Definition at line 184 of file HeTriang.h.

29.107.3.7 `void hed::Edge::setAsLeadingEdge ( bool val = true ) [inline]`

Sets the edge as a leading edge.

Definition at line 181 of file HeTriang.h.

29.107.3.8 `void hed::Edge::setConstrained ( bool val = true ) [inline]`

Sets the edge as a constrained edge.

Definition at line 187 of file HeTriang.h.

29.107.3.9 `void hed::Edge::setNextEdgeInFace ( Edge * edge ) [inline]`

Sets the next edge in face.

Definition at line 175 of file HeTriang.h.

29.107.3.10 `void hed::Edge::setSourceNode ( Node * node ) [inline]`

Sets the source node.

Definition at line 172 of file HeTriang.h.

29.107.3.11 `void hed::Edge::setTwinEdge ( Edge * edge ) [inline]`

Sets the twin edge.

Definition at line 178 of file HeTriang.h.

## 29.107.4 Member Data Documentation

29.107.4.1 `bool hed::Edge::isConstrained_`

Definition at line 160 of file HeTriang.h.

#### 29.107.4.2 bool hed::Edge::isLeadingEdge\_

Definition at line 159 of file HeTriang.h.

The documentation for this class was generated from the following file:

- [ttl/include/ttl/halfedge/HeTriang.h](#)

## 29.108 hetriang::Edge Class Reference

**Edge** class in the half-edge data structure

```
#include <ttlTriang.h>
```

### Public Member Functions

- [Edge](#) ()
- [~Edge](#) ()
- void [setSourceNode](#) (shared\_ptr< [Node](#) > node)
- void [setNextEdgeInFace](#) ([Edge](#) \*edge)
- void [setTwinEdge](#) ([Edge](#) \*edge)
- void [setAsLeadingEdge](#) (bool val=true)
- bool [isLeadingEdge](#) () const
- [Edge](#) \* [getTwinEdge](#) () const
- [Edge](#) \* [getNextEdgeInFace](#) () const
- shared\_ptr< [Node](#) > [getSourceNode](#) ()
- shared\_ptr< [Node](#) > [getTargetNode](#) ()

### 29.108.1 Detailed Description

**Edge** class in the half-edge data structure

Definition at line 90 of file ttlTriang.h.

### 29.108.2 Constructor & Destructor Documentation

#### 29.108.2.1 hetriang::Edge::Edge ( ) [inline]

Definition at line 97 of file ttlTriang.h.

#### 29.108.2.2 hetriang::Edge::~Edge ( ) [inline]

Definition at line 99 of file ttlTriang.h.

### 29.108.3 Member Function Documentation

**29.108.3.1** `Edge* hetriang::Edge::getNextEdgeInFace ( ) const` `[inline]`

Definition at line 107 of file `ttriang.h`.

**29.108.3.2** `shared_ptr<Node> hetriang::Edge::getSourceNode ( )` `[inline]`

Definition at line 108 of file `ttriang.h`.

**29.108.3.3** `shared_ptr<Node> hetriang::Edge::getTargetNode ( )` `[inline]`

Definition at line 109 of file `ttriang.h`.

**29.108.3.4** `Edge* hetriang::Edge::getTwinEdge ( ) const` `[inline]`

Definition at line 106 of file `ttriang.h`.

**29.108.3.5** `bool hetriang::Edge::isLeadingEdge ( ) const` `[inline]`

Definition at line 105 of file `ttriang.h`.

**29.108.3.6** `void hetriang::Edge::setAsLeadingEdge ( bool val = true )` `[inline]`

Definition at line 104 of file `ttriang.h`.

**29.108.3.7** `void hetriang::Edge::setNextEdgeInFace ( Edge * edge )` `[inline]`

Definition at line 102 of file `ttriang.h`.

**29.108.3.8** `void hetriang::Edge::setSourceNode ( shared_ptr< Node > node )` `[inline]`

Definition at line 101 of file `ttriang.h`.

**29.108.3.9** `void hetriang::Edge::setTwinEdge ( Edge * edge )` `[inline]`

Definition at line 103 of file `ttriang.h`.

The documentation for this class was generated from the following file:

- `compositemodel/include/GoTools/compositemodel/ttriang.h`

## 29.109 EdgeType Class Reference

```
#include <PrPathTriangleSeq.h>
```

### Public Member Functions

- void [initEdgeType](#) ()
- bool [isVertex](#) (int node)

### Public Attributes

- int [vertex\\_](#) [2]

### 29.109.1 Detailed Description

[EdgeType](#) - used in [PrPathTriangleSeq](#)

Definition at line 58 of file [PrPathTriangleSeq.h](#).

### 29.109.2 Member Function Documentation

29.109.2.1 void [EdgeType::initEdgeType](#) ( )

29.109.2.2 bool [EdgeType::isVertex](#) ( int *node* )

### 29.109.3 Member Data Documentation

29.109.3.1 int [EdgeType::vertex\\_](#) [2]

Definition at line 61 of file [PrPathTriangleSeq.h](#).

The documentation for this class was generated from the following file:

- parametrization/include/GoTools/parametrization/[PrPathTriangleSeq.h](#)

## 29.110 Go::EdgeVertex Class Reference

```
#include <EdgeVertex.h>
```

## Public Member Functions

- [EdgeVertex](#) (std::vector< [ftEdge](#) \* > edges)  
*Constructor given a number of adjacent half edges.*
- [EdgeVertex](#) ([ftEdge](#) \*edge)  
*Constructor given one half edge.*
- [~EdgeVertex](#) ()  
*Destructor.*
- void [addEdge](#) ([ftEdge](#) \*edge)
- void [removeEdge](#) ([ftEdge](#) \*edge)
- void [addEdgeVertex](#) ([EdgeVertex](#) \*other)  
*Join the content of two edge vertex instances.*
- std::vector< [ftEdge](#) \* > [allEdges](#) () const  
*Returns all edges meeting in this vertex including twins.*
- std::vector< [ftEdge](#) \* > [allEdges](#) ([Body](#) \*bd)
- std::vector< [ftEdge](#) \* > [uniqueEdges](#) ()
- std::vector< [ftEdge](#) \* > [uniqueEdges](#) ([Body](#) \*bd)
- int [nmbUniqueEdges](#) ()
- int [nmbUniqueEdges](#) ([Body](#) \*bd) const
- [ftEdge](#) \* [getEdge](#) (int idx)
- bool [hasEdge](#) ([ftEdge](#) \*edge) const  
*Check if a half edge is collected in this radial edge.*
- bool [hasEdgeSingle](#) ([ftEdge](#) \*edge) const
- std::vector< [ftSurface](#) \* > [getAdjacentFaces](#) () const  
*Get all adjacent faces.*
- std::vector< [ftSurface](#) \* > [getAdjacentFaces](#) ([Body](#) \*bd) const  
*Get all adjacent faces belonging to a given body.*
- std::vector< [Body](#) \* > [getAdjacentBodies](#) () const  
*Get all adjacent bodies.*
- void [organizeTwins](#) ()  
*Reset twin organization.*
- void [reOrganize](#) ()  
*Reorganize edges to get better pairs of twins.*
- void [disconnectTwin](#) ([ftEdge](#) \*e1, [ftEdge](#) \*e2)
- void [splitAtVertex](#) (shared\_ptr< [Vertex](#) > v1, shared\_ptr< [Vertex](#) > v2, shared\_ptr< [Vertex](#) > split\_vx)  
*Split edge vertex.*
- void [averageSplineEdges](#) (double eps)
- bool [checkRadialEdgeTopology](#) ()  
*Debug functionality.*

### 29.110.1 Detailed Description

Represents the radial edge structure in a non-manifold. Edges in a volume model setting. This entity corresponds to [Vertex](#) in the surface model setting.

Definition at line 63 of file [EdgeVertex.h](#).

### 29.110.2 Constructor & Destructor Documentation

#### 29.110.2.1 Go::EdgeVertex::EdgeVertex ( std::vector< [ftEdge](#) \* > edges )

Constructor given a number of adjacent half edges.

29.110.2.2 `Go::EdgeVertex::EdgeVertex ( ftEdge * edge )`

Constructor given one half edge.

29.110.2.3 `Go::EdgeVertex::~~EdgeVertex ( )`

Destructor.

### 29.110.3 Member Function Documentation

29.110.3.1 `void Go::EdgeVertex::addEdge ( ftEdge * edge )`

Add another half edge to this radial edge. Used in topology build for volume models

29.110.3.2 `void Go::EdgeVertex::addEdgeVertex ( EdgeVertex * other )`

Join the content of two edge vertex instances.

29.110.3.3 `std::vector<ftEdge*> Go::EdgeVertex::allEdges ( ) const`

Returns all edges meeting in this vertex including twins.

29.110.3.4 `std::vector<ftEdge*> Go::EdgeVertex::allEdges ( Body * bd )`

Returns all edges belonging to a given body meeting in this vertex including twins

29.110.3.5 `void Go::EdgeVertex::averageSplineEdges ( double eps )`

Average coefficients of all spline surfaces meeting in this radial edge

29.110.3.6 `bool Go::EdgeVertex::checkRadialEdgeTopology ( )`

Debug functionality.

29.110.3.7 `void Go::EdgeVertex::disconnectTwin ( ftEdge * e1, ftEdge * e2 )`

Disconnect twin edges collected into this edge vertex, also in this data structure. Called from [ftEdge](#).

29.110.3.8 `std::vector<Body*> Go::EdgeVertex::getAdjacentBodies ( ) const`

Get all adjacent bodies.



29.110.3.9 `std::vector<ftSurface*> Go::EdgeVertex::getAdjacentFaces ( ) const`

Get all adjacent faces.

29.110.3.10 `std::vector<ftSurface*> Go::EdgeVertex::getAdjacentFaces ( Body * bd ) const`

Get all adjacent faces belonging to a given body.

29.110.3.11 `ftEdge* Go::EdgeVertex::getEdge ( int idx ) [inline]`

Fetch a specified half edge, twins are counted once and an arbitrary member of a twin pair is returned

Definition at line 115 of file EdgeVertex.h.

29.110.3.12 `bool Go::EdgeVertex::hasEdge ( ftEdge * edge ) const`

Check if a half edge is collected in this radial edge.

29.110.3.13 `bool Go::EdgeVertex::hasEdgeSingle ( ftEdge * edge ) const`

Check if a half edge is collected in this radial edge and has no twin information

29.110.3.14 `int Go::EdgeVertex::nmbUniqueEdges ( ) [inline]`

Number of unique edges meeting in this radial edge, twin information is counted only once. The number coincides with the number of bodies meeting in this edge

Definition at line 104 of file EdgeVertex.h.

29.110.3.15 `int Go::EdgeVertex::nmbUniqueEdges ( Body * bd ) const`

Number of unique edges of one body meeting in this radial edge, twin information is counted only once.

29.110.3.16 `void Go::EdgeVertex::organizeTwins ( )`

Reset twin organization.

29.110.3.17 `void Go::EdgeVertex::removeEdge ( ftEdge * edge )`

Remove a half edge from this radial edge. Used in topology modifications for volume models

29.110.3.18 `void Go::EdgeVertex::reOrganize ( )`

Reorganize edges to get better pairs of twins.

29.110.3.19 `void Go::EdgeVertex::splitAtVertex ( shared_ptr< Vertex > v1, shared_ptr< Vertex > v2, shared_ptr< Vertex > split_vx )`

Split edge vertex.

29.110.3.20 `std::vector<ftEdge*> Go::EdgeVertex::uniqueEdges ( )`

Returns all geometrically unique edges meeting in this vertex. One edge is return for a pair of twins.

29.110.3.21 `std::vector<ftEdge*> Go::EdgeVertex::uniqueEdges ( Body * bd )`

Returns all geometrically unique edges belonging to a given body meeting in this vertex. One edge is return for a pair of twins.

The documentation for this class was generated from the following file:

- `compositemodel/include/GoTools/compositemodel/EdgeVertex.h`

## 29.111 Go::Element2D Class Reference

```
#include <Element2D.h>
```

### Public Member Functions

- `Element2D ( )`
- `Element2D (double start_u, double start_v, double stop_u, double stop_v)`
- `~Element2D ( )`
- `void removeSupportFunction (LRBSpline2D *f)`
- `void addSupportFunction (LRBSpline2D *f)`
- `bool hasSupportFunction (LRBSpline2D *f)`
- `Element2D * split (bool split_u, double par_value)`
- `Element2D * copy ( )`
- `double umin ( ) const`
- `double vmin ( ) const`
- `double umax ( ) const`
- `double vmax ( ) const`
- `double area ( ) const`
- `std::vector< LRBSpline2D * >::iterator supportBegin ( )`
- `std::vector< LRBSpline2D * >::iterator supportEnd ( )`
- `std::vector< LRBSpline2D * >::const_iterator supportBegin ( ) const`
- `std::vector< LRBSpline2D * >::const_iterator supportEnd ( ) const`
- `const std::vector< LRBSpline2D * > & getSupport ( ) const`
- `bool contains (double upar, double vpar)`

- [LRBSpline2D](#) \* [supportFunction](#) (int i)
- int [nmbBasisFunctions](#) () const
- void [setUmin](#) (double u)
- void [setVmin](#) (double v)
- void [setUmax](#) (double u)
- void [setVmax](#) (double v)
- bool [isOverloaded](#) () const
- void [resetOverloadCount](#) ()
- int [incrementOverloadCount](#) ()
- int [getOverloadCount](#) () const
- void [updateBasisPointers](#) (std::vector< [LRBSpline2D](#) \* > &basis)
- void [swapParameterDirection](#) ()
- bool [hasDataPoints](#) ()
- int [nmbDataPoints](#) ()
  - Number of scattered data points.*
- int [nmbGhostPoints](#) ()
  - Number of ghost points.*
- void [eraseDataPoints](#) ()
  - Remove data points associated with the element.*
- void [eraseDataPoints](#) (std::vector< double >::iterator start, std::vector< double >::iterator end)
- void [eraseGhostPoints](#) ()
- void [addDataPoints](#) (std::vector< double >::iterator start, std::vector< double >::iterator end, bool sort\_in\_u)
  - Add data points to the element.*
- void [addDataPoints](#) (std::vector< double >::iterator start, std::vector< double >::iterator end, int del, bool sort\_in\_u)
- void [addGhostPoints](#) (std::vector< double >::iterator start, std::vector< double >::iterator end, bool sort\_in\_u)
- void [addGhostPoints](#) (std::vector< double >::iterator start, std::vector< double >::iterator end, int del, bool sort\_in\_u)
- std::vector< double > & [getDataPoints](#) ()
  - Fetch data points.*
- std::vector< double > & [getGhostPoints](#) ()
  - Fetch artificial data points intended for stabilization.*
- void [getOutsidePoints](#) (std::vector< double > &points, [Direction2D](#) d, bool &sort\_in\_u)
- void [getOutsideGhostPoints](#) (std::vector< double > &points, [Direction2D](#) d, bool &sort\_in\_u)
- bool [hasLSMatrix](#) ()
  - Check if a submatrix for least squares approximation exists.*
- void [setLSMatrix](#) ()
- void [getLSMatrix](#) (double \*&LSmat, double \*&LSright, int &ncond)
  - Fetch local least squares arrays.*
- bool [hasAccuracyInfo](#) ()
  - Check if the element has accuracy information.*
- void [getAccuracyInfo](#) (double &average\_error, double &max\_error, int &nmb\_outside\_tol)
  - Fetch accuracy information.*
- int [getNmbOutsideTol](#) ()
- double [getAverageError](#) ()
- double [getAccumulatedError](#) ()
- double [getMaxError](#) ()
- void [setAccuracyInfo](#) (double accumulated\_error, double average\_error, double max\_error, int nmb\_outside\_tol)
  - Store accuracy information.*
- void [resetAccuracyInfo](#) ()
- bool [getDataBoundingBox](#) (double bb[])

- void `makeDataPoints3D` ()
- void `updateAccuracyInfo` ()
- void `updateLSDataParDomain` (double u1, double u2, double v1, double v2, double u1new, double u2new, double v1new, double v2new)
- bool `isModified` ()
  - Check if the element has been modified lately.*
- void `resetModificationFlag` ()
  - Reset modified flag.*
- void `setModified` ()
  - Set modified flag to true.*
- double `sumOfScaledBsplines` (double upar, double vpar)
- std::vector< double > `unitSquareBernsteinBasis` () const
- `SplineCurve` \* `curveOnElement` (double start\_u, double start\_v, double end\_u, double end\_v) const

### 29.111.1 Detailed Description

Definition at line 251 of file Element2D.h.

### 29.111.2 Constructor & Destructor Documentation

29.111.2.1 `Go::Element2D::Element2D` ( )

29.111.2.2 `Go::Element2D::Element2D` ( double start\_u, double start\_v, double stop\_u, double stop\_v )

29.111.2.3 `Go::Element2D::~~Element2D` ( )

### 29.111.3 Member Function Documentation

29.111.3.1 void `Go::Element2D::addDataPoints` ( std::vector< double >::iterator start, std::vector< double >::iterator end, bool sort\_in\_u ) [inline]

Add data points to the element.

Definition at line 341 of file Element2D.h.

29.111.3.2 void `Go::Element2D::addDataPoints` ( std::vector< double >::iterator start, std::vector< double >::iterator end, int del, bool sort\_in\_u ) [inline]

Definition at line 349 of file Element2D.h.

29.111.3.3 void `Go::Element2D::addGhostPoints` ( std::vector< double >::iterator start, std::vector< double >::iterator end, bool sort\_in\_u ) [inline]

Definition at line 359 of file Element2D.h.

29.111.3.4 `void Go::Element2D::addGhostPoints ( std::vector< double >::iterator start, std::vector< double >::iterator end, int del, bool sort_in_u ) [inline]`

Definition at line 367 of file Element2D.h.

29.111.3.5 `void Go::Element2D::addSupportFunction ( LRBSpline2D * f )`

29.111.3.6 `double Go::Element2D::area ( ) const [inline]`

Definition at line 267 of file Element2D.h.

29.111.3.7 `bool Go::Element2D::contains ( double upar, double vpar ) [inline]`

Definition at line 281 of file Element2D.h.

29.111.3.8 `Element2D* Go::Element2D::copy ( )`

29.111.3.9 `SplineCurve* Go::Element2D::curveOnElement ( double start_u, double start_v, double end_u, double end_v ) const`

Get the Bezier curve (as a spline curve) given as the image of a line segment in the parameter space of the spline surface. The parameter domain of the curve will be [0, 1].

#### Parameters

|                            |                                                    |
|----------------------------|----------------------------------------------------|
| <code>start↔<br/>_u</code> | The u-value of the start point of the line segment |
| <code>start↔<br/>_v</code> | The v-value of the start point of the line segment |
| <code>end↔<br/>_u</code>   | The u-value of the start point of the line segment |
| <code>end↔<br/>_v</code>   | The v-value of the start point of the line segment |

#### Returns

The spline curve

29.111.3.10 `void Go::Element2D::eraseDataPoints ( ) [inline]`

Remove data points associated with the element.

Definition at line 321 of file Element2D.h.

29.111.3.11 `void Go::Element2D::eraseDataPoints ( std::vector< double >::iterator start, std::vector< double >::iterator end ) [inline]`

Definition at line 327 of file Element2D.h.

29.111.3.12 `void Go::Element2D::eraseGhostPoints ( ) [inline]`

Definition at line 334 of file Element2D.h.

29.111.3.13 `double Go::Element2D::getAccumulatedError ( ) [inline]`

Definition at line 461 of file Element2D.h.

29.111.3.14 `void Go::Element2D::getAccuracyInfo ( double & average_error, double & max_error, int & nmb_outside_tol ) [inline]`

Fetch accuracy information.

Definition at line 431 of file Element2D.h.

29.111.3.15 `double Go::Element2D::getAverageError ( ) [inline]`

Definition at line 452 of file Element2D.h.

29.111.3.16 `bool Go::Element2D::getDataBoundingBox ( double bb[ ] )`

29.111.3.17 `std::vector<double>& Go::Element2D::getDataPoints ( ) [inline]`

Fetch data points.

Definition at line 377 of file Element2D.h.

29.111.3.18 `std::vector<double>& Go::Element2D::getGhostPoints ( ) [inline]`

Fetch artificial data points intended for stabilization.

Definition at line 385 of file Element2D.h.

29.111.3.19 `void Go::Element2D::getLSMatrix ( double *& LSmat, double *& LSright, int & ncond ) [inline]`

Fetch local least squares arrays.

Definition at line 414 of file Element2D.h.

29.111.3.20 `double Go::Element2D::getMaxError ( ) [inline]`

Definition at line 470 of file Element2D.h.

29.111.3.21 `int Go::Element2D::getNmbOutsideTol ( ) [inline]`

Definition at line 443 of file Element2D.h.

29.111.3.22 `void Go::Element2D::getOutsideGhostPoints ( std::vector< double > & points, Direction2D d, bool & sort_in_u )`

29.111.3.23 `void Go::Element2D::getOutsidePoints ( std::vector< double > & points, Direction2D d, bool & sort_in_u )`

Split point set according to a modified size of the element and return the points lying outside the current element

29.111.3.24 `int Go::Element2D::getOverloadCount ( ) const [inline]`

Definition at line 297 of file Element2D.h.

29.111.3.25 `const std::vector<LRBSpline2D*> & Go::Element2D::getSupport ( ) const [inline]`

Definition at line 272 of file Element2D.h.

29.111.3.26 `bool Go::Element2D::hasAccuracyInfo ( ) [inline]`

Check if the element has accuracy information.

Definition at line 422 of file Element2D.h.

29.111.3.27 `bool Go::Element2D::hasDataPoints ( ) [inline]`

Check if the element is associated data points to be used in least squares approximation

Definition at line 306 of file Element2D.h.

29.111.3.28 `bool Go::Element2D::hasLSMatrix ( ) [inline]`

Check if a submatrix for least squares approximation exists.

Definition at line 401 of file Element2D.h.

29.111.3.29 `bool Go::Element2D::hasSupportFunction ( LRBSpline2D * f )`

29.111.3.30 `int Go::Element2D::incrementOverloadCount ( ) [inline]`

Definition at line 296 of file Element2D.h.

29.111.3.31 **bool** Go::Element2D::isModified ( ) [inline]

Check if the element has been modified lately.

Definition at line 515 of file Element2D.h.

29.111.3.32 **bool** Go::Element2D::isOverloaded ( ) const

29.111.3.33 **void** Go::Element2D::makeDataPoints3D ( )

29.111.3.34 **int** Go::Element2D::nmbBasisFunctions ( ) const [inline]

Definition at line 288 of file Element2D.h.

29.111.3.35 **int** Go::Element2D::nmbDataPoints ( )

Number of scattered data points.

29.111.3.36 **int** Go::Element2D::nmbGhostPoints ( )

Number of ghost points.

29.111.3.37 **void** Go::Element2D::removeSupportFunction ( LRBSpline2D \* f )

29.111.3.38 **void** Go::Element2D::resetAccuracyInfo ( ) [inline]

Definition at line 491 of file Element2D.h.

29.111.3.39 **void** Go::Element2D::resetModificationFlag ( ) [inline]

Reset modified flag.

Definition at line 521 of file Element2D.h.

29.111.3.40 **void** Go::Element2D::resetOverloadCount ( ) [inline]

Definition at line 295 of file Element2D.h.

29.111.3.41 **void** Go::Element2D::setAccuracyInfo ( *double accumulated\_error*, *double average\_error*, *double max\_error*,  
*int nmb\_outside\_tol* ) [inline]

Store accuracy information.

Definition at line 480 of file Element2D.h.



29.111.3.42 `void Go::Element2D::setLSMatrix ( )`

Create scratch to store local least squares approximation arrays

29.111.3.43 `void Go::Element2D::setModified ( )` [inline]

Set modified flag to true.

Definition at line 527 of file Element2D.h.

29.111.3.44 `void Go::Element2D::setUmax ( double u )` [inline]

Definition at line 291 of file Element2D.h.

29.111.3.45 `void Go::Element2D::setUmin ( double u )` [inline]

Definition at line 289 of file Element2D.h.

29.111.3.46 `void Go::Element2D::setVmax ( double v )` [inline]

Definition at line 292 of file Element2D.h.

29.111.3.47 `void Go::Element2D::setVmin ( double v )` [inline]

Definition at line 290 of file Element2D.h.

29.111.3.48 `Element2D* Go::Element2D::split ( bool split_u, double par_value )`

29.111.3.49 `double Go::Element2D::sumOfScaledBsplines ( double upar, double vpar )`

29.111.3.50 `std::vector<LRBSpline2D*>::iterator Go::Element2D::supportBegin ( )` [inline]

Definition at line 268 of file Element2D.h.

29.111.3.51 `std::vector<LRBSpline2D*>::const_iterator Go::Element2D::supportBegin ( ) const` [inline]

Definition at line 270 of file Element2D.h.

29.111.3.52 `std::vector<LRBSpline2D*>::iterator Go::Element2D::supportEnd ( )` [inline]

Definition at line 269 of file Element2D.h.

29.111.3.53 `std::vector<LRBSpline2D*>::const_iterator Go::Element2D::supportEnd ( ) const` `[inline]`

Definition at line 271 of file Element2D.h.

29.111.3.54 `LRBSpline2D* Go::Element2D::supportFunction ( int i )` `[inline]`

Definition at line 287 of file Element2D.h.

29.111.3.55 `void Go::Element2D::swapParameterDirection ( )`

29.111.3.56 `double Go::Element2D::umax ( ) const` `[inline]`

Definition at line 265 of file Element2D.h.

29.111.3.57 `double Go::Element2D::umin ( ) const` `[inline]`

Definition at line 263 of file Element2D.h.

29.111.3.58 `std::vector<double> Go::Element2D::unitSquareBernsteinBasis ( ) const`

Get the coefficients of the underlying Ir-splinesurface on this element, expressed by the Bernstein basis after a linear transformation sending this element to the unit square

#### Returns

a vector of the coefficients of the control points  $p_{ij}$  in order  $p_{00}[0]$ ,  $p_{00}[1]$ , ...,  $p_{10}[0]$ , ...,  $p_{01}[0]$ , ...

29.111.3.59 `void Go::Element2D::updateAccuracyInfo ( )`

29.111.3.60 `void Go::Element2D::updateBasisPointers ( std::vector< LRBSpline2D * > & basis )`

29.111.3.61 `void Go::Element2D::updateLSDataParDomain ( double u1, double u2, double v1, double v2, double u1new, double u2new, double v1new, double v2new )`

29.111.3.62 `double Go::Element2D::vmax ( ) const` `[inline]`

Definition at line 266 of file Element2D.h.

29.111.3.63 `double Go::Element2D::vmin ( ) const` `[inline]`

Definition at line 264 of file Element2D.h.

The documentation for this class was generated from the following file:

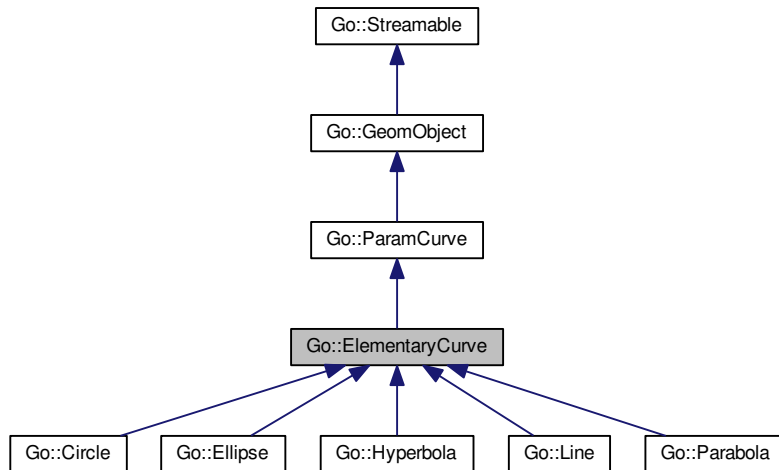
- [Irsplines2D/include/GoTools/Irsplines2D/Element2D.h](#)

## 29.112 Go::ElementaryCurve Class Reference

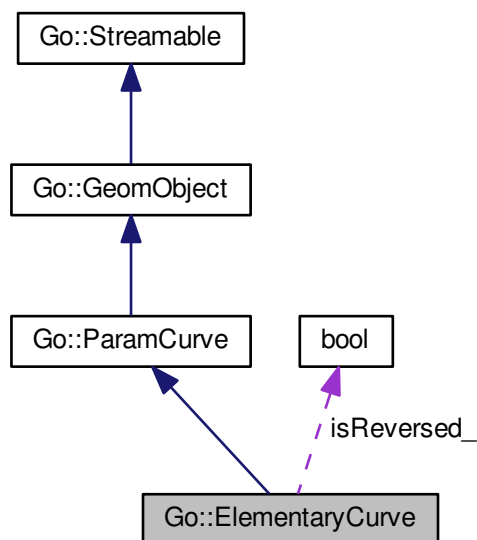
[ElementaryCurve](#) is a base class for elementary curves like lines and circles. Such curves have natural parametrizations and [ElementaryCurve](#) is therefore a subclass of [ParamCurve](#).

```
#include <ElementaryCurve.h>
```

Inheritance diagram for Go::ElementaryCurve:



Collaboration diagram for Go::ElementaryCurve:



## Public Member Functions

- [ElementaryCurve](#) ()
- virtual [~ElementaryCurve](#) ()  
*Virtual destructor, enables safe inheritance.*
- virtual [ElementaryCurve](#) \* [clone](#) () const =0
- virtual void [reverseParameterDirection](#) (bool switchparam=false)
- virtual [SplineCurve](#) \* [createSplineCurve](#) () const =0
- virtual void [setParamBounds](#) (double startpar, double endpar)=0
- virtual void [translateCurve](#) (const [Point](#) &dir)=0
- virtual bool [isReversed](#) () const
- virtual void [swapParameters2D](#) ()=0
- virtual [ElementaryCurve](#) \* [subCurve](#) (double from\_par, double to\_par, double fuzzy=DEFAULT\_PARAMETER\_EPSILON) const =0

## Protected Member Functions

- void [getReversedParameter](#) (double &t) const

## Protected Attributes

- bool [isReversed\\_](#)

## Additional Inherited Members

### 29.112.1 Detailed Description

[ElementaryCurve](#) is a base class for elementary curves like lines and circles. Such curves have natural parametrizations and [ElementaryCurve](#) is therefore a subclass of [ParamCurve](#).

Definition at line 54 of file [ElementaryCurve.h](#).

### 29.112.2 Constructor & Destructor Documentation

29.112.2.1 [Go::ElementaryCurve::ElementaryCurve](#) ( )

29.112.2.2 virtual [Go::ElementaryCurve::~~ElementaryCurve](#) ( ) [virtual]

Virtual destructor, enables safe inheritance.

### 29.112.3 Member Function Documentation

29.112.3.1 virtual [ElementaryCurve](#)\* [Go::ElementaryCurve::clone](#) ( ) const [pure virtual]

The clone-function is inherited from [GeomObject](#), but overridden here to get a covariant return type (for those compilers that allow this).

Implements [Go::ParamCurve](#).

Implemented in [Go::Line](#), [Go::Circle](#), [Go::Hyperbola](#), [Go::Parabola](#), and [Go::Ellipse](#).

29.112.3.2 virtual `SplineCurve*` `Go::ElementaryCurve::createSplineCurve ( ) const` `[pure virtual]`

Implemented in [Go::Circle](#), [Go::Line](#), [Go::Hyperbola](#), [Go::Parabola](#), and [Go::Ellipse](#).

29.112.3.3 void `Go::ElementaryCurve::getReversedParameter ( double & t ) const` `[protected]`

29.112.3.4 virtual `bool` `Go::ElementaryCurve::isReversed ( ) const` `[virtual]`

29.112.3.5 virtual void `Go::ElementaryCurve::reverseParameterDirection ( bool switchparam = false )` `[virtual]`

Set the parameter direction of the curve. The curve's parameter interval will always remain constant, but by flipping the parameter direction, the curve will be traced the opposite way when moving a parameter over the parameter interval.

#### Parameters

|                    |                                                                                                                                                   |
|--------------------|---------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>switchparam</i> | if true, and the curve is 2D, the x and y coordinates should be swapped. This is used when turning the orientation of bounded (trimmed) surfaces. |
|--------------------|---------------------------------------------------------------------------------------------------------------------------------------------------|

Implements [Go::ParamCurve](#).

29.112.3.6 virtual void `Go::ElementaryCurve::setParamBounds ( double startpar, double endpar )` `[pure virtual]`

Set bounds for the parametrization of the curve

#### Parameters

|                 |                 |
|-----------------|-----------------|
| <i>startpar</i> | start parameter |
| <i>endpar</i>   | end parameter   |

Implemented in [Go::Circle](#), [Go::Ellipse](#), [Go::Line](#), [Go::Hyperbola](#), and [Go::Parabola](#).

29.112.3.7 virtual `ElementaryCurve*` `Go::ElementaryCurve::subCurve ( double from_par, double to_par, double fuzzy = DEFAULT_PARAMETER_EPSILON ) const` `[pure virtual]`

Returns a curve which is a part of this curve. The result is NEWed, so the user is responsible for deleting it. NB: It is not guaranteed that the [ParamCurve](#) that is returned is of the same type as the curve itself. Thus, the returned curve might be a [SplineCurve](#).

#### Parameters

|                 |                                                                                                                                                                                                   |
|-----------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>from_par</i> | start value of parameter interval that will define the subcurve                                                                                                                                   |
| <i>to_par</i>   | end value of parameter interval that will define the subcurve                                                                                                                                     |
| <i>fuzzy</i>    | since subCurve works on those curves who are spline-based, this tolerance defines how close the start and end parameter must be to an existing knot in order to be considered <i>on</i> the knot. |

**Returns**

a pointer to a new subcurve which represents the part of the curve between 'from\_par' and 'to\_par'. It will be spline-based and have a k-regular knotvector. The user is responsible for deleting this subcurve when it is no longer needed.

Implements [Go::ParamCurve](#).

Implemented in [Go::Circle](#), [Go::Line](#), [Go::Hyperbola](#), [Go::Parabola](#), and [Go::Ellipse](#).

29.112.3.8 `virtual void Go::ElementaryCurve::swapParameters2D ( ) [pure virtual]`

Implemented in [Go::Circle](#), [Go::Ellipse](#), [Go::Line](#), [Go::Hyperbola](#), and [Go::Parabola](#).

29.112.3.9 `virtual void Go::ElementaryCurve::translateCurve ( const Point & dir ) [pure virtual]`

Implemented in [Go::Circle](#), [Go::Ellipse](#), [Go::Line](#), [Go::Parabola](#), and [Go::Hyperbola](#).

**29.112.4 Member Data Documentation**

29.112.4.1 `bool Go::ElementaryCurve::isReversed_ [protected]`

Definition at line 92 of file ElementaryCurve.h.

The documentation for this class was generated from the following file:

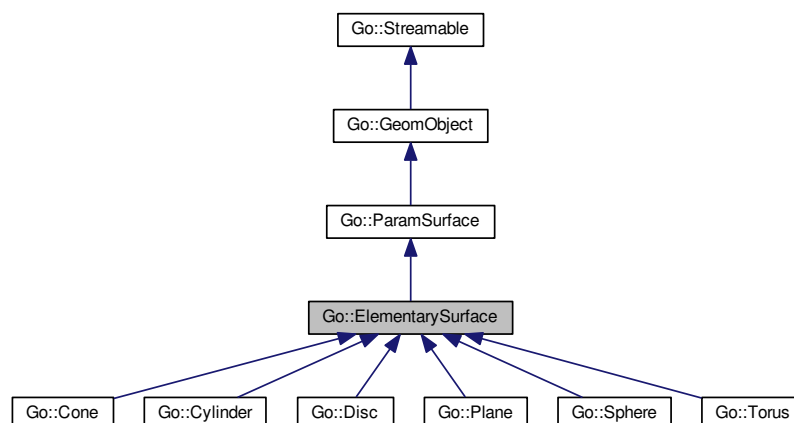
- [gtools-core/include/GoTools/geometry/ElementaryCurve.h](#)

**29.113 Go::ElementarySurface Class Reference**

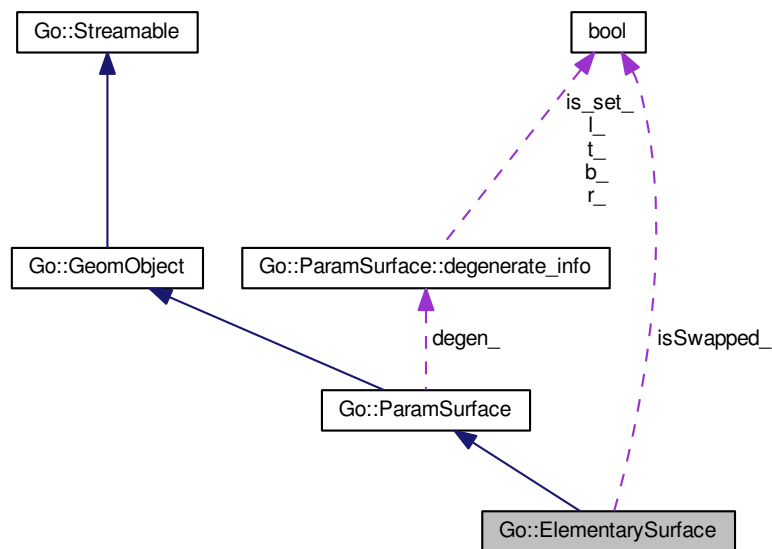
[ElementarySurface](#) is a base class for elementary surfaces like planes and cylinders. Such surfaces have natural parametrizations and [ElementarySurface](#) is therefore a subclass of [ParamSurface](#). These surfaces are non-self-intersecting.

```
#include <ElementarySurface.h>
```

Inheritance diagram for [Go::ElementarySurface](#):



Collaboration diagram for Go::ElementarySurface:



## Public Member Functions

- [ElementarySurface](#) ()
- virtual [~ElementarySurface](#) ()
  - Virtual destructor, enables safe inheritance.*
- virtual [ElementarySurface](#) \* [clone](#) () const =0
- virtual [CurveLoop](#) [outerBoundaryLoop](#) (double [degenerate\\_epsilon](#)=DEFAULT\_SPACE\_EPSILON) const
- [RectDomain](#) [containingDomain](#) () const
- virtual [Point](#) [closestInDomain](#) (double [u](#), double [v](#)) const
- virtual [bool](#) [inDomain](#) (double [u](#), double [v](#), double [eps](#)=1.0e-4) const
  - Check if a parameter pair lies inside the domain of this surface.*
- virtual [int](#) [inDomain2](#) (double [u](#), double [v](#), double [eps](#)=1.0e-4) const
- virtual [bool](#) [onBoundary](#) (double [u](#), double [v](#), double [eps](#)=1.0e-4) const
  - Check if a parameter pair lies at the boundary of this surface.*
- virtual [double](#) [area](#) (double [tol](#)) const
- virtual void [getCornerPoints](#) (std::vector< std::pair< [Point](#), [Point](#) > > &[corners](#)) const
- virtual [SplineSurface](#) \* [asSplineSurface](#) ()
  - Return the spline surface represented by this surface, if any.*
- virtual [bool](#) [isBounded](#) () const
- virtual [bool](#) [isClosed](#) (bool &[closed\\_dir\\_u](#), bool &[closed\\_dir\\_v](#)) const
- virtual [SplineSurface](#) \* [geometrySurface](#) () const =0
- virtual [SplineSurface](#) \* [createSplineSurface](#) () const =0
  - Create a [SplineSurface](#) representation of the elementary surface.*
- virtual void [setParameterBounds](#) (double [from\\_upar](#), double [from\\_vpar](#), double [to\\_upar](#), double [to\\_vpar](#))=0
  - Limit the surface by limiting the parameter domain.*
- virtual [shared\\_ptr](#)< [ElementaryCurve](#) > [getElementaryParamCurve](#) ([ElementaryCurve](#) \*[space\\_crv](#), double [tol](#), const [Point](#) \*[start\\_par\\_pt](#)=NULL, const [Point](#) \*[end\\_par\\_pt](#)=NULL) const
- virtual void [turnOrientation](#) ()

*Turns the direction of the normal of the surface.*

- virtual void [reverseParameterDirection](#) (bool direction\_is\_u)
- virtual void [swapParameterDirection](#) ()

*Swaps the two parameter directions.*

- virtual bool [isSwapped](#) () const

### Protected Member Functions

- void [getOrientedParameters](#) (double &u, double &v) const

### Protected Attributes

- bool [isSwapped\\_](#)

### Additional Inherited Members

#### 29.113.1 Detailed Description

[ElementarySurface](#) is a base class for elementary surfaces like planes and cylinders. Such surfaces have natural parametrizations and [ElementarySurface](#) is therefore a subclass of [ParamSurface](#). These surfaces are non-self-intersecting.

Definition at line 60 of file [ElementarySurface.h](#).

#### 29.113.2 Constructor & Destructor Documentation

29.113.2.1 `Go::ElementarySurface::ElementarySurface ( )`

29.113.2.2 `virtual Go::ElementarySurface::~~ElementarySurface ( ) [virtual]`

Virtual destructor, enables safe inheritance.

#### 29.113.3 Member Function Documentation

29.113.3.1 `virtual double Go::ElementarySurface::area ( double tol ) const [virtual]`

Compute the total area of this surface up to some tolerance

##### Parameters

|            |                                                                                                                                       |
|------------|---------------------------------------------------------------------------------------------------------------------------------------|
| <i>tol</i> | the relative tolerance when approximating the area, i.e. stop iteration when error becomes smaller than $tol / (\text{surface area})$ |
|------------|---------------------------------------------------------------------------------------------------------------------------------------|



**Returns**

the area calculated

Implements [Go::ParamSurface](#).

**29.113.3.2** `virtual SplineSurface* Go::ElementarySurface::asSplineSurface ( ) [virtual]`

Return the spline surface represented by this surface, if any.

Reimplemented from [Go::ParamSurface](#).

**29.113.3.3** `virtual ElementarySurface* Go::ElementarySurface::clone ( ) const [pure virtual]`

make a clone of this surface and return a pointer to it (user is responsible for clearing up memory afterwards).

**Returns**

pointer to cloned object

Implements [Go::ParamSurface](#).

Implemented in [Go::Disc](#), [Go::Torus](#), [Go::Plane](#), [Go::Cone](#), [Go::Cylinder](#), and [Go::Sphere](#).

**29.113.3.4** `virtual Point Go::ElementarySurface::closestInDomain ( double u, double v ) const [virtual]`

Fetch the parameter value in the parameter domain of the surface closest to the parameter pair (u,v)

Implements [Go::ParamSurface](#).

**29.113.3.5** `RectDomain Go::ElementarySurface::containingDomain ( ) const [virtual]`

Get a rectangular parameter domain that is guaranteed to contain the surface's [parameterDomain\(\)](#). It may be the same. There is no guarantee that this is the smallest domain containing the actual domain.

**Returns**

a [RectDomain](#) that is guaranteed to include the surface's total parameter domain.

Implements [Go::ParamSurface](#).

**29.113.3.6** `virtual SplineSurface* Go::ElementarySurface::createSplineSurface ( ) const [pure virtual]`

Create a [SplineSurface](#) representation of the elementary surface.

Implemented in [Go::Torus](#), [Go::Cone](#), [Go::Cylinder](#), [Go::Sphere](#), [Go::Plane](#), and [Go::Disc](#).

29.113.3.7 `virtual SplineSurface* Go::ElementarySurface::geometrySurface ( ) const` [pure virtual]

Implemented in [Go::Torus](#), [Go::Cone](#), [Go::Cylinder](#), [Go::Sphere](#), [Go::Plane](#), and [Go::Disc](#).

29.113.3.8 `virtual void Go::ElementarySurface::getCornerPoints ( std::vector< std::pair< Point, Point > > & corners ) const` [virtual]

Return surface corners, geometric and parametric points in that sequence

Implements [Go::ParamSurface](#).

29.113.3.9 `virtual shared_ptr<ElementaryCurve> Go::ElementarySurface::getElementaryParamCurve ( ElementaryCurve * space_crv, double tol, const Point * start_par_pt=NULL, const Point * end_par_pt=NULL ) const` [virtual]

Fetch the parameter curve in the domain of the elementary surface corresponding to a given elementary curve in geometry space if this curve has a simpler elementary representation. Otherwise, nothing is returned

Reimplemented in [Go::Cone](#), [Go::Cylinder](#), and [Go::Sphere](#).

29.113.3.10 `void Go::ElementarySurface::getOrientedParameters ( double & u, double & v ) const` [protected]

29.113.3.11 `virtual bool Go::ElementarySurface::inDomain ( double u, double v, double eps = 1.0e-4 ) const` [virtual]

Check if a parameter pair lies inside the domain of this surface.

Implements [Go::ParamSurface](#).

29.113.3.12 `virtual int Go::ElementarySurface::inDomain2 ( double u, double v, double eps = 1.0e-4 ) const` [virtual]

Check if a parameter pair lies inside the domain of this surface return value = 0: outside = 1: internal = 2: at the boundary

Implements [Go::ParamSurface](#).

29.113.3.13 `virtual bool Go::ElementarySurface::isBounded ( ) const` [virtual]

Reimplemented in [Go::Plane](#), [Go::Cylinder](#), [Go::Cone](#), [Go::Disc](#), [Go::Torus](#), and [Go::Sphere](#).

29.113.3.14 `virtual bool Go::ElementarySurface::isClosed ( bool & closed_dir_u, bool & closed_dir_v ) const` [virtual]

Reimplemented in [Go::Plane](#), [Go::Cone](#), [Go::Cylinder](#), [Go::Disc](#), [Go::Torus](#), and [Go::Sphere](#).

29.113.3.15 `virtual bool Go::ElementarySurface::isSwapped ( ) const [virtual]`

29.113.3.16 `virtual bool Go::ElementarySurface::onBoundary ( double u, double v, double eps = 1.0e-4 ) const [virtual]`

Check if a parameter pair lies at the boundary of this surface.

Implements [Go::ParamSurface](#).

29.113.3.17 `virtual CurveLoop Go::ElementarySurface::outerBoundaryLoop ( double degenerate_epsilon = DEFAULT_SPACE_EPSILON ) const [virtual]`

Returns the anticlockwise, outer boundary loop of the surface.

Parameters

|                           |                                                            |
|---------------------------|------------------------------------------------------------|
| <i>degenerate_epsilon</i> | edges whose length is smaller than this value are ignored. |
|---------------------------|------------------------------------------------------------|

Returns

a [CurveLoop](#) describing the anticlockwise, outer boundary loop of the surface. A negative *degenerate\_epsilon* indicates that all curves, also the degenerate ones are wanted

Implements [Go::ParamSurface](#).

29.113.3.18 `virtual void Go::ElementarySurface::reverseParameterDirection ( bool direction_is_u ) [virtual]`

Reverses the direction of the basis in input direction.

Parameters

|                       |                                                                                                                       |
|-----------------------|-----------------------------------------------------------------------------------------------------------------------|
| <i>direction_is_u</i> | if 'true', the first parameter direction will be reversed, otherwise, the second parameter direction will be reversed |
|-----------------------|-----------------------------------------------------------------------------------------------------------------------|

Implements [Go::ParamSurface](#).

29.113.3.19 `virtual void Go::ElementarySurface::setParameterBounds ( double from_upar, double from_vpar, double to_upar, double to_vpar ) [pure virtual]`

Limit the surface by limiting the parameter domain.

Implemented in [Go::Torus](#), [Go::Disc](#), [Go::Sphere](#), [Go::Plane](#), [Go::Cylinder](#), and [Go::Cone](#).

29.113.3.20 `virtual void Go::ElementarySurface::swapParameterDirection ( ) [virtual]`

Swaps the two parameter directions.

Implements [Go::ParamSurface](#).

29.113.3.21 `virtual void Go::ElementarySurface::turnOrientation ( ) [virtual]`

Turns the direction of the normal of the surface.

Implements [Go::ParamSurface](#).

#### 29.113.4 Member Data Documentation

29.113.4.1 `bool Go::ElementarySurface::isSwapped_ [protected]`

Definition at line 135 of file `ElementarySurface.h`.

The documentation for this class was generated from the following file:

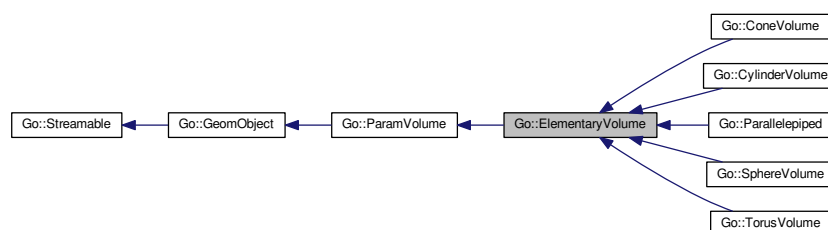
- [gotools-core/include/GoTools/geometry/ElementarySurface.h](#)

### 29.114 Go::ElementaryVolume Class Reference

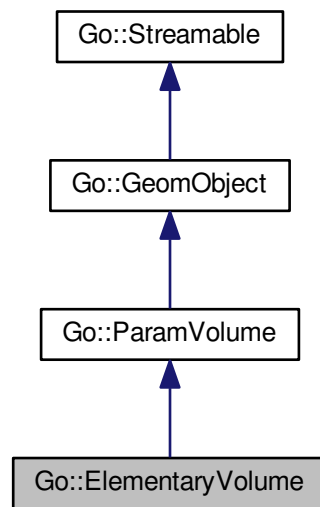
[ElementaryVolume](#) is a base class for elementary volumes like boxes and solid cylinders. Such volumes have natural parametrizations and `ElementaryVolume` is therefore a subclass of [ParamVolume](#). These volumes are non-self-intersecting.

```
#include <ElementaryVolume.h>
```

Inheritance diagram for `Go::ElementaryVolume`:



Collaboration diagram for Go::ElementaryVolume:



## Public Member Functions

- virtual [~ElementaryVolume](#) ()  
*Virtual destructor, enables safe inheritance.*
- virtual [ElementaryVolume](#) \* [clone](#) () const =0
- virtual [SplineVolume](#) \* [geometryVolume](#) () const =0  
*Make a NURBS representation of the object.*

## Additional Inherited Members

### 29.114.1 Detailed Description

[ElementaryVolume](#) is a base class for elementary volumes like boxes and solid cylinders. Such volumes have natural parametrizations and [ElementaryVolume](#) is therefore a subclass of [ParamVolume](#). These volumes are non-self-intersecting.

Definition at line 58 of file [ElementaryVolume.h](#).

### 29.114.2 Constructor & Destructor Documentation

29.114.2.1 virtual [Go::ElementaryVolume::~ElementaryVolume](#) ( ) [[inline](#)],[[virtual](#)]

Virtual destructor, enables safe inheritance.

Definition at line 64 of file [ElementaryVolume.h](#).

### 29.114.3 Member Function Documentation

29.114.3.1 `virtual ElementaryVolume* Go::ElementaryVolume::clone ( ) const` [pure virtual]

make a clone of this volume and return a pointer to it (user is responsible for clearing up memory afterwards).

#### Returns

pointer to cloned object

Implements [Go::ParamVolume](#).

Implemented in [Go::TorusVolume](#), [Go::ConeVolume](#), [Go::CylinderVolume](#), [Go::SphereVolume](#), and [Go::↔Parallelepiped](#).

29.114.3.2 `virtual SplineVolume* Go::ElementaryVolume::geometryVolume ( ) const` [pure virtual]

Make a NURBS representation of the object.

Implemented in [Go::TorusVolume](#), [Go::SphereVolume](#), [Go::ConeVolume](#), [Go::CylinderVolume](#), and [Go::↔Parallelepiped](#).

The documentation for this class was generated from the following file:

- [trivariate/include/GoTools/trivariate/ElementaryVolume.h](#)

## 29.115 Go::LRSplineSurface::ElemKey Struct Reference

```
#include <LRSplineSurface.h>
```

### Public Member Functions

- `bool operator< (const ElemKey &rhs) const`

### Public Attributes

- `double u_min`
- `double v_min`

### 29.115.1 Detailed Description

Definition at line 135 of file `LRSplineSurface.h`.

## 29.115.2 Member Function Documentation

29.115.2.1 `bool Go::LRSplineSurface::ElemKey::operator<( const ElemKey & rhs ) const` `[inline]`

Definition at line 227 of file LRSplineSurface\_impl.h.

## 29.115.3 Member Data Documentation

29.115.3.1 `double Go::LRSplineSurface::ElemKey::u_min`

Definition at line 137 of file LRSplineSurface.h.

29.115.3.2 `double Go::LRSplineSurface::ElemKey::v_min`

Definition at line 137 of file LRSplineSurface.h.

The documentation for this struct was generated from the following files:

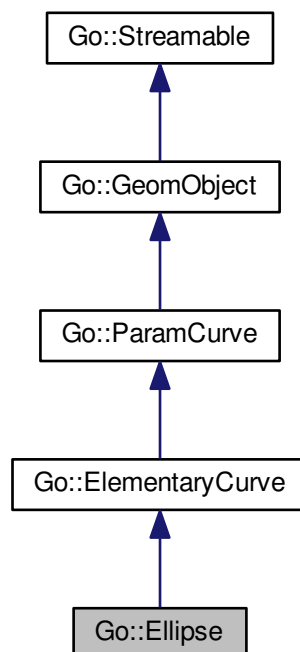
- [Irsplines2D/include/GoTools/Irsplines2D/LRSplineSurface.h](#)
- [Irsplines2D/include/GoTools/Irsplines2D/LRSplineSurface\\_impl.h](#)

## 29.116 Go::Ellipse Class Reference

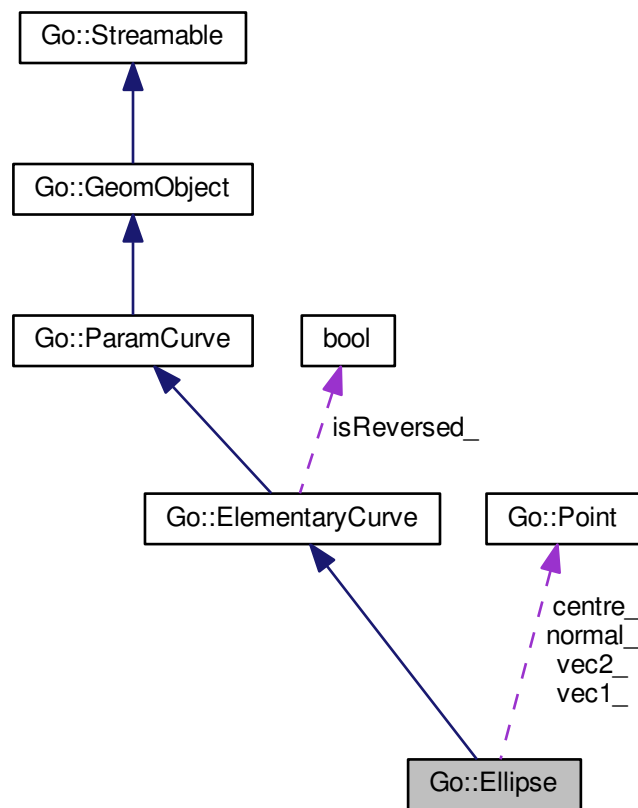
Class that represents an ellipse. It is a subclass of [ElementaryCurve](#) and thus has a parametrization.

```
#include <Ellipse.h>
```

Inheritance diagram for Go::Ellipse:



Collaboration diagram for Go::Ellipse:



## Public Member Functions

- [Ellipse](#) ()
- [Ellipse](#) ([Point](#) centre, [Point](#) direction, [Point](#) normal, [double](#) r1, [double](#) r2, [bool](#) isReversed=false)
- virtual [~Ellipse](#) ()  
*virtual destructor - ensures safe inheritance*
- virtual void [read](#) (std::istream &is)
- virtual void [write](#) (std::ostream &os) **const**
- virtual [BoundingBox](#) boundingBox () **const**  
*Return the object's bounding box.*
- virtual int [dimension](#) () **const**  
*Return the dimension of the space in which the object lies (usually 2 or 3)*
- virtual [ClassType](#) instanceType () **const**  
*Return the class type identifier of a given, derived instance of [GeomObject](#).*
- virtual [Ellipse](#) \* [clone](#) () **const**
- virtual void [point](#) ([Point](#) &pt, [double](#) tpar) **const**
- virtual void [point](#) (std::vector< [Point](#) > &pts, [double](#) tpar, int derivs, [bool](#) from\_right=true) **const**
- virtual [double](#) startparam () **const**
- virtual [double](#) endparam () **const**



- virtual void [setParameterInterval](#) (double t1, double t2)  
*Limit the curve by limiting the parameter domain.*
- virtual [SplineCurve](#) \* [geometryCurve](#) ()
- virtual [SplineCurve](#) \* [createSplineCurve](#) () const  
*Create the spline representation of this curve.*
- virtual bool [isDegenerate](#) (double degenerate\_epsilon)
- virtual [Ellipse](#) \* [subCurve](#) (double from\_par, double to\_par, double fuzzy=DEFAULT\_PARAMETER\_EPSILON) const
- virtual [DirectionCone](#) [directionCone](#) () const
- virtual void [appendCurve](#) ([ParamCurve](#) \*cv, bool reparam=true)
- virtual void [appendCurve](#) ([ParamCurve](#) \*cv, int continuity, double &dist, bool reparam=true)
- virtual void [closestPoint](#) (const [Point](#) &pt, double tmin, double tmax, double &clo\_t, [Point](#) &clo\_pt, double &clo\_dist, double const \*seed=0) const
- virtual double [length](#) (double tol)
- [Point](#) [getCentre](#) () const
- [Point](#) [getNormal](#) () const
- [Point](#) [getXAxis](#) () const
- double [getRadius1](#) () const
- double [getRadius2](#) () const
- virtual void [setParamBounds](#) (double startpar, double endpar)
- virtual void [translateCurve](#) (const [Point](#) &dir)
- virtual bool [isInPlane](#) (const [Point](#) &norm, double eps, [Point](#) &pos) const  
*Check if the ellipse lies in a plane with a given normal.*
- bool [isClosed](#) () const
- virtual void [swapParameters2D](#) ()

### Static Public Member Functions

- static [ClassType](#) [classType](#) ()

### Protected Member Functions

- void [setSpanningVectors](#) ()

### Protected Attributes

- [Point](#) [centre\\_](#)
- [Point](#) [vec1\\_](#)
- [Point](#) [vec2\\_](#)
- [Point](#) [normal\\_](#)
- double [r1\\_](#)
- double [r2\\_](#)
- double [startparam\\_](#)
- double [endparam\\_](#)

### 29.116.1 Detailed Description

Class that represents an ellipse. It is a subclass of [ElementaryCurve](#) and thus has a parametrization.

An ellipse has a natural parametrization in terms of its values:  $f(t) = C + (R_1 * \cos(t))*x + (R_2 * \sin(t)) * y$ . This parametrization is bounded:  $0 \leq t \leq 360$  (degrees).

Definition at line 57 of file [Ellipse.h](#).

## 29.116.2 Constructor & Destructor Documentation

### 29.116.2.1 `Go::Ellipse::Ellipse ( ) [inline]`

Default constructor. Constructs an uninitialized [Ellipse](#) which can only be assigned to or read into.

Definition at line 62 of file `Ellipse.h`.

### 29.116.2.2 `Go::Ellipse::Ellipse ( Point centre, Point direction, Point normal, double r1, double r2, bool isReversed = false )`

Constructor. Input is point, axis direction and lengths of the two semi-axis.

### 29.116.2.3 `virtual Go::Ellipse::~~Ellipse ( ) [virtual]`

virtual destructor - ensures safe inheritance

## 29.116.3 Member Function Documentation

### 29.116.3.1 `virtual void Go::Ellipse::appendCurve ( ParamCurve * cv, bool reparam = true ) [virtual]`

append a curve to this curve, with eventual reparametrization NB: This virtual member function currently only works for `SplineCurves` and `CurveOnSurfaces`. Moreover, 'this' curve and the 'cv' curve must be of the same type.

#### Parameters

|                |                                                          |
|----------------|----------------------------------------------------------|
| <i>cv</i>      | the curve to append to 'this' curve.                     |
| <i>reparam</i> | specify whether or not there should be reparametrization |

Implements [Go::ParamCurve](#).

### 29.116.3.2 `virtual void Go::Ellipse::appendCurve ( ParamCurve * cv, int continuity, double & dist, bool reparam = true ) [virtual]`

append a curve to this curve, with eventual reparametrization

#### Parameters

|                   |                                                                                                       |
|-------------------|-------------------------------------------------------------------------------------------------------|
| <i>cv</i>         | the curve to append to 'this' curve.                                                                  |
| <i>continuity</i> | the required continuity at the transition. Can be $G^{(-1)}$ and upwards.                             |
| <i>dist</i>       | a measure of the local distortion around the transition in order to achieve the specified continuity. |
| <i>reparam</i>    | specify whether or not there should be reparametrization                                              |

Implements [Go::ParamCurve](#).

29.116.3.3 `virtual BoundingBox Go::Ellipse::boundingBox ( ) const [virtual]`

Return the object's bounding box.

Implements [Go::GeomObject](#).

29.116.3.4 `static ClassType Go::Ellipse::classType ( ) [static]`

29.116.3.5 `virtual Ellipse* Go::Ellipse::clone ( ) const [virtual]`

The clone-function is herited from [GeomObject](#), but overridden here to get a covariant return type (for those compilers that allow this).

Implements [Go::ElementaryCurve](#).

29.116.3.6 `virtual void Go::Ellipse::closestPoint ( const Point & pt, double tmin, double tmax, double & clo_t, Point & clo_pt, double & clo_dist, double const * seed = 0 ) const [virtual]`

Compute the closest point from an interval of this curve to a specified point.

Parameters

|                 |                                                                                                                                          |
|-----------------|------------------------------------------------------------------------------------------------------------------------------------------|
| <i>pt</i>       | point we want to find the closest point to                                                                                               |
| <i>tmin</i>     | start parameter of search interval                                                                                                       |
| <i>tmax</i>     | end parameter of search interval                                                                                                         |
| <i>clo_t</i>    | upon function return, 'clo_t' will contain the parameter value of the closest point found.                                               |
| <i>clo_pt</i>   | upon function return, 'clo_pt' will contain the position of the closest point found.                                                     |
| <i>clo_dist</i> | upon function return, 'clo_dist' will contain the distance between 'pt' and the closest point found.                                     |
| <i>seed</i>     | pointer to initial guess value, provided by the user (can be 0, for which the algorithm will determine a (hopefully) reasonable choice). |

Implements [Go::ParamCurve](#).

29.116.3.7 `virtual SplineCurve* Go::Ellipse::createSplineCurve ( ) const [virtual]`

Create the spline representation of this curve.

Implements [Go::ElementaryCurve](#).

29.116.3.8 `virtual int Go::Ellipse::dimension ( ) const [virtual]`

Return the dimension of the space in which the object lies (usually 2 or 3)

Implements [Go::GeomObject](#).

29.116.3.9 `virtual DirectionCone Go::Ellipse::directionCone ( ) const [virtual]`

Creates a [DirectionCone](#) which covers all tangent directions of this curve.

**Returns**

the smallest [DirectionCone](#) containing all tangent directions of this curve.

Implements [Go::ParamCurve](#).

29.116.3.10 `virtual double Go::Ellipse::endparam ( ) const [virtual]`

Query the end parameter of the curve

**Returns**

the curve's end parameter

Implements [Go::ParamCurve](#).

29.116.3.11 `virtual SplineCurve* Go::Ellipse::geometryCurve ( ) [virtual]`

If the definition of this [ParamCurve](#) contains a [SplineCurve](#) describing its spatial shape, then this function will return a pointer to this [SplineCurve](#). Otherwise it will return a null pointer. The returned curve is NEWed, so the user is responsible for deleting it. This function may have side-effects.

**Returns**

a pointer to a [SplineCurve](#) representation of the [ParamCurve](#), if it exists. Null pointer otherwise.

Implements [Go::ParamCurve](#).

29.116.3.12 `Point Go::Ellipse::getCentre ( ) const [inline]`

Definition at line 140 of file Ellipse.h.

29.116.3.13 `Point Go::Ellipse::getNormal ( ) const [inline]`

Definition at line 145 of file Ellipse.h.

29.116.3.14 `double Go::Ellipse::getRadius1 ( ) const [inline]`

Definition at line 155 of file Ellipse.h.

29.116.3.15 **double** Go::Ellipse::getRadius2 ( ) const [inline]

Definition at line 160 of file Ellipse.h.

29.116.3.16 **Point** Go::Ellipse::getXAxis ( ) const [inline]

Definition at line 150 of file Ellipse.h.

29.116.3.17 **virtual ClassType** Go::Ellipse::instanceType ( ) const [virtual]

Return the class type identifier of a given, derived instance of [GeomObject](#).

Implements [Go::GeomObject](#).

29.116.3.18 **bool** Go::Ellipse::isClosed ( ) const

29.116.3.19 **virtual bool** Go::Ellipse::isDegenerate ( **double** *degenerate\_epsilon* ) [virtual]

Query whether the curve is degenerate (collapsed into a single point).

#### Parameters

|                           |                                                                                                                                                   |
|---------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>degenerate_epsilon</i> | the tolerance used in determine whether the curve is degenerate. A curve is considered degenerate if its total length is shorter than this value. |
|---------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------|

#### Returns

`true` if the curve is degenerate, `false` otherwise.

Implements [Go::ParamCurve](#).

29.116.3.20 **virtual bool** Go::Ellipse::isInPlane ( **const Point &** *norm*, **double** *eps*, **Point &** *pos* ) const [virtual]

Check if the ellipse lies in a plane with a given normal.

Reimplemented from [Go::ParamCurve](#).

29.116.3.21 **virtual double** Go::Ellipse::length ( **double** *tol* ) [virtual]

Compute the total length of this curve up to some tolerance

#### Parameters

|            |                                                                                                                                 |
|------------|---------------------------------------------------------------------------------------------------------------------------------|
| <i>tol</i> | the relative tolerance when approximating the length, i.e. stop iteration when error becomes smaller than $tol/(curve\ length)$ |
|------------|---------------------------------------------------------------------------------------------------------------------------------|

**Returns**

the length calculated

Implements [Go::ParamCurve](#).

29.116.3.22 `virtual void Go::Ellipse::point ( Point & pt, double tpar ) const` [virtual]

Evaluate the curve's position at a given parameter

**Parameters**

|             |                                                                      |
|-------------|----------------------------------------------------------------------|
| <i>pt</i>   | the evaluated position will be written to this <a href="#">Point</a> |
| <i>tpar</i> | the parameter for which we wish to evaluate the curve                |

Implements [Go::ParamCurve](#).

29.116.3.23 `virtual void Go::Ellipse::point ( std::vector< Point > & pts, double tpar, int derivs, bool from_right = true ) const` [virtual]

Evaluate the curve's position and a certain number of derivatives at a given parameter.

**Parameters**

|                   |                                                                                                                                                                                                                                                                                                |
|-------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>pts</i>        | the evaluated position and derivatives (tangent, curvature vector, etc.) will be written to this vector. The first entry will be the position, the second entry will be the first derivative, etc. The size of this vector must be set to ' <i>derivs</i> + 1' prior to calling this function. |
| <i>tpar</i>       | the parameter for which we want to evaluate the curve                                                                                                                                                                                                                                          |
| <i>derivs</i>     | the number of derivatives we want to have calculated                                                                                                                                                                                                                                           |
| <i>from_right</i> | specify whether we should calculate derivatives 'from the right' or 'from the left' (default is from the right). This matters only when the curve presents discontinuities in its derivatives.                                                                                                 |

Implements [Go::ParamCurve](#).

29.116.3.24 `virtual void Go::Ellipse::read ( std::istream & is )` [virtual]

Read object from stream

**Parameters**

|           |                                  |
|-----------|----------------------------------|
| <i>is</i> | stream from which object is read |
|-----------|----------------------------------|

Implements [Go::Streamable](#).

29.116.3.25 `virtual void Go::Ellipse::setParamBounds ( double startpar, double endpar )` [virtual]

Set bounds for the parametrization of the [Ellipse](#).

## Parameters

|                 |                 |
|-----------------|-----------------|
| <i>startpar</i> | start parameter |
| <i>endpar</i>   | end parameter   |

Implements [Go::ElementaryCurve](#).

29.116.3.26 `virtual void Go::Ellipse::setParameterInterval ( double t1, double t2 ) [virtual]`

Limit the curve by limiting the parameter domain.

Implements [Go::ParamCurve](#).

29.116.3.27 `void Go::Ellipse::setSpanningVectors ( ) [protected]`

29.116.3.28 `virtual double Go::Ellipse::startparam ( ) const [virtual]`

Query the start parameter of the curve

## Returns

the curve's start parameter

Implements [Go::ParamCurve](#).

29.116.3.29 `virtual Ellipse* Go::Ellipse::subCurve ( double from_par, double to_par, double fuzzy = DEFAULT_PARAMETER_EPSILON ) const [virtual]`

Returns a curve which is a part of this curve. The result is NEWed, so the user is responsible for deleting it. NB: It is not guaranteed that the [ParamCurve](#) that is returned is of the same type as the curve itself. Thus, the returned curve might be a [SplineCurve](#).

## Parameters

|                 |                                                                                                                                                                                                   |
|-----------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>from_par</i> | start value of parameter interval that will define the subcurve                                                                                                                                   |
| <i>to_par</i>   | end value of parameter interval that will define the subcurve                                                                                                                                     |
| <i>fuzzy</i>    | since subCurve works on those curves who are spline-based, this tolerance defines how close the start and end parameter must be to an existing knot in order to be considered <i>on</i> the knot. |

## Returns

a pointer to a new subcurve which represents the part of the curve between 'from\_par' and 'to\_par'. It will be spline-based and have a k-regular knotvector. The user is responsible for deleting this subcurve when it is no longer needed.

Implements [Go::ElementaryCurve](#).

29.116.3.30 `virtual void Go::Ellipse::swapParameters2D ( ) [virtual]`

If the curve is 2 dimensional, x and y coordinates will be swapped. Used when curve is a parameter curve.

Implements [Go::ElementaryCurve](#).

29.116.3.31 `virtual void Go::Ellipse::translateCurve ( const Point & dir ) [virtual]`

Implements [Go::ElementaryCurve](#).

29.116.3.32 `virtual void Go::Ellipse::write ( std::ostream & os ) const [virtual]`

Write object to stream

Parameters

|                 |                                   |
|-----------------|-----------------------------------|
| <code>os</code> | stream to which object is written |
|-----------------|-----------------------------------|

Implements [Go::Streamable](#).

## 29.116.4 Member Data Documentation

29.116.4.1 `Point Go::Ellipse::centre_ [protected]`

Definition at line 185 of file Ellipse.h.

29.116.4.2 `double Go::Ellipse::endparam_ [protected]`

Definition at line 194 of file Ellipse.h.

29.116.4.3 `Point Go::Ellipse::normal_ [protected]`

Definition at line 188 of file Ellipse.h.

29.116.4.4 `double Go::Ellipse::r1_ [protected]`

Definition at line 190 of file Ellipse.h.

29.116.4.5 `double Go::Ellipse::r2_ [protected]`

Definition at line 191 of file Ellipse.h.



29.116.4.6 **double** Go::Ellipse::startparam\_ [protected]

Definition at line 193 of file Ellipse.h.

29.116.4.7 **Point** Go::Ellipse::vec1\_ [protected]

Definition at line 186 of file Ellipse.h.

29.116.4.8 **Point** Go::Ellipse::vec2\_ [protected]

Definition at line 187 of file Ellipse.h.

The documentation for this class was generated from the following file:

- [gotools-core/include/GoTools/geometry/Ellipse.h](#)

## 29.117 Go::EntityList Struct Reference

The entity number of all supported IGES entites.

```
#include <IGESconverter.h>
```

### Public Member Functions

- [EntityList](#) ()
- [bool validEntity](#) (int ent)

### 29.117.1 Detailed Description

The entity number of all supported IGES entites.

Definition at line 118 of file IGESconverter.h.

## 29.117.2 Constructor & Destructor Documentation

### 29.117.2.1 `Go::EntityList::EntityList( )` `[inline]`

24 entities currently supported (072009). Note that for entities with multiple forms some forms may be missing.  
Circular arc.

Composite curve.

Conic arc.

Linear path (form 12).

[Plane](#).

[Line](#).

[Point](#).

Ruled surface.

Surface of revolution.

Tabulated cylinder.

Direction.

Transformation matrix.

[Rational](#) B-spline curve.

[Rational](#) B-spline surface.

Boundary.

Curve on a parametric surface.

Bounded surface.

Trimmed (parametric) surface.

[Plane](#) surface.

Color definition.

Group without back pointers assoc. (form 7).

[Vertex](#) List.

Edge List.

[Loop](#).

Face.

Definition at line 122 of file IGESconverter.h.

### 29.117.3 Member Function Documentation

#### 29.117.3.1 `bool Go::EntityList::validEntity ( int ent )`

The documentation for this struct was generated from the following file:

- [igeslib/include/GoTools/igeslib/IGESconverter.h](#)

## 29.118 Go::Streamable::EofException Class Reference

```
#include <Streamable.h>
```

### 29.118.1 Detailed Description

Definition at line 69 of file Streamable.h.

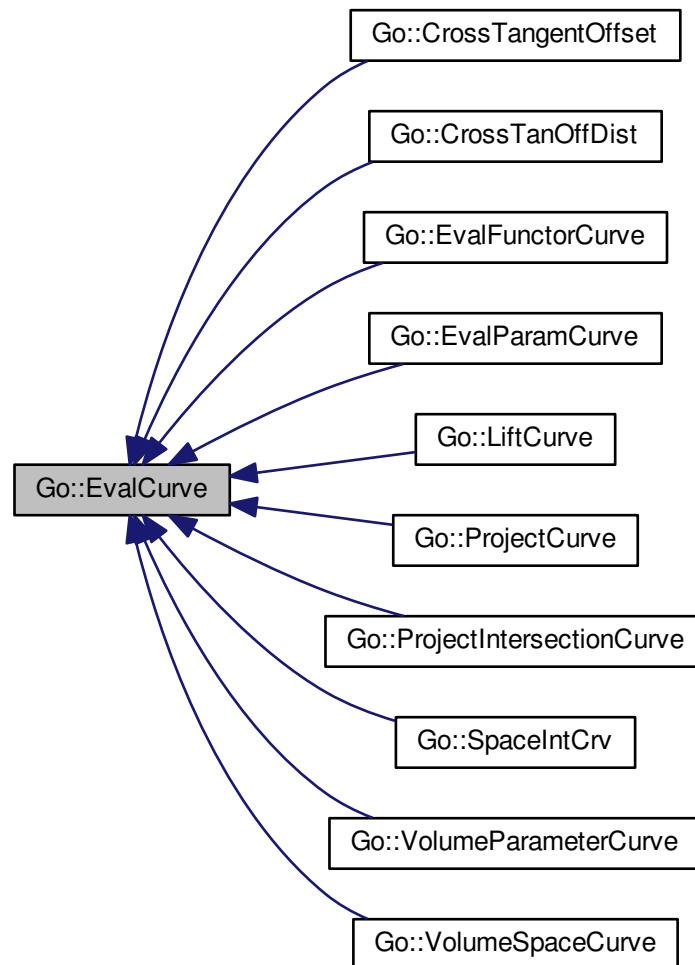
The documentation for this class was generated from the following file:

- [gotools-core/include/GoTools/geometry/Streamable.h](#)

## 29.119 Go::EvalCurve Class Reference

```
#include <EvalCurve.h>
```

Inheritance diagram for Go::EvalCurve:



## Public Member Functions

- virtual `~EvalCurve ()`  
*virtual destructor ensures save inheritance*
- virtual `Point eval (double t) const =0`
- virtual `void eval (double t, int n, Point der[]) const =0`
- virtual `double start () const =0`
- virtual `double end () const =0`
- virtual `int dim () const =0`
- virtual `bool approximationOK (double par, Point approxpos, double tol1, double tol2) const =0`
- virtual `void write (std::ostream &out) const`

### 29.119.1 Detailed Description

This is the abstract base class of a curve type that can be evaluated. Representing the exact geometry, typically used when iteratively approximating the curve.

Definition at line 55 of file EvalCurve.h.

### 29.119.2 Constructor & Destructor Documentation

29.119.2.1 virtual Go::EvalCurve::~EvalCurve ( ) [virtual]

virtual destructor ensures save inheritance

### 29.119.3 Member Function Documentation

29.119.3.1 virtual bool Go::EvalCurve::approximationOK ( double *par*, Point *approxpos*, double *tol1*, double *tol2* ) const [pure virtual]

Check if the curve, evaluated at a given parameter, approximates a given position within a given tolerance.

#### Parameters

|                  |                                                                                          |
|------------------|------------------------------------------------------------------------------------------|
| <i>par</i>       | the parameter at which to check the curve                                                |
| <i>approxpos</i> | the position we want to check whether or not the curve approximates for parameter 'par'. |
| <i>tol1</i>      | approximation tolerance.                                                                 |
| <i>tol2</i>      | another approximation tolerance (its use is defined by some of the derived classes.      |

#### Returns

'true' if the curve approximates the point at the parameter, 'false' otherwise.

Implemented in [Go::CrossTanOffDist](#), [Go::VolumeParameterCurve](#), [Go::ProjectCurve](#), [Go::CrossTangentOffset](#), [Go::ProjectIntersectionCurve](#), [Go::VolumeSpaceCurve](#), [Go::SpaceIntCrv](#), [Go::EvalFunctorCurve](#), [Go::LiftCurve](#), and [Go::EvalParamCurve](#).

29.119.3.2 virtual int Go::EvalCurve::dim ( ) const [pure virtual]

Get the dimension of the space in which the curve lies.

#### Returns

the space dimension of the curve.

Implemented in [Go::CrossTanOffDist](#), [Go::VolumeParameterCurve](#), [Go::ProjectCurve](#), [Go::ProjectIntersectionCurve](#), [Go::SpaceIntCrv](#), [Go::VolumeSpaceCurve](#), [Go::CrossTangentOffset](#), [Go::EvalFunctorCurve](#), [Go::LiftCurve](#), and [Go::EvalParamCurve](#).

29.119.3.3 `virtual double Go::EvalCurve::end ( ) const [pure virtual]`

Get the end parameter of the curve.

#### Returns

the end parameter of the curve.

Implemented in [Go::CrossTanOffDist](#), [Go::VolumeParameterCurve](#), [Go::ProjectCurve](#), [Go::ProjectIntersectionCurve](#), [Go::SpaceIntCrv](#), [Go::VolumeSpaceCurve](#), [Go::CrossTangentOffset](#), [Go::EvalFunctorCurve](#), [Go::LiftCurve](#), and [Go::EvalParamCurve](#).

29.119.3.4 `virtual Point Go::EvalCurve::eval ( double t ) const [pure virtual]`

Evaluate a point on the curve for a given parameter

#### Parameters

|          |                                                |
|----------|------------------------------------------------|
| <i>t</i> | the parameter for which to evaluate the curve. |
|----------|------------------------------------------------|

#### Returns

the evaluated point

Implemented in [Go::ProjectCurve](#), [Go::CrossTanOffDist](#), [Go::CrossTangentOffset](#), [Go::ProjectIntersectionCurve](#), [Go::SpaceIntCrv](#), [Go::VolumeParameterCurve](#), [Go::LiftCurve](#), [Go::VolumeSpaceCurve](#), [Go::EvalParamCurve](#), and [Go::EvalFunctorCurve](#).

29.119.3.5 `virtual void Go::EvalCurve::eval ( double t, int n, Point der[] ) const [pure virtual]`

Evaluate a point and a certain number of derivatives on the curve for a given parameter.

#### Parameters

|          |                                                |
|----------|------------------------------------------------|
| <i>t</i> | the parameter for which to evaluate the curve. |
| <i>n</i> | the number of derivatives (0 or more)          |

#### Return values

|            |                                                                                                                                                                                                                                                                                                                                                                      |
|------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>der</i> | pointer to an array of Points where the result will be written. The position will be stored first, then the first derivative (tangent), then the second, etc.. <b>NB:</b> For most (all) derived classes of 'EvalCurve', the implementation actually only supports the computation of one derivative, i.e. if $n > 1$ , only one derivative will be computed anyway. |
|------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

Implemented in [Go::CrossTanOffDist](#), [Go::ProjectCurve](#), [Go::VolumeParameterCurve](#), [Go::ProjectIntersectionCurve](#), [Go::SpaceIntCrv](#), [Go::CrossTangentOffset](#), [Go::VolumeSpaceCurve](#), [Go::EvalFunctorCurve](#), [Go::LiftCurve](#), and [Go::EvalParamCurve](#).

29.119.3.6 virtual double Go::EvalCurve::start ( ) const [pure virtual]

Get the start parameter of the curve.

#### Returns

the start parameter of the curve.

Implemented in [Go::CrossTanOffDist](#), [Go::ProjectCurve](#), [Go::VolumeParameterCurve](#), [Go::ProjectIntersectionCurve](#), [Go::SpaceIntCrv](#), [Go::VolumeSpaceCurve](#), [Go::CrossTangentOffset](#), [Go::EvalFuncorCurve](#), [Go::LiftCurve](#), and [Go::EvalParamCurve](#).

29.119.3.7 virtual void Go::EvalCurve::write ( std::ostream & out ) const [virtual]

Reimplemented in [Go::EvalParamCurve](#).

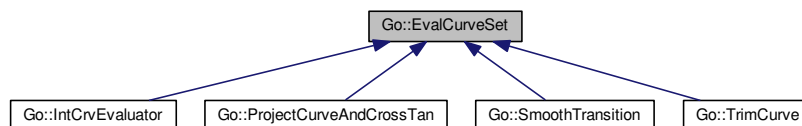
The documentation for this class was generated from the following file:

- [gotools-core/include/GoTools/creators/EvalCurve.h](#)

## 29.120 Go::EvalCurveSet Class Reference

```
#include <EvalCurveSet.h>
```

Inheritance diagram for Go::EvalCurveSet:



### Public Member Functions

- virtual [~EvalCurveSet](#) ()  
*virtual destructor ensures safe inheritance*
- virtual std::vector< [Point](#) > [eval](#) (double t)=0
- virtual void [eval](#) (double t, int n, std::vector< std::vector< [Point](#) > > &der)=0
- virtual double [start](#) ()=0
- virtual double [end](#) ()=0
- virtual int [dim](#) ()=0
- virtual bool [approximationOK](#) (double par, const std::vector< [Point](#) > &approxpos, double tol1, double tol2)=0
- virtual int [nmbCvs](#) ()=0
- virtual void [resetErr](#) ()  
*Reset intermediate error.*

### 29.120.1 Detailed Description

This abstract class provides an interface to a set of curves that can be evaluated. Representing the actual geometry, typically used when iteratively approximating the set of curves on the same basis.

Definition at line 55 of file EvalCurveSet.h.

### 29.120.2 Constructor & Destructor Documentation

29.120.2.1 `virtual Go::EvalCurveSet::~EvalCurveSet ( ) [inline],[virtual]`

virtual destructor ensures safe inheritance

Definition at line 59 of file EvalCurveSet.h.

### 29.120.3 Member Function Documentation

29.120.3.1 `virtual bool Go::EvalCurveSet::approximationOK ( double par, const std::vector< Point > & approxpos, double tol1, double tol2 ) [pure virtual]`

Whether the approximation is within tolerances in input parameter.

#### Parameters

|                  |                                                                                                                |
|------------------|----------------------------------------------------------------------------------------------------------------|
| <i>par</i>       | parameter in which to evaluate.                                                                                |
| <i>approxpos</i> | whether the input points are within tolerance from the evaluated points (as given by <a href="#">eval()</a> ). |
| <i>tol1</i>      | tolerance used to decide approximation accuracy.                                                               |
| <i>tol2</i>      | tolerance used to decide approximation accuracy.                                                               |

#### Returns

whether the approximation is within tolerances in input parameter.

Implemented in [Go::TrimCurve](#), [Go::IntCrvEvaluator](#), and [Go::SmoothTransition](#).

29.120.3.2 `virtual int Go::EvalCurveSet::dim ( ) [pure virtual]`

The geometric dimension of the spline curves.

#### Returns

geometric dimension of the space.

Implemented in [Go::ProjectCurveAndCrossTan](#), [Go::SmoothTransition](#), [Go::TrimCurve](#), and [Go::IntCrvEvaluator](#).



29.120.3.3 virtual double Go::EvalCurveSet::end ( ) [pure virtual]

End parameter of domain.

#### Returns

end parameter of the spline space.

Implemented in [Go::ProjectCurveAndCrossTan](#), [Go::SmoothTransition](#), [Go::TrimCurve](#), and [Go::IntCrvEvaluator](#).

29.120.3.4 virtual std::vector<Point> Go::EvalCurveSet::eval ( double *t* ) [pure virtual]

Evaluate the curves.

#### Parameters

|          |                                 |
|----------|---------------------------------|
| <i>t</i> | parameter in which to evaluate. |
|----------|---------------------------------|

#### Returns

the evaluated points for the curve set.

Implemented in [Go::ProjectCurveAndCrossTan](#), [Go::SmoothTransition](#), [Go::TrimCurve](#), and [Go::IntCrvEvaluator](#).

29.120.3.5 virtual void Go::EvalCurveSet::eval ( double *t*, int *n*, std::vector< std::vector< Point > > & *der* ) [pure virtual]

Evaluate the curve derivatives.

#### Parameters

|            |                                                                   |
|------------|-------------------------------------------------------------------|
| <i>t</i>   | parameter in which to evaluate.                                   |
| <i>n</i>   | number of derivatives to compute.                                 |
| <i>der</i> | the evaluated points up to the n'th derivative for the curve set. |

Implemented in [Go::SmoothTransition](#), [Go::TrimCurve](#), and [Go::IntCrvEvaluator](#).

29.120.3.6 virtual int Go::EvalCurveSet::nmbCvs ( ) [pure virtual]

The number of curves in the curve set.

#### Returns

the number of curves in the curve set.

Implemented in [Go::ProjectCurveAndCrossTan](#), [Go::TrimCurve](#), [Go::IntCrvEvaluator](#), and [Go::SmoothTransition](#).

29.120.3.7 `virtual void Go::EvalCurveSet::resetErr ( ) [inline],[virtual]`

Reset intermediate error.

Reimplemented in [Go::IntCrvEvaluator](#).

Definition at line 100 of file EvalCurveSet.h.

29.120.3.8 `virtual double Go::EvalCurveSet::start ( ) [pure virtual]`

Start parameter of domain.

Returns

start parameter of the spline space.

Implemented in [Go::ProjectCurveAndCrossTan](#), [Go::SmoothTransition](#), [Go::TrimCurve](#), and [Go::IntCrvEvaluator](#).

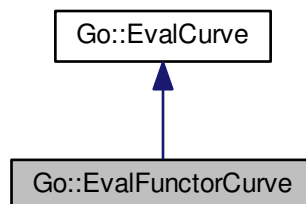
The documentation for this class was generated from the following file:

- [gotools-core/include/GoTools/creators/EvalCurveSet.h](#)

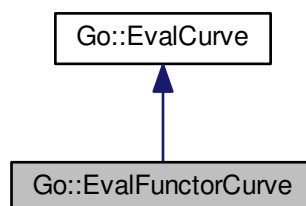
## 29.121 Go::EvalFunctorCurve Class Reference

```
#include <EvalFunctorCurve.h>
```

Inheritance diagram for Go::EvalFunctorCurve:



Collaboration diagram for Go::EvalFunctorCurve:



## Public Member Functions

- [EvalFunctorCurve](#) ([BdCondFunctor](#) \*fbd, [shared\\_ptr](#)< [SplineCurve](#) > geo\_curve, int dimension)
- virtual [~EvalFunctorCurve](#) ()
- virtual [Point](#) eval (double t) const
- virtual void eval (double t, int n, [Point](#) der[]) const
- virtual double start () const
- virtual double end () const
- virtual int dim () const
- virtual [bool](#) approximationOK (double par, [Point](#) approxpos, double tol1, double tol2) const

### 29.121.1 Detailed Description

Definition at line 54 of file EvalFunctorCurve.h.

### 29.121.2 Constructor & Destructor Documentation

29.121.2.1 [Go::EvalFunctorCurve::EvalFunctorCurve](#) ( [BdCondFunctor](#) \* fbd, [shared\\_ptr](#)< [SplineCurve](#) > geo\_curve, int dimension )

29.121.2.2 virtual [Go::EvalFunctorCurve::~EvalFunctorCurve](#) ( ) [virtual]

### 29.121.3 Member Function Documentation

29.121.3.1 virtual [bool](#) [Go::EvalFunctorCurve::approximationOK](#) ( double par, [Point](#) approxpos, double tol1, double tol2 ) const [virtual]

Check if the curve, evaluated at a given parameter, approximates a given position within a given tolerance.

#### Parameters

|                  |                                                                                          |
|------------------|------------------------------------------------------------------------------------------|
| <i>par</i>       | the parameter at which to check the curve                                                |
| <i>approxpos</i> | the position we want to check whether or not the curve approximates for parameter 'par'. |
| <i>tol1</i>      | approximation tolerance.                                                                 |
| <i>tol2</i>      | another approximation tolerance (its use is defined by some of the derived classes.      |

#### Returns

'true' if the curve approximates the point at the parameter, 'false' otherwise.

Implements [Go::EvalCurve](#).

29.121.3.2 virtual int [Go::EvalFunctorCurve::dim](#) ( ) const [virtual]

Get the dimension of the space in which the curve lies.

#### Returns

the space dimension of the curve.

Implements [Go::EvalCurve](#).

29.121.3.3 `virtual double Go::EvalFunctorCurve::end ( ) const` [virtual]

Get the end parameter of the curve.

#### Returns

the end parameter of the curve.

Implements [Go::EvalCurve](#).

29.121.3.4 `virtual Point Go::EvalFunctorCurve::eval ( double t ) const` [virtual]

Evaluate a point on the curve for a given parameter

#### Parameters

|          |                                                |
|----------|------------------------------------------------|
| <i>t</i> | the parameter for which to evaluate the curve. |
|----------|------------------------------------------------|

#### Returns

the evaluated point

Implements [Go::EvalCurve](#).

29.121.3.5 `virtual void Go::EvalFunctorCurve::eval ( double t, int n, Point der[] ) const` [virtual]

Evaluate a point and a certain number of derivatives on the curve for a given parameter.

#### Parameters

|          |                                                |
|----------|------------------------------------------------|
| <i>t</i> | the parameter for which to evaluate the curve. |
| <i>n</i> | the number of derivatives (0 or more)          |

#### Return values

|            |                                                                                                                                                                                                                                                                                                                                                                                        |
|------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>der</i> | pointer to an array of Points where the result will be written. The position will be stored first, then the first derivative (tangent), then the second, etc.. <b>NB:</b> For most (all) derived classes of ' <a href="#">EvalCurve</a> ', the implementation actually only supports the computation of one derivative, i.e. if $n > 1$ , only one derivative will be computed anyway. |
|------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

Implements [Go::EvalCurve](#).

29.121.3.6 `virtual double Go::EvalFunctorCurve::start ( ) const` [virtual]

Get the start parameter of the curve.

**Returns**

the start parameter of the curve.

Implements [Go::EvalCurve](#).

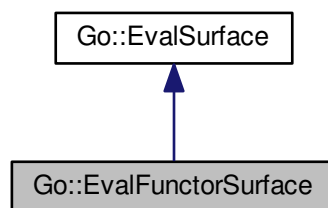
The documentation for this class was generated from the following file:

- isogeometric\_model/include/GoTools/isogeometric\_model/[EvalFunctorCurve.h](#)

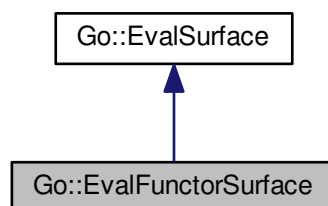
## 29.122 Go::EvalFunctorSurface Class Reference

```
#include <EvalFunctorSurface.h>
```

Inheritance diagram for Go::EvalFunctorSurface:



Collaboration diagram for Go::EvalFunctorSurface:



## Public Member Functions

- [EvalFunctorSurface](#) ([BdCondFunctor](#) \*fbd, [shared\\_ptr](#)< [SplineSurface](#) > geo\_surface, int dimension)
- virtual [~EvalFunctorSurface](#) ()
- virtual [Point](#) eval (double u, double v) const
- virtual void eval (double u, double v, int n, [Point](#) der[]) const
- virtual double start\_u () const
- virtual double start\_v () const
- virtual double end\_u () const
- virtual double end\_v () const
- virtual int dim () const
- virtual bool approximationOK (double par\_u, double par\_v, [Point](#) approxpos, double tol1, double tol2) const

### 29.122.1 Detailed Description

Definition at line 54 of file EvalFunctorSurface.h.

### 29.122.2 Constructor & Destructor Documentation

29.122.2.1 `Go::EvalFunctorSurface::EvalFunctorSurface ( BdCondFunctor * fbd, shared\_ptr< SplineSurface > geo_surface, int dimension )`

29.122.2.2 `virtual Go::EvalFunctorSurface::~~EvalFunctorSurface ( )` [virtual]

### 29.122.3 Member Function Documentation

29.122.3.1 `virtual bool Go::EvalFunctorSurface::approximationOK ( double par_u, double par_v, Point approxpos, double tol1, double tol2 ) const` [virtual]

Check if the surface, evaluated at a given parameter, approximates a given position within a given tolerance.

#### Parameters

|                  |                                                                                            |
|------------------|--------------------------------------------------------------------------------------------|
| <i>par</i>       | the parameter at which to check the surface                                                |
| <i>approxpos</i> | the position we want to check whether or not the surface approximates for parameter 'par'. |
| <i>tol1</i>      | approximation tolerance.                                                                   |
| <i>tol2</i>      | another approximation tolerance (its use is defined by some of the derived classes.        |

#### Returns

'true' if the surface approximates the point at the parameter, 'false' otherwise.

Implements [Go::EvalSurface](#).

29.122.3.2 `virtual int Go::EvalFunctorSurface::dim ( ) const` [virtual]

Get the dimension of the space in which the surface lies.

**Returns**

the space dimension of the surface.

Implements [Go::EvalSurface](#).

**29.122.3.3** `virtual double Go::EvalFunctorSurface::end_u( ) const [virtual]`

Get the end parameter of the surface.

**Returns**

the end parameter of the surface.

Implements [Go::EvalSurface](#).

**29.122.3.4** `virtual double Go::EvalFunctorSurface::end_v( ) const [virtual]`

Implements [Go::EvalSurface](#).

**29.122.3.5** `virtual Point Go::EvalFunctorSurface::eval( double u, double v ) const [virtual]`

Evaluate a point on the surface for a given parameter

**Parameters**

|          |                                                  |
|----------|--------------------------------------------------|
| <i>t</i> | the parameter for which to evaluate the surface. |
|----------|--------------------------------------------------|

**Returns**

the evaluated point

Implements [Go::EvalSurface](#).

**29.122.3.6** `virtual void Go::EvalFunctorSurface::eval( double u, double v, int n, Point der[] ) const [virtual]`

Evaluate a point and a certain number of derivatives on the surface for a given parameter.

**Parameters**

|          |                                                  |
|----------|--------------------------------------------------|
| <i>t</i> | the parameter for which to evaluate the surface. |
| <i>n</i> | the number of derivatives (0 or more)            |

## Return values

|            |                                                                                                                                                                                                                                                                                                                                                                        |
|------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>der</i> | pointer to an array of Points where the result will be written. The position will be stored first, then the first derivative (tangent), then the second, etc.. <b>NB:</b> For most (all) derived classes of 'EvalSurface', the implementation actually only supports the computation of one derivative, i.e. if $n > 1$ , only one derivative will be computed anyway. |
|------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

Implements [Go::EvalSurface](#).

29.122.3.7 `virtual double Go::EvalFunctorSurface::start_u ( ) const [virtual]`

Get the start parameter of the surface.

## Returns

the start parameter of the surface.

Implements [Go::EvalSurface](#).

29.122.3.8 `virtual double Go::EvalFunctorSurface::start_v ( ) const [virtual]`

Implements [Go::EvalSurface](#).

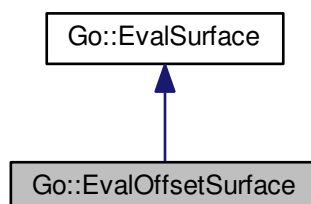
The documentation for this class was generated from the following file:

- [isogeometric\\_model/include/GoTools/isogeometric\\_model/EvalFunctorSurface.h](#)

## 29.123 Go::EvalOffsetSurface Class Reference

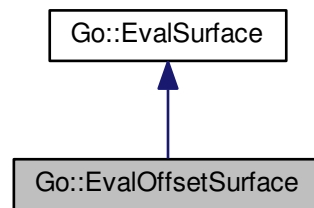
```
#include <EvalOffsetSurface.h>
```

Inheritance diagram for Go::EvalOffsetSurface:





Collaboration diagram for Go::EvalOffsetSurface:



## Public Member Functions

- [EvalOffsetSurface](#) (shared\_ptr< [ftFaceBase](#) > base\_sf, double offset\_dist, double epsgeo)
- virtual [~EvalOffsetSurface](#) ()
- virtual [Point](#) eval (double u, double v) const
- virtual void eval (double u, double v, int n, [Point](#) der[]) const
- virtual double start\_u () const
- virtual double start\_v () const
- virtual double end\_u () const
- virtual double end\_v () const
- virtual int dim () const
- virtual bool approximationOK (double par\_u, double par\_v, [Point](#) approxpos, double tol1, double tol2) const
- void [gridSelfIntersections](#) (const [HermiteGrid2D](#) &grid, std::vector< int > &grid\_self\_intersections, std::vector< double > &radius\_of\_curv) const
- void [gridKinks](#) (const [HermiteGrid2D](#) &grid, const std::vector< shared\_ptr< [SplineCurve](#) > > &kink\_cvs\_2d, std::vector< int > &grid\_kinks) const
- std::vector< shared\_ptr< [SplineCurve](#) > > [getProjKinkCurves](#) (std::vector< pair< shared\_ptr< [ParamCurve](#) >, shared\_ptr< [ParamCurve](#) > > > &par\_cvs, std::vector< pair< shared\_ptr< [ParamSurface](#) >, shared\_ptr< [ParamSurface](#) > > > &sfs)

### 29.123.1 Detailed Description

Definition at line 64 of file [EvalOffsetSurface.h](#).

### 29.123.2 Constructor & Destructor Documentation

29.123.2.1 [Go::EvalOffsetSurface::EvalOffsetSurface](#) ( shared\_ptr< [ftFaceBase](#) > base\_sf, double offset\_dist, double epsgeo )

29.123.2.2 virtual [Go::EvalOffsetSurface::~EvalOffsetSurface](#) ( ) [virtual]

### 29.123.3 Member Function Documentation

29.123.3.1 virtual bool [Go::EvalOffsetSurface::approximationOK](#) ( double par\_u, double par\_v, [Point](#) approxpos, double tol1, double tol2 ) const [virtual]

Check if the curve, evaluated at a given parameter, approximates a given position within a given tolerance.

## Parameters

|                  |                                                                                          |
|------------------|------------------------------------------------------------------------------------------|
| <i>par</i>       | the parameter at which to check the curve                                                |
| <i>approxpos</i> | the position we want to check whether or not the curve approximates for parameter 'par'. |
| <i>tol1</i>      | approximation tolerance.                                                                 |
| <i>tol2</i>      | another approximation tolerance (its use is defined by some of the derived classes.      |

## Returns

'true' if the curve approximates the point at the parameter, 'false' otherwise.

Implements [Go::EvalSurface](#).

**29.123.3.2** `virtual int Go::EvalOffsetSurface::dim ( ) const [virtual]`

Get the dimension of the space in which the curve lies.

## Returns

the space dimension of the curve.

Implements [Go::EvalSurface](#).

**29.123.3.3** `virtual double Go::EvalOffsetSurface::end_u ( ) const [virtual]`

Get the end parameter of the curve.

## Returns

the end parameter of the curve.

Implements [Go::EvalSurface](#).

**29.123.3.4** `virtual double Go::EvalOffsetSurface::end_v ( ) const [virtual]`

Implements [Go::EvalSurface](#).

**29.123.3.5** `virtual Point Go::EvalOffsetSurface::eval ( double u, double v ) const [virtual]`

Evaluate a point on the surface for a given parameter

## Parameters

|          |                                                  |
|----------|--------------------------------------------------|
| <i>t</i> | the parameter for which to evaluate the surface. |
|----------|--------------------------------------------------|

**Returns**

the evaluated point

Implements [Go::EvalSurface](#).

29.123.3.6 `virtual void Go::EvalOffsetSurface::eval ( double u, double v, int n, Point der[ ] ) const` [virtual]

Evaluate a point and a certain number of derivatives on the surface for a given parameter.

**Parameters**

|          |                                                |
|----------|------------------------------------------------|
| <i>t</i> | the parameter for which to evaluate the curve. |
| <i>n</i> | the number of derivatives (0 or more)          |

**Return values**

|            |                                                                                                                                                                                                                                                                                                                                                                        |
|------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>der</i> | pointer to an array of Points where the result will be written. The position will be stored first, then the first derivative (tangent), then the second, etc.. <b>NB:</b> For most (all) derived classes of 'EvalSurface', the implementation actually only supports the computation of one derivative, i.e. if $n > 1$ , only one derivative will be computed anyway. |
|------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

Implements [Go::EvalSurface](#).

29.123.3.7 `std::vector<shared_ptr<SplineCurve>> Go::EvalOffsetSurface::getProjKinkCurves ( std::vector<pair<shared_ptr<ParamCurve>, shared_ptr<ParamCurve>>> & par_cvs, std::vector<pair<shared_ptr<ParamSurface>, shared_ptr<ParamSurface>>> & sfs )`

29.123.3.8 `void Go::EvalOffsetSurface::gridKinks ( const HermiteGrid2D & grid, const std::vector<shared_ptr<SplineCurve>> & kink_cvs_2d, std::vector<int> & grid_kinks ) const`

29.123.3.9 `void Go::EvalOffsetSurface::gridSelfIntersections ( const HermiteGrid2D & grid, std::vector<int> & grid_self_intersections, std::vector<double> & radius_of_curv ) const`

29.123.3.10 `virtual double Go::EvalOffsetSurface::start_u ( ) const` [virtual]

Get the start parameter of the curve.

**Returns**

the start parameter of the curve.

Implements [Go::EvalSurface](#).

29.123.3.11 `virtual double Go::EvalOffsetSurface::start_v ( ) const` [virtual]

Implements [Go::EvalSurface](#).

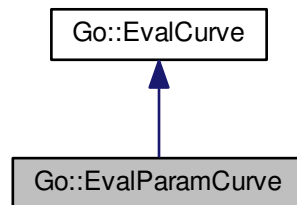
The documentation for this class was generated from the following file:

- `compositemodel/include/GoTools/compositemodel/EvalOffsetSurface.h`

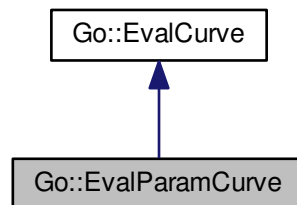
## 29.124 Go::EvalParamCurve Class Reference

```
#include <EvalParamCurve.h>
```

Inheritance diagram for Go::EvalParamCurve:



Collaboration diagram for Go::EvalParamCurve:



### Public Member Functions

- [EvalParamCurve](#) (shared\_ptr< [Go::ParamCurve](#) > &crv)  
*Constructor, taking a parametric curve.*
- virtual [~EvalParamCurve](#) ()  
*virtual destructor enables safe inheritance*
- virtual [Point](#) [eval](#) (double t) const
- virtual void [eval](#) (double t, int n, [Point](#) der[]) const
- virtual [double](#) [start](#) () const
- virtual [double](#) [end](#) () const
- virtual int [dim](#) () const  
*Dimension of the lifted curve (i.e. 3).*
- virtual [bool](#) [approximationOK](#) (double par, [Point](#) approxpos, [double](#) tol1, [double](#) tol2) const
- virtual void [write](#) (std::ostream &out) const

### 29.124.1 Detailed Description

This class represents a interface to a [ParamCurve](#) to make it fit into the evaluator based curve concept

Definition at line 59 of file EvalParamCurve.h.

### 29.124.2 Constructor & Destructor Documentation

29.124.2.1 `Go::EvalParamCurve::EvalParamCurve ( shared_ptr< Go::ParamCurve > & crv )`

Constructor, taking a parametric curve.

29.124.2.2 `virtual Go::EvalParamCurve::~EvalParamCurve ( ) [virtual]`

virtual destructor enables safe inheritance

### 29.124.3 Member Function Documentation

29.124.3.1 `virtual bool Go::EvalParamCurve::approximationOK ( double par, Point approxpos, double tol1, double tol2 ) const [virtual]`

Inherited from [EvalCurve::approximationOK\(\)](#).

#### Parameters

|                  |                                                                                          |
|------------------|------------------------------------------------------------------------------------------|
| <i>par</i>       | the parameter at which to check the curve                                                |
| <i>approxpos</i> | the position we want to check whether or not the curve approximates for parameter 'par'. |
| <i>tol1</i>      | unused                                                                                   |
| <i>tol2</i>      | unused                                                                                   |

#### Returns

'true' if the curve approximates the point at the parameter (within the tolerance given in the constructor, 'eps-geo'). 'false' otherwise.

Implements [Go::EvalCurve](#).

29.124.3.2 `virtual int Go::EvalParamCurve::dim ( ) const [virtual]`

Dimension of the lifted curve (i.e. 3).

Implements [Go::EvalCurve](#).

29.124.3.3 `virtual double Go::EvalParamCurve::end ( ) const [virtual]`

Get the end parameter of the curve.

**Returns**

the end parameter of the curve.

Implements [Go::EvalCurve](#).

29.124.3.4 `virtual Point Go::EvalParamCurve::eval ( double t ) const [virtual]`

Evaluate a point on the curve for a given parameter

**Parameters**

|          |                                                |
|----------|------------------------------------------------|
| <i>t</i> | the parameter for which to evaluate the curve. |
|----------|------------------------------------------------|

**Returns**

the evaluated point

Implements [Go::EvalCurve](#).

29.124.3.5 `virtual void Go::EvalParamCurve::eval ( double t, int n, Point der[] ) const [virtual]`

Evaluate a point and a certain number of derivatives on the curve for a given parameter.

**Parameters**

|          |                                                |
|----------|------------------------------------------------|
| <i>t</i> | the parameter for which to evaluate the curve. |
| <i>n</i> | the number of derivatives (0 or more)          |

**Return values**

|            |                                                                                                                                                                                                                                                                                                                                                                                        |
|------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>der</i> | pointer to an array of Points where the result will be written. The position will be stored first, then the first derivative (tangent), then the second, etc.. <b>NB:</b> For most (all) derived classes of ' <a href="#">EvalCurve</a> ', the implementation actually only supports the computation of one derivative, i.e. if $n > 1$ , only one derivative will be computed anyway. |
|------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

Implements [Go::EvalCurve](#).

29.124.3.6 `virtual double Go::EvalParamCurve::start ( ) const [virtual]`

Get the start parameter of the curve.

**Returns**

the start parameter of the curve.

Implements [Go::EvalCurve](#).

**29.124.3.7** `virtual void Go::EvalParamCurve::write ( std::ostream & out ) const` `[virtual]`

Reimplemented from [Go::EvalCurve](#).

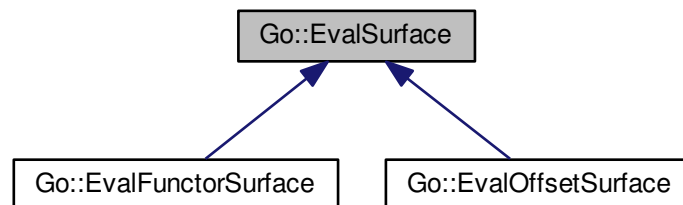
The documentation for this class was generated from the following file:

- `gotools-core/include/GoTools/creators/EvalParamCurve.h`

## 29.125 Go::EvalSurface Class Reference

```
#include <EvalSurface.h>
```

Inheritance diagram for Go::EvalSurface:

**Public Member Functions**

- virtual `~EvalSurface ()`  
*virtual destructor ensures save inheritance*
- virtual `Point eval (double u, double v) const =0`
- virtual `void eval (double u, double v, int n, Point der[]) const =0`
- virtual `double start_u () const =0`
- virtual `double start_v () const =0`
- virtual `double end_u () const =0`
- virtual `double end_v () const =0`
- virtual `int dim () const =0`
- virtual `bool approximationOK (double par_u, double par_v, Point approxpos, double tol1, double tol2) const =0`
- virtual `void write (std::ostream &out) const`

### 29.125.1 Detailed Description

This is the abstract base class of a curve type that can be evaluated. Representing the exact geometry, typically used when iteratively approximating the curve.

Definition at line 56 of file EvalSurface.h.

### 29.125.2 Constructor & Destructor Documentation

29.125.2.1 `virtual Go::EvalSurface::~EvalSurface ( ) [virtual]`

virtual destructor ensures save inheritance

### 29.125.3 Member Function Documentation

29.125.3.1 `virtual bool Go::EvalSurface::approximationOK ( double par_u, double par_v, Point approxpos, double tol1, double tol2 ) const [pure virtual]`

Check if the curve, evaluated at a given parameter, approximates a given position within a given tolerance.

#### Parameters

|                  |                                                                                          |
|------------------|------------------------------------------------------------------------------------------|
| <i>par</i>       | the parameter at which to check the curve                                                |
| <i>approxpos</i> | the position we want to check whether or not the curve approximates for parameter 'par'. |
| <i>tol1</i>      | approximation tolerance.                                                                 |
| <i>tol2</i>      | another approximation tolerance (its use is defined by some of the derived classes.      |

#### Returns

'true' if the curve approximates the point at the parameter, 'false' otherwise.

Implemented in [Go::EvalOffsetSurface](#), and [Go::EvalFunctorSurface](#).

29.125.3.2 `virtual int Go::EvalSurface::dim ( ) const [pure virtual]`

Get the dimension of the space in which the curve lies.

#### Returns

the space dimension of the curve.

Implemented in [Go::EvalOffsetSurface](#), and [Go::EvalFunctorSurface](#).



29.125.3.3 `virtual double Go::EvalSurface::end_u ( ) const [pure virtual]`

Get the end parameter of the curve.

#### Returns

the end parameter of the curve.

Implemented in [Go::EvalOffsetSurface](#), and [Go::EvalFunctorSurface](#).

29.125.3.4 `virtual double Go::EvalSurface::end_v ( ) const [pure virtual]`

Implemented in [Go::EvalOffsetSurface](#), and [Go::EvalFunctorSurface](#).

29.125.3.5 `virtual Point Go::EvalSurface::eval ( double u, double v ) const [pure virtual]`

Evaluate a point on the surface for a given parameter

#### Parameters

|          |                                                |
|----------|------------------------------------------------|
| <i>t</i> | the parameter for which to evaluate the curve. |
|----------|------------------------------------------------|

#### Returns

the evaluated point

Implemented in [Go::EvalOffsetSurface](#), and [Go::EvalFunctorSurface](#).

29.125.3.6 `virtual void Go::EvalSurface::eval ( double u, double v, int n, Point der[] ) const [pure virtual]`

Evaluate a point and a certain number of derivatives on the surface for a given parameter.

#### Parameters

|          |                                                |
|----------|------------------------------------------------|
| <i>t</i> | the parameter for which to evaluate the curve. |
| <i>n</i> | the number of derivatives (0 or more)          |

#### Return values

|            |                                                                                                                                                                                                                                                                                                                                                                        |
|------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>der</i> | pointer to an array of Points where the result will be written. The position will be stored first, then the first derivative (tangent), then the second, etc.. <b>NB:</b> For most (all) derived classes of 'EvalSurface', the implementation actually only supports the computation of one derivative, i.e. if $n > 1$ , only one derivative will be computed anyway. |
|------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

Implemented in [Go::EvalOffsetSurface](#), and [Go::EvalFunctorSurface](#).

29.125.3.7 `virtual double Go::EvalSurface::start_u( ) const` [pure virtual]

Get the start parameter of the curve.

#### Returns

the start parameter of the curve.

Implemented in [Go::EvalOffsetSurface](#), and [Go::EvalFunctorSurface](#).

29.125.3.8 `virtual double Go::EvalSurface::start_v( ) const` [pure virtual]

Implemented in [Go::EvalOffsetSurface](#), and [Go::EvalFunctorSurface](#).

29.125.3.9 `virtual void Go::EvalSurface::write ( std::ostream & out ) const` [virtual]

The documentation for this class was generated from the following file:

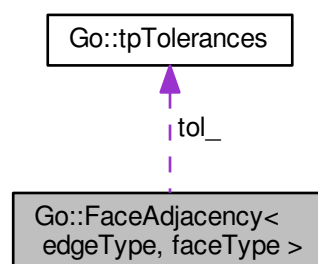
- [gotools-core/include/GoTools/creators/EvalSurface.h](#)

## 29.126 `Go::FaceAdjacency< edgeType, faceType >` Class Template Reference

Compute adjacency information for a set of surfaces.

```
#include <FaceAdjacency.h>
```

Collaboration diagram for `Go::FaceAdjacency< edgeType, faceType >`:



## Public Member Functions

- [FaceAdjacency](#) (double tol\_gap, double tol\_neighbour, double tol\_kink, double tol\_bend)
- [FaceAdjacency](#) (const tpTolerances &tol)
- [~FaceAdjacency](#) ()
- void [setTolerances](#) (const tpTolerances &tol)
 

*Changes the tolerances used in adjacency analysis.*
- [tpTolerances getTolerances](#) ()
 

*Fetch the topological tolerances used in the adjacency analysis.*
- void [computeAdjacency](#) (const std::vector< shared\_ptr< faceType > > &faces, int first\_idx)
 

*Compute the adjacency between the given faces.*
- void [computeAdjacency](#) (const std::vector< shared\_ptr< faceType > > &faces, std::vector< std::pair< faceType \*, faceType \* > > &orient\_inconsistent, int first\_idx)
- void [setConnectivity](#) (const std::vector< shared\_ptr< faceType > > &faces)
- void [setConnectivity](#) (const std::vector< faceType \* > &faces)
- std::vector< edgeType \* > [connectTwins](#) (edgeType \*e1, edgeType \*e2, double min1, double max1, double min2, double max2, int &status)
- void [releaseFaceAdjacency](#) (shared\_ptr< faceType > face)
- void [computeFaceAdjacency](#) (std::vector< shared\_ptr< faceType > > faces, shared\_ptr< faceType > new\_face)
- void [computeFaceAdjacency](#) (std::vector< faceType \* > faces, faceType \*new\_face)
- void [computeFaceAdjacency](#) (std::vector< shared\_ptr< faceType > > faces, shared\_ptr< faceType > new\_face, std::vector< std::pair< faceType \*, faceType \* > > &orient\_inconsistent)
- void [computeFaceAdjacency](#) (std::vector< faceType \* > faces, faceType \*new\_face, std::vector< std::pair< faceType \*, faceType \* > > &orient\_inconsistent)
- void [updateConnectivity](#) (edgeType \*e1, edgeType \*e2)

## Protected Attributes

- [tpTolerances tol\\_](#)
- std::vector< shared\_ptr< edgeType > > [new\\_edges\\_](#)

### 29.126.1 Detailed Description

```
template<class edgeType, class faceType>
class Go::FaceAdjacency< edgeType, faceType >
```

Compute adjacency information for a set of surfaces.

Definition at line 85 of file FaceAdjacency.h.

### 29.126.2 Constructor & Destructor Documentation

29.126.2.1 `template<class edgeType , class faceType > Go::FaceAdjacency< edgeType, faceType >::FaceAdjacency ( double tol_gap, double tol_neighbour, double tol_kink, double tol_bend ) [inline]`

Constructor.

## Parameters

|                      |                                                                                              |
|----------------------|----------------------------------------------------------------------------------------------|
| <i>tol_gap</i>       | Tolerance for when two faces are assumed to be C0 continuous                                 |
| <i>tol_neighbour</i> | Tolerance for when two faces are assumed to be neighbours                                    |
| <i>tol_kink</i>      | Tolerance for when two adjacent faces are assumed to be C1 continous                         |
| <i>tol_bend</i>      | Tolerance for when two adjacent faces are assumed to have an intentionally smooth connection |

Definition at line 104 of file FaceAdjacency.h.

```
29.126.2.2 template<class edgeType , class faceType > Go::FaceAdjacency< edgeType, faceType >::FaceAdjacency (
 const tpTolerances & tol) [inline]
```

Constructor.

## Parameters

|            |                        |
|------------|------------------------|
| <i>tol</i> | Topological tolerances |
|------------|------------------------|

Definition at line 112 of file FaceAdjacency.h.

```
29.126.2.3 template<class edgeType , class faceType > Go::FaceAdjacency< edgeType, faceType
 >::~~FaceAdjacency () [inline]
```

Destructor. Detailed description.

Definition at line 120 of file FaceAdjacency.h.

## 29.126.3 Member Function Documentation

```
29.126.3.1 template<class edgeType , class faceType > void Go::FaceAdjacency< edgeType, faceType
 >::computeAdjacency (const std::vector< shared_ptr< faceType > > & faces, int first_idx) [inline]
```

Compute the adjacency between the given faces.

Definition at line 144 of file FaceAdjacency.h.

```
29.126.3.2 template<class edgeType , class faceType > void Go::FaceAdjacency< edgeType, faceType
 >::computeAdjacency (const std::vector< shared_ptr< faceType > > & faces, std::vector< std::pair<
 faceType *, faceType * > > & orient_inconsistent, int first_idx) [inline]
```

Compute the adjacency between the given faces

## Parameters

|                            |                                                                                      |
|----------------------------|--------------------------------------------------------------------------------------|
| <i>faces</i>               | The face set on which to compute adjacency                                           |
| <i>orient_inconsistent</i> | Information of adjacent faces where the direction of the face normal is inconsistent |

Definition at line 159 of file FaceAdjacency.h.

```
29.126.3.3 template<class edgeType , class faceType > void Go::FaceAdjacency< edgeType, faceType
>::computeFaceAdjacency (std::vector< shared_ptr< faceType > > faces, shared_ptr< faceType > new_face)
[inline]
```

Add one face to the topological structures of a face set

#### Parameters

|                 |                                                                |
|-----------------|----------------------------------------------------------------|
| <i>faces</i>    | The set of faces where all topological information is computed |
| <i>new_face</i> | The face to add to the face set                                |

Definition at line 438 of file FaceAdjacency.h.

```
29.126.3.4 template<class edgeType , class faceType > void Go::FaceAdjacency< edgeType, faceType
>::computeFaceAdjacency (std::vector< faceType * > faces, faceType * new_face) [inline]
```

Definition at line 446 of file FaceAdjacency.h.

```
29.126.3.5 template<class edgeType , class faceType > void Go::FaceAdjacency< edgeType, faceType
>::computeFaceAdjacency (std::vector< shared_ptr< faceType > > faces, shared_ptr< faceType > new_face,
std::vector< std::pair< faceType *, faceType * > > & orient_inconsistent) [inline]
```

Add one face to the topological structures of a face set

#### Parameters

|                            |                                                                                      |
|----------------------------|--------------------------------------------------------------------------------------|
| <i>faces</i>               | The set of faces where all topological information is computed                       |
| <i>new_face</i>            | The face to add to the face set                                                      |
| <i>orient_inconsistent</i> | Information of adjacent faces where the direction of the face normal is inconsistent |

Definition at line 460 of file FaceAdjacency.h.

```
29.126.3.6 template<class edgeType , class faceType > void Go::FaceAdjacency< edgeType, faceType
>::computeFaceAdjacency (std::vector< faceType * > faces, faceType * new_face, std::vector< std::pair<
faceType *, faceType * > > & orient_inconsistent) [inline]
```

Add one face to the topological structures of a face set

#### Parameters

|                            |                                                                                      |
|----------------------------|--------------------------------------------------------------------------------------|
| <i>faces</i>               | The set of faces where all topological information is computed                       |
| <i>new_face</i>            | The face to add to the face set                                                      |
| <i>orient_inconsistent</i> | Information of adjacent faces where the direction of the face normal is inconsistent |

Definition at line 590 of file FaceAdjacency.h.

```
29.126.3.7 template<class edgeType , class faceType > std::vector<edgeType*> Go::FaceAdjacency< edgeType,
faceType >::connectTwins (edgeType * e1, edgeType * e2, double min1, double max1, double min2,
double max2, int & status) [inline]
```

Returns pointer to (vector of) twin edges. Updates topological structure and, if necessary, splits edges.

Definition at line 331 of file FaceAdjacency.h.

```
29.126.3.8 template<class edgeType , class faceType > tpTolerances Go::FaceAdjacency< edgeType, faceType
>::getTolerances () [inline]
```

Fetch the topological tolerances used in the adjacency analysis.

Definition at line 135 of file FaceAdjacency.h.

```
29.126.3.9 template<class edgeType , class faceType > void Go::FaceAdjacency< edgeType, faceType
>::releaseFaceAdjacency (shared_ptr< faceType > face) [inline]
```

Remove one face from the topological structures of the face set it belongs to

Definition at line 389 of file FaceAdjacency.h.

```
29.126.3.10 template<class edgeType , class faceType > void Go::FaceAdjacency< edgeType, faceType
>::setConnectivity (const std::vector< shared_ptr< faceType > > & faces) [inline]
```

Fetch existing adjacency information between faces and set topological pointers representing this adjacency.

Definition at line 271 of file FaceAdjacency.h.

```
29.126.3.11 template<class edgeType , class faceType > void Go::FaceAdjacency< edgeType, faceType
>::setConnectivity (const std::vector< faceType * > & faces) [inline]
```

Fetch existing adjacency information between faces and set topological pointers representing this adjacency.

Definition at line 301 of file FaceAdjacency.h.

```
29.126.3.12 template<class edgeType , class faceType > void Go::FaceAdjacency< edgeType, faceType >::setTolerances
(const tpTolerances & tol) [inline]
```

Changes the tolerances used in adjacency analysis.

Definition at line 127 of file FaceAdjacency.h.

```
29.126.3.13 template<class edgeType , class faceType > void Go::FaceAdjacency< edgeType, faceType
>::updateConnectivity (edgeType * e1, edgeType * e2) [inline]
```

Update topological information related to two neighbouring faces, i.e. recompute information regarding the type of connectivity between the faces

Definition at line 719 of file FaceAdjacency.h.

#### 29.126.4 Member Data Documentation

```
29.126.4.1 template<class edgeType , class faceType > std::vector< shared_ptr<edgeType> > Go::FaceAdjacency<
edgeType, faceType >::new_edges_ [protected]
```

Definition at line 90 of file FaceAdjacency.h.

```
29.126.4.2 template<class edgeType , class faceType > tpTolerances Go::FaceAdjacency< edgeType, faceType
>::tol_ [protected]
```

Definition at line 89 of file FaceAdjacency.h.

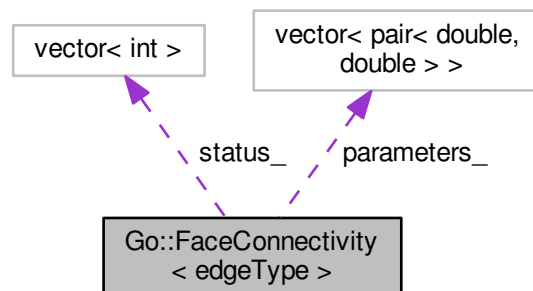
The documentation for this class was generated from the following file:

- [topology/include/GoTools/topology/FaceAdjacency.h](#)

## 29.127 Go::FaceConnectivity< edgeType > Struct Template Reference

```
#include <FaceConnectivity.h>
```

Collaboration diagram for Go::FaceConnectivity< edgeType >:



## Public Member Functions

- [FaceConnectivity](#) (edgeType \*e1, edgeType \*e2)  
*Constructor.*
- void [setEdges](#) (edgeType \*e1, edgeType \*e2)  
*Reset edge info.*
- int [BestStatus](#) () const  
*The highest continuity between the two faces.*
- int [WorstStatus](#) () const  
*The lowest continuity between the two faces.*

## Public Attributes

- edgeType \* [e1\\_](#)  
*Edge of first face along the common boundary.*
- edgeType \* [e2\\_](#)  
*Edge of second face along the common boundary.*
- vector< int > [status\\_](#)
- vector< pair< double, double > > [parameters\\_](#)  
*Parameter intervals limiting the areas of the found state of continuity.*

### 29.127.1 Detailed Description

```
template<class edgeType>
struct Go::FaceConnectivity< edgeType >
```

A structure storing the connectivity information between two adjacent faces

Definition at line 54 of file FaceConnectivity.h.

### 29.127.2 Constructor & Destructor Documentation

29.127.2.1 `template<class edgeType > Go::FaceConnectivity< edgeType >::FaceConnectivity ( edgeType * e1, edgeType * e2 ) [inline]`

Constructor.

Definition at line 74 of file FaceConnectivity.h.

### 29.127.3 Member Function Documentation

29.127.3.1 `template<class edgeType > int Go::FaceConnectivity< edgeType >::BestStatus ( ) const [inline]`

The highest continuity between the two faces.

Definition at line 88 of file FaceConnectivity.h.



29.127.3.2 `template<class edgeType > void Go::FaceConnectivity< edgeType >::setEdges ( edgeType * e1, edgeType * e2 ) [inline]`

Reset edge info.

Definition at line 81 of file FaceConnectivity.h.

29.127.3.3 `template<class edgeType > int Go::FaceConnectivity< edgeType >::WorstStatus ( ) const [inline]`

The lowest continuity between the two faces.

Definition at line 97 of file FaceConnectivity.h.

## 29.127.4 Member Data Documentation

29.127.4.1 `template<class edgeType > edgeType* Go::FaceConnectivity< edgeType >::e1_`

Edge of first face along the common boundary.

Definition at line 58 of file FaceConnectivity.h.

29.127.4.2 `template<class edgeType > edgeType* Go::FaceConnectivity< edgeType >::e2_`

Edge of second face along the common boundary.

Definition at line 60 of file FaceConnectivity.h.

29.127.4.3 `template<class edgeType > vector< pair<double, double> > Go::FaceConnectivity< edgeType >::parameters_`

Parameter intervals limiting the areas of the found state of continuity.

Definition at line 71 of file FaceConnectivity.h.

29.127.4.4 `template<class edgeType > vector<int> Go::FaceConnectivity< edgeType >::status_`

The status is: 0 : edges join smoothly.  $G^1$ . 1 : edges join, but normals are slightly discontinuous. A kink. 2 : edges join, but the normals are discontinuous.  $G^0$ . 3 : edges almost join. A gap. 4 : edges are totally discontinuous. The minimal [tpTopologicalInfo](#) has a one-element status vector and a two-element parameters vector

Definition at line 69 of file FaceConnectivity.h.

The documentation for this struct was generated from the following file:

- [topology/include/GoTools/topology/FaceConnectivity.h](#)

## 29.128 Go::FaceConnectivityUtils< edgeType, faceType > Class Template Reference

Utilities used in adjacency computations of face sets.

```
#include <FaceConnectivityUtils.h>
```

### Public Member Functions

- void [BoundaryLoops](#) (const std::vector< shared\_ptr< faceType > > &faces, std::vector< std::vector< edgeType \* > > &loopvec)
- void [BoundaryLoops](#) (const std::vector< shared\_ptr< faceType > > &faces, std::vector< std::vector< shared\_ptr< edgeType > > > &loopvec)
 

*Returns all boundary loops. Edges are returned as shared pointers.*
- void [disjointObjects](#) (const std::vector< shared\_ptr< faceType > > &faces, std::vector< std::vector< faceType \* > > &grouped\_faces)
 

*Separates objects which are not path connected.*
- void [cornersAndKinks](#) (const std::vector< shared\_ptr< faceType > > &faces, std::vector< edgeType \* > &vec)
- void [smoothEdges](#) (const std::vector< shared\_ptr< faceType > > &faces, std::vector< edgeType \* > &vec)
- std::vector< edgeType \* > [edgesBoundingFace](#) (faceType \*face)
 

*Returns all edges bounding a specific face.*

### 29.128.1 Detailed Description

```
template<class edgeType, class faceType>
class Go::FaceConnectivityUtils< edgeType, faceType >
```

Utilities used in adjacency computations of face sets.

Definition at line 56 of file FaceConnectivityUtils.h.

### 29.128.2 Member Function Documentation

29.128.2.1 `template<class edgeType , class faceType > void Go::FaceConnectivityUtils< edgeType, faceType >::BoundaryLoops ( const std::vector< shared_ptr< faceType > > & faces, std::vector< std::vector< edgeType * > > & loopvec ) [inline]`

Returns all boundary loops. Edges are returned as pointers. Keep in mind that calling next() on elements on the boundary will not iterate around the boundary, but around the face the edge comes from.

Definition at line 66 of file FaceConnectivityUtils.h.

29.128.2.2 `template<class edgeType , class faceType > void Go::FaceConnectivityUtils< edgeType, faceType >::BoundaryLoops ( const std::vector< shared_ptr< faceType > > & faces, std::vector< std::vector< shared_ptr< edgeType > > > & loopvec ) [inline]`

Returns all boundary loops. Edges are returned as shared pointers.

Definition at line 117 of file FaceConnectivityUtils.h.

```
29.128.2.3 template<class edgeType , class faceType > void Go::FaceConnectivityUtils< edgeType, faceType
>::cornersAndKinks (const std::vector< shared_ptr< faceType > > & faces, std::vector< edgeType * > & vec
) [inline]
```

Gives the first of every pair of edges representing a corner edge or kink edge.

Definition at line 203 of file FaceConnectivityUtils.h.

```
29.128.2.4 template<class edgeType , class faceType > void Go::FaceConnectivityUtils< edgeType, faceType
>::disjointObjects (const std::vector< shared_ptr< faceType > > & faces, std::vector< std::vector< faceType
* > > & grouped_faces) [inline]
```

Separates objects which are not path connected.

Definition at line 178 of file FaceConnectivityUtils.h.

```
29.128.2.5 template<class edgeType , class faceType > std::vector<edgeType*> Go::FaceConnectivityUtils<
edgeType, faceType >::edgesBoundingFace (faceType * face) [inline]
```

Returns all edges bounding a specific face.

Definition at line 290 of file FaceConnectivityUtils.h.

```
29.128.2.6 template<class edgeType , class faceType > void Go::FaceConnectivityUtils< edgeType, faceType
>::smoothEdges (const std::vector< shared_ptr< faceType > > & faces, std::vector< edgeType * > & vec)
[inline]
```

Gives the first of every pair of edges representing a smooth edge (edges with gaps excluded)

Definition at line 247 of file FaceConnectivityUtils.h.

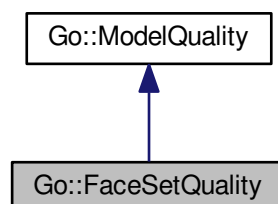
The documentation for this class was generated from the following file:

- [topology/include/GoTools/topology/FaceConnectivityUtils.h](#)

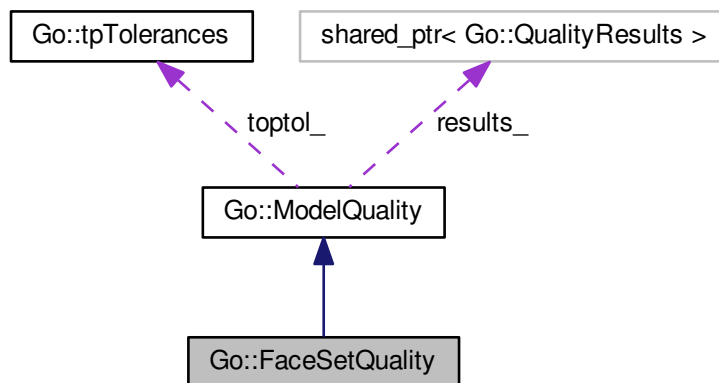
## 29.129 Go::FaceSetQuality Class Reference

```
#include <FaceSetQuality.h>
```

Inheritance diagram for Go::FaceSetQuality:



Collaboration diagram for Go::FaceSetQuality:



## Public Member Functions

- [FaceSetQuality](#) (double gap, double kink, double approx)
- [FaceSetQuality](#) (const [tpTolerances](#) &toptol, double approx)
- [FaceSetQuality](#) (shared\_ptr< [SurfaceModel](#) > sfmodel)
- virtual [~FaceSetQuality](#) ()
- void [attach](#) (shared\_ptr< [SurfaceModel](#) > sfmodel)
- virtual void [degenSurfaces](#) (std::vector< shared\_ptr< [ParamSurface](#) > > &deg\_sfs)
  - *Does not generate new geometry.*
- virtual void [degenerateSfCorners](#) (std::vector< shared\_ptr< [ftPoint](#) > > &deg\_corners)
  - *Generates new points, pointers to existing surfaces.*
- virtual void [identicalVertices](#) (std::vector< std::pair< shared\_ptr< [Vertex](#) >, shared\_ptr< [Vertex](#) > > > &identical\_vertices)
  - *Uses existing topological entity.*
- virtual void [identicalOrEmbeddedEdges](#) (std::vector< std::pair< shared\_ptr< [ftEdge](#) >, shared\_ptr< [ftEdge](#) > > > &identical\_edges, std::vector< std::pair< shared\_ptr< [ftEdge](#) >, shared\_ptr< [ftEdge](#) > > > &embedded\_edges)
  - *Uses existing topological entities. May contain generated curves.*
- virtual void [identicalOrEmbeddedFaces](#) (std::vector< std::pair< shared\_ptr< [ftSurface](#) >, shared\_ptr< [ftSurface](#) > > > &identical\_faces, std::vector< std::pair< shared\_ptr< [ftSurface](#) >, shared\_ptr< [ftSurface](#) > > > &embedded\_faces)
  - *Uses existing topological entities. Points to existing surfaces.*
- virtual void [miniEdges](#) (std::vector< shared\_ptr< [ftEdge](#) > > &mini\_edges)
  - *Uses existing topological entities.*
- virtual void [miniSurfaces](#) (std::vector< shared\_ptr< [ftSurface](#) > > &mini\_surfaces)
  - *Uses existing topological entities. Points to existing surfaces.*
- virtual void [vanishingSurfaceNormal](#) (std::vector< shared\_ptr< [ftPoint](#) > > &singular\_points, std::vector< shared\_ptr< [ftCurve](#) > > &singular\_curves)
  - *Returns ftPoints and ftCurves that will contain generated geometry.*
- virtual void [vanishingCurveTangent](#) (std::vector< shared\_ptr< [PointOnCurve](#) > > &sing\_points, std::vector< std::pair< shared\_ptr< [PointOnCurve](#) >, shared\_ptr< [PointOnCurve](#) > > > &sing\_curves)

- Contains generated points and curves fetched from topological entities.*

  - virtual void [sliverSurfaces](#) (std::vector< shared\_ptr< [ParamSurface](#) > > &sliver\_sfs, double thickness, double factor=2.0)

*Uses existing geometry.*
- virtual void [narrowRegion](#) (std::vector< std::pair< shared\_ptr< [PointOnEdge](#) >, shared\_ptr< [PointOnEdge](#) > > &narrow\_regions)

*Uses existing topological entities. Contains generated geometry.*
- virtual void [edgeVertexDistance](#) (std::vector< std::pair< [ftEdge](#) \*, shared\_ptr< [Vertex](#) > > &edge\_↔vertices)

*Uses existing topological entities.*
- virtual void [faceVertexDistance](#) (std::vector< std::pair< [ftSurface](#) \*, shared\_ptr< [Vertex](#) > > &face\_↔vertices)

*Existing topology. Partly generated geometry, existing surfaces.*
- virtual void [faceEdgeDistance](#) (std::vector< std::pair< [ftSurface](#) \*, [ftEdge](#) \* > > &face\_edges)

*Existing topology. Partly generated geometry, existing surfaces.*
- virtual void [edgePosAndTangDiscontinuity](#) (std::vector< std::pair< [ftEdge](#) \*, [ftEdge](#) \* > > &pos\_disconts, std::vector< std::pair< [ftEdge](#) \*, [ftEdge](#) \* > > &tang\_disconts)

*Uses existing topological entities.*
- virtual void [facePositionDiscontinuity](#) (std::vector< std::pair< [ftEdge](#) \*, [ftEdge](#) \* > > &pos\_disconts)

*Uses existing topological entities. Existing surfaces available.*
- virtual void [faceTangentDiscontinuity](#) (std::vector< std::pair< [ftEdge](#) \*, [ftEdge](#) \* > > &tangent\_disconts)

*Uses existing topological entities. Existing surfaces available.*
- virtual void [loopOrientationConsistency](#) (std::vector< shared\_ptr< [Loop](#) > > &inconsistent\_loops)

*Uses existing topological entities.*
- virtual void [faceNormalConsistency](#) (std::vector< shared\_ptr< [ftSurface](#) > > &inconsistent\_faces)

*Uses existing topological entities. Existing surfaces available.*
- virtual void [sfG1Discontinuity](#) (std::vector< shared\_ptr< [ftSurface](#) > > &discont\_sfs)

*Uses existing topological entities. Existing surfaces available.*
- virtual void [sfC1Discontinuity](#) (std::vector< shared\_ptr< [ftSurface](#) > > &discont\_sfs)

*Uses existing topological entities. Existing surfaces available.*
- virtual void [cvC1G1Discontinuity](#) (std::vector< shared\_ptr< [ParamCurve](#) > > &c1\_discont, std::vector< shared\_ptr< [ParamCurve](#) > > &g1\_discont)

*Contains curves fetched from topological entities.*
- virtual void [cvCurvatureRadius](#) (std::vector< std::pair< shared\_ptr< [PointOnCurve](#) >, double > > &small\_↔\_curv\_rad, std::pair< shared\_ptr< [PointOnCurve](#) >, double > &minimum\_curv\_rad)

*Contains generated points and curves fetched from topological entities.*
- virtual void [sfCurvatureRadius](#) (std::vector< std::pair< shared\_ptr< [ftPoint](#) >, double > > &small\_curv\_rad, std::pair< shared\_ptr< [ftPoint](#) >, double > &minimum\_curv\_rad)

*Generates new points, pointers to existing surfaces.*
- virtual void [acuteEdgeAngle](#) (std::vector< std::pair< [ftEdge](#) \*, [ftEdge](#) \* > > &edge\_acute)

*Uses existing topological entities.*
- virtual void [acuteFaceAngle](#) (std::vector< std::pair< [ftSurface](#) \*, [ftSurface](#) \* > > &face\_acute)

*Uses existing topological entities. Existing surfaces available.*
- virtual void [loopIntersection](#) (std::vector< std::pair< shared\_ptr< [PointOnEdge](#) >, shared\_ptr< [PointOnEdge](#) > > &loop\_intersection)

*Uses existing topological entities. Contains generated geometry.*
- virtual void [loopSelfIntersection](#) (std::vector< std::pair< shared\_ptr< [PointOnEdge](#) >, shared\_ptr< [PointOnEdge](#) > > &loop\_self\_intersection)

*Uses existing topological entities. Contains generated geometry.*
- virtual void [indistinctKnots](#) (std::vector< shared\_ptr< [ParamCurve](#) > > &cv\_knots, std::vector< shared\_ptr< [ParamSurface](#) > > &sf\_knots, double tol=1.0e-8)

*Uses existing surface and possibly generated curves.*
- shared\_ptr< [SurfaceModel](#) > [getAssociatedSfModel](#) ()

## Additional Inherited Members

### 29.129.1 Detailed Description

Definition at line 50 of file FaceSetQuality.h.

### 29.129.2 Constructor & Destructor Documentation

29.129.2.1 `Go::FaceSetQuality::FaceSetQuality ( double gap, double kink, double approx )`

29.129.2.2 `Go::FaceSetQuality::FaceSetQuality ( const tpTolerances & toptol, double approx )`

29.129.2.3 `Go::FaceSetQuality::FaceSetQuality ( shared_ptr< SurfaceModel > smodel )`

29.129.2.4 `virtual Go::FaceSetQuality::~FaceSetQuality ( ) [virtual]`

### 29.129.3 Member Function Documentation

29.129.3.1 `virtual void Go::FaceSetQuality::acuteEdgeAngle ( std::vector< std::pair< ftEdge *, ftEdge * > > & edge_acute ) [virtual]`

Uses existing topological entities.

Reimplemented from [Go::ModelQuality](#).

29.129.3.2 `virtual void Go::FaceSetQuality::acuteFaceAngle ( std::vector< std::pair< ftSurface *, ftSurface * > > & face_acute ) [virtual]`

Uses existing topological entities. Existing surfaces available.

Reimplemented from [Go::ModelQuality](#).

29.129.3.3 `void Go::FaceSetQuality::attach ( shared_ptr< SurfaceModel > smodel )`

29.129.3.4 `virtual void Go::FaceSetQuality::cvC1G1Discontinuity ( std::vector< shared_ptr< ParamCurve > > & c1_discont, std::vector< shared_ptr< ParamCurve > > & g1_discont ) [virtual]`

Contains curves fetched from topological entities.

Reimplemented from [Go::ModelQuality](#).

29.129.3.5 `virtual void Go::FaceSetQuality::cvCurvatureRadius ( std::vector< std::pair< shared_ptr< PointOnCurve >, double > > & small_curv_rad, std::pair< shared_ptr< PointOnCurve >, double > & minimum_curv_rad ) [virtual]`

Contains generated points and curves fetched from topological entities.

Reimplemented from [Go::ModelQuality](#).

```
29.129.3.6 virtual void Go::FaceSetQuality::degenerateSfCorners (std::vector< shared_ptr< ftPoint > > & deg_corners)
 [virtual]
```

Generates new points, pointers to existing surfaces.

Reimplemented from [Go::ModelQuality](#).

```
29.129.3.7 virtual void Go::FaceSetQuality::degenSurfaces (std::vector< shared_ptr< ParamSurface > > & deg_sfs)
 [virtual]
```

Does not generate new geometry.

Reimplemented from [Go::ModelQuality](#).

```
29.129.3.8 virtual void Go::FaceSetQuality::edgePosAndTangDiscontinuity (std::vector< std::pair< ftEdge *, ftEdge * >
 > & pos_disconts, std::vector< std::pair< ftEdge *, ftEdge * > > & tang_disconts) [virtual]
```

Uses existing topological entities.

Reimplemented from [Go::ModelQuality](#).

```
29.129.3.9 virtual void Go::FaceSetQuality::edgeVertexDistance (std::vector< std::pair< ftEdge *, shared_ptr< Vertex >
 > > & edge_vertices) [virtual]
```

Uses existing topological entities.

Reimplemented from [Go::ModelQuality](#).

```
29.129.3.10 virtual void Go::FaceSetQuality::faceEdgeDistance (std::vector< std::pair< ftSurface *, ftEdge * > > &
 face_edges) [virtual]
```

Existing topology. Partly generated geometry, existing surfaces.

Reimplemented from [Go::ModelQuality](#).

```
29.129.3.11 virtual void Go::FaceSetQuality::faceNormalConsistency (std::vector< shared_ptr< ftSurface > > &
 inconsistent_faces) [virtual]
```

Uses existing topological entities. Existing surfaces available.

Reimplemented from [Go::ModelQuality](#).

```
29.129.3.12 virtual void Go::FaceSetQuality::facePositionDiscontinuity (std::vector< std::pair< ftEdge *, ftEdge * > > &
 pos_disconts) [virtual]
```

Uses existing topological entities. Existing surfaces available.

Reimplemented from [Go::ModelQuality](#).

29.129.3.13 `virtual void Go::FaceSetQuality::faceTangentDiscontinuity ( std::vector< std::pair< ftEdge *, ftEdge * > > & tangent_disconts ) [virtual]`

Uses existing topological entities. Existing surfaces available.

Reimplemented from [Go::ModelQuality](#).

29.129.3.14 `virtual void Go::FaceSetQuality::faceVertexDistance ( std::vector< std::pair< ftSurface *, shared_ptr< Vertex > > > & face_vertices ) [virtual]`

Existing topology. Partly generated geometry, existing surfaces.

Reimplemented from [Go::ModelQuality](#).

29.129.3.15 `shared_ptr<SurfaceModel> Go::FaceSetQuality::getAssociatedSfModel ( ) [inline]`

Definition at line 178 of file FaceSetQuality.h.

29.129.3.16 `virtual void Go::FaceSetQuality::identicalOrEmbeddedEdges ( std::vector< std::pair< shared_ptr< ftEdge >, shared_ptr< ftEdge > > > & identical_edges, std::vector< std::pair< shared_ptr< ftEdge >, shared_ptr< ftEdge > > > & embedded_edges ) [virtual]`

Uses existing topological entities. May contain generated curves.

Reimplemented from [Go::ModelQuality](#).

29.129.3.17 `virtual void Go::FaceSetQuality::identicalOrEmbeddedFaces ( std::vector< std::pair< shared_ptr< ftSurface >, shared_ptr< ftSurface > > > & identical_faces, std::vector< std::pair< shared_ptr< ftSurface >, shared_ptr< ftSurface > > > & embedded_faces ) [virtual]`

Uses existing topological entities. Points to existing surfaces.

Reimplemented from [Go::ModelQuality](#).

29.129.3.18 `virtual void Go::FaceSetQuality::identicalVertices ( std::vector< std::pair< shared_ptr< Vertex >, shared_ptr< Vertex > > > & identical_vertices ) [virtual]`

Uses existing topological entity.

Reimplemented from [Go::ModelQuality](#).

29.129.3.19 `virtual void Go::FaceSetQuality::indistinctKnots ( std::vector< shared_ptr< ParamCurve > > & cv_knots, std::vector< shared_ptr< ParamSurface > > & sf_knots, double tol=1.0e-8 ) [virtual]`

Uses existing surface and possibly generated curves.

Reimplemented from [Go::ModelQuality](#).



29.129.3.20 `virtual void Go::FaceSetQuality::loopIntersection ( std::vector< std::pair< shared_ptr< PointOnEdge >, shared_ptr< PointOnEdge > > > & loop_intersection ) [virtual]`

Uses existing topological entities. Contains generated geometry.

Reimplemented from [Go::ModelQuality](#).

29.129.3.21 `virtual void Go::FaceSetQuality::loopOrientationConsistency ( std::vector< shared_ptr< Loop > > & inconsistent_loops ) [virtual]`

Uses existing topological entities.

Reimplemented from [Go::ModelQuality](#).

29.129.3.22 `virtual void Go::FaceSetQuality::loopSelfIntersection ( std::vector< std::pair< shared_ptr< PointOnEdge >, shared_ptr< PointOnEdge > > > & loop_self_intersection ) [virtual]`

Uses existing topological entities. Contains generated geometry.

Reimplemented from [Go::ModelQuality](#).

29.129.3.23 `virtual void Go::FaceSetQuality::miniEdges ( std::vector< shared_ptr< ftEdge > > & mini_edges ) [virtual]`

Uses existing topological entities.

Reimplemented from [Go::ModelQuality](#).

29.129.3.24 `virtual void Go::FaceSetQuality::miniSurfaces ( std::vector< shared_ptr< ftSurface > > & mini_surfaces ) [virtual]`

Uses existing topological entities. Points to existing surfaces.

Reimplemented from [Go::ModelQuality](#).

29.129.3.25 `virtual void Go::FaceSetQuality::narrowRegion ( std::vector< std::pair< shared_ptr< PointOnEdge >, shared_ptr< PointOnEdge > > > & narrow_regions ) [virtual]`

Uses existing topological entities. Contains generated geometry.

Reimplemented from [Go::ModelQuality](#).

29.129.3.26 `virtual void Go::FaceSetQuality::sfC1Discontinuity ( std::vector< shared_ptr< ftSurface > > & scont_sfs ) [virtual]`

Uses existing topological entities. Existing surfaces available.

Reimplemented from [Go::ModelQuality](#).

```
29.129.3.27 virtual void Go::FaceSetQuality::sfCurvatureRadius (std::vector< std::pair< shared_ptr< ftPoint >, double > > & small_curv_rad, std::pair< shared_ptr< ftPoint >, double > & minimum_curv_rad) [virtual]
```

Generates new points, pointers to existing surfaces.

Reimplemented from [Go::ModelQuality](#).

```
29.129.3.28 virtual void Go::FaceSetQuality::sfG1Discontinuity (std::vector< shared_ptr< ftSurface > > & discont_sfs) [virtual]
```

Uses existing topological entities. Existing surfaces available.

Reimplemented from [Go::ModelQuality](#).

```
29.129.3.29 virtual void Go::FaceSetQuality::sliverSurfaces (std::vector< shared_ptr< ParamSurface > > & sliver_sfs, double thickness, double factor = 2.0) [virtual]
```

Uses existing geometry.

Reimplemented from [Go::ModelQuality](#).

```
29.129.3.30 virtual void Go::FaceSetQuality::vanishingCurveTangent (std::vector< shared_ptr< PointOnCurve > > & sing_points, std::vector< std::pair< shared_ptr< PointOnCurve >, shared_ptr< PointOnCurve > > > & sing_curves) [virtual]
```

Contains generated points and curves fetched from topological entities.

Reimplemented from [Go::ModelQuality](#).

```
29.129.3.31 virtual void Go::FaceSetQuality::vanishingSurfaceNormal (std::vector< shared_ptr< ftPoint > > & singular_points, std::vector< shared_ptr< ftCurve > > & singular_curves) [virtual]
```

Returns ftPoints and ftCurves that will contain generated geometry.

Reimplemented from [Go::ModelQuality](#).

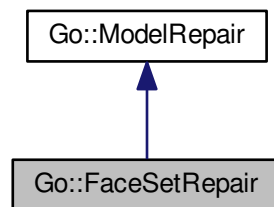
The documentation for this class was generated from the following file:

- [qualitymodule/include/GoTools/qualitymodule/FaceSetQuality.h](#)

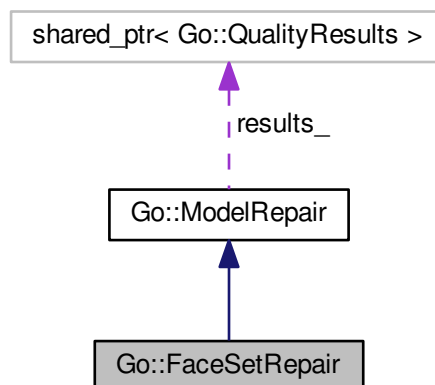
## 29.130 Go::FaceSetRepair Class Reference

```
#include <FaceSetRepair.h>
```

Inheritance diagram for Go::FaceSetRepair:



Collaboration diagram for Go::FaceSetRepair:



### Public Member Functions

- [FaceSetRepair](#) (shared\_ptr< [SurfaceModel](#) > sfmodel)  
*Constructor.*
- [FaceSetRepair](#) (shared\_ptr< [FaceSetQuality](#) > quality)  
*Constructor, given quality check class.*
- [~FaceSetRepair](#) ()  
*Destructor.*
- virtual void [mendGaps](#) ()  
*Remove gaps in the model.*

- virtual void [identicalAndEmbeddedFaces](#) ()  
*Remove identical and embedded faces.*
- virtual void [identicalVertices](#) ()  
*Handle identical vertices.*
- virtual void [consistentFaceNormal](#) ()  
*Handle inconsistent face normals.*
- virtual void [optimizeVertexPosition](#) ()
- virtual void [mendEdgeDistance](#) ()
- shared\_ptr< [SurfaceModel](#) > [getAssociatedSfModel](#) ()

## Additional Inherited Members

### 29.130.1 Detailed Description

Definition at line 53 of file FaceSetRepair.h.

### 29.130.2 Constructor & Destructor Documentation

29.130.2.1 `Go::FaceSetRepair::FaceSetRepair ( shared_ptr< SurfaceModel > sfmodel )`

Constructor.

29.130.2.2 `Go::FaceSetRepair::FaceSetRepair ( shared_ptr< FaceSetQuality > quality )`

Constructor, given quality check class.

29.130.2.3 `Go::FaceSetRepair::~~FaceSetRepair ( )`

Destructor.

### 29.130.3 Member Function Documentation

29.130.3.1 `virtual void Go::FaceSetRepair::consistentFaceNormal ( ) [virtual]`

[Handle](#) inconsistent face normals.

Implements [Go::ModelRepair](#).

29.130.3.2 `shared_ptr<SurfaceModel> Go::FaceSetRepair::getAssociatedSfModel ( ) [inline]`

Definition at line 82 of file FaceSetRepair.h.

29.130.3.3 virtual void Go::FaceSetRepair::identicalAndEmbeddedFaces ( ) [virtual]

Remove identical and embedded faces.

Implements [Go::ModelRepair](#).

29.130.3.4 virtual void Go::FaceSetRepair::identicalVertices ( ) [virtual]

[Handle](#) identical vertices.

Implements [Go::ModelRepair](#).

29.130.3.5 virtual void Go::FaceSetRepair::mendEdgeDistance ( ) [virtual]

Implements [Go::ModelRepair](#).

29.130.3.6 virtual void Go::FaceSetRepair::mendGaps ( ) [virtual]

Remove gaps in the model.

Implements [Go::ModelRepair](#).

29.130.3.7 virtual void Go::FaceSetRepair::optimizeVertexPosition ( ) [virtual]

Implements [Go::ModelRepair](#).

The documentation for this class was generated from the following file:

- [qualitymodule/include/GoTools/qualitymodule/FaceSetRepair.h](#)

## 29.131 Go::Factorial< N > Struct Template Reference

Compile-time factorial calculations.

```
#include <Factorial.h>
```

### Public Types

- enum { [value](#) = N \* Factorial<N-1>::value }

### 29.131.1 Detailed Description

```
template<int N>
struct Go::Factorial< N >
```

Compile-time factorial calculations.

Calculate the factorial of a given integer N.

Definition at line 52 of file Factorial.h.

### 29.131.2 Member Enumeration Documentation

#### 29.131.2.1 template<int N> anonymous enum

Enumerator

**value**

Definition at line 53 of file Factorial.h.

The documentation for this struct was generated from the following file:

- [gotools-core/include/GoTools/Utils/Factorial.h](#)

## 29.132 Go::Factorial< 1 > Struct Template Reference

```
#include <Factorial.h>
```

### Public Types

- enum { [value](#) = 1 }

### 29.132.1 Detailed Description

```
template<>
struct Go::Factorial< 1 >
```

Definition at line 59 of file Factorial.h.

## 29.132.2 Member Enumeration Documentation

### 29.132.2.1 anonymous enum

Enumerator

***value***

Definition at line 60 of file Factorial.h.

The documentation for this struct was generated from the following file:

- [gootools-core/include/GoTools/Utils/Factorial.h](#)

## 29.133 Go::Factory Class Reference

```
#include <Factory.h>
```

### Public Member Functions

- [~Factory](#) ()
- void [registerClass](#) ([ClassType](#) class\_type, [Creator](#) \*c)

### Static Public Member Functions

- static [GeomObject](#) \* [createObject](#) ([ClassType](#) class\_type)
- static [Factory](#) \* [globalFactory](#) ()

### 29.133.1 Detailed Description

This is the [Factory](#) for creating instances of [GeomObjects](#) of type requested by the user. There will be one, global [Factory](#) object, called `Go::global_factory_`. However, the user will not have to deal with this directly. The user need only interact with the [Factory](#) using the static member function '[createObject\(ClassType\)](#)' and the template [Register\(\)](#) function (or [Registrar](#) class, if the compiler misbehaves. In practice, this means calling '[GoTools::init\(\)](#)' early in the code.

Definition at line 83 of file [Factory.h](#).

### 29.133.2 Constructor & Destructor Documentation

#### 29.133.2.1 `Go::Factory::~~Factory ( )` `[inline]`

Definition at line 86 of file [Factory.h](#).

### 29.133.3 Member Function Documentation

#### 29.133.3.1 `static GeomObject* Go::Factory::createObject ( ClassType class_type )` `[inline]`, `[static]`

Makes a new [GeomObject](#) instance of the specified [ClassType](#) and returns a pointer to it. The user assumes ownership over the created object.

## Parameters

|                   |                                                                       |
|-------------------|-----------------------------------------------------------------------|
| <i>class_type</i> | the class type of the object that the user wants to have constructed. |
|-------------------|-----------------------------------------------------------------------|

Definition at line 97 of file Factory.h.

**29.133.3.2** `static Factory* Go::Factory::globalFactory( ) [inline],[static]`

This function returns a pointer to the unique, global [Factory](#).

## Returns

pointer to global [Factory](#).

Definition at line 118 of file Factory.h.

**29.133.3.3** `void Go::Factory::registerClass ( ClassType class_type, Creator * c ) [inline]`

Register a ClassType with the [Factory](#). This amounts to provide the [Factory](#) with the [Creator](#) object that is used to generate a new [GeomObject](#) of type ClassType. Usually, the user would not want to call this function directly, but through the (template) function [Register\(\)](#) or by using a [Registrar](#)

## Parameters

|                   |                                                                                                                                  |
|-------------------|----------------------------------------------------------------------------------------------------------------------------------|
| <i>class_type</i> | the ClassType for which we want the <a href="#">Factory</a> to associate a particular <a href="#">Creator</a> .                  |
| <i>c</i>          | pointer to the <a href="#">Creator</a> object to be used by the <a href="#">Factory</a> when creating objects of type ClassType. |

Definition at line 111 of file Factory.h.

The documentation for this class was generated from the following file:

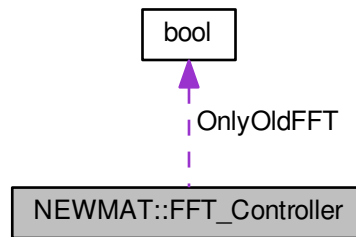
- [gotools-core/include/GoTools/geometry/Factory.h](#)

## 29.134 NEWMAT::FFT\_Controller Class Reference

```
#include <newmatap.h>
```



Collaboration diagram for NEWMAT::FFT\_Controller:



### Static Public Member Functions

- static `bool ar_1d_ft` (int *PTS*, `Real *X`, `Real *Y`)
- static `bool CanFactor` (int *PTS*)

### Static Public Attributes

- static `bool OnlyOldFFT`

#### 29.134.1 Detailed Description

Definition at line 81 of file `newmatap.h`.

#### 29.134.2 Member Function Documentation

**29.134.2.1** `bool FFT_Controller::ar_1d_ft ( int PTS, Real * X, Real * Y )` `[static]`

Definition at line 145 of file `newfft.cpp`.

**29.134.2.2** `bool FFT_Controller::CanFactor ( int PTS )` `[static]`

Definition at line 987 of file `newfft.cpp`.

#### 29.134.3 Member Data Documentation

**29.134.3.1** `bool FFT_Controller::OnlyOldFFT` `[static]`

Definition at line 84 of file `newmatap.h`.

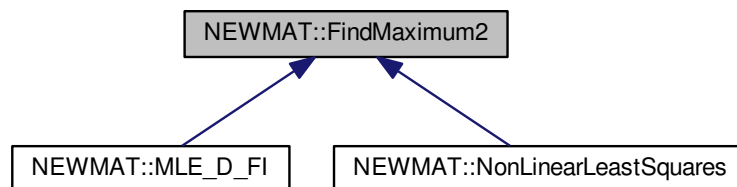
The documentation for this class was generated from the following files:

- `newmat/include/newmatap.h`
- `newmat/src/newfft.cpp`

## 29.135 NEWMAT::FindMaximum2 Class Reference

```
#include <newmatnl.h>
```

Inheritance diagram for NEWMAT::FindMaximum2:



### Public Member Functions

- void [Fit](#) ([ColumnVector](#) &, int)
- virtual [~FindMaximum2](#) ()

#### 29.135.1 Detailed Description

Definition at line 171 of file newmatnl.h.

#### 29.135.2 Constructor & Destructor Documentation

29.135.2.1 virtual NEWMAT::FindMaximum2::~~FindMaximum2 ( ) [inline],[virtual]

Definition at line 178 of file newmatnl.h.

#### 29.135.3 Member Function Documentation

29.135.3.1 void FindMaximum2::Fit ( [ColumnVector](#) & *Theta*, int *n\_it* )

Definition at line 18 of file newmatnl.cpp.

The documentation for this class was generated from the following files:

- newmat/include/newmatnl.h
- newmat/src/newmatnl.cpp

## 29.136 NEWMAT::FloatingPointPrecision Class Reference

Floating point precision (type double).

```
#include <precisio.h>
```

### Static Public Member Functions

- static int [Dig](#) ()
- static [Real Epsilon](#) ()
- static int [Mantissa](#) ()
- static [Real Maximum](#) ()
- static int [MaximumDecimalExponent](#) ()
- static int [MaximumExponent](#) ()
- static [Real LnMaximum](#) ()
- static [Real Minimum](#) ()
- static int [MinimumDecimalExponent](#) ()
- static int [MinimumExponent](#) ()
- static [Real LnMinimum](#) ()
- static int [Radix](#) ()
- static int [Rounds](#) ()

### 29.136.1 Detailed Description

Floating point precision (type double).

Definition at line 136 of file precisio.h.

### 29.136.2 Member Function Documentation

**29.136.2.1** static int NEWMAT::FloatingPointPrecision::Dig ( ) [inline],[static]

Definition at line 140 of file precisio.h.

**29.136.2.2** static [Real](#) NEWMAT::FloatingPointPrecision::Epsilon ( ) [inline],[static]

Definition at line 143 of file precisio.h.

**29.136.2.3** static [Real](#) NEWMAT::FloatingPointPrecision::LnMaximum ( ) [inline],[static]

Definition at line 158 of file precisio.h.

**29.136.2.4** static [Real](#) NEWMAT::FloatingPointPrecision::LnMinimum ( ) [inline],[static]

Definition at line 176 of file precisio.h.

29.136.2.5 `static int NEWMAT::FloatingPointPrecision::Mantissa ( ) [inline],[static]`

Definition at line 146 of file `precisio.h`.

29.136.2.6 `static Real NEWMAT::FloatingPointPrecision::Maximum ( ) [inline],[static]`

Definition at line 149 of file `precisio.h`.

29.136.2.7 `static int NEWMAT::FloatingPointPrecision::MaximumDecimalExponent ( ) [inline],[static]`

Definition at line 152 of file `precisio.h`.

29.136.2.8 `static int NEWMAT::FloatingPointPrecision::MaximumExponent ( ) [inline],[static]`

Definition at line 155 of file `precisio.h`.

29.136.2.9 `static Real NEWMAT::FloatingPointPrecision::Minimum ( ) [inline],[static]`

Definition at line 161 of file `precisio.h`.

29.136.2.10 `static int NEWMAT::FloatingPointPrecision::MinimumDecimalExponent ( ) [inline],[static]`

Definition at line 170 of file `precisio.h`.

29.136.2.11 `static int NEWMAT::FloatingPointPrecision::MinimumExponent ( ) [inline],[static]`

Definition at line 173 of file `precisio.h`.

29.136.2.12 `static int NEWMAT::FloatingPointPrecision::Radix ( ) [inline],[static]`

Definition at line 180 of file `precisio.h`.

29.136.2.13 `static int NEWMAT::FloatingPointPrecision::Rounds ( ) [inline],[static]`

Definition at line 183 of file `precisio.h`.

The documentation for this class was generated from the following file:

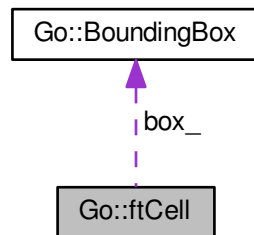
- [newmat/include/precisio.h](#)

## 29.137 Go::ftCell Class Reference

Helper class handling one of the cells in class [CellDivision](#).

```
#include <CellDivision.h>
```

Collaboration diagram for Go::ftCell:



### Public Member Functions

- [ftCell](#) ()
- [~ftCell](#) ()
- void [setBox](#) (const [BoundingBox](#) &box)
- void [addFace](#) (ftSurface \*f)
- void [addFaceBox](#) ([BoundingBox](#) &box)
- const [BoundingBox](#) & [box](#) () const
- int [num\\_faces](#) () const
- ftSurface \* [face](#) (int i) const
- const [BoundingBox](#) & [faceBox](#) (int i) const
- void [removeFace](#) (ftSurface \*face)

### Protected Attributes

- std::vector< ftSurface \* > [faces\\_](#)
- std::vector< [BoundingBox](#) > [face\\_boxes\\_](#)
- [BoundingBox](#) [box\\_](#)

#### 29.137.1 Detailed Description

Helper class handling one of the cells in class [CellDivision](#).

Definition at line 55 of file [CellDivision.h](#).

## 29.137.2 Constructor & Destructor Documentation

29.137.2.1 `Go::ftCell::ftCell ( )` [inline]

Definition at line 62 of file CellDivision.h.

29.137.2.2 `Go::ftCell::~~ftCell ( )` [inline]

Definition at line 63 of file CellDivision.h.

## 29.137.3 Member Function Documentation

29.137.3.1 `void Go::ftCell::addFace ( ftSurface * f )` [inline]

Definition at line 66 of file CellDivision.h.

29.137.3.2 `void Go::ftCell::addFaceBox ( BoundingBox & box )` [inline]

Definition at line 68 of file CellDivision.h.

29.137.3.3 `const BoundingBox& Go::ftCell::box ( ) const` [inline]

Definition at line 70 of file CellDivision.h.

29.137.3.4 `ftSurface* Go::ftCell::face ( int i ) const` [inline]

Definition at line 74 of file CellDivision.h.

29.137.3.5 `const BoundingBox& Go::ftCell::faceBox ( int i ) const` [inline]

Definition at line 76 of file CellDivision.h.

29.137.3.6 `int Go::ftCell::num_faces ( ) const` [inline]

Definition at line 72 of file CellDivision.h.

29.137.3.7 `void Go::ftCell::removeFace ( ftSurface * face )` [inline]

Definition at line 79 of file CellDivision.h.

29.137.3.8 void Go::ftCell::setBox ( const BoundingBox & box ) [inline]

Definition at line 64 of file CellDivision.h.

## 29.137.4 Member Data Documentation

29.137.4.1 BoundingBox Go::ftCell::box\_ [protected]

Definition at line 60 of file CellDivision.h.

29.137.4.2 std::vector<BoundingBox> Go::ftCell::face\_boxes\_ [protected]

Definition at line 59 of file CellDivision.h.

29.137.4.3 std::vector<ftSurface\*> Go::ftCell::faces\_ [protected]

Definition at line 58 of file CellDivision.h.

The documentation for this class was generated from the following file:

- [compositemodel/include/GoTools/compositemodel/CellDivision.h](#)

## 29.138 Go::ftCellInfo Class Reference

Function object for sorting ftCells.

```
#include <CellDivision.h>
```

### Public Member Functions

- [ftCellInfo \(\)](#)
- [ftCellInfo \(int i, double d\)](#)
- [bool operator< \(const ftCellInfo &ci\) const](#)

### Public Attributes

- int [index\\_](#)
- double [dist\\_](#)

## 29.138.1 Detailed Description

Function object for sorting ftCells.

Definition at line 90 of file CellDivision.h.

## 29.138.2 Constructor & Destructor Documentation

29.138.2.1 `Go::ftCellInfo::ftCellInfo ( )` `[inline]`

Definition at line 95 of file CellDivision.h.

29.138.2.2 `Go::ftCellInfo::ftCellInfo ( int i, double d )` `[inline]`

Definition at line 96 of file CellDivision.h.

## 29.138.3 Member Function Documentation

29.138.3.1 `bool Go::ftCellInfo::operator< ( const ftCellInfo & ci ) const` `[inline]`

Definition at line 97 of file CellDivision.h.

## 29.138.4 Member Data Documentation

29.138.4.1 `double Go::ftCellInfo::dist_`

Definition at line 94 of file CellDivision.h.

29.138.4.2 `int Go::ftCellInfo::index_`

Definition at line 93 of file CellDivision.h.

The documentation for this class was generated from the following file:

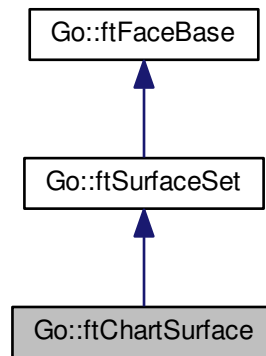
- `compositemodel/include/GoTools/compositemodel/CellDivision.h`



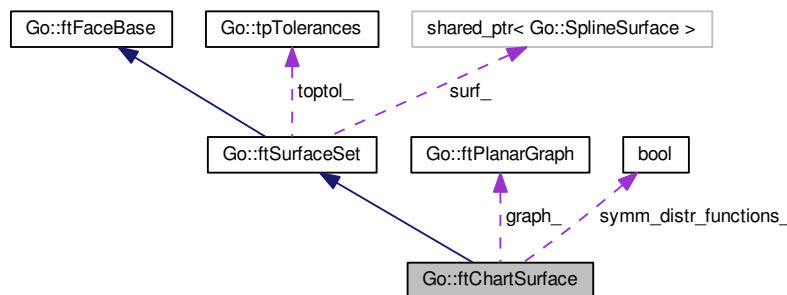
## 29.139 Go::ftChartSurface Class Reference

```
#include <ftChartSurface.h>
```

Inheritance diagram for Go::ftChartSurface:



Collaboration diagram for Go::ftChartSurface:



### Public Member Functions

- `ftChartSurface` (`const` `std::vector`< `shared_ptr`< `ParamSurface` > > &`surfaces`, `tpTolerances` &`topeps`, `double` `aproxeps`, `double` `curvature_tol`, `int` `m=0`, `int` `n=0`)  
*Constructor.*
- `~ftChartSurface` ()  
*Destructor.*
- `virtual` `std::vector`< `shared_ptr`< `ftEdgeBase` > > `createInitialEdges` (`double` `degenerate_epsilon=DEFAULT_SPACE_EPSILON`, `double` `kink=0.00015`, `bool` `no_split=false`)
- `virtual` `Point` `point` (`double` `u`, `double` `v`) `const`  
*Evaluation and interrogation. Overridden from `ftFaceBase`.*

- [Point](#) [point](#) ([double](#) &u, [double](#) &v, [shared\\_ptr](#)< [ftFaceBase](#) > &face, [double](#) \*seed=NULL, [bool](#) use\_input↔\_face=false) [const](#)
- virtual [Point](#) [normal](#) ([double](#) u, [double](#) v) [const](#)  
*Evaluate surface normal.*
- virtual [ftMessage](#) [createSurf](#) ([double](#) &max\_error, [double](#) &mean\_error)  
*Merge the current surface patches to produce one [SplineSurface](#).*
- virtual void [getError](#) ([double](#) &max\_error, [double](#) &mean\_error)
- virtual [ftTangPriority](#) [getPrioType](#) () [const](#)  
*Priority type.*
- void [setPrioType](#) ([ftTangPriority](#) type)
- void [setGridResolution](#) (int m, int n)
- [bool](#) [gridCreated](#) (int &m, int &n) [const](#)  
*We may also decide to add grid pts along an edge.*
- void [setEdgeScales](#) ([std::vector](#)< [std::pair](#)< [Point](#), [double](#) > > &edge\_scales)
- void [setSymmDistrFunctions](#) ([bool](#) symm\_distr\_functions)
- [ftMessage](#) [prepareGrid](#) ([RotationInfo](#) \*rot\_info=NULL, [ftCurve](#) \*outer\_bd=NULL)
- [ftMessage](#) [createGrid](#) ([RotationInfo](#) \*rot\_info=NULL, [ftCurve](#) \*outer\_bd=NULL)
- [std::vector](#)< [shared\\_ptr](#)< [ftSamplePoint](#) > > [getEdgeGrid](#) (int edge\_ind)
- void [writeGrid](#) ([std::ostream](#) &os) [const](#)
- void [writeDebugGrid](#) ([std::ostream](#) &os) [const](#)  
*Write to file [LineCloud](#) describing grid. Still awaiting exchange format.*
- [vector](#)< [shared\\_ptr](#)< [FaceConnectivity](#)< [ftEdgeBase](#) > > > [getInnerEdgeCont](#) () [const](#)

### Protected Member Functions

- virtual [bool](#) [sampleOnlyEdges](#) () [const](#)  
*Approximate only the edges in the group of surfaces.*

### Protected Attributes

- [ftTangPriority](#) prio\_type\_
- [double](#) curvature\_tol\_
- [ftPlanarGraph](#) graph\_
- [double](#) maxerror\_
- [double](#) meanerror\_
- int m\_
- int n\_
- [std::vector](#)< [shared\\_ptr](#)< [SplineCurve](#) > > grid\_distr\_functions\_
- [std::vector](#)< [std::pair](#)< [Point](#), [double](#) > > edge\_scales\_
- [bool](#) symm\_distr\_functions\_
- [std::vector](#)< [bool](#) > secn\_distr\_
- [std::vector](#)< [std::vector](#)< [shared\\_ptr](#)< [ftSurfaceSetPoint](#) > > > grid\_pts\_

### 29.139.1 Detailed Description

[ftChartSurface](#) - Short description. Detailed description.

Definition at line 52 of file [ftChartSurface.h](#).

## 29.139.2 Constructor & Destructor Documentation

29.139.2.1 `Go::ftChartSurface::ftChartSurface ( const std::vector< shared_ptr< ParamSurface > > & surfaces, tpTolerances & topeps, double approxeps, double curvature_tol, int m = 0, int n = 0 )`

Constructor.

29.139.2.2 `Go::ftChartSurface::~~ftChartSurface ( )`

Destructor.

## 29.139.3 Member Function Documentation

29.139.3.1 `ftMessage Go::ftChartSurface::createGrid ( RotationInfo * rot_info = NULL, ftCurve * outer_bd = NULL )`

User may choose to let `bd_edges` of all sfs in the topology be in correspondence through a rotation of the entire set. For all members of topology it is expected that both [createSurf\(\)](#) & [prepareGrid\(\)](#) have been called.

29.139.3.2 `virtual std::vector<shared_ptr<ftEdgeBase> > Go::ftChartSurface::createInitialEdges ( double degenerate_epsilon = DEFAULT_SPACE_EPSILON, double kink = 0.00015, bool no_split = false ) [virtual]`

Warning! Assumes the return edges are taken care of, i.e. stored for further use. Otherwise `first_edges_` will be invalid! To be called when topology is set up; essentially performed from constructor.

Implements [Go::ftSurfaceSet](#).

29.139.3.3 `virtual ftMessage Go::ftChartSurface::createSurf ( double & max_error, double & mean_error ) [virtual]`

Merge the current surface patches to produce one [SplineSurface](#).

Implements [Go::ftSurfaceSet](#).

29.139.3.4 `std::vector<shared_ptr<ftSamplePoint> > Go::ftChartSurface::getEdgeGrid ( int edge_ind )`

The created grid is returned (copied). The pts are returned in ccw order. Ordering is: b, t, l, r

29.139.3.5 `virtual void Go::ftChartSurface::getError ( double & max_error, double & mean_error ) [virtual]`

Fetch the error due to approximations in `createSurf`. Not applicable in normal configurations

Implements [Go::ftSurfaceSet](#).

29.139.3.6 `vector<shared_ptr<FaceConnectivity<ftEdgeBase>>> Go::ftChartSurface::getInnerEdgeCont ( ) const`

29.139.3.7 `virtual ftTangPriority Go::ftChartSurface::getPrioType ( ) const [virtual]`

Priority type.

Implements [Go::ftSurfaceSet](#).

29.139.3.8 `bool Go::ftChartSurface::gridCreated ( int & m, int & n ) const`

We may also decide to add grid pts along an edge.

29.139.3.9 `virtual Point Go::ftChartSurface::normal ( double u, double v ) const [virtual]`

Evaluate surface normal.

Implements [Go::ftSurfaceSet](#).

29.139.3.10 `virtual Point Go::ftChartSurface::point ( double u, double v ) const [virtual]`

Evaluation and interrogation. Overridden from [ftFaceBase](#).

Implements [Go::ftSurfaceSet](#).

29.139.3.11 `Point Go::ftChartSurface::point ( double & u, double & v, shared_ptr< ftFaceBase > & face, double * seed = NULL, bool use_input_face = false ) const`

Given par values in surf\_, return corresponding par values in faces\_[i]. If use\_input\_face == true, we do not search in graph but go directly to face.

29.139.3.12 `ftMessage Go::ftChartSurface::prepareGrid ( RotationInfo * rot_info = NULL, ftCurve * outer_bd = NULL )`

Flicking on grid\_distr\_functions\_ so that they both agree along and towards common bd. For all members of topology it is expected that both [createSurf\(\)](#) has been called.

29.139.3.13 `virtual bool Go::ftChartSurface::sampleOnlyEdges ( ) const [protected],[virtual]`

Approximate only the edges in the group of surfaces.

Implements [Go::ftSurfaceSet](#).

29.139.3.14 void Go::ftChartSurface::setEdgeScales ( std::vector< std::pair< Point, double > > & edge\_scales )

29.139.3.15 void Go::ftChartSurface::setGridResolution ( int m, int n )

29.139.3.16 void Go::ftChartSurface::setPrioType ( ftTangPriority type )

29.139.3.17 void Go::ftChartSurface::setSymmDistrFunctions ( bool symm\_distr\_functions )

29.139.3.18 void Go::ftChartSurface::writeDebugGrid ( std::ostream & os ) const

Write to file [LineCloud](#) describing grid. Still awaiting exchange format.

29.139.3.19 void Go::ftChartSurface::writeGrid ( std::ostream & os ) const

## 29.139.4 Member Data Documentation

29.139.4.1 double Go::ftChartSurface::curvature\_tol\_ [protected]

Definition at line 127 of file ftChartSurface.h.

29.139.4.2 std::vector<std::pair<Point, double> > Go::ftChartSurface::edge\_scales\_ [protected]

Definition at line 143 of file ftChartSurface.h.

29.139.4.3 ftPlanarGraph Go::ftChartSurface::graph\_ [protected]

Definition at line 128 of file ftChartSurface.h.

29.139.4.4 std::vector<shared\_ptr<SplineCurve> > Go::ftChartSurface::grid\_distr\_functions\_ [protected]

Definition at line 139 of file ftChartSurface.h.

29.139.4.5 std::vector<std::vector<shared\_ptr<ftSurfaceSetPoint> > > Go::ftChartSurface::grid\_pts\_  
[protected]

Definition at line 148 of file ftChartSurface.h.

29.139.4.6 int Go::ftChartSurface::m\_ [protected]

Definition at line 136 of file ftChartSurface.h.

29.139.4.7 **double** `Go::ftChartSurface::maxerror_` [protected]

Definition at line 130 of file `ftChartSurface.h`.

29.139.4.8 **double** `Go::ftChartSurface::meanerror_` [protected]

Definition at line 131 of file `ftChartSurface.h`.

29.139.4.9 **int** `Go::ftChartSurface::n_` [protected]

Definition at line 137 of file `ftChartSurface.h`.

29.139.4.10 **ftTangPriority** `Go::ftChartSurface::prio_type_` [protected]

Definition at line 122 of file `ftChartSurface.h`.

29.139.4.11 **std::vector<bool>** `Go::ftChartSurface::secn_distr_` [protected]

Definition at line 146 of file `ftChartSurface.h`.

29.139.4.12 **bool** `Go::ftChartSurface::symm_distr_functions_` [protected]

Definition at line 145 of file `ftChartSurface.h`.

The documentation for this class was generated from the following file:

- [compositemodel/include/GoTools/compositemodel/ftChartSurface.h](#)

## 29.140 `Go::ftCurve` Class Reference

```
#include <ftCurve.h>
```

## Public Member Functions

- [ftCurve](#) ()  
*Constructor.*
- [ftCurve](#) ([ftCurveType](#) t)
- [ftCurve](#) & [operator+=](#) ([const ftCurve](#) &cv)  
*Append curve.*
- void [appendCurve](#) ([const ftCurve](#) &cv)  
*Append the ftCurve cv after this curve.*
- void [prependCurve](#) ([const ftCurve](#) &cv)  
*Place the ftCurve cv before this curve.*
- void [appendSegment](#) ([const ftCurveSegment](#) &seg)  
*A new segment after this curve.*
- void [prependSegment](#) ([const ftCurveSegment](#) &seg)  
*A new segment prior to this curve.*
- void [joinSegments](#) ([double](#) gap\_tol, [double](#) neighbour\_tol, [double](#) kink\_tol, [double](#) bend\_tol)
- void [orientSegments](#) ([double](#) neighbour\_tol, [bool](#) assume\_manifold=true)
- void [reverse](#) ()  
*Reverse the curve and maintain correct joint types between segments.*
- void [chopOff](#) ([const BoundingBox](#) &box)  
*Remove the parts of this curve lying outside the specified bounding box.*
- int [numSegments](#) () [const](#)  
*Number of segments in ftCurve.*
- [double](#) [startOfSegment](#) (int segment) [const](#)  
*Start parameter of segment number segment.*
- [double](#) [endOfSegment](#) (int segment) [const](#)  
*End parameter of segment number segment.*
- [const ftCurveSegment](#) & [segment](#) (int segment) [const](#)  
*Fetch segment number segment.*
- [ftCurveType](#) [curveType](#) () [const](#)
- [tpJointType](#) [jointAfter](#) (int segment) [const](#)
- [tpJointType](#) [worstJointType](#) () [const](#)  
*Least continuous joint between segments.*
- int [numDisjointSubcurves](#) () [const](#)  
*Number of disjoint subcurves.*
- void [point](#) ([double](#) t, int segment, [Point](#) &pt) [const](#)  
*Compute position given segment number and parameter within segment.*
- void [tangent](#) ([double](#) t, int segment, [Point](#) &tan) [const](#)  
*Compute tangent given segment number and parameter within segment.*
- void [normal](#) ([double](#) t, int segment, int side, [Point](#) &normal, [double](#) eps) [const](#)
- [double](#) [arcLength](#) ([double](#) t1, int seg1, [double](#) t2, int seg2) [const](#)  
*Curve length.*
- void [reparametrize](#) ([double](#) eps\_go=1.0e-6)  
*Reparametrization.*
- void [tesselate](#) ([std::vector](#)< [shared\\_ptr](#)< [LineStrip](#) > > &meshes) [const](#)  
*Tesselation with default number of line segments.*
- void [tesselate](#) (int resolution, [std::vector](#)< [shared\\_ptr](#)< [LineStrip](#) > > &meshes) [const](#)  
*Tesselation, number of line segments for each ftCurveSegment given.*
- void [tesselate](#) ([double](#) density, [std::vector](#)< [shared\\_ptr](#)< [LineStrip](#) > > &meshes) [const](#)
- void [write](#) ([std::ostream](#) &os) [const](#)  
*Debug.*
- void [writeSpaceCurve](#) ([std::ostream](#) &os) [const](#)  
*Debug.*
- void [deleteSpaceCurveRepresentation](#) ()  
*Debug.*

## Protected Attributes

- [ftCurveType type\\_](#)
- `std::vector< ftCurveSegment > segments_`

### 29.140.1 Detailed Description

[ftCurve](#) - contains a number of (possibly disjoint) curve segments. The [ftCurve](#) class has interfaces for most operations (such as evaluation) that one might want to perform on them

#### Author

Atgeirr F Rasmussen [atgeirr@sintef.no](mailto:atgeirr@sintef.no)

Definition at line 236 of file [ftCurve.h](#).

### 29.140.2 Constructor & Destructor Documentation

#### 29.140.2.1 `Go::ftCurve::ftCurve ( )` `[inline]`

Constructor.

Definition at line 240 of file [ftCurve.h](#).

#### 29.140.2.2 `Go::ftCurve::ftCurve ( ftCurveType t )` `[inline]`

Constructor of curve given curve type -1 = no specified type, 0 = intersection curve, 1 = edge curve, 2 = kink curve, 3 = corner curve, 4 = gap curve, 5 = singularity curve

Definition at line 245 of file [ftCurve.h](#).

### 29.140.3 Member Function Documentation

#### 29.140.3.1 `void Go::ftCurve::appendCurve ( const ftCurve & cv )` `[inline]`

Append the [ftCurve](#) `cv` after this curve.

Definition at line 360 of file [ftCurve.h](#).

#### 29.140.3.2 `void Go::ftCurve::appendSegment ( const ftCurveSegment & seg )` `[inline]`

A new segment after this curve.

Definition at line 378 of file [ftCurve.h](#).



29.140.3.3 `double Go::ftCurve::arcLength ( double t1, int seg1, double t2, int seg2 ) const`

Curve length.

29.140.3.4 `void Go::ftCurve::chopOff ( const BoundingBox & box ) [inline]`

Remove the parts of this curve lying outside the specified bounding box.

Definition at line 408 of file ftCurve.h.

29.140.3.5 `ftCurveType Go::ftCurve::curveType ( ) const [inline]`

Type of curve -1 = no specified type, 0 = intersection curve, 1 = edge curve, 2 = kink curve, 3 = corner curve, 4 = gap curve, 5 = singularity curve

Definition at line 453 of file ftCurve.h.

29.140.3.6 `void Go::ftCurve::deleteSpaceCurveRepresentation ( ) [inline]`

Debug.

Definition at line 339 of file ftCurve.h.

29.140.3.7 `double Go::ftCurve::endOfSegment ( int segment ) const [inline]`

End parameter of segment number segment.

Definition at line 439 of file ftCurve.h.

29.140.3.8 `void Go::ftCurve::joinSegments ( double gap_tol, double neighbour_tol, double kink_tol, double bend_tol )`

Flag the connections between segments according to their continuity. This function assumes that the segments have already been sorted and oriented. Sorting and orienting of segments can be done with [orientSegments\(\)](#)

29.140.3.9 `tpJointType Go::ftCurve::jointAfter ( int segment ) const [inline]`

Type of joint between segment number segment and the next segment 0 = g1 continuity, 1 = not quite g1, 2 = g0 continuity, 3 = gap, 4 = discontinuous, 5 = last segment

Definition at line 460 of file ftCurve.h.

29.140.3.10 `void Go::ftCurve::normal ( double t, int segment, int side, Point & normal, double eps ) const [inline]`

Compute normal given segment number, parameter within segment and the number of the associated face

Definition at line 503 of file ftCurve.h.

29.140.3.11 `int Go::ftCurve::numDisjointSubcurves ( ) const [inline]`

Number of disjoint subcurves.

Definition at line 478 of file ftCurve.h.

29.140.3.12 `int Go::ftCurve::numSegments ( ) const [inline]`

Number of segments in [ftCurve](#).

Definition at line 425 of file ftCurve.h.

29.140.3.13 `ftCurve & Go::ftCurve::operator+=( const ftCurve & cv ) [inline]`

Append curve.

Definition at line 352 of file ftCurve.h.

29.140.3.14 `void Go::ftCurve::orientSegments ( double neighbour_tol, bool assume_manifold = true )`

Assures that the segments are consistently oriented and sorted such that the end point of segment 'k' is coincident (within the specified tolerance) with the start point of segment 'k+1' if those segments are connected. If we cannot a priori be sure that our curve is describing an 1-manifold (no 'branches' on the curve), the last argument should be set to 'false'. This may effect performance, though not severely unless the number of segments is very high.

29.140.3.15 `void Go::ftCurve::point ( double t, int segment, Point & pt ) const [inline]`

Compute position given segment number and parameter within segment.

Definition at line 489 of file ftCurve.h.

29.140.3.16 `void Go::ftCurve::prependCurve ( const ftCurve & cv ) [inline]`

Place the [ftCurve](#) cv before this curve.

Definition at line 369 of file ftCurve.h.

29.140.3.17 `void Go::ftCurve::prependSegment ( const ftCurveSegment & seg ) [inline]`

A new segment prior to this curve.

Definition at line 385 of file ftCurve.h.

29.140.3.18 void Go::ftCurve::reparametrize ( double *eps\_go* = 1.0e-6 )

Reparametrization.

The *eps\_go* argument is used as a tolerance if ever this function has to regenerate the spacecurve from its underlying surface and parametric curve

29.140.3.19 void Go::ftCurve::reverse ( ) [inline]

Reverse the curve and maintain correct joint types between segments.

Definition at line 392 of file ftCurve.h.

29.140.3.20 const ftCurveSegment & Go::ftCurve::segment ( int *segment* ) const [inline]

Fetch segment number segment.

Definition at line 446 of file ftCurve.h.

29.140.3.21 double Go::ftCurve::startOfSegment ( int *segment* ) const [inline]

Start parameter of segment number segment.

Definition at line 432 of file ftCurve.h.

29.140.3.22 void Go::ftCurve::tangent ( double *t*, int *segment*, Point & *tan* ) const [inline]

Compute tangent given segment number and parameter within segment.

Definition at line 496 of file ftCurve.h.

29.140.3.23 void Go::ftCurve::tessellate ( std::vector< shared\_ptr< LineStrip > > & *meshes* ) const

Tessellation with default number of line segments.

29.140.3.24 void Go::ftCurve::tessellate ( int *resolution*, std::vector< shared\_ptr< LineStrip > > & *meshes* ) const

Tessellation, number of line segments for each [ftCurveSegment](#) given.

29.140.3.25 void Go::ftCurve::tessellate ( double *density*, std::vector< shared\_ptr< LineStrip > > & *meshes* ) const

Tessellation, number of line segments for each [ftCurveSegment](#) depends on the length of the segment and the given density

29.140.3.26 `tpJointType Go::ftCurve::worstJointType ( ) const [inline]`

Least continuous joint between segments.

Definition at line 467 of file `ftCurve.h`.

29.140.3.27 `void Go::ftCurve::write ( std::ostream & os ) const`

Debug.

29.140.3.28 `void Go::ftCurve::writeSpaceCurve ( std::ostream & os ) const`

Debug.

## 29.140.4 Member Data Documentation

29.140.4.1 `std::vector<ftCurveSegment> Go::ftCurve::segments_ [protected]`

Definition at line 347 of file `ftCurve.h`.

29.140.4.2 `ftCurveType Go::ftCurve::type_ [protected]`

Definition at line 346 of file `ftCurve.h`.

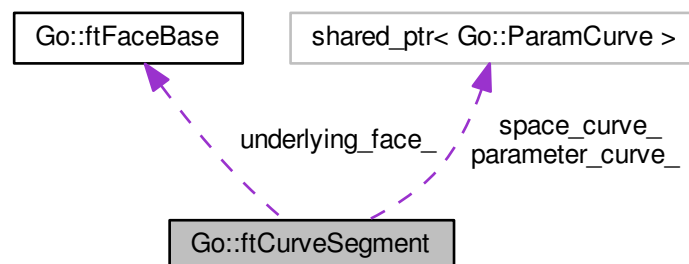
The documentation for this class was generated from the following file:

- `compositemodel/include/GoTools/compositemodel/ftCurve.h`

## 29.141 Go::ftCurveSegment Class Reference

```
#include <ftCurve.h>
```

Collaboration diagram for `Go::ftCurveSegment`:



## Public Member Functions

- [ftCurveSegment](#) ([ftCurveType](#) type, [tpJointType](#) joint, [ftFaceBase](#) \*f0, [ftFaceBase](#) \*f1, [shared\\_ptr](#)< [ParamCurve](#) > paramcv0, [shared\\_ptr](#)< [ParamCurve](#) > paramcv1, [shared\\_ptr](#)< [ParamCurve](#) > spacecv, [double](#) eps\_geo=1.0e-6)
- [ftCurveType](#) [segmentType](#) () [const](#)
- void [setJointAfter](#) ([tpJointType](#) jt)
- [tpJointType](#) [jointAfter](#) () [const](#)  
*Fetch joint type between this curve segment and the next.*
- [ftFaceBase](#) \* [face](#) (int number) [const](#)  
*Fetch face number number associated with this curve segment.*
- [double](#) [startOfSegment](#) () [const](#)  
*Return start parameter of segment.*
- [double](#) [endOfSegment](#) () [const](#)  
*Return end parameter of segment.*
- void [point](#) ([double](#) t, [Point](#) &pt) [const](#)
- void [tangent](#) ([double](#) t, [Point](#) &tan) [const](#)
- void [paramcurvePoint](#) (int number, [double](#) t, [Point](#) &pt) [const](#)
- void [paramcurveTangent](#) (int number, [double](#) t, [Point](#) &pt) [const](#)  
*Tangent to the curve in the parameter domain of face number.*
- void [reverse](#) ()
- [std::vector](#)< [ftCurveSegment](#) > [chopOff](#) ([const](#) [BoundingBox](#) &box, [bool](#) &eraseme)  
*Internal use.*
- [shared\\_ptr](#)< [ParamCurve](#) > [spaceCurve](#) () [const](#)  
*Return the space curve corresponding to this segment.*
- [shared\\_ptr](#)< [ParamCurve](#) > [parameterCurve](#) (int number) [const](#)  
*Return the curve in the parameter domain of face number.*
- [Point](#) [startPoint](#) () [const](#)  
*Start point of segment.*
- [Point](#) [endPoint](#) () [const](#)  
*End point of segment.*
- void [normal](#) ([double](#) t, int side, [Point](#) &normal, [double](#) eps) [const](#)
- [double](#) [arcLength](#) ([double](#) t1, [double](#) t2) [const](#)  
*The arc length of this curve segment limited by the interval [t1, t2].*
- void [reparametrize](#) ([double](#) eps\_go=1.0e-6)
- [shared\\_ptr](#)< [LineStrip](#) > [tessellate](#) (int resolution) [const](#)  
*Tessellate curve segment to return a linear approximation.*
- void [deleteSpaceCurveRepresentation](#) ()  
*Debug purpose.*

## Protected Member Functions

- void [redefineSpaceCurve](#) ([double](#) eps\_go)

## Protected Attributes

- [ftCurveType](#) [segment\\_type\\_](#)
- [tpJointType](#) [joint\\_](#)
- [ftFaceBase](#) \* [underlying\\_face\\_](#) [2]  
*Surface(s) on which curve is lying.*
- [shared\\_ptr](#)< [ParamCurve](#) > [parameter\\_curve\\_](#) [2]  
*Curves in surface domain.*
- [shared\\_ptr](#)< [ParamCurve](#) > [space\\_curve\\_](#)  
*Curve in space.*

### 29.141.1 Detailed Description

[ftCurveSegment](#) - One segment describing a curve lying on one or two surfaces. The curve can be of type intersection, edge curve, kink curve, corner curve or gap curve. The curve is an output of a computation corresponding to the curve type.

Definition at line 78 of file ftCurve.h.

### 29.141.2 Constructor & Destructor Documentation

29.141.2.1 `Go::ftCurveSegment::ftCurveSegment ( ftCurveType type, tpJointType joint, ftFaceBase * f0, ftFaceBase * f1, shared_ptr< ParamCurve > paramcv0, shared_ptr< ParamCurve > paramcv1, shared_ptr< ParamCurve > spacecv, double eps_geo = 1.0e-6 )`

Constructor. The segment owns the nurbscurves, but not the faces

### 29.141.3 Member Function Documentation

29.141.3.1 `double Go::ftCurveSegment::arcLength ( double t1, double t2 ) const`

The arc length of this curve segment limited by the interval [t1, t2].

29.141.3.2 `std::vector<ftCurveSegment> Go::ftCurveSegment::chopOff ( const BoundingBox & box, bool & eraseme )`

Internal use.

29.141.3.3 `void Go::ftCurveSegment::deleteSpaceCurveRepresentation ( ) [inline]`

Debug purpose.

Definition at line 160 of file ftCurve.h.

29.141.3.4 `double Go::ftCurveSegment::endOfSegment ( ) const`

Return end parameter of segment.

29.141.3.5 `Point Go::ftCurveSegment::endPoint ( ) const`

End point of segment.

29.141.3.6 `ftFaceBase * Go::ftCurveSegment::face ( int number ) const [inline]`

Fetch face number number associated with this curve segment.

Definition at line 208 of file ftCurve.h.

29.141.3.7 **tpJointType** Go::ftCurveSegment::jointAfter ( ) const [inline]

Fetch joint type between this curve segment and the next.

Definition at line 201 of file ftCurve.h.

29.141.3.8 void Go::ftCurveSegment::normal ( double t, int side, Point & normal, double eps ) const

The normal of the surface corresponding to this curve.

29.141.3.9 void Go::ftCurveSegment::paramcurvePoint ( int number, double t, Point & pt ) const

Comput point on curve in the parameter domain of face number in the parameter t. If the parameter curve does not exist, the closest point in the specified face to the space curve point corresponding to t is returned.

29.141.3.10 void Go::ftCurveSegment::paramcurveTangent ( int number, double t, Point & pt ) const

Tangent to the curve in the parameter domain of face number.

29.141.3.11 shared\_ptr< ParamCurve > Go::ftCurveSegment::parameterCurve ( int number ) const [inline]

Return the curve in the parameter domain of face number.

Definition at line 222 of file ftCurve.h.

29.141.3.12 void Go::ftCurveSegment::point ( double t, Point & pt ) const

Evaluate a point on the curve.

29.141.3.13 void Go::ftCurveSegment::redefineSpaceCurve ( double eps\_go ) [protected]

29.141.3.14 void Go::ftCurveSegment::reparametrize ( double eps\_go = 1.0e-6 )

The eps\_go argument is used as a tolerance if ever this function has to regenerate the spacecurve from its underlying surface and parametric curve

29.141.3.15 void Go::ftCurveSegment::reverse ( )

Reverse the direction of the curve segment. NB! The joint types will not be correct

**29.141.3.16** `ftCurveType` `Go::ftCurveSegment::segmentType ( ) const` `[inline]`

Fetch info about the type of this curve segment -1 = no specified type, 0 = intersection curve, 1 = edge curve, 2 = kink curve, 3 = corner curve, 4 = gap curve, 5 = singularity curve

Definition at line 187 of file `ftCurve.h`.

**29.141.3.17** `void` `Go::ftCurveSegment::setJointAfter ( tpJointType jt )` `[inline]`

Set joint type between this curve segment and the next 0 = g1 continuity, 1 = not quite g1, 2 = g0 continuity, 3 = gap, 4 = discontinuous, 5 = last segment

Definition at line 194 of file `ftCurve.h`.

**29.141.3.18** `shared_ptr< ParamCurve >` `Go::ftCurveSegment::spaceCurve ( ) const` `[inline]`

Return the space curve corresponding to this segment.

Definition at line 215 of file `ftCurve.h`.

**29.141.3.19** `double` `Go::ftCurveSegment::startOfSegment ( ) const`

Return start parameter of segment.

**29.141.3.20** `Point` `Go::ftCurveSegment::startPoint ( ) const`

Start point of segment.

**29.141.3.21** `void` `Go::ftCurveSegment::tangent ( double t, Point & tan ) const`

Get the curve tangent in a given parameter.

**29.141.3.22** `shared_ptr< LineStrip >` `Go::ftCurveSegment::tessellate ( int resolution ) const`

Tessellate curve segment to return a linear approximation.

## 29.141.4 Member Data Documentation

**29.141.4.1** `tpJointType` `Go::ftCurveSegment::joint_` `[protected]`

Definition at line 173 of file `ftCurve.h`.



29.141.4.2 `shared_ptr<ParamCurve> Go::ftCurveSegment::parameter_curve_[2]` [protected]

Curves in surface domain.

Definition at line 177 of file `ftCurve.h`.

29.141.4.3 `ftCurveType Go::ftCurveSegment::segment_type_` [protected]

Definition at line 172 of file `ftCurve.h`.

29.141.4.4 `shared_ptr<ParamCurve> Go::ftCurveSegment::space_curve_` [protected]

Curve in space.

Definition at line 179 of file `ftCurve.h`.

29.141.4.5 `ftFaceBase* Go::ftCurveSegment::underlying_face_[2]` [protected]

Surface(s) on which curve is lying.

Definition at line 175 of file `ftCurve.h`.

The documentation for this class was generated from the following file:

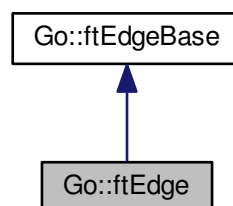
- `compositemodel/include/GoTools/compositemodel/ftCurve.h`

## 29.142 Go::ftEdge Class Reference

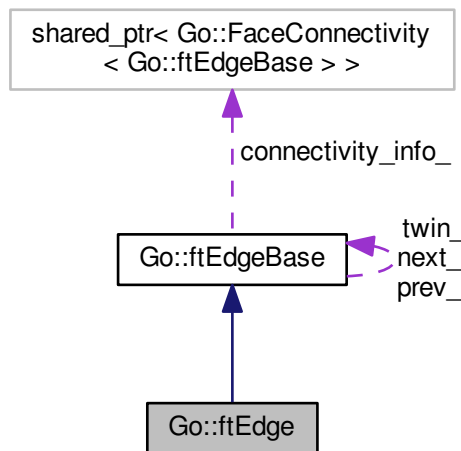
The `ftEdge` is a half-edge implementation of a topological data structure.

```
#include <ftEdge.h>
```

Inheritance diagram for `Go::ftEdge`:



Collaboration diagram for Go::ftEdge:



## Public Member Functions

- `ftEdge` (`ftFaceBase *face`, `shared_ptr< ParamCurve > cv`, `double tmin`, `double tmax`, `int entry_id=-1`)
- `ftEdge` (`shared_ptr< ParamCurve > cv`, `double tmin`, `double tmax`, `int entry_id=-1`)
- `ftEdge` (`ftFaceBase *face`, `shared_ptr< ParamCurve > cv`, `shared_ptr< Vertex > v1`, `shared_ptr< Vertex > v2`, `bool is_reversed=false`, `int entry_id=-1`)
- `ftEdge` (`shared_ptr< ParamCurve > cv`, `shared_ptr< Vertex > v1`, `shared_ptr< Vertex > v2`, `bool is_reversed=false`, `int entry_id=-1`)
- `~ftEdge` ()
  - Destructor.*
- virtual `double tMin` () `const`
  - Minimum parameter of curve restricted to edge.*
- virtual `double tMax` () `const`
  - Maximum parameter of curve restricted to edge.*
- virtual void `setReversed` (`bool is_reversed`)
- virtual `bool isReversed` ()
- void `reverseGeomCurve` ()
- virtual `ftFaceBase * face` ()
  - The face associated to this edge.*
- virtual void `setFace` (`ftFaceBase *face`)
  - Set pointer to the face associated to this edge.*
- virtual `BoundingBox boundingBox` ()
  - Bounding box surrounding this edge. The box may be too large.*
- virtual `ftEdge * split` (`double t`)
  - Split edge in a given parameter.*
- `shared_ptr< ftEdge > split2` (`double t`)
  - Split edge and update associated edge loop.*
- `shared_ptr< ftEdge > splitAtVertex` (`shared_ptr< Vertex > vx`)
  - Split according to an already existing vertex.*

- virtual int `entryId` ()  
*Fetch Id corresponding to this edge. It is not necessarily uniquely set.*
- virtual void `setEntryId` (int id)  
*Set Id corresponding to this edge.*
- virtual `Point` `point` (double t) const  
*Evaluate position given edge parameter.*
- virtual `Point` `tangent` (double t) const  
*Evaluate tangent of edge given edge parameter.*
- virtual `Point` `normal` (double t) const  
*Evaluate normal of associated face given edge parameter.*
- virtual `Point` `normal` (double t, `Point` &face\_par\_pt, double \*face\_seed) const
- virtual void `closestPoint` (const `Point` &pt, double &clo\_t, `Point` &clo\_pt, double &clo\_dist, double const \*seed=0) const  
*Closest point on edge to a given point.*
- virtual void `connectAfter` (ftEdgeBase \*edge)  
*Connect this edge to the given one.*
- virtual void `closeLoop` (ftEdgeBase \*last)  
*Closing of loop.*
- virtual void `disconnectThis` ()
- virtual void `connectTwin` (ftEdgeBase \*twin, int &status)
- virtual void `disconnectTwin` ()  
*Remove twin information in this edge.*
- void `joinVertices` (ftEdgeBase \*twin)  
*Update associated vertices when two edges are identified as twins.*
- virtual ftEdge \* `geomEdge` ()  
*Return edge pointer.*
- virtual bool `orientationOK` () const  
*Compare orientation of curve and edge.*
- void `point` (double t, int der, std::vector< `Point` > &derivs) const
- shared\_ptr< `ParamCurve` > `geomCurve` ()  
*Fetch the geometry curve associated with this edge.*
- double `estimatedCurveLength` ()  
*Estimate the length of the curve corresponding to this edge.*
- double `estimatedCurveLength` (double min\_par, double max\_par)
- `Point` `faceParameter` (double t, double \*seed=NULL) const
- void `setGeomCurve` (shared\_ptr< `ParamCurve` > geom\_curve)
- virtual void `updateGeomCurve` (double tol)
- bool `updateEdgeInfo` (double tol)  
*Update geometry info if possible.*
- int `getCurveIndex` () const
- shared\_ptr< `Vertex` > `getVertex` (bool at\_start)
- void `getVertices` (shared\_ptr< `Vertex` > &v1, shared\_ptr< `Vertex` > &v2)
- shared\_ptr< `Vertex` > `getOtherVertex` (const `Vertex` \*vx)
- shared\_ptr< `Vertex` > `getCommonVertex` (ftEdge \*other)
- bool `hasVertex` (`Vertex` \*vx)  
*Check if the vertex vx belongs to this edge.*
- void `setVertices` (shared\_ptr< `Vertex` > v1, shared\_ptr< `Vertex` > v2)  
*Assign vertices to this edge.*
- double `parAtVertex` (const `Vertex` \*vx) const
- void `replaceVertex` (shared\_ptr< `Vertex` > &this\_vertex, shared\_ptr< `Vertex` > &other\_vertex)  
*Interchange one of this edge's vertex. Used in topology build.*
- void `addEdgeMultiplicityInstance` (ftEdge \*other)

- Functionality related to non-manifold models.*

  - [bool hasEdgeMultiplicity \(\)](#)
  - Check if this edge belongs to a radial edge ([EdgeVertex](#))*
  - [shared\\_ptr< \[EdgeVertex\]\(#\) > getEdgeMultiplicityInstance \(\)](#)
  - Fetch the radial edge to which this edge belongs.*
  - [void setEdgeVertex \(shared\\_ptr< \[EdgeVertex\]\(#\) > radial\\_edge\)](#)
  - [void joinEdgeVertex \(shared\\_ptr< \[EdgeVertex\]\(#\) > radial\\_edge\)](#)
  - [void removeEdgeVertex \(\)](#)
  - Remove radial edge instance.*
  - [void joinVertex \(shared\\_ptr< \[Vertex\]\(#\) > this\\_vertex, shared\\_ptr< \[Vertex\]\(#\) > other\\_vertex\)](#)
  - Connect to adjacent vertices including updating edge information.*
  - [std::vector< \[ftSurface\]\(#\) \\* > getAdjacentFaces \(\) const](#)
  - Get all (up to 2) faces meeting in this edge.*
  - [std::vector< \[ftSurface\]\(#\) \\* > getAllAdjacentFaces \(\) const](#)
  - Get all faces meeting in this edge, including non-manifold cases.*
  - [bool hasCommonVertices \(\[ftEdge\]\(#\) \\*other\) const](#)
  - Check for common vertices between two edges.*
  - [bool commonVertex \(\[ftEdge\]\(#\) \\*other\) const](#)
  - [bool hasCommonRadialEdge \(\[ftEdge\]\(#\) \\*other\) const](#)
  - [bool checkEdgeTopology \(\)](#)
  - Debug functionality.*

## Additional Inherited Members

### 29.142.1 Detailed Description

The [ftEdge](#) is a half-edge implementation of a topological data structure.

[ftEdge](#) - topological edge for Fantastic

The [ftEdge](#) is a half-edge implementation of a topological data structure. It implements the [ftEdgeBase](#) interface by using the [Go](#) geometry library.

#### Author

Atgeirr F Rasmussen [atgeirr@sintef.no](mailto:atgeirr@sintef.no)

#### See also

[ftEdgeBase](#)

Definition at line 68 of file [ftEdge.h](#).

### 29.142.2 Constructor & Destructor Documentation

29.142.2.1 [Go::ftEdge::ftEdge \( \[ftFaceBase\]\(#\) \\* face, shared\\_ptr< \[ParamCurve\]\(#\) > cv, double tmin, double tmax, int entry\\_id = -1 \)](#)

Constructor. Detailed description.

29.142.2.2 `Go::ftEdge::ftEdge ( shared_ptr< ParamCurve > cv, double tmin, double tmax, int entry_id = -1 )`

Constructor. Input is the curve representing the geometry of the edge and parameter values for the two vertices. There is no reference to a face object.

29.142.2.3 `Go::ftEdge::ftEdge ( ftFaceBase * face, shared_ptr< ParamCurve > cv, shared_ptr< Vertex > v1, shared_ptr< Vertex > v2, bool is_reversed = false, int entry_id = -1 )`

Constructor.

Parameters

|                    |                                                                                                                       |
|--------------------|-----------------------------------------------------------------------------------------------------------------------|
| <i>face</i>        | the face to which this edge is associate                                                                              |
| <i>cv</i>          | geometric curve                                                                                                       |
| <i>v1</i>          | vertex in the start of the edge                                                                                       |
| <i>v2</i>          | vertex in the end of the edge                                                                                         |
| <i>is_reversed</i> | whether the geometric curve is reversed with respect to the start and end vertices. Needed to sort out closed curves. |

29.142.2.4 `Go::ftEdge::ftEdge ( shared_ptr< ParamCurve > cv, shared_ptr< Vertex > v1, shared_ptr< Vertex > v2, bool is_reversed = false, int entry_id = -1 )`

Constructor. Input is the curve representing the geometry of the edge and two vertices. There is no reference to a face object.

29.142.2.5 `Go::ftEdge::~ftEdge ( )`

Destructor.

### 29.142.3 Member Function Documentation

29.142.3.1 `void Go::ftEdge::addEdgeMultiplicityInstance ( ftEdge * other )`

Functionality related to non-manifold models.

29.142.3.2 `virtual BoundingBox Go::ftEdge::boundingBox ( ) [virtual]`

Bounding box surrounding this edge. The box may be too large.

Implements [Go::ftEdgeBase](#).

29.142.3.3 `bool Go::ftEdge::checkEdgeTopology ( )`

Debug functionality.

29.142.3.4 `virtual void Go::ftEdge::closeLoop ( ftEdgeBase * last ) [virtual]`

Closing of loop.

Reimplemented from [Go::ftEdgeBase](#).

29.142.3.5 `virtual void Go::ftEdge::closestPoint ( const Point & pt, double & clo_t, Point & clo_pt, double & clo_dist, double const * seed = 0 ) const [virtual]`

Closest point on edge to a given point.

Implements [Go::ftEdgeBase](#).

29.142.3.6 `bool Go::ftEdge::commonVertex ( ftEdge * other ) const [inline]`

Definition at line 369 of file ftEdge.h.

29.142.3.7 `virtual void Go::ftEdge::connectAfter ( ftEdgeBase * edge ) [virtual]`

Connect this edge to the given one.

Reimplemented from [Go::ftEdgeBase](#).

29.142.3.8 `virtual void Go::ftEdge::connectTwin ( ftEdgeBase * twin, int & status ) [virtual]`

Represent adjacency by connecting this edge to its twin edge on the adjace face

Reimplemented from [Go::ftEdgeBase](#).

29.142.3.9 `virtual void Go::ftEdge::disconnectThis ( ) [virtual]`

Disconnect this edge from its neighbours in the loop of edges limiting a face

Reimplemented from [Go::ftEdgeBase](#).

29.142.3.10 `virtual void Go::ftEdge::disconnectTwin ( ) [virtual]`

Remove twin information in this edge.

Reimplemented from [Go::ftEdgeBase](#).

29.142.3.11 `virtual int Go::ftEdge::entryId ( ) [inline],[virtual]`

Fetch Id corresponding to this edge. It is not necessarily uniquely set.

Implements [Go::ftEdgeBase](#).

Definition at line 162 of file ftEdge.h.

29.142.3.12 **double** Go::ftEdge::estimatedCurveLength ( )

Estimate the length of the curve corresponding to this edge.

29.142.3.13 **double** Go::ftEdge::estimatedCurveLength ( **double** *min\_par*, **double** *max\_par* )

Estimate the length of the curve corresponding to this edge limited by edge parameters

29.142.3.14 **virtual ftFaceBase\*** Go::ftEdge::face ( ) [virtual]

The face associated to this edge.

Implements [Go::ftEdgeBase](#).

29.142.3.15 **Point** Go::ftEdge::faceParameter ( **double** *t*, **double** \* *seed* = NULL ) const

Fetch the parameter in the associated face, given the edge parameter

29.142.3.16 **shared\_ptr<ParamCurve>** Go::ftEdge::geomCurve ( ) [inline]

Fetch the geometry curve associated with this edge.

Definition at line 219 of file ftEdge.h.

29.142.3.17 **virtual ftEdge\*** Go::ftEdge::geomEdge ( ) [virtual]

Return edge pointer.

Implements [Go::ftEdgeBase](#).

29.142.3.18 **std::vector<ftSurface\*>** Go::ftEdge::getAdjacentFaces ( ) const

Get all (up to 2) faces meeting in this edge.

29.142.3.19 **std::vector<ftSurface\*>** Go::ftEdge::getAllAdjacentFaces ( ) const

Get all faces meeting in this edge, including non-manifold cases.

29.142.3.20 **shared\_ptr<Vertex>** Go::ftEdge::getCommonVertex ( **ftEdge** \* *other* ) [inline]

Fetch the vertex that is common to this edge and the edge other. If no such vertex exist, an empty shared pointer is returned

Definition at line 287 of file ftEdge.h.

29.142.3.21 `int Go::ftEdge::getCurveIndex ( ) const`

Fetch index of geometry curve with respect to associated surface, if this information exists: -1 : No info or not a rectangular surface 0 : umin 1 : umax 2 : vmin 3 : vmax

29.142.3.22 `shared_ptr<EdgeVertex> Go::ftEdge::getEdgeMultiplicityInstance ( ) [inline]`

Fetch the radial edge to which this edge belongs.

Definition at line 328 of file ftEdge.h.

29.142.3.23 `shared_ptr<Vertex> Go::ftEdge::getOtherVertex ( const Vertex * vx ) [inline]`

Given one of the vertices belonging to this edge, fetch the other

Definition at line 274 of file ftEdge.h.

29.142.3.24 `shared_ptr<Vertex> Go::ftEdge::getVertex ( bool at_start )`

Access function for vertices. The function takes into account whether or not the orientation is reversed.

29.142.3.25 `void Go::ftEdge::getVertices ( shared_ptr< Vertex > & v1, shared_ptr< Vertex > & v2 ) [inline]`

Access function for vertices. This function does not take into account the orientation of the edge.

Definition at line 265 of file ftEdge.h.

29.142.3.26 `bool Go::ftEdge::hasCommonRadialEdge ( ftEdge * other ) const [inline]`

Check if this edge and the edge other have a common radial edge ([EdgeVertex](#))

Definition at line 380 of file ftEdge.h.

29.142.3.27 `bool Go::ftEdge::hasCommonVertices ( ftEdge * other ) const [inline]`

Check for common vertices between two edges.

Definition at line 360 of file ftEdge.h.

29.142.3.28 `bool Go::ftEdge::hasEdgeMultiplicity ( ) [inline]`

Check if this edge belongs to a radial edge ([EdgeVertex](#))

Definition at line 322 of file ftEdge.h.



29.142.3.29 **bool** Go::ftEdge::hasVertex ( **Vertex** \* vx ) [inline]

Check if the vertex vx belongs to this edge.

Definition at line 299 of file ftEdge.h.

29.142.3.30 **virtual bool** Go::ftEdge::isReversed ( ) [virtual]

Get the value of the flag indicating that the orientation of the edge is reversed with respect to the parametrization of the underlying [ParamCurve](#).

**Returns**

*true* if orientation is reversed, *false* otherwise

Implements [Go::ftEdgeBase](#).

29.142.3.31 **void** Go::ftEdge::joinEdgeVertex ( **shared\_ptr**< **EdgeVertex** > radial\_edge )

Connect to adjacent radial edges. Used in topology build for volumes

29.142.3.32 **void** Go::ftEdge::joinVertex ( **shared\_ptr**< **Vertex** > this\_vertex, **shared\_ptr**< **Vertex** > other\_vertex )

Connect to adjacent vertices including updating edge information.

29.142.3.33 **void** Go::ftEdge::joinVertices ( **ftEdgeBase** \* twin )

Update associated vertices when two edges are identified as twins.

29.142.3.34 **virtual Point** Go::ftEdge::normal ( **double** t ) **const** [virtual]

Evaluate normal of associated face given edge parameter.

Implements [Go::ftEdgeBase](#).

29.142.3.35 **virtual Point** Go::ftEdge::normal ( **double** t, **Point** & face\_par\_pt, **double** \* face\_seed ) **const** [virtual]

Evaluate normal of associated face given a seed to the search for the corresponding face parameter

Implements [Go::ftEdgeBase](#).

29.142.3.36 **virtual bool** Go::ftEdge::orientationOK ( ) **const** [virtual]

Compare orientation of curve and edge.

Reimplemented from [Go::ftEdgeBase](#).

29.142.3.37 **double** `Go::ftEdge::parAtVertex ( const Vertex * vx ) const`

Fetch the parameter of the curve associated to this edge at a given vertex

29.142.3.38 **virtual Point** `Go::ftEdge::point ( double t ) const` [virtual]

Evaluate position given edge parameter.

Implements [Go::ftEdgeBase](#).

29.142.3.39 **void** `Go::ftEdge::point ( double t, int der, std::vector< Point > & derivs ) const`

Evaluate this edge at the parameter t, including derivatives up to der

29.142.3.40 **void** `Go::ftEdge::removeEdgeVertex ( )`

Remove radial edge instance.

29.142.3.41 **void** `Go::ftEdge::replaceVertex ( shared_ptr< Vertex > & this_vertex, shared_ptr< Vertex > & other_vertex )`

Interchange one of this edge's vertex. Used in topology build.

29.142.3.42 **void** `Go::ftEdge::reverseGeomCurve ( )`

Reverse the geometric curve. A call to this function will reverse the direction of the geometry curve, swap the vertices, and change the 'is\_reversed' flag. The geometry curve will be cloned.

29.142.3.43 **void** `Go::ftEdge::setEdgeVertex ( shared_ptr< EdgeVertex > radial_edge )` [inline]

Assign a radial edge to this edge. Used in topology build for volumes

Definition at line 335 of file ftEdge.h.

29.142.3.44 **virtual void** `Go::ftEdge::setEntryId ( int id )` [inline],[virtual]

Set Id corresponding to this edge.

Implements [Go::ftEdgeBase](#).

Definition at line 165 of file ftEdge.h.

29.142.3.45 **virtual void** `Go::ftEdge::setFace ( ftFaceBase * face )` [inline],[virtual]

Set pointer to the face associated to this edge.

Implements [Go::ftEdgeBase](#).

Definition at line 136 of file ftEdge.h.

29.142.3.46 **void** `Go::ftEdge::setGeomCurve ( shared_ptr< ParamCurve > geom_curve )`

Convenience function to set or reset the geometry representation of the edge.

## Parameters

|                         |                                                                                                             |
|-------------------------|-------------------------------------------------------------------------------------------------------------|
| <code>geom_curve</code> | The curve representing the geometry of the edge. This will frequently be a <a href="#">CurveOnSurface</a> . |
|-------------------------|-------------------------------------------------------------------------------------------------------------|

29.142.3.47 `virtual void Go::ftEdge::setReversed ( bool is_reversed ) [virtual]`

Set a flag to indicate that the orientation of the edge is reversed with respect to the parametrization of the underlying [ParamCurve](#).

## Parameters

|                          |                               |
|--------------------------|-------------------------------|
| <code>is_reversed</code> | value of the orientation flag |
|--------------------------|-------------------------------|

Implements [Go::ftEdgeBase](#).

29.142.3.48 `void Go::ftEdge::setVertices ( shared_ptr< Vertex > v1, shared_ptr< Vertex > v2 )`

Assign vertices to this edge.

29.142.3.49 `virtual ftEdge* Go::ftEdge::split ( double t ) [virtual]`

Split edge in a given parameter.

Implements [Go::ftEdgeBase](#).

29.142.3.50 `shared_ptr<ftEdge> Go::ftEdge::split2 ( double t )`

Split edge and update associated edge loop.

29.142.3.51 `shared_ptr<ftEdge> Go::ftEdge::splitAtVertex ( shared_ptr< Vertex > vx )`

Split according to an already existing vertex.

29.142.3.52 `virtual Point Go::ftEdge::tangent ( double t ) const [virtual]`

Evaluate tangent of edge given edge parameter.

Implements [Go::ftEdgeBase](#).

29.142.3.53 `virtual double Go::ftEdge::tMax ( ) const [inline],[virtual]`

Maximum parameter of curve restricted to edge.

Implements [Go::ftEdgeBase](#).

Definition at line 115 of file `ftEdge.h`.

29.142.3.54 `virtual double Go::ftEdge::tMin ( ) const [inline],[virtual]`

Minimum parameter of curve restricted to edge.

Implements [Go::ftEdgeBase](#).

Definition at line 110 of file `ftEdge.h`.

29.142.3.55 `bool Go::ftEdge::updateEdgeInfo ( double tol )`

Update geometry info if possible.

29.142.3.56 `virtual void Go::ftEdge::updateGeomCurve ( double tol ) [virtual]`

Update pointer to geometry curve associated to this edge after changes to the associated face. To be used when topology changes are applied to a model.

The documentation for this class was generated from the following file:

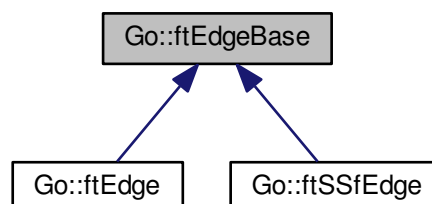
- `compositemodel/include/GoTools/compositemodel/ftEdge.h`

## 29.143 Go::ftEdgeBase Class Reference

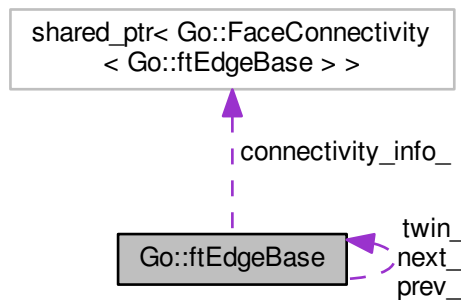
Base class for edges. Defines the interface used in topology analysis.

```
#include <ftEdgeBase.h>
```

Inheritance diagram for `Go::ftEdgeBase`:



Collaboration diagram for Go::ftEdgeBase:



## Public Member Functions

- [ftEdgeBase](#) ()  
*Constructor.*
- virtual [~ftEdgeBase](#) ()  
*Destructor.*
- [ftEdgeBase \\* next](#) ()  
*Next edge in the edge loop.*
- [ftEdgeBase \\* prev](#) ()  
*Previous edge in the edge loop.*
- [ftEdgeBase \\* twin](#) ()
- virtual [double tMin](#) () **const** =0  
*Start parameter value of edge.*
- virtual [double tMax](#) () **const** =0  
*End parameter value of edge.*
- virtual void [setReversed](#) (**bool** is\_reversed)=0
- virtual **bool** [isReversed](#) ()=0
- virtual [ftFaceBase \\* face](#) ()=0  
*The face corresponding to this edge.*
- virtual void [setFace](#) ([ftFaceBase \\*face](#))=0  
*Set face pointer.*
- virtual [Go::BoundingBox boundingBox](#) ()=0  
*Coordinate box surrounding this edge.*
- virtual **int** [entryId](#) ()=0  
*Id of edge. Default set to -1.*
- virtual void [setEntryId](#) (**int** id)=0  
*Set id of edge.*
- virtual [ftEdgeBase \\* split](#) (**double** t)=0
- virtual [Go::Point point](#) (**double** t) **const** =0  
*Evaluate position.*
- virtual [Go::Point tangent](#) (**double** t) **const** =0  
*Evaluate tangent.*
- virtual [Go::Point normal](#) (**double** t) **const** =0

*Evaluate normal of associated face.*

- virtual `Go::Point normal (double t, Go::Point &face_par_pt, double *face_seed) const =0`
- virtual void `closestPoint (const Go::Point &pt, double &clo_t, Go::Point &clo_pt, double &clo_dist, double const *seed=0) const =0`

*Closest point on this edge to a given point.*

- virtual void `connectAfter (ftEdgeBase *edge)`

*Add the edge edge to the edge loop after this edge.*

- virtual void `closeLoop (ftEdgeBase *last)`
- virtual void `disconnectThis ()`

*Remove this edge from the edge loop it belongs to.*

- virtual void `connectTwin (ftEdgeBase *twin, int &status)`
- virtual void `disconnectTwin ()`
- `bool onBoundary ()`
- void `adjacentEdges (bool at_start_of_edge, std::vector< ftEdgeBase * > &adjacent, std::vector< bool > &at_start)`
- `tpJointType checkContinuity (ftEdgeBase *nextedge, double neighbour, double gap, double bend, double kink) const`
- virtual `bool orientationOK () const`

*Compare orientation of curve and edge.*

- virtual `ftEdge * geomEdge ()=0`

*Return edge pointer.*

- `bool checkOverlap (ftEdgeBase *other, double tol, int nmb-sample, double &t1, double &t2, double &t3, double &t4, bool &same_dir, bool no_snap=true) const`

*Compute an eventual overlap between this edge and the edge other.*

- `bool hasConnectivityInfo ()`
- `shared_ptr< FaceConnectivity< ftEdgeBase > > getConnectivityInfo ()`
- void `setConnectivityInfo (shared_ptr< FaceConnectivity< ftEdgeBase > > info)`
- void `resetConnectivityInfo ()`

*Mark the connectivity information associated to this edge as outdated.*

## Protected Attributes

- `ftEdgeBase * next_`
- `ftEdgeBase * prev_`
- `ftEdgeBase * twin_`
- `shared_ptr< FaceConnectivity< ftEdgeBase > > connectivity_info_`

### 29.143.1 Detailed Description

Base class for edges. Defines the interface used in topology analysis.

Definition at line 58 of file `ftEdgeBase.h`.

### 29.143.2 Constructor & Destructor Documentation

#### 29.143.2.1 `Go::ftEdgeBase::ftEdgeBase ( )`

Constructor.

29.143.2.2 `virtual Go::ftEdgeBase::~~ftEdgeBase ( ) [virtual]`

Destructor.

### 29.143.3 Member Function Documentation

29.143.3.1 `void Go::ftEdgeBase::adjacentEdges ( bool at_start_of_edge, std::vector< ftEdgeBase * > & adjacent, std::vector< bool > & at_start )`

Consider the 'node' or 'corner' implied by the startpoint (if *at\_start\_of\_edge* is true) or endpoint (if it is false). The vector *adjacent* is filled with all edges with that 'node' as a startpoint (indicated by a true at the corresponding place in the *at\_start* vector) or endpoint (false in *at\_start*). The edge originally asked for is not included.

29.143.3.2 `virtual Go::BoundingBox Go::ftEdgeBase::boundingBox ( ) [pure virtual]`

Coordinate box surrounding this edge.

Implemented in [Go::ftEdge](#), and [Go::ftSSfEdge](#).

29.143.3.3 `tpJointType Go::ftEdgeBase::checkContinuity ( ftEdgeBase * nextedge, double neighbour, double gap, double bend, double kink ) const`

Compute the continuity between this edge and *nextedge* given the necessary continuity parameters. 0 = g1 continuity, 1 = not quite g1, 2 = g0 continuity, 3 = gap, 4 = discontinuous, 5 = last segment

29.143.3.4 `bool Go::ftEdgeBase::checkOverlap ( ftEdgeBase * other, double tol, int nmbsample, double & t1, double & t2, double & t3, double & t4, bool & same_dir, bool no_snap = true ) const`

Compute an eventual overlap between this edge and the edge *other*.

29.143.3.5 `virtual void Go::ftEdgeBase::closeLoop ( ftEdgeBase * last ) [virtual]`

Close edge loop by setting the appropriate pointers between this edge and last edge

Reimplemented in [Go::ftEdge](#).

29.143.3.6 `virtual void Go::ftEdgeBase::closestPoint ( const Go::Point & pt, double & clo_t, Go::Point & clo_pt, double & clo_dist, double const * seed = 0 ) const [pure virtual]`

Closest point on this edge to a given point.

Implemented in [Go::ftEdge](#), and [Go::ftSSfEdge](#).

29.143.3.7 `virtual void Go::ftEdgeBase::connectAfter ( ftEdgeBase * edge ) [virtual]`

Add the edge `edge` to the edge loop after this edge.

Reimplemented in [Go::ftEdge](#).

29.143.3.8 `virtual void Go::ftEdgeBase::connectTwin ( ftEdgeBase * twin, int & status ) [virtual]`

Set adjacency between the face connected to this edge and the face connected to `twin` by setting twin pointers

Reimplemented in [Go::ftEdge](#).

29.143.3.9 `virtual void Go::ftEdgeBase::disconnectThis ( ) [virtual]`

Remove this edge from the edge loop it belongs to.

Reimplemented in [Go::ftEdge](#).

29.143.3.10 `virtual void Go::ftEdgeBase::disconnectTwin ( ) [virtual]`

Disconnect this edge from its twin, i.e. remove adjacency information between the face connected to this edge and the face connected to the twin edge

Reimplemented in [Go::ftEdge](#).

29.143.3.11 `virtual int Go::ftEdgeBase::entryId ( ) [pure virtual]`

Id of edge. Default set to -1.

Implemented in [Go::ftEdge](#), and [Go::ftSSfEdge](#).

29.143.3.12 `virtual ftFaceBase* Go::ftEdgeBase::face ( ) [pure virtual]`

The face corresponding to this edge.

Implemented in [Go::ftEdge](#), and [Go::ftSSfEdge](#).

29.143.3.13 `virtual ftEdge* Go::ftEdgeBase::geomEdge ( ) [pure virtual]`

Return edge pointer.

Implemented in [Go::ftEdge](#), and [Go::ftSSfEdge](#).



29.143.3.14 `shared_ptr<FaceConnectivity<ftEdgeBase>> Go::ftEdgeBase::getConnectivityInfo ( ) [inline]`

Fetch the continuity information regarding the joint between the face corresponding to this edge and the adjacent face along this edge

Definition at line 181 of file ftEdgeBase.h.

29.143.3.15 `bool Go::ftEdgeBase::hasConnectivityInfo ( ) [inline]`

Check if this edge has access to continuity information regarding the joint between the face corresponding to this edge and the adjacent face along this edge

Definition at line 174 of file ftEdgeBase.h.

29.143.3.16 `virtual bool Go::ftEdgeBase::isReversed ( ) [pure virtual]`

Get the value of the flag indicating that the orientation of the edge is reversed with respect to the parametrization of the underlying [ParamCurve](#).

Returns

*true* if orientation is reversed, *false* otherwise

Implemented in [Go::ftEdge](#), and [Go::ftSSfEdge](#).

29.143.3.17 `ftEdgeBase* Go::ftEdgeBase::next ( )`

Next edge in the edge loop.

29.143.3.18 `virtual Go::Point Go::ftEdgeBase::normal ( double t ) const [pure virtual]`

Evaluate normal of associated face.

Implemented in [Go::ftEdge](#), and [Go::ftSSfEdge](#).

29.143.3.19 `virtual Go::Point Go::ftEdgeBase::normal ( double t, Go::Point & face_par_pt, double * face_seed ) const [pure virtual]`

Evaluate normal of associated face given a seed to the search for the corresponding face parameter

Implemented in [Go::ftEdge](#), and [Go::ftSSfEdge](#).

29.143.3.20 `bool Go::ftEdgeBase::onBoundary ( ) [inline]`

Check if this edge lies on the boundary of the associated face set, i.e. it has no neighbour

Definition at line 140 of file ftEdgeBase.h.

29.143.3.21 `virtual bool Go::ftEdgeBase::orientationOK ( ) const` [virtual]

Compare orientation of curve and edge.

Reimplemented in [Go::ftEdge](#).

29.143.3.22 `virtual Go::Point Go::ftEdgeBase::point ( double t ) const` [pure virtual]

Evaluate position.

Implemented in [Go::ftEdge](#), and [Go::ftSSfEdge](#).

29.143.3.23 `ftEdgeBase* Go::ftEdgeBase::prev ( )`

Previous edge in the edge loop.

29.143.3.24 `void Go::ftEdgeBase::resetConnectivityInfo ( )` [inline]

Mark the connectivity information associated to this edge as outdated.

Definition at line 194 of file `ftEdgeBase.h`.

29.143.3.25 `void Go::ftEdgeBase::setConnectivityInfo ( shared_ptr< FaceConnectivity< ftEdgeBase > > info )`  
[inline]

Store the continuity information regarding the joint between the face corresponding to this edge and the adjacent face along this edge

Definition at line 188 of file `ftEdgeBase.h`.

29.143.3.26 `virtual void Go::ftEdgeBase::setEntryId ( int id )` [pure virtual]

Set id of edge.

Implemented in [Go::ftEdge](#), and [Go::ftSSfEdge](#).

29.143.3.27 `virtual void Go::ftEdgeBase::setFace ( ftFaceBase * face )` [pure virtual]

Set face pointer.

Implemented in [Go::ftEdge](#), and [Go::ftSSfEdge](#).

29.143.3.28 `virtual void Go::ftEdgeBase::setReversed ( bool is_reversed )` [pure virtual]

Set a flag to indicate that the orientation of the edge is reversed with respect to the parametrization of the underlying [ParamCurve](#).

## Parameters

|                    |                               |
|--------------------|-------------------------------|
| <i>is_reversed</i> | value of the orientation flag |
|--------------------|-------------------------------|

Implemented in [Go::ftEdge](#), and [Go::ftSSfEdge](#).

**29.143.3.29** `virtual ftEdgeBase* Go::ftEdgeBase::split ( double t ) [pure virtual]`

Split the edge at the given parameter t. This edge will then represent the first part of the original edge, the second part is returned to the caller.

Implemented in [Go::ftEdge](#), and [Go::ftSSfEdge](#).

**29.143.3.30** `virtual Go::Point Go::ftEdgeBase::tangent ( double t ) const [pure virtual]`

Evaluate tangent.

Implemented in [Go::ftEdge](#), and [Go::ftSSfEdge](#).

**29.143.3.31** `virtual double Go::ftEdgeBase::tMax ( ) const [pure virtual]`

End parameter value of edge.

Implemented in [Go::ftEdge](#), and [Go::ftSSfEdge](#).

**29.143.3.32** `virtual double Go::ftEdgeBase::tMin ( ) const [pure virtual]`

Start parameter value of edge.

Implemented in [Go::ftEdge](#), and [Go::ftSSfEdge](#).

**29.143.3.33** `ftEdgeBase* Go::ftEdgeBase::twin ( )`

The corresponding edge on the neighbouring face. If no such neighbour exists, the function returns null.

## 29.143.4 Member Data Documentation

**29.143.4.1** `shared_ptr<FaceConnectivity<ftEdgeBase>> Go::ftEdgeBase::connectivity_info_ [protected]`

Definition at line 204 of file ftEdgeBase.h.

**29.143.4.2** `ftEdgeBase* Go::ftEdgeBase::next_ [protected]`

Definition at line 201 of file ftEdgeBase.h.

29.143.4.3 `ftEdgeBase*` `Go::ftEdgeBase::prev_` [protected]

Definition at line 202 of file `ftEdgeBase.h`.

29.143.4.4 `ftEdgeBase*` `Go::ftEdgeBase::twin_` [protected]

Definition at line 203 of file `ftEdgeBase.h`.

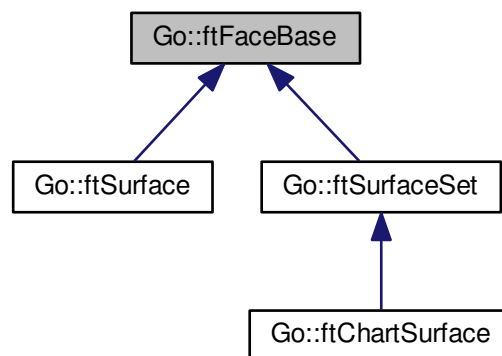
The documentation for this class was generated from the following file:

- `compositemodel/include/GoTools/compositemodel/ftEdgeBase.h`

## 29.144 Go::ftFaceBase Class Reference

```
#include <ftFaceBase.h>
```

Inheritance diagram for `Go::ftFaceBase`:



### Public Member Functions

- `ftFaceBase ()`  
*Constructor.*
- `ftFaceBase (int id)`  
*Constructor.*
- `virtual ~ftFaceBase ()`  
*Destructor.*
- `virtual ftSurface * asFtSurface ()`
- `virtual void clearInitialEdges ()`  
*Reset loop information.*

- virtual `std::vector< shared_ptr< ftEdgeBase > > createInitialEdges` (`double` degenerate\_epsilon=DEFAULT\_SPACE\_EPSILON, `double` kink=0.00015, `bool` no\_split=false)=0
- virtual `std::vector< shared_ptr< ftEdgeBase > > startEdges` ()=0  
*Return pointers to first part of all bd cvs.*
- virtual `Point point` (`double` u, `double` v) `const` =0  
*Evaluate point on face.*
- virtual `Point normal` (`double` u, `double` v) `const` =0  
*Evaluate surface normal.*
- virtual `BoundingBox boundingBox` ()=0  
*The bounding box corresponding to this face.*
- virtual `void setId` (`int` id)  
*Set id for this face.*
- virtual `int getId` ()  
*Return id, default id is -1.*
- virtual `shared_ptr< ParamSurface > surface` ()=0  
*Fetch geometric surface.*
- virtual `ftMessage createSurf` (`double` &max\_error, `double` &mean\_error)=0
- virtual `void getError` (`double` &max\_error, `double` &mean\_error)=0
- virtual `ftTangPriority getPrioType` () `const` =0  
*Priority of current face. Not applicable in normal configurations.*
- virtual `void updateBoundaryLoops` (`shared_ptr< ftEdgeBase >` new\_edge)
- virtual `void isolateFace` ()  
*Remove all adjacency information related to this face.*
- virtual `ftMessage removeGap` (`ftEdgeBase *e1`, `ftEdgeBase *e2`, `ftFaceBase *other`)  
*Close gap between adjacent faces.*
- virtual `void closestPoint` (`const Point` &pt, `double` &clo\_u, `double` &clo\_v, `Point` &clo\_pt, `double` &clo\_dist, `double` epsilon) `const` =0  
*Closest point between this face and a point.*

## Protected Attributes

- `int id_`

### 29.144.1 Detailed Description

`ftFaceBase` - An abstract interface to a topological face

Definition at line 60 of file `ftFaceBase.h`.

### 29.144.2 Constructor & Destructor Documentation

#### 29.144.2.1 Go::ftFaceBase::ftFaceBase ( )

Constructor.

#### 29.144.2.2 Go::ftFaceBase::ftFaceBase ( int id )

Constructor.

29.144.2.3 virtual `Go::ftFaceBase::~~ftFaceBase ( )` [virtual]

Destructor.

### 29.144.3 Member Function Documentation

29.144.3.1 virtual `ftSurface* Go::ftFaceBase::asFtSurface ( )` [virtual]

Reimplemented in [Go::ftSurface](#).

29.144.3.2 virtual `BoundingBox Go::ftFaceBase::boundingBox ( )` [pure virtual]

The bounding box corresponding to this face.

Implemented in [Go::ftSurface](#), and [Go::ftSurfaceSet](#).

29.144.3.3 virtual `void Go::ftFaceBase::clearInitialEdges ( )` [inline],[virtual]

Reset loop information.

Reimplemented in [Go::ftSurface](#).

Definition at line 74 of file `ftFaceBase.h`.

29.144.3.4 virtual `void Go::ftFaceBase::closestPoint ( const Point & pt, double & clo_u, double & clo_v, Point & clo_pt, double & clo_dist, double epsilon ) const` [pure virtual]

Closest point between this face and a point.

Implemented in [Go::ftSurface](#), and [Go::ftSurfaceSet](#).

29.144.3.5 virtual `std::vector<shared_ptr<ftEdgeBase>> Go::ftFaceBase::createInitialEdges ( double degenerate_epsilon = DEFAULT_SPACE_EPSILON, double kink = 0.00015, bool no_split = false )` [pure virtual]

Compute the edges associated to this face or fetch already existing edges

Implemented in [Go::ftSurface](#), [Go::ftChartSurface](#), and [Go::ftSurfaceSet](#).

29.144.3.6 virtual `ftMessage Go::ftFaceBase::createSurf ( double & max_error, double & mean_error )` [pure virtual]

Make sure that a geometric surface exists. Not applicable in normal configurations

Implemented in [Go::ftSurface](#), [Go::ftSurfaceSet](#), and [Go::ftChartSurface](#).

29.144.3.7 `virtual void Go::ftFaceBase::getError ( double & max_error, double & mean_error ) [pure virtual]`

Fetch the error due to approximations in createSurf. Not applicable in normal configurations

Implemented in [Go::ftSurface](#), [Go::ftSurfaceSet](#), and [Go::ftChartSurface](#).

29.144.3.8 `virtual int Go::ftFaceBase::getId ( ) [virtual]`

Return id, default id is -1.

Reimplemented in [Go::ftSurface](#), and [Go::ftSurfaceSet](#).

29.144.3.9 `virtual ftTangPriority Go::ftFaceBase::getPrioType ( ) const [pure virtual]`

Priority of current face. Not applicable in normal configurations.

Implemented in [Go::ftSurface](#), [Go::ftSurfaceSet](#), and [Go::ftChartSurface](#).

29.144.3.10 `virtual void Go::ftFaceBase::isolateFace ( ) [inline],[virtual]`

Remove all adjacency information related to this face.

Reimplemented in [Go::ftSurface](#).

Definition at line 116 of file ftFaceBase.h.

29.144.3.11 `virtual Point Go::ftFaceBase::normal ( double u, double v ) const [pure virtual]`

Evaluate surface normal.

Implemented in [Go::ftSurface](#), [Go::ftChartSurface](#), and [Go::ftSurfaceSet](#).

29.144.3.12 `virtual Point Go::ftFaceBase::point ( double u, double v ) const [pure virtual]`

Evaluate point on face.

Implemented in [Go::ftSurface](#), [Go::ftSurfaceSet](#), and [Go::ftChartSurface](#).

29.144.3.13 `virtual ftMessage Go::ftFaceBase::removeGap ( ftEdgeBase * e1, ftEdgeBase * e2, ftFaceBase * other ) [inline],[virtual]`

Close gap between adjacent faces.

Definition at line 123 of file ftFaceBase.h.

29.144.3.14 `virtual void Go::ftFaceBase::setId ( int id ) [virtual]`

Set id for this face.

Reimplemented in [Go::ftSurface](#), and [Go::ftSurfaceSet](#).

29.144.3.15 `virtual std::vector<shared_ptr<ftEdgeBase>> Go::ftFaceBase::startEdges ( ) [pure virtual]`

Return pointers to first part of all bd cvs.

Implemented in [Go::ftSurface](#), and [Go::ftSurfaceSet](#).

29.144.3.16 `virtual shared_ptr<ParamSurface> Go::ftFaceBase::surface ( ) [pure virtual]`

Fetch geometric surface.

Implemented in [Go::ftSurface](#), and [Go::ftSurfaceSet](#).

29.144.3.17 `virtual void Go::ftFaceBase::updateBoundaryLoops ( shared_ptr< ftEdgeBase > new_edge ) [virtual]`

Update the boundary loops corresponding to this face with a new edge, i.e. one edge is split and all related topology information must be updated accordingly

Reimplemented in [Go::ftSurface](#).

## 29.144.4 Member Data Documentation

29.144.4.1 `int Go::ftFaceBase::id_ [protected]`

Definition at line 137 of file `ftFaceBase.h`.

The documentation for this class was generated from the following file:

- `compositemodel/include/GoTools/compositemodel/ftFaceBase.h`

## 29.145 Go::ftGraphEdge Class Reference

```
#include <ftPlanarGraph.h>
```



## Public Member Functions

- [ftGraphEdge \( \)](#)  
*Constructor.*
- [ftGraphEdge \(ftSurfaceSetPoint \\*&lower, ftSurfaceSetPoint \\*&upper\)](#)
- [~ftGraphEdge \( \)](#)  
*Destructor.*
- [double startparam \( \) const](#)
- [double endparam \( \) const](#)
- [Vector2D startPoint \( \) const](#)
- [Vector2D endPoint \( \) const](#)
- [Vector2D point \(double v\\_par\)](#)
- [Vector2D point \(double v\\_par, shared\\_ptr< ftFaceBase > face\)](#)  
*Return parameter point in domain of face, given by its's v\_par on graph edge.*
- [const std::vector< shared\\_ptr< ftFaceBase > > & getFaces \( \) const](#)  
*Return the face(s) to which the edge belongs.*
- [Vector2D lower \( \) const](#)
- [Vector2D upper \( \) const](#)

### 29.145.1 Detailed Description

Edge is defined by two end points. The face(s) to which it belongs is also known. y-value of startpoint() is less than that of endpoint().

Definition at line 39 of file ftPlanarGraph.h.

### 29.145.2 Constructor & Destructor Documentation

#### 29.145.2.1 Go::ftGraphEdge::ftGraphEdge ( )

Constructor.

#### 29.145.2.2 Go::ftGraphEdge::ftGraphEdge ( ftSurfaceSetPoint \*& lower, ftSurfaceSetPoint \*& upper )

#### 29.145.2.3 Go::ftGraphEdge::~~ftGraphEdge ( )

Destructor.

### 29.145.3 Member Function Documentation

#### 29.145.3.1 double Go::ftGraphEdge::endparam ( ) const

#### 29.145.3.2 Vector2D Go::ftGraphEdge::endPoint ( ) const

#### 29.145.3.3 const std::vector<shared\_ptr<ftFaceBase> > & Go::ftGraphEdge::getFaces ( ) const

Return the face(s) to which the edge belongs.

29.145.3.4 **Vector2D** `Go::ftGraphEdge::lower ( ) const`

29.145.3.5 **Vector2D** `Go::ftGraphEdge::point ( double v_par )`

29.145.3.6 **Vector2D** `Go::ftGraphEdge::point ( double v_par, shared_ptr< ftFaceBase > face )`

Return parameter point in domain of face, given by its's v\_par on graph edge.

29.145.3.7 **double** `Go::ftGraphEdge::startparam ( ) const`

29.145.3.8 **Vector2D** `Go::ftGraphEdge::startPoint ( ) const`

29.145.3.9 **Vector2D** `Go::ftGraphEdge::upper ( ) const`

The documentation for this class was generated from the following file:

- [compositemodel/include/GoTools/compositemodel/ftPlanarGraph.h](#)

## 29.146 Go::ftGroupGeom Class Reference

A group of geometrical objects.

```
#include <ftGroupGeom.h>
```

### Public Member Functions

- [ftGroupGeom](#) ( )
- [~ftGroupGeom](#) ( )
- [shared\\_ptr< GeomObject > operator\[\]](#) (int idx) const  
*Fetch object number idx.*
- int [size](#) ( ) const  
*The number of objects in the group.*
- [ftTangPriority](#) [getType](#) ( ) const
- void [setType](#) (ftTangPriority type)
- void [addGeomObj](#) (shared\_ptr< [GeomObject](#) > obj)  
*Add a new geometry entity to the group.*

### Protected Attributes

- std::vector< shared\_ptr< [GeomObject](#) > > [geomobj\\_](#)
- [ftTangPriority](#) type\_

#### 29.146.1 Detailed Description

A group of geometrical objects.

Definition at line 53 of file [ftGroupGeom.h](#).

## 29.146.2 Constructor & Destructor Documentation

### 29.146.2.1 Go::ftGroupGeom::ftGroupGeom ( ) `[inline]`

Definition at line 57 of file ftGroupGeom.h.

### 29.146.2.2 Go::ftGroupGeom::~~ftGroupGeom ( ) `[inline]`

Definition at line 62 of file ftGroupGeom.h.

## 29.146.3 Member Function Documentation

### 29.146.3.1 void Go::ftGroupGeom::addGeomObj ( shared\_ptr< GeomObject > *obj* ) `[inline]`

Add a new geometry entity to the group.

Definition at line 86 of file ftGroupGeom.h.

### 29.146.3.2 ftTangPriority Go::ftGroupGeom::getType ( ) const `[inline]`

Whether the group of objects (surfaces) are a master, a slave or not specified. Related to tangent plane continuity between groups of surfaces

Definition at line 76 of file ftGroupGeom.h.

### 29.146.3.3 shared\_ptr<GeomObject> Go::ftGroupGeom::operator[] ( int *idx* ) const `[inline]`

Fetch object number *idx*.

Definition at line 66 of file ftGroupGeom.h.

### 29.146.3.4 void Go::ftGroupGeom::setType ( ftTangPriority *type* ) `[inline]`

Set whether the group of objects (surfaces) are a master, a slave or not specified. Related to tangent plane continuity between groups of surfaces

Definition at line 82 of file ftGroupGeom.h.

### 29.146.3.5 int Go::ftGroupGeom::size ( ) const `[inline]`

The number of objects in the group.

Definition at line 70 of file ftGroupGeom.h.

## 29.146.4 Member Data Documentation

29.146.4.1 `std::vector<shared_ptr<GeomObject>>` `Go::ftGroupGeom::geomobj_` [protected]

Definition at line 91 of file `ftGroupGeom.h`.

29.146.4.2 `ftTangPriority` `Go::ftGroupGeom::type_` [protected]

Definition at line 92 of file `ftGroupGeom.h`.

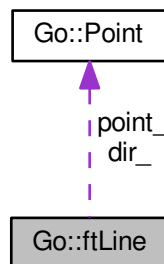
The documentation for this class was generated from the following file:

- `igeslib/include/GoTools/igeslib/ftGroupGeom.h`

## 29.147 Go::ftLine Class Reference

```
#include <ftLine.h>
```

Collaboration diagram for `Go::ftLine`:



### Public Member Functions

- `ftLine ()`  
*Default constructor.*
- `ftLine (const Point &dir, const Point &pnt)`
- `~ftLine ()`  
*Destructor.*
- `void getTwoPlanes (ftPlane &plane1, ftPlane &plane2) const`  
*Find two planes intersecting in this line.*
- `bool closeToScaledPoint (const Point &pt, const Point &scale, double dist2) const`
- `ftLine scaled (const Point &scale) const`  
*Return a new line corresponding to this line after a rescaling of the coordinates.*
- `bool intersectsSphereOfBox (const BoundingBox &box) const`
- `bool intersectsBox (const BoundingBox &box) const`  
*Determine if the line intersects a box.*
- `bool planesIntersectBox (const BoundingBox &box) const`
- `const Point &direction () const`  
*The direction of this line.*
- `const Point &point () const`  
*A point on this line.*

## Protected Attributes

- [Point dir\\_](#)
- [Point point\\_](#)

### 29.147.1 Detailed Description

[ftLine](#) - A line

Definition at line 59 of file ftLine.h.

### 29.147.2 Constructor & Destructor Documentation

#### 29.147.2.1 `Go::ftLine::ftLine ( )` [inline]

Default constructor.

Definition at line 70 of file ftLine.h.

#### 29.147.2.2 `Go::ftLine::ftLine ( const Point & dir, const Point & pnt )` [inline]

Constructor. First parameter is a vector normal to the plane, second parameter is a point in the plane.

Definition at line 76 of file ftLine.h.

#### 29.147.2.3 `Go::ftLine::~~ftLine ( )`

Destructor.

### 29.147.3 Member Function Documentation

#### 29.147.3.1 `bool Go::ftLine::closeToScaledPoint ( const Point & pt, const Point & scale, double dist2 ) const`

Determine if this line comes close enough to a given point after a rescaling of pt's coordinates

#### 29.147.3.2 `const Point& Go::ftLine::direction ( ) const` [inline]

The direction of this line.

Definition at line 108 of file ftLine.h.

#### 29.147.3.3 `void Go::ftLine::getTwoPlanes ( ftPlane & plane1, ftPlane & plane2 ) const`

Find two planes intersecting in this line.

29.147.3.4 **bool** Go::ftLine::intersectsBox ( **const** BoundingBox & *box* ) **const**

Determine if the line intersects a box.

29.147.3.5 **bool** Go::ftLine::intersectsSphereOfBox ( **const** BoundingBox & *box* ) **const**

Determine if the line intersects the smallest eliposide containing a box Is about 2 times faster than intersectsBox, but returns true about 50% more often than intersectsBox

29.147.3.6 **bool** Go::ftLine::planesIntersectBox ( **const** BoundingBox & *box* ) **const**

Determine if two surfaces of the line intersect a box Returns true about 45% more often than intersectsBox

29.147.3.7 **const Point&** Go::ftLine::point ( ) **const** [inline]

A point on this line.

Definition at line 110 of file ftLine.h.

29.147.3.8 **ftLine** Go::ftLine::scaled ( **const** Point & *scale* ) **const**

Return a new line corresponding to this line after a rescaling of the coordinates.

## 29.147.4 Member Data Documentation

29.147.4.1 **Point** Go::ftLine::dir\_ [protected]

Definition at line 65 of file ftLine.h.

29.147.4.2 **Point** Go::ftLine::point\_ [protected]

Definition at line 66 of file ftLine.h.

The documentation for this class was generated from the following file:

- [compositemodel/include/GoTools/compositemodel/ftLine.h](#)

## 29.148 Go::ftMessage Class Reference

```
#include <ftMessage.h>
```

## Public Member Functions

- [ftMessage](#) ()  
*Constructor.*
- [ftMessage](#) ([ftmessages](#) message)  
*Constructor.*
- [~ftMessage](#) ()  
*Destructor.*
- void [setError](#) ([ftmessages](#) error)  
*Set message.*
- void [addWarning](#) ([ftmessages](#) warning)  
*Add a warning.*
- bool [isOk](#) ()  
*Check for error and warning messages.*
- [ftmessages](#) [getMessage](#) ()  
*Return current message.*
- int [noOfWarnings](#) ()  
*Check number of warnings.*
- [ftmessages](#) [getWarning](#) (int i)  
*Return requested warning.*

### 29.148.1 Detailed Description

[ftMessage](#) - Error and warning messages. Used only in some contexts

Definition at line 92 of file [ftMessage.h](#).

### 29.148.2 Constructor & Destructor Documentation

#### 29.148.2.1 [Go::ftMessage::ftMessage](#) ( ) [[inline](#)]

Constructor.

Definition at line 97 of file [ftMessage.h](#).

#### 29.148.2.2 [Go::ftMessage::ftMessage](#) ( [ftmessages](#) message ) [[inline](#)]

Constructor.

Definition at line 102 of file [ftMessage.h](#).

#### 29.148.2.3 [Go::ftMessage::~ftMessage](#) ( ) [[inline](#)]

Destructor.

Definition at line 107 of file [ftMessage.h](#).

### 29.148.3 Member Function Documentation

29.148.3.1 `void Go::ftMessage::addWarning ( ftmessages warning )` [inline]

Add a warning.

Definition at line 114 of file `ftMessage.h`.

29.148.3.2 `ftmessages Go::ftMessage::getMessage ( )` [inline]

Return current message.

Definition at line 123 of file `ftMessage.h`.

29.148.3.3 `ftmessages Go::ftMessage::getWarning ( int i )` [inline]

Return requested warning.

Definition at line 131 of file `ftMessage.h`.

29.148.3.4 `bool Go::ftMessage::isOK ( )` [inline]

Check for error and warning messages.

Definition at line 119 of file `ftMessage.h`.

29.148.3.5 `int Go::ftMessage::noOfWarnings ( )` [inline]

Check number of warnings.

Definition at line 127 of file `ftMessage.h`.

29.148.3.6 `void Go::ftMessage::setError ( ftmessages error )` [inline]

Set message.

Definition at line 110 of file `ftMessage.h`.

The documentation for this class was generated from the following file:

- [compositemodel/include/GoTools/compositemodel/ftMessage.h](#)

### 29.149 Go::ftPlanarGraph Class Reference

```
#include <ftPlanarGraph.h>
```



## Public Member Functions

- [ftPlanarGraph](#) ()  
*Constructor.*
- [ftPlanarGraph](#) (std::vector< [ftSamplePoint](#) \* > &nodes)
- [~ftPlanarGraph](#) ()  
*Destructor.*
- void [setGraph](#) (std::vector< [ftSamplePoint](#) \* > &nodes)
- shared\_ptr< [ftFaceBase](#) > [locateInGraph](#) (double u, double v) **const**  
*Given input of parameter points, return the corresponding ftFaceBase.*
- void [getLocalParameters](#) (double &u, double &v, shared\_ptr< [ftFaceBase](#) > &face) **const**

### 29.149.1 Detailed Description

The chosen structure of graph is, in essence, taken from "Computational Geometry, 2.2.2.1, F. P. Preparata & M. I. Shamos, 1985", and is well suited for fast searching of points inside graph.

Definition at line 123 of file ftPlanarGraph.h.

### 29.149.2 Constructor & Destructor Documentation

#### 29.149.2.1 Go::ftPlanarGraph::ftPlanarGraph ( )

Constructor.

#### 29.149.2.2 Go::ftPlanarGraph::ftPlanarGraph ( std::vector< ftSamplePoint \* > & nodes )

Warning! points are really of type ftSurfaceSetPoint! We cast... We expect that every pair of points differ in their parameter values.

#### 29.149.2.3 Go::ftPlanarGraph::~~ftPlanarGraph ( )

Destructor.

### 29.149.3 Member Function Documentation

#### 29.149.3.1 void Go::ftPlanarGraph::getLocalParameters ( double & u, double & v, shared\_ptr< ftFaceBase > & face ) **const**

Return local parameter values in corresponding patch. We locate a bounding trapezoid inside a patch, and then make a qualified guess from corner values.

#### 29.149.3.2 shared\_ptr<ftFaceBase> Go::ftPlanarGraph::locateInGraph ( double u, double v ) **const**

Given input of parameter points, return the corresponding [ftFaceBase](#).

29.149.3.3 void Go::ftPlanarGraph::setGraph ( std::vector< ftSamplePoint \* > & nodes )

Warning! points are really of type ftSurfaceSetPoint! We cast... We expect that every pair of points differ in their parameter values. We only use those points which are on a subsurface-boundary.

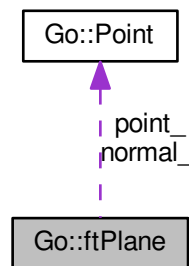
The documentation for this class was generated from the following file:

- [compositemodel/include/GoTools/compositemodel/ftPlanarGraph.h](#)

## 29.150 Go::ftPlane Class Reference

```
#include <ftPlane.h>
```

Collaboration diagram for Go::ftPlane:



### Public Member Functions

- [ftPlane \(\)](#)
- [ftPlane \(const Point &n, const Point &p\)](#)
- [ftPlane \(const Point &x, const Point &y, const Point &z\)](#)
- [ftPlane \(const ftPlane &plane\)](#)  
*Copy constructor.*
- [~ftPlane \(\)](#)  
*Destructor.*
- [const Point & normal \(\) const](#)  
*Get plane normal.*
- [const Point & point \(\) const](#)  
*Get point in plane.*
- [bool intersectsBox \(const BoundingBox &box\) const](#)  
*Check if the plane intersects a bounding box.*

### Protected Attributes

- [Point normal\\_](#)
- [Point point\\_](#)

### 29.150.1 Detailed Description

[ftPlane](#) - Representing a plane

Author

Atgeirr F Rasmussen [atgeirr@sintef.no](mailto:atgeirr@sintef.no)

Definition at line 60 of file ftPlane.h.

### 29.150.2 Constructor & Destructor Documentation

29.150.2.1 `Go::ftPlane::ftPlane ( )` [\[inline\]](#)

Definition at line 71 of file ftPlane.h.

29.150.2.2 `Go::ftPlane::ftPlane ( const Point & n, const Point & p )` [\[inline\]](#)

Constructor. First parameter is a vector normal to the plane, second parameter is a point in the plane.

Definition at line 77 of file ftPlane.h.

29.150.2.3 `Go::ftPlane::ftPlane ( const Point & x, const Point & y, const Point & z )` [\[inline\]](#)

Constructor. Parameters are three points in the plane

Definition at line 84 of file ftPlane.h.

29.150.2.4 `Go::ftPlane::ftPlane ( const ftPlane & plane )`

Copy constructor.

29.150.2.5 `Go::ftPlane::~~ftPlane ( )` [\[inline\]](#)

Destructor.

Definition at line 95 of file ftPlane.h.

### 29.150.3 Member Function Documentation

29.150.3.1 `bool Go::ftPlane::intersectsBox ( const BoundingBox & box ) const`

Check if the plane intersects a bounding box.

29.150.3.2 `const Point& Go::ftPlane::normal( )const` [inline]

Get plane normal.

Definition at line 98 of file ftPlane.h.

29.150.3.3 `const Point& Go::ftPlane::point( )const` [inline]

Get point in plane.

Definition at line 100 of file ftPlane.h.

#### 29.150.4 Member Data Documentation

29.150.4.1 `Point Go::ftPlane::normal_` [protected]

Definition at line 66 of file ftPlane.h.

29.150.4.2 `Point Go::ftPlane::point_` [protected]

Definition at line 67 of file ftPlane.h.

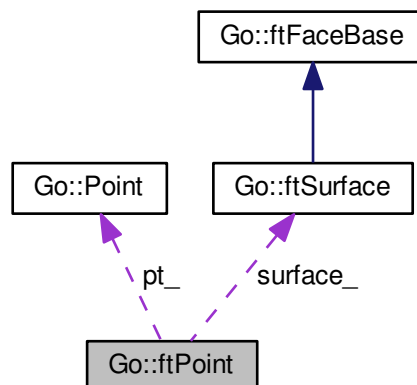
The documentation for this class was generated from the following file:

- [compositemodel/include/GoTools/compositemodel/ftPlane.h](#)

### 29.151 Go::ftPoint Class Reference

```
#include <ftPoint.h>
```

Collaboration diagram for Go::ftPoint:



## Public Member Functions

- [ftPoint](#) ([Point](#) pt, [ftSurface](#) \*sf=0, [double](#) u=0, [double](#) v=0)
- [ftPoint](#) ([double](#) x, [double](#) y, [double](#) z)
 

*Constructor. Point in 3d given by space coordinates.*
- [const Point](#) & [position](#) () [const](#)

*Position of point.*
- [ftSurface](#) \* [face](#) () [const](#)

*Associated face/surface.*
- [double](#) [u](#) () [const](#)
- [double](#) [v](#) () [const](#)
- [Point](#) [normal](#) () [const](#)

*Normal of surface in point.*

## Protected Attributes

- [Point](#) pt\_
- [ftSurface](#) \* [surface\\_](#)
- [double](#) u\_
- [double](#) v\_

### 29.151.1 Detailed Description

[ftPoint](#) - represents a point, possibly lying on a surface [ftPoint](#) contains representations for both a 3D point in space and possibly also parameters for the point on a specific surface.

#### Author

Atgeirr F Rasmussen [atgeirr@sintef.no](mailto:atgeirr@sintef.no)

Definition at line 59 of file [ftPoint.h](#).

### 29.151.2 Constructor & Destructor Documentation

29.151.2.1 [Go::ftPoint::ftPoint](#) ( [Point](#) pt, [ftSurface](#) \* sf=0, [double](#) u=0, [double](#) v=0 ) [\[inline\]](#)

#### Constructor

##### Parameters

|           |                                                    |
|-----------|----------------------------------------------------|
| <i>pt</i> | <a href="#">Point</a>                              |
| <i>sf</i> | Associated surface                                 |
| <i>u</i>  | First parameter in surface corresponding to point  |
| <i>v</i>  | Second parameter in surface corresponding to point |

Definition at line 72 of file [ftPoint.h](#).

29.151.2.2 `Go::ftPoint::ftPoint ( double x, double y, double z ) [inline]`

Constructor. [Point](#) in 3d given by space coordinates.

Definition at line 75 of file ftPoint.h.

### 29.151.3 Member Function Documentation

29.151.3.1 `ftSurface* Go::ftPoint::face ( ) const [inline]`

Associated face/surface.

Definition at line 80 of file ftPoint.h.

29.151.3.2 `Point Go::ftPoint::normal ( ) const`

Normal of surface in point.

29.151.3.3 `const Point& Go::ftPoint::position ( ) const [inline]`

Position of point.

Definition at line 78 of file ftPoint.h.

29.151.3.4 `double Go::ftPoint::u ( ) const [inline]`

1. parameter in surface

Definition at line 82 of file ftPoint.h.

29.151.3.5 `double Go::ftPoint::v ( ) const [inline]`

1. parameter in surface

Definition at line 84 of file ftPoint.h.

### 29.151.4 Member Data Documentation

29.151.4.1 `Point Go::ftPoint::pt_ [protected]`

Definition at line 62 of file ftPoint.h.

29.151.4.2 `ftSurface*` `Go::ftPoint::surface_` [protected]

Definition at line 63 of file `ftPoint.h`.

29.151.4.3 `double` `Go::ftPoint::u_` [protected]

Definition at line 64 of file `ftPoint.h`.

29.151.4.4 `double` `Go::ftPoint::v_` [protected]

Definition at line 64 of file `ftPoint.h`.

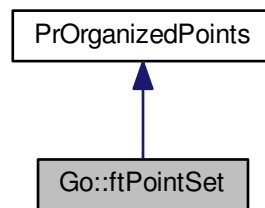
The documentation for this class was generated from the following file:

- `compositemodel/include/GoTools/compositemodel/ftPoint.h`

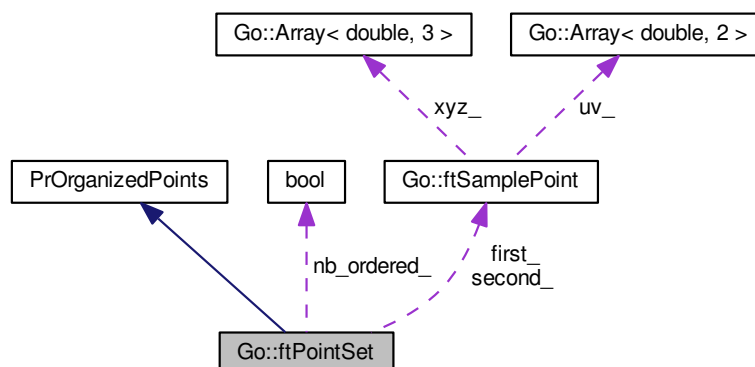
## 29.152 Go::ftPointSet Class Reference

```
#include <ftPointSet.h>
```

Inheritance diagram for `Go::ftPointSet`:



Collaboration diagram for `Go::ftPointSet`:



## Public Member Functions

- [ftPointSet](#) ()  
*Constructor.*
- virtual [~ftPointSet](#) ()  
*Destructor.*
- int [size](#) () **const**  
*Get the number of points.*
- [ftSamplePoint \\* operator\[\]](#) (int idx)  
*Operator overload [].*
- **const** [ftSamplePoint \\* operator\[\]](#) (int idx) **const**  
*Operator overload [].*
- [PointIter addEntry](#) (shared\_ptr< [ftSamplePoint](#) > point)  
*Add a new point to the point set.*
- void [removePoint](#) ([PointIter](#) point)  
*Remove a point from the point set.*
- void [setFirst](#) ([PointIter](#) point)  
*Mark the first point around the boundary of this point set.*
- void [setSecond](#) ([PointIter](#) point)  
*Mark the second point around the boundary of this point set.*
- [PointIter lastAdded](#) ()  
*Mark the last point added to this point set.*
- **double** [getMaxDist](#) () **const**  
*Return the maximum distance in the pointset.*
- **double** [getMeanDist](#) () **const**  
*Return the medium distance in the pointset.*
- void [computeParametricDist](#) (shared\_ptr< [ParamSurface](#) > surf)
- void [computeDist](#) (shared\_ptr< [ParamSurface](#) > surf)
- void [computeDistAndRepar](#) (shared\_ptr< [ParamSurface](#) > surf)
- **double** [reparBdy](#) (shared\_ptr< [ParamSurface](#) > surf, **bool** use\_seed=true)  
*Reparameterize points at the boundary of the point set.*
- **double** [reparInnerPoints](#) (shared\_ptr< [ParamSurface](#) > surf, **bool** use\_seed=true)  
*Reparameterize points in the inner of the point set.*
- void [orderNeighbours](#) ()  
*Reorganize the sequence of neighbours.*
- void [append](#) (shared\_ptr< [ftPointSet](#) > triang)
- void [cleanNodeIdentity](#) (**double** tol)  
*Remove identical boundary nodes.*
- void [mergeBoundary](#) (shared\_ptr< [ftFaceBase](#) > face1, int range1\_idx1, int range1\_idx2, shared\_ptr< [ftFaceBase](#) > face2, int range2\_idx1, int range2\_idx2, **double** eps)
- void [identifyBdPnts](#) (std::vector< [Point](#) > &points, std::vector< int > &pnt\_ix)  
*Fetch index of specified points at the boundary (closest to given point)*
- void [getTriangles](#) (std::vector< std::vector< int > > &triangles) **const**  
*Fetch all triangles in the connectivity graph.*
- void [getPoints](#) (std::vector< [Vector3D](#) > &positions) **const**  
*Get the position of all points.*
- void [checkAndUpdateTriangCorners](#) ()  
*Avoid boundary points being connected to three other boundary points.*
- void [getOrientedTriangles](#) (std::vector< std::vector< int > > &triangles)
- void [write](#) (std::ostream &os) **const**  
*Write point set to stream.*



- void [write2D](#) (std::ostream &os) **const**  
*Write parameter points to stream.*
- void [printPoints](#) (std::ostream &os) **const**  
*Debug.*
- virtual int [getNumNodes](#) () **const**  
*Number of points in point set.*
- virtual [Vector3D](#) [get3dNode](#) (int i) **const**  
*Get content of point number i.*
- virtual void [set3dNode](#) (int i, **const** [Vector3D](#) &p)  
*Change content on point number i.*
- virtual void [getNeighbours](#) (int i, std::vector< int > &neighbours) **const**  
*Fetch all neighbours to point number i.*
- virtual **bool** [isBoundary](#) (int i) **const**  
*Check if point number i lies at the boundary of the point set.*
- virtual **double** [getU](#) (int i) **const**  
*Fetch 1. parameter of point number i.*
- virtual **double** [getV](#) (int i) **const**  
*Fetch 2. parameter of point number i.*
- virtual void [setU](#) (int i, **double** u)  
*Set 1. parameter of point number i.*
- virtual void [setV](#) (int i, **double** v)  
*Set 2. parameter of point number i.*

### Protected Attributes

- [PointList](#) [points\\_](#)
- std::vector< [PointIter](#) > [index\\_to\\_iter\\_](#)
- **bool** [nb\\_ordered\\_](#)
- [PointIter](#) [first\\_](#)
- [PointIter](#) [second\\_](#)

### Additional Inherited Members

#### 29.152.1 Detailed Description

[ftPointSet](#) - A set of sample points used as data in surface approximation

Definition at line 181 of file [ftPointSet.h](#).

#### 29.152.2 Constructor & Destructor Documentation

##### 29.152.2.1 Go::ftPointSet::ftPointSet ( )

Constructor.

##### 29.152.2.2 virtual Go::ftPointSet::~~ftPointSet ( ) [virtual]

Destructor.

### 29.152.3 Member Function Documentation

29.152.3.1 `PointIter Go::ftPointSet::addEntry ( shared_ptr< ftSamplePoint > point ) [inline]`

Add a new point to the point set.

Definition at line 209 of file ftPointSet.h.

29.152.3.2 `void Go::ftPointSet::append ( shared_ptr< ftPointSet > triang )`

Extend the point set with points from another set. Avoid points with no connectivity

29.152.3.3 `void Go::ftPointSet::checkAndUpdateTriangCorners ( )`

Avoid boundary points being connected to three other boundary points.

29.152.3.4 `void Go::ftPointSet::cleanNodeIdentity ( double tol )`

Remove identical boundary nodes.

29.152.3.5 `void Go::ftPointSet::computeDist ( shared_ptr< ParamSurface > surf )`

Compute the distances from the points in the point set to the given surface.

29.152.3.6 `void Go::ftPointSet::computeDistAndRepar ( shared_ptr< ParamSurface > surf )`

Compute the distances from the points in the point set to the given surface, and reparametrize.

29.152.3.7 `void Go::ftPointSet::computeParametricDist ( shared_ptr< ParamSurface > surf )`

Compute the distances from the points in the point set to the given surface, at the same parameter value.

29.152.3.8 `virtual Vector3D Go::ftPointSet::get3dNode ( int i ) const [virtual]`

Get content of point number i.

Implements [PrOrganizedPoints](#).

29.152.3.9 `double Go::ftPointSet::getMaxDist ( ) const`

Return the maximum distance in the pointset.

29.152.3.10 `double Go::ftPointSet::getMeanDist ( ) const`

Return the medium distance in the pointset.

29.152.3.11 `virtual void Go::ftPointSet::getNeighbours ( int i, std::vector< int > & neighbours ) const [virtual]`

Fetch all neighbours to point number i.

Implements [PrOrganizedPoints](#).

29.152.3.12 `virtual int Go::ftPointSet::getNumNodes ( ) const [virtual]`

Number of points in point set.

Implements [PrOrganizedPoints](#).

29.152.3.13 `void Go::ftPointSet::getOrientedTriangles ( std::vector< std::vector< int > > & triangles )`

Fetch all triangles in the connectivity graph and make sure that the triangle orientation is consistent, i.e. opposite directions of edges between the same two nodes

29.152.3.14 `void Go::ftPointSet::getPoints ( std::vector< Vector3D > & positions ) const`

Get the position of all points.

29.152.3.15 `void Go::ftPointSet::getTriangles ( std::vector< std::vector< int > > & triangles ) const`

Fetch all triangles in the connectivity graph.

29.152.3.16 `virtual double Go::ftPointSet::getU ( int i ) const [virtual]`

Fetch 1. parameter of point number i.

Implements [PrOrganizedPoints](#).

29.152.3.17 `virtual double Go::ftPointSet::getV ( int i ) const [virtual]`

Fetch 2. parameter of point number i.

Implements [PrOrganizedPoints](#).

29.152.3.18 `void Go::ftPointSet::identifyBdPnts ( std::vector< Point > & points, std::vector< int > & pnt_ix )`

Fetch index of specified points at the boundary (closest to given point)

29.152.3.19 `virtual bool Go::ftPointSet::isBoundary ( int i ) const [virtual]`

Check if point number *i* lies at the boundary of the point set.

Implements [PrOrganizedPoints](#).

29.152.3.20 `PointIter Go::ftPointSet::lastAdded ( ) [inline]`

Mark the last point added to this point set.

Definition at line 234 of file `ftPointSet.h`.

29.152.3.21 `void Go::ftPointSet::mergeBoundary ( shared_ptr< ftFaceBase > face1, int range1_idx1, int range1_idx2, shared_ptr< ftFaceBase > face2, int range2_idx1, int range2_idx2, double eps )`

Given two faces, rearrange points on the common boundary between these two faces

29.152.3.22 `ftSamplePoint* Go::ftPointSet::operator[] ( int idx ) [inline]`

Operator overload [].

Definition at line 196 of file `ftPointSet.h`.

29.152.3.23 `const ftSamplePoint* Go::ftPointSet::operator[] ( int idx ) const [inline]`

Operator overload [].

Definition at line 203 of file `ftPointSet.h`.

29.152.3.24 `void Go::ftPointSet::orderNeighbours ( )`

Reorganize the sequence of neighbours.

29.152.3.25 `void Go::ftPointSet::printPoints ( std::ostream & os ) const`

Debug.

29.152.3.26 `void Go::ftPointSet::removePoint ( PointIter point )`

Remove a point from the point set.

29.152.3.27 `double Go::ftPointSet::reparBdy ( shared_ptr< ParamSurface > surf, bool use_seed = true )`

Reparameterize points at the boundary of the point set.

29.152.3.28 `double Go::ftPointSet::reparInnerPoints ( shared_ptr< ParamSurface > surf, bool use_seed = true )`

Reparameterize points in the inner of the point set.

29.152.3.29 `virtual void Go::ftPointSet::set3dNode ( int i, const Vector3D & p ) [virtual]`

Change content on point number i.

29.152.3.30 `void Go::ftPointSet::setFirst ( PointIter point ) [inline]`

Mark the first point around the boundary of this point set.

Definition at line 222 of file ftPointSet.h.

29.152.3.31 `void Go::ftPointSet::setSecond ( PointIter point ) [inline]`

Mark the second point around the boundary of this point set.

Definition at line 228 of file ftPointSet.h.

29.152.3.32 `virtual void Go::ftPointSet::setU ( int i, double u ) [virtual]`

Set 1. parameter of point number i.

Implements [PrOrganizedPoints](#).

29.152.3.33 `virtual void Go::ftPointSet::setV ( int i, double v ) [virtual]`

Set 2. parameter of point number i.

Implements [PrOrganizedPoints](#).

29.152.3.34 `int Go::ftPointSet::size ( ) const [inline]`

Get the number of points.

Definition at line 192 of file ftPointSet.h.

29.152.3.35 `void Go::ftPointSet::write ( std::ostream & os ) const`

Write point set to stream.

29.152.3.36 `void Go::ftPointSet::write2D ( std::ostream & os ) const`

Write parameter points to stream.

## 29.152.4 Member Data Documentation

29.152.4.1 `PointIter Go::ftPointSet::first_ [protected]`

Definition at line 326 of file `ftPointSet.h`.

29.152.4.2 `std::vector<PointIter> Go::ftPointSet::index_to_iter_ [protected]`

Definition at line 324 of file `ftPointSet.h`.

29.152.4.3 `bool Go::ftPointSet::nb_ordered_ [mutable], [protected]`

Definition at line 325 of file `ftPointSet.h`.

29.152.4.4 `PointList Go::ftPointSet::points_ [protected]`

Definition at line 323 of file `ftPointSet.h`.

29.152.4.5 `PointIter Go::ftPointSet::second_ [protected]`

Definition at line 327 of file `ftPointSet.h`.

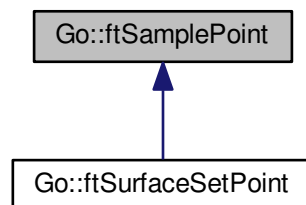
The documentation for this class was generated from the following file:

- `compositemodel/include/GoTools/compositemodel/ftPointSet.h`

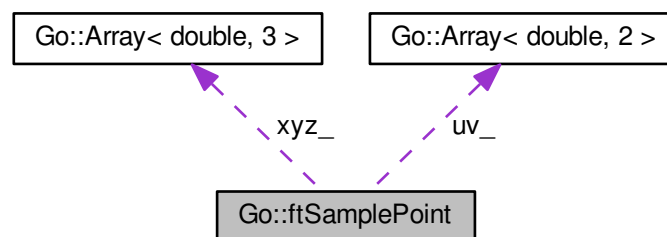
## 29.153 Go::ftSamplePoint Class Reference

```
#include <ftPointSet.h>
```

Inheritance diagram for Go::ftSamplePoint:



Collaboration diagram for Go::ftSamplePoint:



### Public Member Functions

- `ftSamplePoint` (`Vector3D xyz`, `int bnd`)  
*Constructor.*
- `virtual ~ftSamplePoint` ()  
*Destructor.*
- `void setPoint` (`Vector3D xyz`)  
*Set point.*
- `void setPar` (`Vector2D uv`)  
*Set parameter value.*
- `void setDist` (`double d`)  
*Set dist.*
- `void setIndex` (`int i`)  
*Set index.*

- void [setBoundary](#) (int bd\_info)  
*Set boundary information.*
- virtual [ftSurfaceSetPoint](#) \* [asSurfaceSetPoint](#) ()  
*Return pointer to sub class entity if the point is of that type.*
- virtual [bool](#) [containsFace](#) ([ftFaceBase](#) \*face) [const](#)  
*Check if a face is associated this point.*
- void [addNeighbour](#) ([PointIter](#) next)  
*Add a new neighbouring point.*
- void [removeNeighbour](#) ([PointIter](#) neighbour)  
*If neighbour exists it is removed from neighbour vector.*
- [bool](#) [isOnBoundary](#) () [const](#)  
*Am I on the boundary?*
- [bool](#) [isOnSubSurfaceBoundary](#) () [const](#)  
*Am I on sub surface boundary?*
- int [getNmbNeighbour](#) () [const](#)  
*Get number of neighbours.*
- [const](#) std::vector< [PointIter](#) > & [getNeighbours](#) () [const](#)  
*Fetch all neighbouring points.*
- [bool](#) [isConnected](#) ([PointIter](#) pnt)  
*Check if the point pnt is a neighbour to this point.*
- [Vector3D](#) [getPoint](#) () [const](#)  
*Return 3D point value.*
- [Vector2D](#) [getPar](#) () [const](#)  
*Return parameter value.*
- [double](#) [getDist](#) () [const](#)  
*Return distance.*
- int [getIndex](#) () [const](#)  
*Return index.*
- void [orderNeighbours](#) ([ftSamplePoint](#) \*nextpoint, [bool](#) forward)  
*Order the neighbours for one given point.*
- [PointIter](#) [getFirstNeighbour](#) ()  
*Get first neighbour.*
- [double](#) [pntDist](#) ([ftSamplePoint](#) \*other) [const](#)  
*Distance between sample points.*
- void [getAttachedTriangles](#) (std::vector< std::vector< int > > &triangles) [const](#)  
*Fetch all triangles containing this point.*
- virtual void [write2Dval](#) (std::ostream &os) [const](#)  
*Debug.*

## Protected Attributes

- [Vector3D](#) xyz\_
- [Vector2D](#) uv\_
- [double](#) dist\_
- int index\_
- int at\_boundary\_
- std::vector< [PointIter](#) > next\_



### 29.153.1 Detailed Description

[ftSamplePoint](#) - One point in a set of sample points

Definition at line 69 of file ftPointSet.h.

### 29.153.2 Constructor & Destructor Documentation

#### 29.153.2.1 Go::ftSamplePoint::ftSamplePoint ( **Vector3D** *xyz*, int *bn* )

Constructor.

#### 29.153.2.2 virtual Go::ftSamplePoint::~ftSamplePoint ( ) [inline],[virtual]

Destructor.

Definition at line 76 of file ftPointSet.h.

### 29.153.3 Member Function Documentation

#### 29.153.3.1 void Go::ftSamplePoint::addNeighbour ( **PointIter** *next* )

Add a new neighbouring point.

#### 29.153.3.2 virtual **ftSurfaceSetPoint\*** Go::ftSamplePoint::asSurfaceSetPoint ( ) [inline],[virtual]

Return pointer to sub class entity if the point is of that type.

Reimplemented in [Go::ftSurfaceSetPoint](#).

Definition at line 97 of file ftPointSet.h.

#### 29.153.3.3 virtual **bool** Go::ftSamplePoint::containsFace ( **ftFaceBase** \* *face* ) const [inline],[virtual]

Check if a face is associated this point.

Reimplemented in [Go::ftSurfaceSetPoint](#).

Definition at line 103 of file ftPointSet.h.

#### 29.153.3.4 void Go::ftSamplePoint::getAttachedTriangles ( **std::vector**< **std::vector**< int > > & *triangles* ) const

Fetch all triangles containing this point.

29.153.3.5 **double** Go::ftSamplePoint::getDist ( ) const [inline]

Return distance.

Definition at line 142 of file ftPointSet.h.

29.153.3.6 **PointIter** Go::ftSamplePoint::getFirstNeighbour ( ) [inline]

Get first neighbour.

Definition at line 150 of file ftPointSet.h.

29.153.3.7 **int** Go::ftSamplePoint::getIndex ( ) const [inline]

Return index.

Definition at line 145 of file ftPointSet.h.

29.153.3.8 **const** std::vector<PointIter>& Go::ftSamplePoint::getNeighbours ( ) const [inline]

Fetch all neighbouring points.

Definition at line 123 of file ftPointSet.h.

29.153.3.9 **int** Go::ftSamplePoint::getNmbNeighbour ( ) const [inline]

Get number of neighbours.

Definition at line 120 of file ftPointSet.h.

29.153.3.10 **Vector2D** Go::ftSamplePoint::getPar ( ) const [inline]

Return parameter value.

Definition at line 139 of file ftPointSet.h.

29.153.3.11 **Vector3D** Go::ftSamplePoint::getPoint ( ) const [inline]

Return 3D point value.

Definition at line 136 of file ftPointSet.h.

29.153.3.12 **bool** Go::ftSamplePoint::isConnected ( PointIter *pnt* ) [inline]

Check if the point *pnt* is a neighbour to this point.

Definition at line 127 of file ftPointSet.h.

29.153.3.13 **bool** Go::ftSamplePoint::isOnBoundary ( ) const [inline]

Am I on the boundary?

Definition at line 114 of file ftPointSet.h.

29.153.3.14 **bool** Go::ftSamplePoint::isOnSubSurfaceBoundary ( ) const [inline]

Am I on sub surface boundary?

Definition at line 117 of file ftPointSet.h.

29.153.3.15 **void** Go::ftSamplePoint::orderNeighbours ( ftSamplePoint \* *nextpoint*, **bool** *forward* )

Order the neighbours for one given point.

29.153.3.16 **double** Go::ftSamplePoint::pntDist ( ftSamplePoint \* *other* ) const

Distance between sample points.

29.153.3.17 **void** Go::ftSamplePoint::removeNeighbour ( PointIter *neighbour* )

If neighbour exists it is removed from neighbour vector.

29.153.3.18 **void** Go::ftSamplePoint::setBoundary ( int *bd\_info* ) [inline]

Set boundary information.

Definition at line 91 of file ftPointSet.h.

29.153.3.19 **void** Go::ftSamplePoint::setDist ( double *d* ) [inline]

Set dist.

Definition at line 85 of file ftPointSet.h.

29.153.3.20 **void** Go::ftSamplePoint::setIndex ( int *i* ) [inline]

Set index.

Definition at line 88 of file ftPointSet.h.

29.153.3.21 `void Go::ftSamplePoint::setPar ( Vector2D uv ) [inline]`

Set parameter value.

Definition at line 82 of file `ftPointSet.h`.

29.153.3.22 `void Go::ftSamplePoint::setPoint ( Vector3D xyz ) [inline]`

Set point.

Definition at line 79 of file `ftPointSet.h`.

29.153.3.23 `virtual void Go::ftSamplePoint::write2Dval ( std::ostream & os ) const [virtual]`

Debug.

Reimplemented in [Go::ftSurfaceSetPoint](#).

#### 29.153.4 Member Data Documentation

29.153.4.1 `int Go::ftSamplePoint::at_boundary_ [protected]`

Definition at line 168 of file `ftPointSet.h`.

29.153.4.2 `double Go::ftSamplePoint::dist_ [protected]`

Definition at line 166 of file `ftPointSet.h`.

29.153.4.3 `int Go::ftSamplePoint::index_ [protected]`

Definition at line 167 of file `ftPointSet.h`.

29.153.4.4 `std::vector<PointIter> Go::ftSamplePoint::next_ [protected]`

Definition at line 170 of file `ftPointSet.h`.

29.153.4.5 `Vector2D Go::ftSamplePoint::uv_ [protected]`

Definition at line 165 of file `ftPointSet.h`.

## 29.153.4.6 Vector3D Go::ftSamplePoint::xyz\_ [protected]

Definition at line 164 of file ftPointSet.h.

The documentation for this class was generated from the following file:

- [compositemodel/include/GoTools/compositemodel/ftPointSet.h](#)

## 29.154 Go::ftSearchNode Class Reference

```
#include <ftPlanarGraph.h>
```

### Public Member Functions

- [ftSearchNode](#) ([ftSurfaceSetPoint](#) \*node)  
*Constructor.*
- [~ftSearchNode](#) ()  
*Destructor.*
- [Vector2D](#) node () const  
*As we allow to set param value of point, function nor return val are const.*
- void [setOrderedSegments](#) (const std::vector< [ftGraphEdge](#) > &edges)
- const std::vector< [ftGraphEdge](#) > & [getOrderedSegments](#) () const

### 29.154.1 Detailed Description

Node inside a graph. A [ftSearchNode](#) contains all edges in graph crossing horizontal  $y_0$ -line (point = (x0, y0)), ordered from left to right, enabling fast searching of points inside graph. We also store [ftFaceBase](#)'s of point.

Definition at line 91 of file ftPlanarGraph.h.

### 29.154.2 Constructor & Destructor Documentation

#### 29.154.2.1 Go::ftSearchNode::ftSearchNode ( [ftSurfaceSetPoint](#) \* node )

Constructor.

#### 29.154.2.2 Go::ftSearchNode::~~ftSearchNode ( )

Destructor.

### 29.154.3 Member Function Documentation

29.154.3.1 `const std::vector<ftGraphEdge> & Go::ftSearchNode::getOrderedSegments ( ) const`

29.154.3.2 `Vector2D Go::ftSearchNode::node ( ) const`

As we allow to set param value of point, function nor return val are const.

29.154.3.3 `void Go::ftSearchNode::setOrderedSegments ( const std::vector< ftGraphEdge > & edges )`

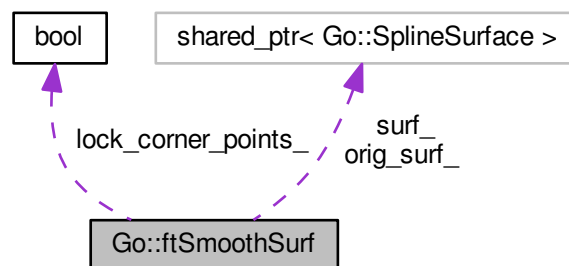
The documentation for this class was generated from the following file:

- [compositemodel/include/GoTools/compositemodel/ftPlanarGraph.h](#)

### 29.155 Go::ftSmoothSurf Class Reference

```
#include <ftSmoothSurf.h>
```

Collaboration diagram for Go::ftSmoothSurf:



#### Public Member Functions

- `ftSmoothSurf` (`shared_ptr< SplineSurface > surf`, `double approx_tol`, `double approx_orig_tol`, `std::vector< int > ccw_edge_derivs`, `int maxiter`, `bool lock_corner_points=false`)
- `~ftSmoothSurf` ()  
*Destructor.*
- `void setSmoothU` (`int k`)
- `void setSmoothV` (`int k`)
- `void refineSurf` (`ftPointSet &points`, `bool reparam=true`)  
*Express the surface to be modified on a refined knot vector.*
- `bool update` (`ftPointSet &points`, `double gapeps`, `bool reparam=true`)
- `void setApproxWeight` (`double weight`)  
*Weight on point approximation.*
- `double getApproxWeight` ()  
*Fetch weight on point approximation.*
- `void getError` (`double &max_error`, `double &mean_error`)  
*Fetch maximum and average error in approximation.*

## Protected Attributes

- `shared_ptr< SplineSurface > surf_`
- `shared_ptr< SplineSurface > orig_surf_`
- `double approx_tol_`
- `double approx_orig_tol_`
- `double init_approx_weight_`
- `std::vector< int > ccw_edge_derivs_`
- `double max_error_`
- `double mean_error_`
- `int maxiter_`
- `bool lock_corner_points_`
- `int seem_[2]`

### 29.155.1 Detailed Description

`ftSmoothSurf` - Surface smoothing and approximation. Interface to `SmoothSurf` in `gotools-core/creators`. An iterative process with parameter iteration and refinement of the initial surface with respect to approximation errors at each step.

Definition at line 61 of file `ftSmoothSurf.h`.

### 29.155.2 Constructor & Destructor Documentation

29.155.2.1 `Go::ftSmoothSurf::ftSmoothSurf ( shared_ptr< SplineSurface > surf, double approx_tol, double approx_orig_tol, std::vector< int > ccw_edge_derivs, int maxiter, bool lock_corner_points = false )`

Constructor

Parameters

|                           |                                                                                                                                                   |
|---------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>surf</i>               | initial surface                                                                                                                                   |
| <i>approx_tol</i>         | tolerance in point approximation                                                                                                                  |
| <i>approx_orig_tol</i>    | allowed deviance from the original surface                                                                                                        |
| <i>ccw_edge_derivs</i>    | number of derivatives to keep fixed along the boundaries, sequence: <code>umin</code> , <code>vmax</code> , <code>umax</code> , <code>vmin</code> |
| <i>maxiter</i>            | maximum allowed number of iterations in approximation                                                                                             |
| <i>lock_corner_points</i> | indicates if the surface corners are fixed                                                                                                        |

29.155.2.2 `Go::ftSmoothSurf::~~ftSmoothSurf ( )`

Destructor.

### 29.155.3 Member Function Documentation

29.155.3.1 `double Go::ftSmoothSurf::getApproxWeight ( ) [inline]`

Fetch weight on point approximation.

Definition at line 101 of file `ftSmoothSurf.h`.

29.155.3.2 `void Go::ftSmoothSurf::getError ( double & max_error, double & mean_error ) [inline]`

Fetch maximum and average error in approximation.

Definition at line 105 of file ftSmoothSurf.h.

29.155.3.3 `void Go::ftSmoothSurf::refineSurf ( ftPointSet & points, bool reparam = true )`

Express the surface to be modified on a refined knot vector.

29.155.3.4 `void Go::ftSmoothSurf::setApproxWeight ( double weight ) [inline]`

Weight on point approximation.

Definition at line 97 of file ftSmoothSurf.h.

29.155.3.5 `void Go::ftSmoothSurf::setSmoothU ( int k )`

Set expected continuity of seem in 1. parameter direction  $0 \leq k \leq 1$ , C0 and C1 continuity possible

29.155.3.6 `void Go::ftSmoothSurf::setSmoothV ( int k )`

Set expected continuity of seem in 2. parameter direction  $0 \leq k \leq 1$ , C0 and C1 continuity possible

29.155.3.7 `bool Go::ftSmoothSurf::update ( ftPointSet & points, double gapeps, bool reparam = true )`

Modify the surface. User may turn off reparametrization of points. If smoothing was a success, true is returned.

#### 29.155.4 Member Data Documentation

29.155.4.1 `double Go::ftSmoothSurf::approx_orig_tol_ [protected]`

Definition at line 115 of file ftSmoothSurf.h.

29.155.4.2 `double Go::ftSmoothSurf::approx_tol_ [protected]`

Definition at line 114 of file ftSmoothSurf.h.

29.155.4.3 `std::vector<int> Go::ftSmoothSurf::ccw_edge_derivs_ [protected]`

Definition at line 117 of file ftSmoothSurf.h.



29.155.4.4 **double** Go::ftSmoothSurf::init\_approx\_weight\_ [protected]

Definition at line 116 of file ftSmoothSurf.h.

29.155.4.5 **bool** Go::ftSmoothSurf::lock\_corner\_points\_ [protected]

Definition at line 122 of file ftSmoothSurf.h.

29.155.4.6 **double** Go::ftSmoothSurf::max\_error\_ [protected]

Definition at line 119 of file ftSmoothSurf.h.

29.155.4.7 **int** Go::ftSmoothSurf::maxiter\_ [protected]

Definition at line 121 of file ftSmoothSurf.h.

29.155.4.8 **double** Go::ftSmoothSurf::mean\_error\_ [protected]

Definition at line 120 of file ftSmoothSurf.h.

29.155.4.9 **shared\_ptr<SplineSurface>** Go::ftSmoothSurf::orig\_surf\_ [protected]

Definition at line 113 of file ftSmoothSurf.h.

29.155.4.10 **int** Go::ftSmoothSurf::seem\_[2] [protected]

Definition at line 123 of file ftSmoothSurf.h.

29.155.4.11 **shared\_ptr<SplineSurface>** Go::ftSmoothSurf::surf\_ [protected]

Definition at line 112 of file ftSmoothSurf.h.

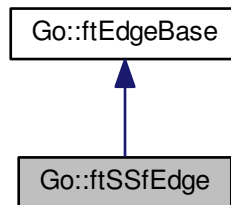
The documentation for this class was generated from the following file:

- compositemodel/include/GoTools/compositemodel/[ftSmoothSurf.h](#)

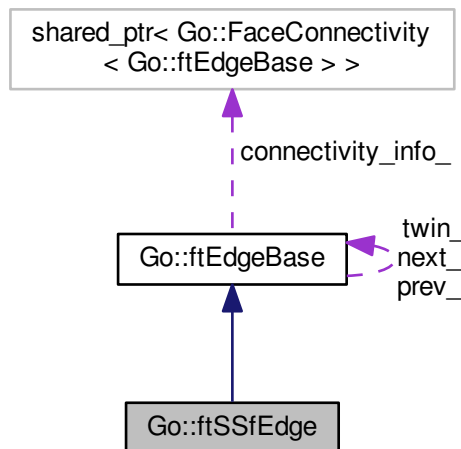
## 29.156 Go::ftSSfEdge Class Reference

```
#include <ftSSfEdge.h>
```

Inheritance diagram for Go::ftSSfEdge:



Collaboration diagram for Go::ftSSfEdge:



### Public Member Functions

- `ftSSfEdge` (`ftFaceBase` \*face, `ftEdge` \*edge, int entry\_id=-1)
- `~ftSSfEdge` ()  
*Empty destructor.*
- virtual `double tMin` () const  
*Start parameter value of edge.*
- virtual `double tMax` () const

- End parameter value of edge.*
- virtual void [setReversed](#) (bool is\_reversed)
- virtual [bool isReversed](#) ()
- virtual void [setFace](#) (ftFaceBase \*face)
- Set face pointer.*
- virtual [ftFaceBase \\* face](#) ()
- The face corresponding to this edge.*
- virtual [BoundingBox boundingBox](#) ()
- Coordinate box surrounding this edge.*
- virtual [ftSSfEdge \\* split](#) (double t)
- virtual int [entryId](#) ()
- Id of edge. Default set to -1.*
- virtual void [setEntryId](#) (int id)
- Set id of edge.*
- virtual [Point point](#) (double t) const
- Evaluate position.*
- virtual [Point tangent](#) (double t) const
- Evaluate tangent.*
- virtual [Point normal](#) (double t) const
- Evaluate normal of associated face.*
- virtual [Point normal](#) (double t, [Point](#) &face\_par\_pt, double \*face\_seed) const
- virtual void [closestPoint](#) (const [Point](#) &pt, double &clo\_t, [Point](#) &clo\_pt, double &clo\_dist, double const \*seed=0) const
- Closest point on this edge to a given point.*
- virtual [ftEdge \\* geomEdge](#) ()
- Return edge pointer.*

## Additional Inherited Members

### 29.156.1 Detailed Description

[ftSSfEdge](#) - topological edge for Fantastic corresponding to [ftSuperSurface](#) Detailed description.

The [ftSSfEdge](#) is a half-edge implementation of a topological data structure. It implements the [ftEdgeBase](#) interface by using the [Go](#) geometry library.

#### Author

Atgeirr F Rasmussen [atgeirr@sintef.no](mailto:atgeirr@sintef.no)

**Bug** Not tested

#### See also

[ftEdgeBase](#)

Definition at line 41 of file [ftSSfEdge.h](#).

## 29.156.2 Constructor & Destructor Documentation

29.156.2.1 `Go::ftSSfEdge::ftSSfEdge ( ftFaceBase * face, ftEdge * edge, int entry_id = -1 )`

Constructor. Detailed description.

29.156.2.2 `Go::ftSSfEdge::~~ftSSfEdge ( )`

Empty destructor.

## 29.156.3 Member Function Documentation

29.156.3.1 `virtual BoundingBox Go::ftSSfEdge::boundingBox ( ) [virtual]`

Coordinate box surrounding this edge.

Implements [Go::ftEdgeBase](#).

29.156.3.2 `virtual void Go::ftSSfEdge::closestPoint ( const Point & pt, double & clo_t, Point & clo_pt, double & clo_dist, double const * seed = 0 ) const [virtual]`

Closest point on this edge to a given point.

Implements [Go::ftEdgeBase](#).

29.156.3.3 `virtual int Go::ftSSfEdge::entryId ( ) [inline],[virtual]`

Id of edge. Default set to -1.

Implements [Go::ftEdgeBase](#).

Definition at line 93 of file ftSSfEdge.h.

29.156.3.4 `virtual ftFaceBase* Go::ftSSfEdge::face ( ) [virtual]`

The face corresponding to this edge.

Implements [Go::ftEdgeBase](#).

29.156.3.5 `virtual ftEdge* Go::ftSSfEdge::geomEdge ( ) [virtual]`

Return edge pointer.

Implements [Go::ftEdgeBase](#).

29.156.3.6 `virtual bool Go::ftSSfEdge::isReversed ( ) [inline],[virtual]`

Get the value of the flag indicating that the orientation of the edge is reversed with respect to the parametrization of the underlying [ParamCurve](#).

#### Returns

*true* if orientation is reversed, *false* otherwise

Implements [Go::ftEdgeBase](#).

Definition at line 69 of file `ftSSfEdge.h`.

29.156.3.7 `virtual Point Go::ftSSfEdge::normal ( double t ) const [virtual]`

Evaluate normal of associated face.

Implements [Go::ftEdgeBase](#).

29.156.3.8 `virtual Point Go::ftSSfEdge::normal ( double t, Point & face_par_pt, double * face_seed ) const [virtual]`

Evaluate normal of associated face given a seed to the search for the corresponding face parameter

Implements [Go::ftEdgeBase](#).

29.156.3.9 `virtual Point Go::ftSSfEdge::point ( double t ) const [virtual]`

Evaluate position.

Implements [Go::ftEdgeBase](#).

29.156.3.10 `virtual void Go::ftSSfEdge::setEntryId ( int id ) [inline],[virtual]`

Set id of edge.

Implements [Go::ftEdgeBase](#).

Definition at line 94 of file `ftSSfEdge.h`.

29.156.3.11 `virtual void Go::ftSSfEdge::setFace ( ftFaceBase * face ) [inline],[virtual]`

Set face pointer.

Implements [Go::ftEdgeBase](#).

Definition at line 74 of file `ftSSfEdge.h`.

29.156.3.12 `virtual void Go::ftSSfEdge::setReversed ( bool is_reversed ) [inline],[virtual]`

Set a flag to indicate that the orientation of the edge is reversed with respect to the parametrization of the underlying [ParamCurve](#).

## Parameters

|                    |                               |
|--------------------|-------------------------------|
| <i>is_reversed</i> | value of the orientation flag |
|--------------------|-------------------------------|

Implements [Go::ftEdgeBase](#).

Definition at line 64 of file ftSSfEdge.h.

**29.156.3.13** virtual `ftSSfEdge*` `Go::ftSSfEdge::split ( double t )` [virtual]

Split the edge at the given parameter t. This edge will then represent the first part of the original edge, the second part is returned to the caller.

Implements [Go::ftEdgeBase](#).

**29.156.3.14** virtual `Point` `Go::ftSSfEdge::tangent ( double t ) const` [virtual]

Evaluate tangent.

Implements [Go::ftEdgeBase](#).

**29.156.3.15** virtual `double` `Go::ftSSfEdge::tMax ( ) const` [inline],[virtual]

End parameter value of edge.

Implements [Go::ftEdgeBase](#).

Definition at line 56 of file ftSSfEdge.h.

**29.156.3.16** virtual `double` `Go::ftSSfEdge::tMin ( ) const` [inline],[virtual]

Start parameter value of edge.

Implements [Go::ftEdgeBase](#).

Definition at line 52 of file ftSSfEdge.h.

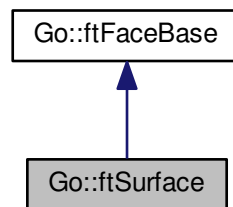
The documentation for this class was generated from the following file:

- [compositemodel/include/GoTools/compositemodel/ftSSfEdge.h](#)

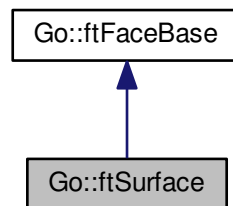
## 29.157 Go::ftSurface Class Reference

```
#include <ftSurface.h>
```

Inheritance diagram for Go::ftSurface:



Collaboration diagram for Go::ftSurface:



### Public Member Functions

- [ftSurface](#) (shared\_ptr< [ParamSurface](#) > sf, int id)
- [ftSurface](#) (shared\_ptr< [ParamSurface](#) > sf, shared\_ptr< [Loop](#) > loop, int id=-1)
- virtual [~ftSurface](#) ()  
*Empty destructor.*
- virtual [ftSurface](#) \* [asFtSurface](#) ()  
*Return as type [ftSurface](#).*
- virtual void [clearInitialEdges](#) ()  
*Reset loop information.*
- virtual std::vector< shared\_ptr< [ftEdgeBase](#) > > [createInitialEdges](#) (double degenerate\_epsilon=DEFAULT\_SPACE\_EPSILON, double kink=0.00015, bool no\_split=false)
- virtual std::vector< shared\_ptr< [ftEdgeBase](#) > > [startEdges](#) ()  
*Return pointers to first part of all bd cvs.*
- virtual [Point](#) [point](#) (double u, double v) const

- Evaluate point on face.*

  - virtual [Point](#) normal (double u, double v) const
- Evaluate surface normal.*

  - virtual [BoundingBox](#) boundingBox ()

*The bounding box corresponding to this face.*

  - virtual void setId (int id)

*Set Id of this face.*

  - virtual int getId ()

*Fetch Id of this face, may not be uniquely set.*

  - virtual shared\_ptr< [ParamSurface](#) > surface ()

*Turn orientation of the corresponding surface.*

  - virtual [ftMessage](#) createSurf (double &max\_error, double &mean\_error)

*By inheritance, not relevant for this entity.*

  - virtual void getError (double &max\_error, double &mean\_error)

*By inheritance, not relevant for this entity.*

  - virtual [ftTangPriority](#) getPrioType () const

*Priority type, not relevant for this entity.*

  - void setPrioType ([ftTangPriority](#) type)

*Priority type, not relevant for this entity.*

  - virtual void updateBoundaryLoops (shared\_ptr< [ftEdgeBase](#) > new\_edge)

*Update the information stored in the boundary loops.*

  - virtual void isolateFace ()

*Remove all adjacency information related to this face.*

  - void addBoundaryLoops (std::vector< shared\_ptr< [Loop](#) > > &bd\_loops)
  - void addOuterBoundaryLoop (shared\_ptr< [Loop](#) > outer\_loop)
  - int nmbBoundaryLoops ()
  - shared\_ptr< [Loop](#) > getBoundaryLoop (int idx)

*Get the specified boundary loops.*

  - int nmbEdges () const

*Number of edges in all loops.*

  - std::vector< shared\_ptr< [ftEdge](#) > > getAllEdges () const

*Fetch all edges in all loops.*

  - std::vector< shared\_ptr< [ftEdge](#) > > getAllEdges (int loop\_idx) const

*Fetch all edges in given loop.*

  - std::vector< [ftEdge](#) \* > getAllEdgePtrs () const

*Fetch pointers to all edges in all loops.*

  - std::vector< [ftEdge](#) \* > getAllEdgePtrs (int loop\_idx) const

*Fetch pointers to all edges in specified loop.*

  - bool onlyOuterTrim () const

*Check if this face contains any holes.*

  - int nmbOuterBdCrvs (double gap, double neighbour, double angtol) const
  - shared\_ptr< [ParamSurface](#) > getUntrimmed (double gap, double neighbour, double angtol, bool only\_↔ corner=false)
  - virtual void closestPoint (const [Point](#) &pt, double &clo\_u, double &clo\_v, [Point](#) &clo\_pt, double &clo\_dist, double epsilon) const

*Closest point between this face and a point.*

  - void closestPoint (const [Point](#) &pt, double &clo\_u, double &clo\_v, [Point](#) &clo\_pt, double &clo\_dist, double epsilon, const [RectDomain](#) \*domain\_of\_interest, double \*seed) const

*Closest point between a domain of this face and a point.*

  - [ftEdgeBase](#) \* closestBoundaryPoint (const [Point](#) &pt, const [Point](#) &in\_vec, double &clo\_u, double &clo\_v, [Point](#) &clo\_pt, double &clo\_dist, double &clo\_par) const



- Closest point between a boundary on this face and a point.*
- `ftEdgeBase * closestOuterBoundaryPoint (const Point &pt, const Point &in_vec, double &clo_u, double &clo_v, Point &clo_pt, double &clo_dist, double &clo_par) const`  
*Closest point between the outer boundary on this face and a point.*
  - `ftEdgeBase * edgeClosestToPoint (double u, double v)`  
*Return the edge on this face closest to a point.*
  - `void write (std::ostream &os)`  
*Write surface to stream.*
  - `ftMessage smoothOutFace (int edge_cont, double approx_orig_tol, double deg_tol, double &maxerr, double &meanerr, double approx_weight=0.8)`
  - `shared_ptr< SplineCurve > getBoundaryPiece (Point &pt1, Point &pt2, double eps)`
  - `bool getSurfaceKinks (double angtol, std::vector< double > &g1_disc_u, std::vector< double > &g1_disc_v)`  
*Check for constant parameter line kinks.*
  - `bool getSurfaceDisconts (double tol, std::vector< double > &disc_u, std::vector< double > &disc_v)`  
*Check for constant parameter line kinks, i.e. C1 discontinuities.*
  - `std::vector< shared_ptr< ftSurface > > splitAlongKinks (double angtol)`
  - `virtual ftMessage removeGap (ftEdgeBase *e1, ftEdgeBase *e2, ftFaceBase *other, double epsge)`  
*Close the gap between adjacent surfaces if the configuration allows it.*
  - `bool isClose (shared_ptr< Vertex > v, double tol) const`  
*Test if a Vertex is close to the surface within some tolerance.*
  - `std::vector< shared_ptr< Vertex > > vertices () const`  
*Get all vertices, duplicates removed.*
  - `std::vector< shared_ptr< Vertex > > getNonCornerVertices (double kink) const`  
*Get non corner vertices.*
  - `std::vector< shared_ptr< Vertex > > getNonCornerVertices (double kink, int loop_idx) const`  
*Get non corner vertices restricted to a particular loop.*
  - `std::vector< shared_ptr< Vertex > > getCornerVertices (double kink) const`  
*Get corner vertices.*
  - `std::vector< shared_ptr< Vertex > > getCornerVertices (double kink, int loop_idx) const`  
*Get corner vertices restricted to a particular loop.*
  - `std::vector< shared_ptr< Vertex > > getCommonVertices (ftSurface *other) const`  
*Get all vertices common to this face and another face.*
  - `std::vector< shared_ptr< ftEdge > > getCommonEdges (ftSurface *other) const`
  - `shared_ptr< Vertex > getClosestVertex (const Point &pnt) const`  
*Get the vertex closest to a given point.*
  - `void getBadDistance (std::vector< std::pair< ftSurface *, shared_ptr< Vertex > > > &badPairs, double tol)`  
*Collect all pairs of surface and vertex points where distance is greater than a tolerance.*
  - `void getBadDistance (std::vector< std::pair< ftSurface *, ftEdge * > > &badPairs, double tol) const`
  - `void getBadDistance (std::vector< std::pair< ftEdge *, shared_ptr< Vertex > > > &badPairs, double tol) const`
  - `void getPosTangentSurfaceDiscont (std::vector< ftEdge * > &badPos, std::vector< ftEdge * > &badTangent, double tol, double kink, double bend, int leastSurfIndex, shared_ptr< SurfaceModel > sm) const`
  - `bool checkLoopOrientation (std::vector< shared_ptr< Loop > > &inconsistent_loops) const`  
*Check consistency of boundary loops.*
  - `bool hasAcuteAngle (ftEdge *along_edge, double angtol) const`  
*Used in quality checing of a surface model.*
  - `void getNarrowRegion (double gap_tol, double tol, std::vector< std::pair< shared_ptr< PointOnEdge >, shared_ptr< PointOnEdge > > > &narrow_pt)`  
*Get pairs of close boundary loop points.*
  - `bool checkAndFixBoundaries ()`
  - `double area (double tol) const`  
*Compute face area.*

- void [setBoundaryConditions](#) (int bd\_type, int bd)  
*Set boundary conditions (related to isogeometric analysis)*
- [bool hasBoundaryConditions](#) () const
- void [getBoundaryConditions](#) (int &bd\_type, int &bd) const
- void [getAdjacentFaces](#) (std::vector< [ftSurface](#) \* > &neighbours) const  
*Get neighbouring faces.*
- [double getCurrEps](#) () const  
*Fetch tolerance used in check for degenerace.*
- void [setBody](#) ([Body](#) \*body)  
*Boundary model / volume model, set associated body.*
- [Body](#) \* [getBody](#) () const  
*Fetch associated body, if any.*
- [bool hasBody](#) () const  
*Whether or not this face belongs to a solid.*
- [ftSurface](#) \* [twin](#) ()  
*Non-manifold, set adjacency information between two bodies.*
- [bool hasTwin](#) ()
- std::vector< [Body](#) \* > [getAdjacentBodies](#) () const  
*Get all (up to 2) adjacent bodies.*
- void [setTwin](#) ([ftSurface](#) \*newtwin)
- [bool connectTwin](#) ([ftSurface](#) \*newtwin, double tol, bool no\_snap=true)
- void [connectTwin](#) ([ftSurface](#) \*newtwin, std::vector< [ftEdgeBase](#) \* > first1, std::vector< [ftEdgeBase](#) \* > first2, std::vector< int > forward)
- void [disconnectTwin](#) ()  
*Disconnect twin info. NB radial edge information is NOT removed.*
- [bool areNeighbours](#) ([ftSurface](#) \*other, shared\_ptr< [ftEdge](#) > &edge1, shared\_ptr< [ftEdge](#) > &edge2, int adj\_idx=0) const  
*the edge along which this acjacency occurs*
- int [nmbAdjacencies](#) ([ftSurface](#) \*other) const  
*Number of edges along which the two faces are adjacent.*
- int [nmbNextNeighbours](#) ([ftSurface](#) \*other) const  
*Number neighbouring faces common to the two given faces.*
- [bool isSpline](#) () const  
*Check if the face is represented as a spline.*
- [bool commonSplineSpace](#) ([ftSurface](#) \*other, double tol)
- void [makeCommonSplineSpace](#) ([ftSurface](#) \*other)
- [bool isCornerToCorner](#) ([ftSurface](#) \*other, double tol, int adj\_idx=0)
- void [splitAtInternalCorner](#) ([ftSurface](#) \*other, std::vector< shared\_ptr< [ftSurface](#) > > &new\_face1, std::vector< shared\_ptr< [ftSurface](#) > > &new\_face2, double tol=DEFAULT\_SPACE\_EPSILON)
- [bool getAdjacencyInfo](#) ([ftSurface](#) \*other, double tol, int &bd1, int &bd2, bool &same\_orient)
- [AdjacencyInfo](#) [getAdjacencyInfo](#) ([ftSurface](#) \*other, double tol, int adj\_idx=0, bool test\_corner=false)  
*Fetch info on adjacency between neighbouring faces.*
- [AdjacencyInfo](#) [getAdjacencyInfo](#) ([ftEdge](#) \*edge, [ftSurface](#) \*other, double tol)
- [bool checkDegAdjacency](#) ([ftSurface](#) \*other, shared\_ptr< [Vertex](#) > vx, double tol, shared\_ptr< [ParamCurve](#) > &bdcv1, shared\_ptr< [ParamCurve](#) > &bdcv2)  
*Check if two degenerate surface boundaries meet in a vertex.*
- [bool getCorrCoefEnumeration](#) ([ftSurface](#) \*other, double tol, std::vector< std::pair< int, int > > &enumeration)
- [bool getFreeBoundaryInfo](#) (double tol, std::vector< int > &free\_boundaries)
- [bool getBoundaryCoefEnumeration](#) (int bd, std::vector< int > &enumeration)
- std::vector< shared\_ptr< [EdgeVertex](#) > > [getRadialEdges](#) () const  
*Get all instances of radial edges.*
- [bool hasRadialEdges](#) () const

- [bool hasRealRadialEdges \(\) const](#)
- [bool allRadialEdges \(\) const](#)  
*All edges is connected a radial edge.*
- [std::vector< ftSurface \\* > fetchCorrespondingFaces \(\) const](#)
- [bool pointInFace \(double u, double v, double tol\)](#)  
*Point in face testing.*
- [int pointInFace2 \(double u, double v, double tol\)](#)
- [bool pointOnBd \(double u, double v, double tol\)](#)  
*Point on surface boundary.*
- [int ElementOnBoundary \(int elem\\_ix, double eps\)](#)
- [int ElementBoundaryStatus \(int elem\\_ix, double eps\)](#)
- [bool checkFaceTopology \(\)](#)  
*Debug functionality.*

### Protected Member Functions

- void [replaceSurf](#) (shared\_ptr< [ParamSurface](#) > sf)

### Additional Inherited Members

#### 29.157.1 Detailed Description

[ftSurface](#) - A topological face. The class also provides an interface for operations on the associated surface. Detailed description.

#### Author

Vibeke Skytt

#### See also

[ftFaceBase](#)

Definition at line 91 of file [ftSurface.h](#).

#### 29.157.2 Constructor & Destructor Documentation

##### 29.157.2.1 [Go::ftSurface::ftSurface \( shared\\_ptr< \[ParamSurface\]\(#\) > sf, int id \)](#)

Constructor. Typically, the [ParamSurface](#) in this constructor will be a [BoundedSurface](#). A loop will be built from this, but if we already have a loop and the edges in the loop has twins, then we must explicitly set this loop with [addBoundaryLoops\(\)](#). Note: In STEP, [ftSurface](#) corresponds to the entity 'advanced\_face'.

##### 29.157.2.2 [Go::ftSurface::ftSurface \( shared\\_ptr< \[ParamSurface\]\(#\) > sf, shared\\_ptr< \[Loop\]\(#\) > loop, int id = -1 \)](#)

Constructor. Input is the geometrical surface representing the underlying face, and the topological loop that bounds the face. Typically, the [ParamSurface](#) in this constructor will be a [SplineSurface](#) or an [ElementarySurface](#). Note: In STEP, [ftSurface](#) corresponds to the entity 'advanced\_face'.

29.157.2.3 `virtual Go::ftSurface::~~ftSurface ( ) [virtual]`

Empty destructor.

### 29.157.3 Member Function Documentation

29.157.3.1 `void Go::ftSurface::addBoundaryLoops ( std::vector< shared_ptr< Loop > > & bd_loops )`

Set information about boundary loops. Intended for use when topology information exist prior to building a [GoTools](#) face set (i.e. [SurfaceModel](#)). The function removes any existing loops prior to setting the input. Note that the function will throw if the loop information is inconsistent with the already existing information in `surf_` or the given loops are not consistent with the rules. If more than one loop is given, the first loop is the outer one. Subsequent loops must lie inside the outer loop. The loops may not intersect.

29.157.3.2 `void Go::ftSurface::addOuterBoundaryLoop ( shared_ptr< Loop > outer_loop )`

Set the outer boundary loop. Any existing loops are removed prior to setting the `outer_loop`.

29.157.3.3 `bool Go::ftSurface::allRadialEdges ( ) const`

All edges is connected a radial edge.

29.157.3.4 `double Go::ftSurface::area ( double tol ) const`

Compute face area.

29.157.3.5 `bool Go::ftSurface::areNeighbours ( ftSurface * other, shared_ptr< ftEdge > & edge1, shared_ptr< ftEdge > & edge2, int adj_idx = 0 ) const`

the edge along which this adjacency occurs

Check if two faces are adjacent, and return information about

29.157.3.6 `virtual ftSurface* Go::ftSurface::asFtSurface ( ) [virtual]`

Return as type [ftSurface](#).

Reimplemented from [Go::ftFaceBase](#).

29.157.3.7 `virtual BoundingBox Go::ftSurface::boundingBox ( ) [virtual]`

The bounding box corresponding to this face.

Implements [Go::ftFaceBase](#).

29.157.3.8 `bool Go::ftSurface::checkAndFixBoundaries ( )`

Check and fix orientation of boundary loops of trimmed surfaces

Returns

`true` if the orientation of a boundary loop was fixed, `false` otherwise

29.157.3.9 `bool Go::ftSurface::checkDegAdjacency ( ftSurface * other, shared_ptr< Vertex > vx, double tol, shared_ptr< ParamCurve > & bdcv1, shared_ptr< ParamCurve > & bdcv2 )`

Check if two degenerate surface boundaries meet in a vertex.

29.157.3.10 `bool Go::ftSurface::checkFaceTopology ( )`

Debug functionality.

29.157.3.11 `bool Go::ftSurface::checkLoopOrientation ( std::vector< shared_ptr< Loop > > & inconsistent_loops ) const`

Check consistency of boundary loops.

29.157.3.12 `virtual void Go::ftSurface::clearInitialEdges ( ) [virtual]`

Reset loop information.

Reimplemented from [Go::ftFaceBase](#).

29.157.3.13 `ftEdgeBase* Go::ftSurface::closestBoundaryPoint ( const Point & pt, const Point & in_vec, double & clo_u, double & clo_v, Point & clo_pt, double & clo_dist, double & clo_par ) const`

Closest point between a boundary on this face and a point.

29.157.3.14 `ftEdgeBase* Go::ftSurface::closestOuterBoundaryPoint ( const Point & pt, const Point & in_vec, double & clo_u, double & clo_v, Point & clo_pt, double & clo_dist, double & clo_par ) const`

Closest point between the outer boundary on this face and a point.

29.157.3.15 `virtual void Go::ftSurface::closestPoint ( const Point & pt, double & clo_u, double & clo_v, Point & clo_pt, double & clo_dist, double epsilon ) const [virtual]`

Closest point between this face and a point.

Implements [Go::ftFaceBase](#).

29.157.3.16 `void Go::ftSurface::closestPoint ( const Point & pt, double & clo_u, double & clo_v, Point & clo_pt, double & clo_dist, double epsilon, const RectDomain * domain_of_interest, double * seed ) const`

Closest point between a domain of this face and a point.

29.157.3.17 `bool Go::ftSurface::commonSplineSpace ( ftSurface * other, double tol )`

Check if two bodies are neighbours, are splines and have common a spline space at the interface

29.157.3.18 `bool Go::ftSurface::connectTwin ( ftSurface * newtwin, double tol, bool no_snap = true )`

Set adjacency between bodies Radial edge information is set

29.157.3.19 `void Go::ftSurface::connectTwin ( ftSurface * newtwin, std::vector< ftEdgeBase * > first1, std::vector< ftEdgeBase * > first2, std::vector< int > forward )`

Set adjacency between bodies Correspondence of surface loops is given as input

29.157.3.20 `virtual std::vector<shared_ptr<ftEdgeBase> > Go::ftSurface::createInitialEdges ( double degenerate_epsilon = DEFAULT_SPACE_EPSILON, double kink = 0.00015, bool no_split = false ) [virtual]`

Compute the edges associated to this face or fetch already existing edges

Implements [Go::ftFaceBase](#).

29.157.3.21 `virtual ftMessage Go::ftSurface::createSurf ( double & max_error, double & mean_error ) [virtual]`

By inheritance, not relevant for this entity.

Implements [Go::ftFaceBase](#).

29.157.3.22 `void Go::ftSurface::disconnectTwin ( )`

Disconnect twin info. NB radial edge information is NOT removed.

29.157.3.23 `ftEdgeBase* Go::ftSurface::edgeClosestToPoint ( double u, double v )`

Return the edge on this face closest to a point.

29.157.3.24 `int Go::ftSurface::ElementBoundaryStatus ( int elem_ix, double eps ) [inline]`

Check if a polynomial element (for spline surfaces) intersects the (trimming) boundaries of this [ftSurface](#), is inside or outside

## Parameters

|                |                                                                                                          |
|----------------|----------------------------------------------------------------------------------------------------------|
| <i>elem_ix</i> | Element index counted according to distinct knot values. Sequence of coordinates: x runs fastest, then y |
| <i>eps</i>     | Intersection tolerance                                                                                   |

## Returns

-1: Not a spline surface or element index out of range 0: Outside trimmed volume 1: On boundary (intersection with boundary found) 2: Internal to trimmed surfaces Note that a touch with the boundaries of the underlying surface is not considered a boundary intersection while touching a trimming curve is seen as an intersection

Definition at line 586 of file ftSurface.h.

29.157.3.25 `int Go::ftSurface::ElementOnBoundary ( int elem_ix, double eps ) [inline]`

Check if a polynomial element (for spline surfaces) intersects the (trimming) boundaries of this [ftSurface](#)

## Parameters

|                |                                                                                                          |
|----------------|----------------------------------------------------------------------------------------------------------|
| <i>elem_ix</i> | Element index counted according to distinct knot values. Sequence of coordinates: x runs fastest, then y |
| <i>eps</i>     | Intersection tolerance                                                                                   |

## Returns

-1: Not a spline surface or element index out of range 0: Not on boundary or touching a boundary curve 1: On boundary (intersection with boundary found) Note that a touch with the boundaries of the underlying surfaces is not considered a boundary intersection while touching a trimming curve is seen as an intersection

Definition at line 569 of file ftSurface.h.

29.157.3.26 `std::vector<ftSurface*> Go::ftSurface::fetchCorrespondingFaces ( ) const`

Fetch all faces that belongs to all radial edges corresponding to this face

29.157.3.27 `bool Go::ftSurface::getAdjacencyInfo ( ftSurface * other, double tol, int & bd1, int & bd2, bool & same_orient )`

DEPRECATED METHOD. USE THE OTHER [getAdjacencyInfo\(\)](#) instead Fetch info on adjacency between neighbouring faces

## Parameters

|                    |                                                                               |
|--------------------|-------------------------------------------------------------------------------|
| <i>other</i>       | The other face                                                                |
| <i>tol</i>         | Adjacency tolerance                                                           |
| <i>bd1</i>         | Boundary index of first surface 0 = umin, 1 = umax, 2 = vmin, 3 = vmax        |
| <i>bd2</i>         | Boundary index of second surface 0 = umin, 1 = umax, 2 = vmin, 3 = vmax       |
| <i>same_orient</i> | Indicates whether the surfaces are equally oriented along the common boundary |

**Returns**

*true* if adjacency is found and both faces are non-trimmed or boundary trimmed

**29.157.3.28 AdjacencyInfo** `Go::ftSurface::getAdjacencyInfo ( ftSurface * other, double tol, int adj_idx = 0, bool test_corner = false )`

Fetch info on adjacency between neighbouring faces.

**29.157.3.29 AdjacencyInfo** `Go::ftSurface::getAdjacencyInfo ( ftEdge * edge, ftSurface * other, double tol )`

Fetch info on adjacency between neighbouring faces given information about the common edge

**29.157.3.30** `std::vector<Body*> Go::ftSurface::getAdjacentBodies ( ) const`

Get all (up to 2) adjacent bodies.

**29.157.3.31** `void Go::ftSurface::getAdjacentFaces ( std::vector< ftSurface * > & neighbours ) const`

Get neighbouring faces.

**29.157.3.32** `std::vector<ftEdge*> Go::ftSurface::getAllEdgePtrs ( ) const`

Fetch pointers to all edges in all loops.

**29.157.3.33** `std::vector<ftEdge*> Go::ftSurface::getAllEdgePtrs ( int loop_idx ) const`

Fetch pointers to all edges in specified loop.

**29.157.3.34** `std::vector<shared_ptr<ftEdge> > Go::ftSurface::getAllEdges ( ) const`

Fetch all edges in all loops.

**29.157.3.35** `std::vector<shared_ptr<ftEdge> > Go::ftSurface::getAllEdges ( int loop_idx ) const`

Fetch all edges in given loop.

**29.157.3.36** `void Go::ftSurface::getBadDistance ( std::vector< std::pair< ftSurface *, shared_ptr< Vertex > > > & badPairs, double tol )`

Collect all pairs of surface and vertex points where distance is greater than a tolerance.



29.157.3.37 `void Go::ftSurface::getBadDistance ( std::vector< std::pair< ftSurface *, ftEdge * > > & badPairs, double tol ) const`

Collect all pairs of surface and edge where some part of the edge has a distance to the surface greater than a tolerance

29.157.3.38 `void Go::ftSurface::getBadDistance ( std::vector< std::pair< ftEdge *, shared_ptr< Vertex > > > & badPairs, double tol ) const`

Collect all pairs of edge and vertex where the vertex has a distance to the edge greater than a tolerance

29.157.3.39 `Body* Go::ftSurface::getBody ( ) const [inline]`

Fetch associated body, if any.

Definition at line 398 of file ftSurface.h.

29.157.3.40 `bool Go::ftSurface::getBoundaryCoefEnumeration ( int bd, std::vector< int > & enumeration )`

For a spline surface, get the local enumeration of coefficients along a specified boundary return value: true if spline, false if not

29.157.3.41 `void Go::ftSurface::getBoundaryConditions ( int & bd_type, int & bd ) const [inline]`

Definition at line 375 of file ftSurface.h.

29.157.3.42 `shared_ptr<Loop> Go::ftSurface::getBoundaryLoop ( int idx ) [inline]`

Get the specified boundary loops.

Definition at line 185 of file ftSurface.h.

29.157.3.43 `shared_ptr<SplineCurve> Go::ftSurface::getBoundaryPiece ( Point & pt1, Point & pt2, double eps )`

Return a part of the boundary of the current surface, specified by two points belonging to this boundary curve

29.157.3.44 `shared_ptr<Vertex> Go::ftSurface::getClosestVertex ( const Point & pnt ) const`

Get the vertex closest to a given point.

29.157.3.45 `std::vector<shared_ptr<ftEdge> > Go::ftSurface::getCommonEdges ( ftSurface * other ) const`

Get all edges common to this face and another face represented by the half edges in this face

29.157.3.46 `std::vector<shared_ptr<Vertex>> Go::ftSurface::getCommonVertices ( ftSurface * other ) const`

Get all vertices common to this face and another face.

29.157.3.47 `std::vector<shared_ptr<Vertex>> Go::ftSurface::getCornerVertices ( double kink ) const`

Get corner vertices.

29.157.3.48 `std::vector<shared_ptr<Vertex>> Go::ftSurface::getCornerVertices ( double kink, int loop_idx ) const`

Get corner vertices restricted to a particular loop.

29.157.3.49 `bool Go::ftSurface::getCorrCoefEnumeration ( ftSurface * other, double tol, std::vector< std::pair< int, int > > & enumeration )`

Given two adjacent spline faces represented in the same spline space, return local enumeration of the coefficients. This surface is given first and the other surface second

29.157.3.50 `double Go::ftSurface::getCurrEps ( ) const [inline]`

Fetch tolerance used in check for degenerace.

Definition at line 385 of file ftSurface.h.

29.157.3.51 `virtual void Go::ftSurface::getError ( double & max_error, double & mean_error ) [virtual]`

By inheritance, not relevant for this entity.

Implements [Go::ftFaceBase](#).

29.157.3.52 `bool Go::ftSurface::getFreeBoundaryInfo ( double tol, std::vector< int > & free_boundaries )`

Return the boundary number of outer boundary edges that follow surface boundaries return value = true: All free boundaries are associated with surface boundaries = false:Some free boundaries are not associated with surface boundaries

29.157.3.53 `virtual int Go::ftSurface::getId ( ) [virtual]`

Fetch Id of this face, may not be uniquely set.

Reimplemented from [Go::ftFaceBase](#).

29.157.3.54 `void Go::ftSurface::getNarrowRegion ( double gap_tol, double tol, std::vector< std::pair< shared_ptr< PointOnEdge >, shared_ptr< PointOnEdge > > > & narrow_pt )`

Get pairs of close boundary loop points.

29.157.3.55 `std::vector<shared_ptr<Vertex> > Go::ftSurface::getNonCornerVertices ( double kink ) const`

Get non corner vertices.

29.157.3.56 `std::vector<shared_ptr<Vertex> > Go::ftSurface::getNonCornerVertices ( double kink, int loop_idx ) const`

Get non corner vertices restricted to a particular loop.

29.157.3.57 `void Go::ftSurface::getPosTangentSurfaceDiscont ( std::vector< ftEdge * > & badPos, std::vector< ftEdge * > & badTangent, double tol, double kink, double bend, int leastSurfIndex, shared_ptr< SurfaceModel > sm ) const`

Information about discontinuities, used in quality checking of a surface model

29.157.3.58 `virtual ftTangPriority Go::ftSurface::getPrioType ( ) const [virtual]`

Priority type, not relevant for this entity.

Implements [Go::ftFaceBase](#).

29.157.3.59 `std::vector<shared_ptr<EdgeVertex> > Go::ftSurface::getRadialEdges ( ) const`

Get all instances of radial edges.

29.157.3.60 `bool Go::ftSurface::getSurfaceDisconts ( double tol, std::vector< double > & disc_u, std::vector< double > & disc_v )`

Check for constant parameter line kinks, i.e. C1 discontinuities.

29.157.3.61 `bool Go::ftSurface::getSurfaceKinks ( double angtol, std::vector< double > & g1_disc_u, std::vector< double > & g1_disc_v )`

Check for constant parameter line kinks.

29.157.3.62 `shared_ptr<ParamSurface> Go::ftSurface::getUntrimmed ( double gap, double neighbour, double angtol, bool only_corner = false )`

Approximate a regular face with a non-trimmed spline surface If the initial surface is not regular, no output is created

29.157.3.63 **bool** Go::ftSurface::hasAcuteAngle ( **ftEdge** \* *along\_edge*, **double** *angtol* ) **const**

Used in quality checing of a surface model.

29.157.3.64 **bool** Go::ftSurface::hasBody ( ) **const** `[inline]`

Whether or not this face belongs to a solid.

Definition at line 404 of file ftSurface.h.

29.157.3.65 **bool** Go::ftSurface::hasBoundaryConditions ( ) **const** `[inline]`

Definition at line 370 of file ftSurface.h.

29.157.3.66 **bool** Go::ftSurface::hasRadialEdges ( ) **const**

Check for radial edges Existence

29.157.3.67 **bool** Go::ftSurface::hasRealRadialEdges ( ) **const**

29.157.3.68 **bool** Go::ftSurface::hasTwin ( ) `[inline]`

Definition at line 415 of file ftSurface.h.

29.157.3.69 **bool** Go::ftSurface::isClose ( **shared\_ptr**< **Vertex** > *v*, **double** *tol* ) **const**

Test if a [Vertex](#) is close to the surface within some tolerance.

29.157.3.70 **bool** Go::ftSurface::isCornerToCorner ( **ftSurface** \* *other*, **double** *tol*, **int** *adj\_idx* = 0 )

Check if two volumes are splines and meet in a corner-to-corner configuration

29.157.3.71 **virtual void** Go::ftSurface::isolateFace ( ) `[virtual]`

Remove all adjacency information related to this face.

Reimplemented from [Go::ftFaceBase](#).

29.157.3.72 **bool** Go::ftSurface::isSpline ( ) **const**

Check if the face is represented as a spline.

29.157.3.73 `void Go::ftSurface::makeCommonSplineSpace ( ftSurface * other )`

Ensure that two neighbouring spline surfaces have a common spline space at the interface Averaging of coefficients at the common boundary is also performed

29.157.3.74 `int Go::ftSurface::nmbAdjacencies ( ftSurface * other ) const`

Number of edges along which the two faces are adjacent.

29.157.3.75 `int Go::ftSurface::nmbBoundaryLoops ( ) [inline]`

Number of loops, the first is the outer boundary loop, further loops represents holes

Definition at line 179 of file ftSurface.h.

29.157.3.76 `int Go::ftSurface::nmbEdges ( ) const`

Number of edges in all loops.

29.157.3.77 `int Go::ftSurface::nmbNextNeighbours ( ftSurface * other ) const`

Number neighbouring faces common to the two given faces.

29.157.3.78 `int Go::ftSurface::nmbOuterBdCrvs ( double gap, double neighbour, double angtol ) const`

Count the number of curves in the outer boundary loop with regard to how many corners there is in this loop, but without regard to how the loop is actually divided into curves

29.157.3.79 `virtual Point Go::ftSurface::normal ( double u, double v ) const [virtual]`

Evaluate surface normal.

Implements [Go::ftFaceBase](#).

29.157.3.80 `bool Go::ftSurface::onlyOuterTrim ( ) const [inline]`

Check if this face contains any holes.

Definition at line 208 of file ftSurface.h.

29.157.3.81 `virtual Point Go::ftSurface::point ( double u, double v ) const [virtual]`

Evaluate point on face.

Implements [Go::ftFaceBase](#).

29.157.3.82 `bool Go::ftSurface::pointInFace ( double u, double v, double tol )` [inline]

[Point](#) in face testing.

Definition at line 538 of file `ftSurface.h`.

29.157.3.83 `int Go::ftSurface::pointInFace2 ( double u, double v, double tol )` [inline]

[Point](#) in face testing return value = 0 : Not in face = 1 : Internal in face = 2 : On boundary

Definition at line 547 of file `ftSurface.h`.

29.157.3.84 `bool Go::ftSurface::pointOnBd ( double u, double v, double tol )` [inline]

[Point](#) on surface boundary.

Definition at line 553 of file `ftSurface.h`.

29.157.3.85 `virtual ftMessage Go::ftSurface::removeGap ( ftEdgeBase * e1, ftEdgeBase * e2, ftFaceBase * other, double epsge )` [virtual]

Close the gap between adjacent surfaces if the configuration allows it.

29.157.3.86 `void Go::ftSurface::replaceSurf ( shared_ptr< ParamSurface > sf )` [inline],[protected]

Definition at line 595 of file `ftSurface.h`.

29.157.3.87 `void Go::ftSurface::setBody ( Body * body )` [inline]

Boundary model / volume model, set associated body.

Definition at line 392 of file `ftSurface.h`.

29.157.3.88 `void Go::ftSurface::setBoundaryConditions ( int bd_type, int bd )` [inline]

Set boundary conditions (related to isogeometric analysis)

Definition at line 364 of file `ftSurface.h`.

29.157.3.89 `virtual void Go::ftSurface::setId ( int id )` [virtual]

Set Id of this face.

Reimplemented from [Go::ftFaceBase](#).

29.157.3.90 `void Go::ftSurface::setPrioType ( ftTangPriority type ) [inline]`

Priority type, not relevant for this entity.

Definition at line 153 of file ftSurface.h.

29.157.3.91 `void Go::ftSurface::setTwin ( ftSurface * newtwin )`

Set adjacency between bodies Radial edge information is existed to exist

29.157.3.92 `ftMessage Go::ftSurface::smoothOutFace ( int edge_cont, double approx_orig_tol, double deg_tol, double & maxerr, double & meanerr, double approx_weight = 0.8 )`

Smooth the face, the associated surface is required to be of type [SplineSurface](#).

29.157.3.93 `std::vector<shared_ptr<ftSurface>> Go::ftSurface::splitAlongKinks ( double angtol )`

Split a surface along constant parameter line kinks. Currently only spline surface is implemented.

29.157.3.94 `void Go::ftSurface::splitAtInternalCorner ( ftSurface * other, std::vector< shared_ptr< ftSurface >> & new_face1, std::vector< shared_ptr< ftSurface >> & new_face2, double tol = DEFAULT_SPACE_EPSILON )`

Ensure that two spline volumes meet in a corner to corner configuration

29.157.3.95 `virtual std::vector<shared_ptr<ftEdgeBase>> Go::ftSurface::startEdges ( ) [virtual]`

Return pointers to first part of all bd cvs.

Implements [Go::ftFaceBase](#).

29.157.3.96 `virtual shared_ptr<ParamSurface> Go::ftSurface::surface ( ) [inline],[virtual]`

Turn orientation of the corresponding surface.

Get underlying surface

Implements [Go::ftFaceBase](#).

Definition at line 142 of file ftSurface.h.

29.157.3.97 `ftSurface* Go::ftSurface::twin ( ) [inline]`

Non-manifold, set adjacency information between two bodies.

Definition at line 410 of file ftSurface.h.

29.157.3.98 `virtual void Go::ftSurface::updateBoundaryLoops ( shared_ptr< ftEdgeBase > new_edge )` [virtual]

Update the information stored in the boundary loops.

Reimplemented from [Go::ftFaceBase](#).

29.157.3.99 `std::vector<shared_ptr<Vertex>> Go::ftSurface::vertices ( ) const`

Get all vertices, duplicates removed.

29.157.3.100 `void Go::ftSurface::write ( std::ostream & os )`

Write surface to stream.

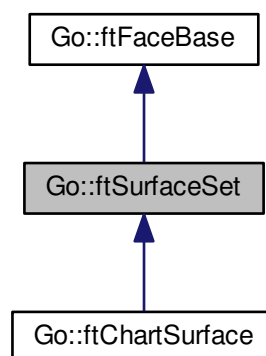
The documentation for this class was generated from the following file:

- [compositemodel/include/GoTools/compositemodel/ftSurface.h](#)

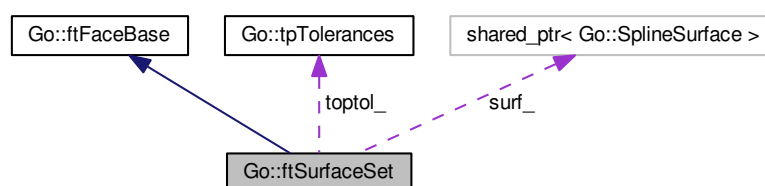
## 29.158 Go::ftSurfaceSet Class Reference

```
#include <ftSurfaceSet.h>
```

Inheritance diagram for Go::ftSurfaceSet:



Collaboration diagram for Go::ftSurfaceSet:





## Public Member Functions

- [ftSurfaceSet](#) ([const](#) [std::vector](#)< [shared\\_ptr](#)< [ParamSurface](#) > > &[surfaces](#), [tpTolerances](#) &[topeps](#), [double](#) [approxeps](#))  
*Constructor.*
- [~ftSurfaceSet](#) ()  
*Destructor.*
- virtual [std::vector](#)< [shared\\_ptr](#)< [ftEdgeBase](#) > > [createInitialEdges](#) ([double](#) [degenerate\\_epsilon](#)=[DEFAULT\\_SPACE\\_EPSILON](#), [double](#) [kink](#)=0.00015, [bool](#) [no\\_split](#)=[false](#))=0
- virtual [std::vector](#)< [shared\\_ptr](#)< [ftEdgeBase](#) > > [startEdges](#) ()  
*Return pointers to first part of all bd cvs.*
- virtual [Point](#) [point](#) ([double](#) [u](#), [double](#) [v](#)) [const](#) =0  
*Evaluate point on face.*
- virtual [Point](#) [normal](#) ([double](#) [u](#), [double](#) [v](#)) [const](#) =0  
*Evaluate surface normal.*
- virtual [BoundingBox](#) [boundingBox](#) ()  
*The bounding box corresponding to this face.*
- virtual void [setId](#) ([int](#) [id](#))  
*Set id for this face.*
- virtual [int](#) [getId](#) ()  
*Return id, default id is -1.*
- virtual [shared\\_ptr](#)< [ParamSurface](#) > [surface](#) ()  
*Fetch geometric surface.*
- virtual [ftMessage](#) [createSurf](#) ([double](#) &[max\\_error](#), [double](#) &[mean\\_error](#))=0
- virtual void [getError](#) ([double](#) &[max\\_error](#), [double](#) &[mean\\_error](#))=0
- virtual [ftTangPriority](#) [getPrioType](#) () [const](#) =0  
*Priority type.*
- [ftMessage](#) [trimWithPlane](#) ([const](#) [ftPlane](#) &[plane](#))
- void [getSurfaces](#) ([vector](#)< [shared\\_ptr](#)< [ParamSurface](#) > > &[surfaces](#))  
*Return the surfaces belonging to the surface set.*
- [ftMessage](#) [modifySurface](#) ([int](#) [maxiter](#), [int](#) [max\\_update\\_iter](#), [std::vector](#)< [int](#) > &[edge\\_derivs](#), [ftPointSet](#) &[points](#), [double](#) &[max\\_error](#), [double](#) &[mean\\_error](#))
- [double](#) [getApproxweight](#) ()
- void [setApproxweight](#) ([double](#) [approxweight](#))
- void [setAdditionalCornerPts](#) ([std::vector](#)< [Point](#) > &[add\\_corner\\_pts](#))
- virtual void [closestPoint](#) ([const](#) [Point](#) &[pt](#), [double](#) &[clo\\_u](#), [double](#) &[clo\\_v](#), [Point](#) &[clo\\_pt](#), [double](#) &[clo\\_dist](#), [double](#) [epsilon](#)) [const](#)  
*Closest point between this face and a point.*

## Protected Member Functions

- virtual [bool](#) [sampleOnlyEdges](#) () [const](#) =0
- [ftMessage](#) [getOuterLoop](#) ([std::vector](#)< [ftEdgeBase](#) \* > &[outer\\_loop](#), [std::vector](#)< [int](#) > &[corners](#), [double](#) [bend\\_tol](#), [double](#) [kink\\_tol](#))
- [ftMessage](#) [fetchSamplePoints](#) ([const](#) [std::vector](#)< [ftEdgeBase](#) \* > &[edgeloop](#), [std::vector](#)< [int](#) > &[corner](#), [std::vector](#)< [int](#) > &[cn](#), [ftPointSet](#) &[points](#))
- void [getInitBndData](#) ([std::vector](#)< [ftEdgeBase](#) \* > &[edgc](#), [ftPointSet](#) &[points](#), [int](#) [cn\[ \]](#))
- [ftMessage](#) [getInitInnerData2](#) ([ftPointSet](#) &[points](#))
- [ftMessage](#) [getInitInnerData](#) ([ftPointSet](#) &[points](#), [int](#) [max\\_sample](#))
- [ftMessage](#) [updatePointTopology](#) ([ftPointSet](#) &[points](#))
- [ftMessage](#) [merge](#) ([double](#) &[max\\_error](#), [double](#) &[mean\\_error](#), [double](#) [bend\\_tol](#), [double](#) [kink\\_tol](#))
- [std::vector](#)< [double](#) > [estimateSurfSides](#) ([SplineSurface](#) \*[surf](#))

- [ftMessage reparametrizeSurf](#) (`const std::vector< ftEdgeBase * > &first_edges`, `const std::vector< int > &corners`, `ftPointSet &points`, `double umin=0`, `double umax=0`, `double vmin=0`, `double vmax=0`)
- [ftMessage getBoundaryConditions](#) (`const std::vector< ftEdgeBase * > &edgeloop`, `std::vector< int > &corner`, `std::vector< shared_ptr< SplineCurve > > &bd_curves`, `std::vector< shared_ptr< SplineCurve > > &cross_curves`, `bool compute_cross_curves`, `std::vector< BoundaryPiece > &bdpiece`, `bool require_cord←_length_param=false`)
- [ftMessage getNextEdgePieceInfo](#) (`std::vector< ftEdgeBase * >::iterator &first_edge`, `std::vector< ftEdge←Base * >::iterator &last_edge`, `ftFaceBase * &adjsurf`, `SplineCurve * &cv`, `Point &ptmin`, `Point &ptmax`, `double &length`)
- `shared_ptr< SplineCurve > joinCrossCurves` (`std::vector< shared_ptr< SplineCurve > > &cross_curves`, `double startpar`, `double endpar`)
- `shared_ptr< SplineCurve > getBoundaryPiece` (`Point &ptmin`, `Point &ptmax`, `double eps`)

## Protected Attributes

- `tpTolerances toptol_`
- `std::vector< shared_ptr< ftFaceBase > > faces_`
- `shared_ptr< SplineSurface > surf_`
- `std::vector< shared_ptr< Loop > > boundary_loops_`
- `double approxtol_`
- `double approxweight_`
- `std::vector< shared_ptr< SplineCurve > > approx_bd_curves_`
- `std::vector< Point > additional_corner_pts_`

### 29.158.1 Detailed Description

A set of surfaces is viewed as one surface. Member functions assume object is handled by a [tpTopologyTable](#).

Definition at line 53 of file `ftSurfaceSet.h`.

### 29.158.2 Constructor & Destructor Documentation

29.158.2.1 `Go::ftSurfaceSet::ftSurfaceSet ( const std::vector< shared_ptr< ParamSurface > > & surfaces, tpTolerances & topeps, double approxeps )`

Constructor.

29.158.2.2 `Go::ftSurfaceSet::~~ftSurfaceSet ( )`

Destructor.

### 29.158.3 Member Function Documentation

29.158.3.1 `virtual BoundingBox Go::ftSurfaceSet::boundingBox ( ) [virtual]`

The bounding box corresponding to this face.

Implements [Go::ftFaceBase](#).

29.158.3.2 `virtual void Go::ftSurfaceSet::closestPoint ( const Point & pt, double & clo_u, double & clo_v, Point & clo_pt, double & clo_dist, double epsilon ) const [inline], [virtual]`

Closest point between this face and a point.

Implements [Go::ftFaceBase](#).

Definition at line 122 of file ftSurfaceSet.h.

29.158.3.3 `virtual std::vector<shared_ptr<ftEdgeBase> > Go::ftSurfaceSet::createInitialEdges ( double degenerate_epsilon = DEFAULT_SPACE_EPSILON, double kink = 0.00015, bool no_split = false ) [pure virtual]`

Compute the edges associated to this face or fetch already existing edges

Implements [Go::ftFaceBase](#).

Implemented in [Go::ftChartSurface](#).

29.158.3.4 `virtual ftMessage Go::ftSurfaceSet::createSurf ( double & max_error, double & mean_error ) [pure virtual]`

Make sure that a geometric surface exists. Not applicable in normal configurations

Implements [Go::ftFaceBase](#).

Implemented in [Go::ftChartSurface](#).

29.158.3.5 `std::vector<double> Go::ftSurfaceSet::estimateSurfSides ( SplineSurface * surf ) [protected]`

29.158.3.6 `ftMessage Go::ftSurfaceSet::fetchSamplePoints ( const std::vector< ftEdgeBase * > & edgeloop, std::vector< int > & corner, std::vector< int > & cn, ftPointSet & points ) [protected]`

29.158.3.7 `double Go::ftSurfaceSet::getApproxweight ( ) [inline]`

Definition at line 112 of file ftSurfaceSet.h.

29.158.3.8 `ftMessage Go::ftSurfaceSet::getBoundaryConditions ( const std::vector< ftEdgeBase * > & edgeloop, std::vector< int > & corner, std::vector< shared_ptr< SplineCurve > > & bd_curves, std::vector< shared_ptr< SplineCurve > > & cross_curves, bool compute_cross_curves, std::vector< BoundaryPiece > & bdpiece, bool require_cord_length_param = false ) [protected]`

29.158.3.9 `shared_ptr<SplineCurve> Go::ftSurfaceSet::getBoundaryPiece ( Point & ptmin, Point & ptmax, double eps ) [protected]`

29.158.3.10 `virtual void Go::ftSurfaceSet::getError ( double & max_error, double & mean_error ) [pure virtual]`

Fetch the error due to approximations in createSurf. Not applicable in normal configurations

Implements [Go::ftFaceBase](#).

Implemented in [Go::ftChartSurface](#).

29.158.3.11 `virtual int Go::ftSurfaceSet::getId ( ) [virtual]`

Return id, default id is -1.

Reimplemented from [Go::ftFaceBase](#).

29.158.3.12 `void Go::ftSurfaceSet::getInitBndData ( std::vector< ftEdgeBase * > & edgc, ftPointSet & points, int cn[ ] ) [protected]`

29.158.3.13 `ftMessage Go::ftSurfaceSet::getInitInnerData ( ftPointSet & points, int max_sample ) [protected]`

29.158.3.14 `ftMessage Go::ftSurfaceSet::getInitInnerData2 ( ftPointSet & points ) [protected]`

29.158.3.15 `ftMessage Go::ftSurfaceSet::getNextEdgePieceInfo ( std::vector< ftEdgeBase * >::iterator & first_edge, std::vector< ftEdgeBase * >::iterator & last_edge, ftFaceBase * & adjsurf, SplineCurve * & cv, Point & ptmin, Point & ptmax, double & length ) [protected]`

29.158.3.16 `ftMessage Go::ftSurfaceSet::getOuterLoop ( std::vector< ftEdgeBase * > & outer_loop, std::vector< int > & corners, double bend_tol, double kink_tol ) [protected]`

29.158.3.17 `virtual ftTangPriority Go::ftSurfaceSet::getPrioType ( ) const [pure virtual]`

Priority type.

Implements [Go::ftFaceBase](#).

Implemented in [Go::ftChartSurface](#).

29.158.3.18 `void Go::ftSurfaceSet::getSurfaces ( vector< shared_ptr< ParamSurface > > & surfaces )`

Return the surfaces belonging to the surface set.

29.158.3.19 `shared_ptr< SplineCurve > Go::ftSurfaceSet::joinCrossCurves ( std::vector< shared_ptr< SplineCurve > > & cross_curves, double startpar, double endpar ) [protected]`

29.158.3.20 `ftMessage Go::ftSurfaceSet::merge ( double & max_error, double & mean_error, double bend_tol, double kink_tol ) [protected]`

29.158.3.21 `ftMessage Go::ftSurfaceSet::modifySurface ( int maxiter, int max_update_iter, std::vector< int > & edge_derivs, ftPointSet & points, double & max_error, double & mean_error )`

29.158.3.22 `virtual Point Go::ftSurfaceSet::normal ( double u, double v ) const [pure virtual]`

Evaluate surface normal.

Implements [Go::ftFaceBase](#).

Implemented in [Go::ftChartSurface](#).

29.158.3.23 `virtual Point Go::ftSurfaceSet::point ( double u, double v ) const` [pure virtual]

Evaluate point on face.

Implements [Go::ftFaceBase](#).

Implemented in [Go::ftChartSurface](#).

29.158.3.24 `ftMessage Go::ftSurfaceSet::reparametrizeSurf ( const std::vector< ftEdgeBase * > & first_edges, const std::vector< int > & corners, ftPointSet & points, double umin = 0, double umax = 0, double vmin = 0, double vmax = 0 )` [protected]

29.158.3.25 `virtual bool Go::ftSurfaceSet::sampleOnlyEdges ( ) const` [protected],[pure virtual]

Implemented in [Go::ftChartSurface](#).

29.158.3.26 `void Go::ftSurfaceSet::setAdditionalCornerPts ( std::vector< Point > & add_corner_pts )` [inline]

Definition at line 118 of file `ftSurfaceSet.h`.

29.158.3.27 `void Go::ftSurfaceSet::setApproxweight ( double approxweight )` [inline]

Definition at line 115 of file `ftSurfaceSet.h`.

29.158.3.28 `virtual void Go::ftSurfaceSet::setid ( int id )` [virtual]

Set id for this face.

Reimplemented from [Go::ftFaceBase](#).

29.158.3.29 `virtual std::vector<shared_ptr<ftEdgeBase>> Go::ftSurfaceSet::startEdges ( )` [virtual]

Return pointers to first part of all bd cvs.

Implements [Go::ftFaceBase](#).

29.158.3.30 `virtual shared_ptr<ParamSurface> Go::ftSurfaceSet::surface ( )` [inline],[virtual]

Fetch geometric surface.

Implements [Go::ftFaceBase](#).

Definition at line 87 of file `ftSurfaceSet.h`.

29.158.3.31 **ftMessage** Go::ftSurfaceSet::trimWithPlane ( const ftPlane & plane )

Faces or parts of faces in faces\_ are trimmed if they lie above plane. Remember to update top\_table\_ and first\_↔ edges\_!

29.158.3.32 **ftMessage** Go::ftSurfaceSet::updatePointTopology ( ftPointSet & points ) [protected]

## 29.158.4 Member Data Documentation

29.158.4.1 **std::vector<Point>** Go::ftSurfaceSet::additional\_corner\_pts\_ [protected]

Definition at line 150 of file ftSurfaceSet.h.

29.158.4.2 **std::vector<shared\_ptr<SplineCurve>>** Go::ftSurfaceSet::approx\_bd\_curves\_ [protected]

Definition at line 147 of file ftSurfaceSet.h.

29.158.4.3 **double** Go::ftSurfaceSet::aproxtol\_ [protected]

Definition at line 144 of file ftSurfaceSet.h.

29.158.4.4 **double** Go::ftSurfaceSet::approxweight\_ [protected]

Definition at line 145 of file ftSurfaceSet.h.

29.158.4.5 **std::vector<shared\_ptr<Loop>>** Go::ftSurfaceSet::boundary\_loops\_ [protected]

The boundary loops limiting this face, the first loop represents the initial boundary

Definition at line 142 of file ftSurfaceSet.h.

29.158.4.6 **std::vector<shared\_ptr<ftFaceBase>>** Go::ftSurfaceSet::faces\_ [protected]

Definition at line 137 of file ftSurfaceSet.h.

29.158.4.7 **shared\_ptr<SplineSurface>** Go::ftSurfaceSet::surf\_ [protected]

Definition at line 138 of file ftSurfaceSet.h.

29.158.4.8 `tpTolerances` `Go::ftSurfaceSet::toptol_` [protected]

Definition at line 135 of file `ftSurfaceSet.h`.

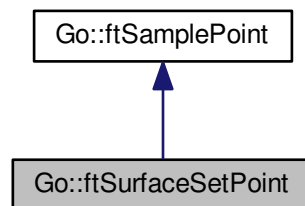
The documentation for this class was generated from the following file:

- `compositemodel/include/GoTools/compositemodel/ftSurfaceSet.h`

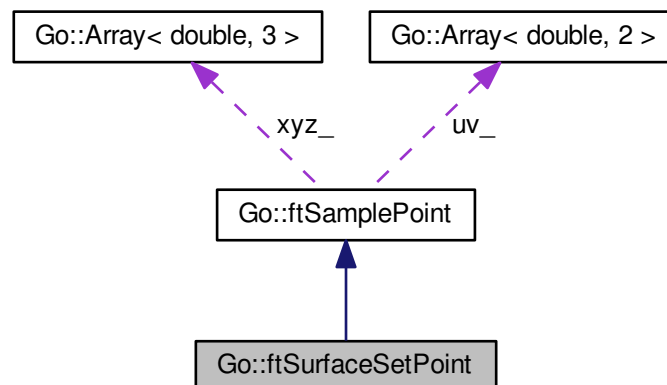
## 29.159 Go::ftSurfaceSetPoint Class Reference

```
#include <ftSurfaceSetPoint.h>
```

Inheritance diagram for `Go::ftSurfaceSetPoint`:



Collaboration diagram for `Go::ftSurfaceSetPoint`:



## Public Member Functions

- [ftSurfaceSetPoint](#) ([Vector3D](#) xyz, int bnd)  
*Constructor.*
- [ftSurfaceSetPoint](#) ([Vector3D](#) xyz, int bnd, [shared\\_ptr](#)< [ftFaceBase](#) > &face, [Vector2D](#) par\_pt)  
*Constructor.*
- virtual [~ftSurfaceSetPoint](#) ()  
*Destructor.*
- virtual [ftSurfaceSetPoint](#) \* [asSurfaceSetPoint](#) ()  
*Return pointer to this point as a surface set point.*
- virtual [bool](#) [containsFace](#) ([ftFaceBase](#) \*face) [const](#)  
*Check if a face is associated this point.*
- void [addPair](#) ([shared\\_ptr](#)< [ftFaceBase](#) > &face, [const](#) [Vector2D](#) &par\_pt)  
*For a given point, add information about param values on another ftFaceBase.*
- void [addFace](#) ([shared\\_ptr](#)< [ftFaceBase](#) > &face)  
*For a given point, add another face. Parameter values are computed.*
- int [nmbFaces](#) ()  
*Return the number of faces in which point is member of.*
- [shared\\_ptr](#)< [ftFaceBase](#) > [face](#) (int i)  
*Return pointer to face number i (error if not enough faces).*
- [Vector2D](#) [parValue](#) (int i)  
*Fetch the parameter vale in face number i of this point.*
- [Vector2D](#) [getPar](#) ([ftFaceBase](#) \*face)  
*Fetch the parameter value in face face of this point.*
- void [addInfo](#) ([ftSurfaceSetPoint](#) \*other)
- void [resetPosition](#) ([Vector3D](#) pos, int bnd)  
*Change position of point and update parameter values accordingly.*
- void [resetPosition](#) ([Vector3D](#) pos)
- virtual void [write2Dval](#) ([std::ostream](#) &os) [const](#)  
*Write parameter information to stream.*

## Additional Inherited Members

### 29.159.1 Detailed Description

Subclass of [ftSamplePoint](#); intended for use when a point is member of multiple [ftFaceBase](#)'s.

Definition at line 54 of file [ftSurfaceSetPoint.h](#).

### 29.159.2 Constructor & Destructor Documentation

#### 29.159.2.1 [Go::ftSurfaceSetPoint::ftSurfaceSetPoint](#) ( [Vector3D](#) xyz, int bnd )

Constructor.

#### 29.159.2.2 [Go::ftSurfaceSetPoint::ftSurfaceSetPoint](#) ( [Vector3D](#) xyz, int bnd, [shared\\_ptr](#)< [ftFaceBase](#) > & face, [Vector2D](#) par\_pt )

Constructor.



29.159.2.3 `virtual Go::ftSurfaceSetPoint::~~ftSurfaceSetPoint ( ) [virtual]`

Destructor.

### 29.159.3 Member Function Documentation

29.159.3.1 `void Go::ftSurfaceSetPoint::addFace ( shared_ptr< ftFaceBase > & face )`

For a given point, add another face. Parameter values are computed.

29.159.3.2 `void Go::ftSurfaceSetPoint::addInfo ( ftSurfaceSetPoint * other )`

Add face, parameter and connectivity info from the point other to this point NB! It is assumed that the position information is consistent

29.159.3.3 `void Go::ftSurfaceSetPoint::addPair ( shared_ptr< ftFaceBase > & face, const Vector2D & par_pt )`

For a given point, add information about param values on another [ftFaceBase](#).

29.159.3.4 `virtual ftSurfaceSetPoint* Go::ftSurfaceSetPoint::asSurfaceSetPoint ( ) [inline],[virtual]`

Return pointer to this point as a surface set point.

Reimplemented from [Go::ftSamplePoint](#).

Definition at line 67 of file `ftSurfaceSetPoint.h`.

29.159.3.5 `virtual bool Go::ftSurfaceSetPoint::containsFace ( ftFaceBase * face ) const [virtual]`

Check if a face is associated this point.

Reimplemented from [Go::ftSamplePoint](#).

29.159.3.6 `shared_ptr<ftFaceBase> Go::ftSurfaceSetPoint::face ( int i )`

Return pointer to face number i (error if not enough faces).

29.159.3.7 `Vector2D Go::ftSurfaceSetPoint::getPar ( ftFaceBase * face )`

Fetch the parameter value in face face of this point.

29.159.3.8 `int Go::ftSurfaceSetPoint::nmbFaces ( )`

Return the number of faces in which point is member of.

29.159.3.9 `Vector2D Go::ftSurfaceSetPoint::parValue ( int i )`

Fetch the parameter vale in face number i of this point.

29.159.3.10 `void Go::ftSurfaceSetPoint::resetPosition ( Vector3D pos, int bnd )`

Change position of point and update parameter values accordingly.

29.159.3.11 `void Go::ftSurfaceSetPoint::resetPosition ( Vector3D pos )`

29.159.3.12 `virtual void Go::ftSurfaceSetPoint::write2Dval ( std::ostream & os ) const` [virtual]

Write parameter information to stream.

Reimplemented from [Go::ftSamplePoint](#).

The documentation for this class was generated from the following file:

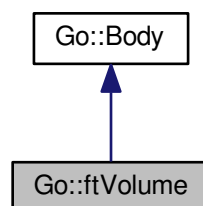
- [compositemodel/include/GoTools/compositemodel/ftSurfaceSetPoint.h](#)

## 29.160 Go::ftVolume Class Reference

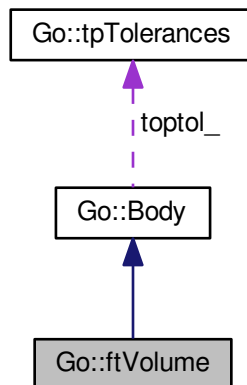
A topological solid with a trivariate geometry description.

```
#include <ftVolume.h>
```

Inheritance diagram for Go::ftVolume:



Collaboration diagram for Go::ftVolume:



## Public Member Functions

- [ftVolume](#) (shared\_ptr< [ParamVolume](#) > vol, int id=-1)  
*Constructor given a trivariate volume description.*
- [ftVolume](#) (shared\_ptr< [ParamVolume](#) > vol, double gap\_eps, double kink\_eps, int id=-1)
- [ftVolume](#) (shared\_ptr< [ParamVolume](#) > vol, double gap\_eps, double neighbour, double kink\_eps, double bend, int id=-1)
- [ftVolume](#) (shared\_ptr< [ParamVolume](#) > vol, shared\_ptr< [SurfaceModel](#) > shell, int id=-1)
- [ftVolume](#) (shared\_ptr< [ParamVolume](#) > vol, std::vector< shared\_ptr< [SurfaceModel](#) > > shells, int id=-1)
- [ftVolume](#) (shared\_ptr< [SurfaceModel](#) > shell, int id=-1)  
*Create a [ftVolume](#) when no geometry description is known yet.*
- [~ftVolume](#) ()  
*Destructor.*
- shared\_ptr< [ParamVolume](#) > [getVolume](#) ()  
*Fetch geometric description.*
- int [getId](#) ()  
*Fetch Id, not necessarily uniquely set.*
- void [closestPoint](#) (Point &pt, double &clo\_u, double &clo\_v, double &clo\_w, Point &clo\_pt, double &clo\_dist, double epsilon, double \*seed=NULL) const  
*Closest point between this possibly trimmed volume and a point.*
- [ftFaceBase](#) \* [closestBoundaryPoint](#) (Point &pt, double &clo\_u, double &clo\_v, double &clo\_w, Point &clo\_pt, double &clo\_dist, double &clo\_par\_u, double &clo\_par\_v, double epsilon) const
- virtual [BoundingBox](#) [boundingBox](#) () const  
*The bounding box corresponding to this solid.*
- bool [isSpline](#) () const  
*Check if the volume is represented as a spline.*
- std::vector< shared\_ptr< [EdgeVertex](#) > > [radialEdges](#) () const  
*Fetch all radial edges belonging to this model.*
- std::vector< shared\_ptr< [EdgeVertex](#) > > [getCommonEdges](#) ([ftVolume](#) \*other) const  
*Fetch all radial edges common to this model and another one.*
- std::vector< shared\_ptr< [ftEdge](#) > > [uniqueNonRadialEdges](#) () const

- void [getAdjacentBodies](#) (std::vector< [ftVolume](#) \* > &neighbours)
  - Get neighbouring bodies.*
- [bool commonSplineSpace](#) ([ftVolume](#) \*other, [double](#) tol)
- [bool makeCommonSplineSpace](#) ([ftVolume](#) \*other)
- [bool isCornerToCorner](#) (shared\_ptr< [ftVolume](#) > other, [double](#) tol)
- void [splitAtInternalCorner](#) ([ftVolume](#) \*other, std::vector< shared\_ptr< [ftVolume](#) > > &new\_vol1, std::vector< shared\_ptr< [ftVolume](#) > > &new\_vol2, [double](#) tol=DEFAULT\_SPACE\_EPSILON)
- [bool getAdjacencyInfo](#) ([ftVolume](#) \*other, [double](#) tol, int &bd1, int &bd2, int &orientation, [bool](#) &same\_seq)
- [VolumeAdjacencyInfo](#) [getAdjacencyInfo](#) ([ftVolume](#) \*other, [double](#) tol, int adj\_idx=0, [bool](#) test\_corner=false)
  - Fetch info on adjacency between neighbouring bodies.*
- [VolumeAdjacencyInfo](#) [getCornerAdjacencyInfo](#) ([ftVolume](#) \*other, [EdgeVertex](#) \*evx, [double](#) tol, int adj\_idx=0)
- [VolumeAdjacencyInfo](#) [getCornerAdjacencyInfo](#) ([ftVolume](#) \*other, [double](#) tol, int adj\_idx=0)
- [bool checkDegAdjacency](#) ([ftVolume](#) \*other, shared\_ptr< [EdgeVertex](#) > evx, [double](#) tol, shared\_ptr< [ParamSurface](#) > &bdsf1, shared\_ptr< [ParamSurface](#) > &bdsf2)
- [bool getCorrCoefEnumeration](#) ([ftVolume](#) \*other, [double](#) tol, std::vector< std::pair< int, int > > &enumeration)
- [bool getVertexPosition](#) (shared\_ptr< [Vertex](#) > vx, [Point](#) &param) [const](#)
- [bool getVertexEnumeration](#) (shared\_ptr< [Vertex](#) > vx, [Point](#) &param, int &corner, int &coef\_nmb) [const](#)
- [bool getFreeBoundaryInfo](#) ([double](#) tol, std::vector< int > &free\_boundaries)
- [bool getBoundaryCoefEnumeration](#) (int bd, std::vector< int > &enumeration)
- int [ElementOnBoundary](#) (int elem\_ix)
- int [ElementBoundaryStatus](#) (int elem\_ix)
- [bool isBoundaryTrimmed](#) () [const](#)
- [bool isIsoTrimmed](#) () [const](#)
- [bool ParamInVolume](#) ([double](#) u, [double](#) v, [double](#) w)
- [bool regularizeBdShells](#) (std::vector< std::pair< [Point](#), [Point](#) > > &corr\_vx\_pts, std::vector< [SurfaceModel](#) \* > &modified\_adjacent, int split\_mode=1, [bool](#) pattern\_split=false)
- [bool isRegularized](#) () [const](#)
- [bool untrimRegular](#) (int degree)
  - Modify a regularized, possibly trimmed volume to become non-trimmed.*
- std::vector< shared\_ptr< [ftVolume](#) > > [replaceWithRegVolumes](#) (int degree, std::vector< [SurfaceModel](#) \* > &modified\_adjacent, [bool](#) performe\_step2=true, int split\_mode=1, [bool](#) pattern\_split=false)
- void [updateBoundaryInfo](#) ()
- [bool checkBodyTopology](#) ()
  - Debug.*

## Additional Inherited Members

### 29.160.1 Detailed Description

A topological solid with a trivariate geometry description.

Definition at line 91 of file [ftVolume.h](#).

### 29.160.2 Constructor & Destructor Documentation

#### 29.160.2.1 Go::ftVolume::ftVolume ( shared\_ptr< [ParamVolume](#) > vol, int id = -1 )

Constructor given a trivariate volume description.

29.160.2.2 `Go::ftVolume::ftVolume ( shared_ptr< ParamVolume > vol, double gap_eps, double kink_eps, int id = -1 )`

Constructor given a trivariate volume description and tolerances for topology analysis of the boundary shell

29.160.2.3 `Go::ftVolume::ftVolume ( shared_ptr< ParamVolume > vol, double gap_eps, double neighbour, double kink_eps, double bend, int id = -1 )`

29.160.2.4 `Go::ftVolume::ftVolume ( shared_ptr< ParamVolume > vol, shared_ptr< SurfaceModel > shell, int id = -1 )`

Given a volume and boundary surfaces, create a possibly trimmed [ftVolume](#)

29.160.2.5 `Go::ftVolume::ftVolume ( shared_ptr< ParamVolume > vol, std::vector< shared_ptr< SurfaceModel > > shells, int id = -1 )`

Given a volume and a number of boundary shells, create a possibly trimmed [ftVolume](#)

29.160.2.6 `Go::ftVolume::ftVolume ( shared_ptr< SurfaceModel > shell, int id = -1 )`

Create a [ftVolume](#) when no geometry description is known yet.

29.160.2.7 `Go::ftVolume::~~ftVolume ( )`

Destructor.

### 29.160.3 Member Function Documentation

29.160.3.1 `virtual BoundingBox Go::ftVolume::boundingBox ( ) const` [virtual]

The bounding box corresponding to this solid.

Reimplemented from [Go::Body](#).

29.160.3.2 `bool Go::ftVolume::checkBodyTopology ( )`

Debug.

29.160.3.3 `bool Go::ftVolume::checkDegAdjacency ( ftVolume * other, shared_ptr< EdgeVertex > evx, double tol, shared_ptr< ParamSurface > & bdsf1, shared_ptr< ParamSurface > & bdsf2 )`

Look for adjacency between degenerate volumes in a degenerate boundary surface

29.160.3.4 `ftFaceBase* Go::ftVolume::closestBoundaryPoint ( Point & pt, double & clo_u, double & clo_v, double & clo_w, Point & clo_pt, double & clo_dist, double & clo_par_u, double & clo_par_v, double epsilon ) const`

Closest point between a boundary on this possibly trimmed volume and a point

29.160.3.5 `void Go::ftVolume::closestPoint ( Point & pt, double & clo_u, double & clo_v, double & clo_w, Point & clo_pt, double & clo_dist, double epsilon, double * seed = NULL ) const`

Closest point between this possibly trimmed volume and a point.

29.160.3.6 `bool Go::ftVolume::commonSplineSpace ( ftVolume * other, double tol )`

Check if two bodies are neighbours, are splines and have a common spline space at the interface

29.160.3.7 `int Go::ftVolume::ElementBoundaryStatus ( int elem_ix )`

Check if a polynomial element (for spline volumes) intersects the (trimming) boundaries of this [ftVolume](#), is inside or outside

#### Parameters

|                      |                                                                                                                        |
|----------------------|------------------------------------------------------------------------------------------------------------------------|
| <code>elem_ix</code> | Element index counted according to distinct knot values. Sequence of coordinates: x runs fastest, then y and at last z |
|----------------------|------------------------------------------------------------------------------------------------------------------------|

#### Returns

-1: Not a spline volume or element index out of range 0: Outside trimmed volume 1: On boundary (intersection with boundary found) 2: Internal to trimmed volume Note that a touch with the boundaries of the underlying volume is not considered a boundary intersection while touching a trimming surface is seen as an intersection

29.160.3.8 `int Go::ftVolume::ElementOnBoundary ( int elem_ix )`

Check if a polynomial element (for spline volumes) intersects the (trimming) boundaries of this [ftVolume](#)

#### Parameters

|                      |                                                                                                                        |
|----------------------|------------------------------------------------------------------------------------------------------------------------|
| <code>elem_ix</code> | Element index counted according to distinct knot values. Sequence of coordinates: x runs fastest, then y and at last z |
|----------------------|------------------------------------------------------------------------------------------------------------------------|

#### Returns

-1: Not a spline volume or element index out of range 0: Not on boundary or touching a boundary surface 1: On boundary (intersection with boundary found) Note that a touch with the boundaries of the underlying volume is not considered a boundary intersection while touching a trimming surface is seen as an intersection

29.160.3.9 **bool** Go::ftVolume::getAdjacencyInfo ( **ftVolume** \* *other*, **double** *tol*, **int** & *bd1*, **int** & *bd2*, **int** & *orientation*, **bool** & *same\_seq* )

DEPRECATED METHOD. USE THE OTHER [getAdjacencyInfo\(\)](#) instead Fetch info on adjacency between neighbouring bodies return value true if adjacency is found and both bodies are non-trimmed or boundary trimmed bd1: boundary index of first volume 0 = umin, 1 = umax, 2 = vmin, 3 = vmax, 4 = wmin, 5 = wmax bd2: boundary index of second volume 0 = umin, 1 = umax, 2 = vmin, 3 = vmax, 4 = wmin, 5 = wmax orientation: Indicates whether the surfaces are equally oriented along the common boundary 0 = same orientation 1 = first parameter of the common boundary surface is opposite 2 = second parameter of the common boundary surface is opposite 3 = both parameters of the common boundary surface is opposite same\_seq: Indicates if the two parameter directions of the common boundary surface are given in the same sequence

29.160.3.10 **VolumeAdjacencyInfo** Go::ftVolume::getAdjacencyInfo ( **ftVolume** \* *other*, **double** *tol*, **int** *adj\_idx* = 0, **bool** *test\_corner* = false )

Fetch info on adjacency between neighbouring bodies.

29.160.3.11 **void** Go::ftVolume::getAdjacentBodies ( **std::vector**< **ftVolume** \* > & *neighbours* )

Get neighbouring bodies.

29.160.3.12 **bool** Go::ftVolume::getBoundaryCoefEnumeration ( **int** *bd*, **std::vector**< **int** > & *enumeration* )

For a spline volume, get the local enumeration of coefficients along a specified boundary return value: true if spline, false if not

29.160.3.13 **std::vector**< **shared\_ptr**< **EdgeVertex** > > Go::ftVolume::getCommonEdges ( **ftVolume** \* *other* ) **const**

Fetch all radial edges common to this model and another one.

29.160.3.14 **VolumeAdjacencyInfo** Go::ftVolume::getCornerAdjacencyInfo ( **ftVolume** \* *other*, **EdgeVertex** \* *evx*, **double** *tol*, **int** *adj\_idx* = 0 )

29.160.3.15 **VolumeAdjacencyInfo** Go::ftVolume::getCornerAdjacencyInfo ( **ftVolume** \* *other*, **double** *tol*, **int** *adj\_idx* = 0 )

Fetch info on adjacence between bodies meeting an a common edge (no common face)

29.160.3.16 **bool** Go::ftVolume::getCorrCoefEnumeration ( **ftVolume** \* *other*, **double** *tol*, **std::vector**< **std::pair**< **int**, **int** > > & *enumeration* )

Given two adjacent spline volumes represented in the same spline space, return local enumeration of the coefficients. This surface is given first and the other volume second

29.160.3.17 **bool** Go::ftVolume::getFreeBoundaryInfo ( **double** *tol*, **std::vector**< **int** > & *free\_boundaries* )

Return the boundary number of outer boundary surfaces that follow volume boundaries return value = true: All free boundaries are associated with volume boundaries = false:Some free boundaries are not associated with volume boundaries

29.160.3.18 **int** Go::ftVolume::getId ( ) [*inline*]

Fetch Id, not necessarily uniquely set.

Definition at line 131 of file ftVolume.h.

29.160.3.19 **bool** Go::ftVolume::getVertexEnumeration ( **shared\_ptr**< **Vertex** > *vx*, **Point** & *param*, **int** & *corner*, **int** & *coef\_nmb* ) **const**

Given a vertex, find if this vertex is associated this body, compute the parameter value of the body associated with the vertex, and if this body is a spline volume, return the corner number and the number of the associated spline coefficient corner: 0=(umin,vmin,wmin), 1=(umax,vmin,wmin), 2=(umin,vmax,wmin), 3=(umax,vmax,wmin), 4=(umin,vmin,wmax), 5=(umax,vmin,wmax), 6=(umin,vmax,wmax), 7=(umax,vmax,wmax)

29.160.3.20 **bool** Go::ftVolume::getVertexPosition ( **shared\_ptr**< **Vertex** > *vx*, **Point** & *param* ) **const**

Given a vertex, find if this vertex is associated this body and compute the parameter value of the body associated with the vertex

29.160.3.21 **shared\_ptr**<**ParamVolume**> Go::ftVolume::getVolume ( ) [*inline*]

Fetch geometric description.

Definition at line 125 of file ftVolume.h.

29.160.3.22 **bool** Go::ftVolume::isBoundaryTrimmed ( ) **const**

Information about whether or not the volume is trimmed and how it is trimmed Check if the volume is boundary trimmed (not trimmed). The boundary surfaces themselves may be trimmed

29.160.3.23 **bool** Go::ftVolume::isCornerToCorner ( **shared\_ptr**< **ftVolume** > *other*, **double** *tol* )

Check if two volumes are splines and meet in a corner-to-corner configuration

29.160.3.24 **bool** Go::ftVolume::isIsoTrimmed ( ) **const**

Check if all boundary surfaces correspond to an iso-parameter in the volume



29.160.3.25 **bool** Go::ftVolume::isRegularized ( ) const

Check if this volume has 6 boundary surfaces that may act as the boundary surfaces of a non-trimmed spline volume

29.160.3.26 **bool** Go::ftVolume::isSpline ( ) const

Check if the volume is represented as a spline.

29.160.3.27 **bool** Go::ftVolume::makeCommonSplineSpace ( ftVolume \* other )

Ensure that two neighbouring spline volumes have a common spline space at the interface

29.160.3.28 **bool** Go::ftVolume::ParamInVolume ( double u, double v, double w )

Check if all boundary surfaces approximately correspond to an iso-parameter in the volume Check if a given parameter tripple is inside the possibly trimmed volume

29.160.3.29 **std::vector<shared\_ptr<EdgeVertex> >** Go::ftVolume::radialEdges ( ) const

Fetch all radial edges belonging to this model.

29.160.3.30 **bool** Go::ftVolume::regularizeBdShells ( std::vector< std::pair< Point, Point > > & corr\_vx\_pts, std::vector< SurfaceModel \* > & modified\_adjacent, int split\_mode = 1, bool pattern\_split = false )

Update the volume by regularizing all boundary shells, i.e. all faces in all boundary shells of all connected volumes should be 4-sided, and no T-joints are allowed

29.160.3.31 **std::vector<shared\_ptr<ftVolume> >** Go::ftVolume::replaceWithRegVolumes ( int degree, std::vector< SurfaceModel \* > & modified\_adjacent, bool performe\_step2 = true, int split\_mode = 1, bool pattern\_split = false )

Divide a trimmed volume into a set of regular volumes NB! The boundary shells of the corresponding volume model (or this volume) must be regular. Not all configurations are handled. If an unknown configuration occur, nothing is returned Should be called from [VolumeModel](#). Ruins the current [ftVolume](#).

29.160.3.32 **void** Go::ftVolume::splitAtInternalCorner ( ftVolume \* other, std::vector< shared\_ptr< ftVolume > > & new\_vol1, std::vector< shared\_ptr< ftVolume > > & new\_vol2, double tol = DEFAULT\_SPACE\_EPSILON )

Ensure that two spline volumes meet in a corner to corner configuration

29.160.3.33 `std::vector<shared_ptr<ftEdge> > Go::ftVolume::uniqueNonRadialEdges ( ) const`

Fetch edges where no radial edge exist, i.e. there is no adjacent volume along any boundary surface meeting in this edge. Only one occurrence in an edge, twin-edge pair is returned

29.160.3.34 `bool Go::ftVolume::untrimRegular ( int degree )`

Modify a regularized, possibly trimmed volume to become non-trimmed.

29.160.3.35 `void Go::ftVolume::updateBoundaryInfo ( )`

Update boundary shells to reflect changes in the geometric volume while maintaining topology information

The documentation for this class was generated from the following file:

- [trivariatemodel/include/GoTools/trivariatemodel/ftVolume.h](#)

## 29.161 Go::Fun2Fun< Functor > Class Template Reference

```
#include <brent_minimize.h>
```

### Public Member Functions

- [Fun2Fun](#) (`const Functor &f`, `double a`, `double b`)
- [double operator\(\)](#) (`const double *arg`) `const`  
*Evaluate functor.*
- [double minPar](#) (`int n`) `const`  
*Return start parameter of functor.*
- [double maxPar](#) (`int n`) `const`  
*Return end parameter.*

### 29.161.1 Detailed Description

```
template<class Functor>
class Go::Fun2Fun< Functor >
```

Functor evaluation

Definition at line 51 of file `brent_minimize.h`.

### 29.161.2 Constructor & Destructor Documentation

29.161.2.1 `template<class Functor> Go::Fun2Fun< Functor >::Fun2Fun ( const Functor & f, double a, double b )`  
`[inline]`

Constructor

## Parameters

|          |                 |
|----------|-----------------|
| <i>f</i> | the functor     |
| <i>a</i> | start parameter |
| <i>b</i> | end parameter   |

Definition at line 58 of file brent\_minimize.h.

### 29.161.3 Member Function Documentation

29.161.3.1 `template<class Functor> double Go::Fun2Fun< Functor >::maxPar ( int n ) const` `[inline]`

Return end parameter.

Definition at line 69 of file brent\_minimize.h.

29.161.3.2 `template<class Functor> double Go::Fun2Fun< Functor >::minPar ( int n ) const` `[inline]`

Return start parameter of functor.

Definition at line 67 of file brent\_minimize.h.

29.161.3.3 `template<class Functor> double Go::Fun2Fun< Functor >::operator() ( const double * arg ) const`  
`[inline]`

Evaluate functor.

Definition at line 61 of file brent\_minimize.h.

The documentation for this class was generated from the following file:

- [gotools-core/include/GoTools/utils/brent\\_minimize.h](#)

## 29.162 Go::FunctionMinimizer< Functor > Class Template Reference

```
#include <GeneralFunctionMinimizer.h>
```

## Public Member Functions

- [FunctionMinimizer](#) (int num\_param, const Functor &fun, const double \*const seed, double tol=std::sqrt(std::numeric\_limits< double >::epsilon()))
- [~FunctionMinimizer](#) ()
- [double minimize](#) (const Point &dir, bool &hit\_domain\_edge, bool rerun=false)
- [double linminBrent](#) (const Point &dir, const double \*bracket, const double \*fval\_brak)
- [double fval](#) () const  
*evaluating distance function at the current point of evaluation*
- [double fval](#) (const Point &param) const
- [void grad](#) (Point &result) const
- [void grad](#) (const Point &param, Point &result) const
- [void moveUV](#) (const Point &dir, double multiplier)
- [bool atMin](#) (int param\_ix) const
- [bool atMax](#) (int param\_ix) const
- [double getPar](#) (int param\_ix) const  
*Get the current value for parameter nb. 'param\_ix'.*
- [const double \\* getPar](#) () const  
*Get a pointer to the array containing the current values of the parameters.*
- [int numPars](#) () const  
*Get the number of function parameters.*

### 29.162.1 Detailed Description

```
template<class Functor>
class Go::FunctionMinimizer< Functor >
```

This is the [FunctionMinimizer](#) class that can be used ex. with [minimise\\_conjugated\\_gradient](#). Together, they provide functionality for minimizing a function with an arbitrary number of parameters, using linear minimization along a set of directions which is chosen by the conjugated gradient algorithm, although the [FunctionMinimizer](#) can also be used independently for minimizing a function along any user-defined direction. The function to minimize must be presented in the form of a Functor, which has the following member functions:

```
/// double operator()(const double* arg) const; // to evaluate the function at the parameters
/// // pointed to by 'arg'.
/// void grad(const double* arg, double* grad) const; // to evaluate the function
/// // gradient at the parameters
/// // pointed to by 'arg'. The resulting
/// // vector will be written to 'grad'.
/// double minPar(int n) const; // return the lower bound of the n'th function parameter.
/// double maxPar(int n) const; // return the upper bound of the n'th function parameter.
///
```

Minimizing a function whose domain is  $R^n$  is a two-fold problem. One is of linear minimization along a specified direction. Another is to choose a set of directions to minimize along. The first problem is taken care of by the class [FunctionMinimizer](#), which wraps around the user-defined function and contains information about the current point of evaluation and minimizing along a given direction. The second problem can be taken care of by the supplied function '[minimise\\_conjugated\\_gradient](#)', or the user can use [FunctionMinimizer](#) in combination with his or her own direction-choosing algorithm.

The user first should wrap his or hers Functor into a [FunctionMinimizer](#), where the starting seed and tolerances should also be specified. The [FunctionMinimizer](#) can be used separately to minimize along directions that the user specify (using the '[minimize](#)' member function), or alternatively can be given to the supplied nonmember function '[minimise\\_conjugated\\_gradient](#)', which will try to find a local minimum starting from the supplied seed.

Example:

Let us assume that the user has a function taking four arguments. He defines the functor [MyFunction](#) containing the required member functions, and instanciates it:

```
/// MyFunction f(...constructor arguments...)
///
```

He now wraps it in a [FunctionMinimizer](#), giving the seed where he wants to start the search:

```
/// FunctionMinimizer<MyFunction> fmini(4, f, seed_p, 1.0e-8);
///
```

This function minimizer can now be used with the 'minimise\_conjugated\_gradient'-algorithm in order to search for a (local) minimum:

```
/// minimise_conjugated_gradient(fmini, 5);
///
```

The obtained 4-tuple of parameters corresponding to the found minimum can be obtained by:

```
/// const double* par_ptr = fmini.getPar();
///
```

Definition at line 49 of file GeneralFunctionMinimizer.h.

## 29.162.2 Constructor & Destructor Documentation

**29.162.2.1** `template<class Functor > Go::FunctionMinimizer< Functor >::FunctionMinimizer ( int num_param, const Functor & fun, const double *const seed, double tol = std::sqrt(std::numeric_limits<double>::epsilon()) ) [inline]`

Constructor. 'minimization\_tol' should in general be kept no lower than the square root of machine precision. 'seed' is a pointer to an array containing the start values for the search in the 'num\_param' parameters.

Definition at line 168 of file GeneralFunctionMinimizer\_implementation.h.

**29.162.2.2** `template<class Functor> Go::FunctionMinimizer< Functor >::~~FunctionMinimizer ( ) [inline]`

Definition at line 125 of file GeneralFunctionMinimizer.h.

## 29.162.3 Member Function Documentation

**29.162.3.1** `template<class Functor> bool Go::FunctionMinimizer< Functor >::atMax ( int param_ix ) const [inline]`

Definition at line 178 of file GeneralFunctionMinimizer.h.

**29.162.3.2** `template<class Functor> bool Go::FunctionMinimizer< Functor >::atMin ( int param_ix ) const [inline]`

Returns 'true' if the current parameter for the first/second curve is equal to the start/end-parameter for the curve.

Definition at line 177 of file GeneralFunctionMinimizer.h.

29.162.3.3 `template<class Functor > double Go::FunctionMinimizer< Functor >::fval ( ) const [inline]`

evaluating distance function at the current point of evaluation

Definition at line 730 of file GeneralFunctionMinimizer\_implementation.h.

29.162.3.4 `template<class Functor > double Go::FunctionMinimizer< Functor >::fval ( const Point & param ) const [inline]`

evaluating the distance function for an arbitrary parameter value (not the parameter pair).

Definition at line 742 of file GeneralFunctionMinimizer\_implementation.h.

29.162.3.5 `template<class Functor> double Go::FunctionMinimizer< Functor >::getPar ( int param_ix ) const [inline]`

Get the current value for parameter nb. 'param\_ix'.

Definition at line 181 of file GeneralFunctionMinimizer.h.

29.162.3.6 `template<class Functor> const double* Go::FunctionMinimizer< Functor >::getPar ( ) const [inline]`

Get a pointer to the array containing the current values of the parameters.

Definition at line 184 of file GeneralFunctionMinimizer.h.

29.162.3.7 `template<class Functor > void Go::FunctionMinimizer< Functor >::grad ( Point & result ) const [inline]`

evaluate the gradient (2 components) of the distance function at the currently set parameter pair of evaluation.

Definition at line 750 of file GeneralFunctionMinimizer\_implementation.h.

29.162.3.8 `template<class Functor > void Go::FunctionMinimizer< Functor >::grad ( const Point & param, Point & result ) const [inline]`

evaluate the gradient (2 components) of the distance function for an arbitrary parameter value (not the current parameter pair).

Definition at line 762 of file GeneralFunctionMinimizer\_implementation.h.

```
29.162.3.9 template<class Functor > double Go::FunctionMinimizer< Functor >::linminBrent (const Point & dir,
 const double * bracket, const double * fval_brak) [inline]
```

Move the current parameter point in order to minimize the function along a ray. The ray is given by the current parameter point and the parameter 'dir'. In contrast to [minimize\(\)](#), [linminBrent\(\)](#) requires that the minimum is bracketed. The method uses an algorithm inspired by Brent's method (with certain modifications). This method is rapid when the function is well-behaved. Otherwise it will use a slow but robust 'golden mean'-search. The current parameter point is moved to the minimum found along the given ray. Before calling this function, the minimum must be bracketed, and the brackets must be given to the function through the 'brackets' variable, and the corresponding function values through the 'fval\_brak' variable. The function value at the new minimum is returned.

Definition at line 324 of file GeneralFunctionMinimizer\_implementation.h.

```
29.162.3.10 template<class Functor > double Go::FunctionMinimizer< Functor >::minimize (const Point & dir,
 bool & hit_domain_edge, bool rerun = false) [inline]
```

Move the current parameter point in order to minimize the function along an unoriented line. The line is given by the current parameter point and the parameter 'dir'.

#### Parameters

|            |                                         |
|------------|-----------------------------------------|
| <i>dir</i> | a direction parallel to the search line |
|------------|-----------------------------------------|

#### Return values

|                        |                                                                       |
|------------------------|-----------------------------------------------------------------------|
| <i>hit_domain_edge</i> | indicates whether the found minimum is on the boundary of the domain. |
|------------------------|-----------------------------------------------------------------------|

Definition at line 209 of file GeneralFunctionMinimizer\_implementation.h.

```
29.162.3.11 template<class Functor > void Go::FunctionMinimizer< Functor >::moveUV (const Point & dir, double
 multiplier) [inline]
```

move the current parameter pair a certain distance ('multiplier') in a certain direction ('dir', which has 2 components)

Definition at line 770 of file GeneralFunctionMinimizer\_implementation.h.

```
29.162.3.12 template<class Functor> int Go::FunctionMinimizer< Functor >::numPars () const [inline]
```

Get the number of function parameters.

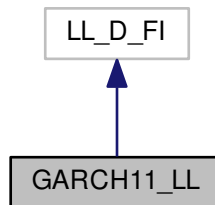
Definition at line 187 of file GeneralFunctionMinimizer.h.

The documentation for this class was generated from the following files:

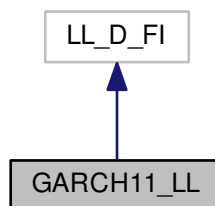
- [gotools-core/include/GoTools/utis/GeneralFunctionMinimizer.h](#)
- [gotools-core/include/GoTools/utis/GeneralFunctionMinimizer\\_implementation.h](#)

## 29.163 GARCH11\_LL Class Reference

Inheritance diagram for GARCH11\_LL:



Collaboration diagram for GARCH11\_LL:



### Public Member Functions

- [GARCH11\\_LL](#) ([const](#) ColumnVector &y, [const](#) ColumnVector &x)
- void [Set](#) ([const](#) ColumnVector &p)
- [bool](#) [IsValid](#) ()
- Real [LogLikelihood](#) ()
- [ReturnMatrix Derivatives](#) ()
- [ReturnMatrix FI](#) ()

### 29.163.1 Detailed Description

Definition at line 31 of file garch.cpp.

### 29.163.2 Constructor & Destructor Documentation

29.163.2.1 [GARCH11\\_LL::GARCH11\\_LL](#) ([const](#) ColumnVector & y, [const](#) ColumnVector & x ) [[inline](#)]

Definition at line 43 of file garch.cpp.



### 29.163.3 Member Function Documentation

#### 29.163.3.1 ReturnMatrix GARCH11\_LL::Derivatives ( )

Definition at line 130 of file garch.cpp.

#### 29.163.3.2 ReturnMatrix GARCH11\_LL::FI ( )

Definition at line 133 of file garch.cpp.

#### 29.163.3.3 bool GARCH11\_LL::IsValid ( )

Definition at line 60 of file garch.cpp.

#### 29.163.3.4 Real GARCH11\_LL::LogLikelihood ( )

Definition at line 63 of file garch.cpp.

#### 29.163.3.5 void GARCH11\_LL::Set ( const ColumnVector & p ) [inline]

Definition at line 47 of file garch.cpp.

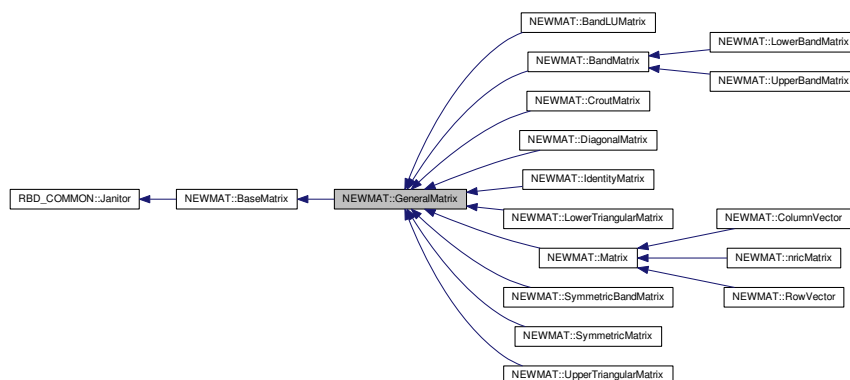
The documentation for this class was generated from the following file:

- [newmat/app/garch.cpp](#)

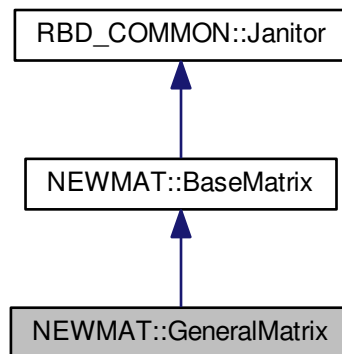
## 29.164 NEWMAT::GeneralMatrix Class Reference

```
#include <newmat.h>
```

Inheritance diagram for NEWMAT::GeneralMatrix:



Collaboration diagram for NEWMAT::GeneralMatrix:



## Public Member Functions

- [GeneralMatrix \\* Evaluate](#) ([MatrixType](#) mt=[MatrixTypeUnSp](#))
- virtual [MatrixType Type](#) () const =0
- int [Nrows](#) () const
- int [Ncols](#) () const
- int [Storage](#) () const
- [Real \\* Store](#) () const
- virtual [~GeneralMatrix](#) ()
- void [tDelete](#) ()
- [bool reuse](#) ()
- void [Protect](#) ()
- int [Tag](#) () const
- [bool IsZero](#) () const
- void [Release](#) ()
- void [Release](#) (int t)
- void [ReleaseAndDelete](#) ()
- void [operator<<](#) (const [Real](#) \*)
- void [operator<<](#) (const [BaseMatrix](#) &X)
- void [Inject](#) (const [GeneralMatrix](#) &)
- void [operator+=](#) (const [BaseMatrix](#) &)
- void [operator-=](#) (const [BaseMatrix](#) &)
- void [operator\\*=\[BaseMatrix\]\(#\) &](#)
- void [operator|=](#) (const [BaseMatrix](#) &)
- void [operator&=\[BaseMatrix\]\(#\) &](#)
- void [operator+=](#) ([Real](#))
- void [operator-=](#) ([Real](#) r)
- void [operator\\*=\[Real\]\(#\)](#)
- void [operator/=](#) ([Real](#) r)
- virtual [GeneralMatrix \\* MakeSolver](#) ()
- virtual void [Solver](#) ([MatrixColX](#) &, const [MatrixColX](#) &)
- virtual void [GetRow](#) ([MatrixRowCol](#) &)=0
- virtual void [RestoreRow](#) ([MatrixRowCol](#) &)

- virtual void [NextRow](#) ([MatrixRowCol](#) &)
- virtual void [GetCol](#) ([MatrixRowCol](#) &)=0
- virtual void [GetCol](#) ([MatrixColX](#) &)=0
- virtual void [RestoreCol](#) ([MatrixRowCol](#) &)
- virtual void [RestoreCol](#) ([MatrixColX](#) &)
- virtual void [NextCol](#) ([MatrixRowCol](#) &)
- virtual void [NextCol](#) ([MatrixColX](#) &)
- [Real SumSquare](#) () const
- [Real SumAbsoluteValue](#) () const
- [Real Sum](#) () const
- [Real MaximumAbsoluteValue1](#) (int &i) const
- [Real MinimumAbsoluteValue1](#) (int &i) const
- [Real Maximum1](#) (int &i) const
- [Real Minimum1](#) (int &i) const
- [Real MaximumAbsoluteValue](#) () const
- [Real MaximumAbsoluteValue2](#) (int &i, int &j) const
- [Real MinimumAbsoluteValue](#) () const
- [Real MinimumAbsoluteValue2](#) (int &i, int &j) const
- [Real Maximum](#) () const
- [Real Maximum2](#) (int &i, int &j) const
- [Real Minimum](#) () const
- [Real Minimum2](#) (int &i, int &j) const
- [LogAndSign LogDeterminant](#) () const
- virtual [bool IsEqual](#) (const [GeneralMatrix](#) &) const
- void [CheckStore](#) () const
- virtual void [SetParameters](#) (const [GeneralMatrix](#) \*)
- [operator ReturnMatrix](#) () const
- [ReturnMatrix ForReturn](#) () const
- virtual [bool SameStorageType](#) (const [GeneralMatrix](#) &A) const
- virtual void [ReSizeForAdd](#) (const [GeneralMatrix](#) &A, const [GeneralMatrix](#) &B)
- virtual void [ReSizeForSP](#) (const [GeneralMatrix](#) &A, const [GeneralMatrix](#) &B)
- virtual void [ReSize](#) (const [GeneralMatrix](#) &A)
- [MatrixInput operator<<](#) ([Real](#))
- [MatrixInput operator<<](#) (int f)
- void [CleanUp](#) ()

### Protected Member Functions

- [GeneralMatrix](#) ()
- [GeneralMatrix](#) ([ArrayLengthSpecifier](#))
- void [Add](#) ([GeneralMatrix](#) \*, [Real](#))
- void [Add](#) ([Real](#))
- void [NegAdd](#) ([GeneralMatrix](#) \*, [Real](#))
- void [NegAdd](#) ([Real](#))
- void [Multiply](#) ([GeneralMatrix](#) \*, [Real](#))
- void [Multiply](#) ([Real](#))
- void [Negate](#) ([GeneralMatrix](#) \*)
- void [Negate](#) ()
- void [ReverseElements](#) ()
- void [ReverseElements](#) ([GeneralMatrix](#) \*)
- void [operator=](#) ([Real](#))
- [Real \\* GetStore](#) ()
- [GeneralMatrix \\* BorrowStore](#) ([GeneralMatrix](#) \*, [MatrixType](#))
- void [GetMatrix](#) (const [GeneralMatrix](#) \*)

- void [Eq](#) (const [BaseMatrix](#) &, [MatrixType](#))
- void [Eq](#) (const [BaseMatrix](#) &, [MatrixType](#), bool)
- void [Eq2](#) (const [BaseMatrix](#) &, [MatrixType](#))
- int [search](#) (const [BaseMatrix](#) \*) const
- virtual [GeneralMatrix](#) \* [Transpose](#) ([TransposedMatrix](#) \*, [MatrixType](#))
- void [CheckConversion](#) (const [BaseMatrix](#) &)
- void [ReSize](#) (int, int, int)
- virtual short [SimpleAddOK](#) (const [GeneralMatrix](#) \*gm)

### Protected Attributes

- int [tag](#)
- int [nrows](#)
- int [ncols](#)
- int [storage](#)
- [Real](#) \* [store](#)

### Friends

- class [Matrix](#)
- class [nricMatrix](#)
- class [SymmetricMatrix](#)
- class [UpperTriangularMatrix](#)
- class [LowerTriangularMatrix](#)
- class [DiagonalMatrix](#)
- class [CroutMatrix](#)
- class [RowVector](#)
- class [ColumnVector](#)
- class [BandMatrix](#)
- class [LowerBandMatrix](#)
- class [UpperBandMatrix](#)
- class [SymmetricBandMatrix](#)
- class [BaseMatrix](#)
- class [AddedMatrix](#)
- class [MultipliedMatrix](#)
- class [SubtractedMatrix](#)
- class [SPMatrix](#)
- class [KPMatrix](#)
- class [ConcatenatedMatrix](#)
- class [StackedMatrix](#)
- class [SolvedMatrix](#)
- class [ShiftedMatrix](#)
- class [NegShiftedMatrix](#)
- class [ScaledMatrix](#)
- class [TransposedMatrix](#)
- class [ReversedMatrix](#)
- class [NegatedMatrix](#)
- class [InvertedMatrix](#)
- class [RowedMatrix](#)
- class [ColedMatrix](#)
- class [DiagedMatrix](#)
- class [MatedMatrix](#)
- class [GetSubMatrix](#)
- class [ReturnMatrixX](#)
- class [LinearEquationSolver](#)
- class [GenericMatrix](#)

### 29.164.1 Detailed Description

Definition at line 415 of file newmat.h.

### 29.164.2 Constructor & Destructor Documentation

#### 29.164.2.1 GeneralMatrix::GeneralMatrix ( ) [protected]

Definition at line 33 of file newmat4.cpp.

#### 29.164.2.2 NEWMAT::GeneralMatrix::GeneralMatrix ( **ArrayLengthSpecifier** ) [protected]

#### 29.164.2.3 GeneralMatrix::~GeneralMatrix ( ) [virtual]

Definition at line 132 of file newmat4.cpp.

### 29.164.3 Member Function Documentation

#### 29.164.3.1 void NEWMAT::GeneralMatrix::Add ( **GeneralMatrix** \* , **Real** ) [protected]

#### 29.164.3.2 void NEWMAT::GeneralMatrix::Add ( **Real** ) [protected]

#### 29.164.3.3 **GeneralMatrix** \* GeneralMatrix::BorrowStore ( **GeneralMatrix** \* *gmx*, **MatrixType** *mt* ) [protected]

Definition at line 587 of file newmat4.cpp.

#### 29.164.3.4 void NEWMAT::GeneralMatrix::CheckConversion ( **const BaseMatrix** & ) [protected]

#### 29.164.3.5 void GeneralMatrix::CheckStore ( ) const

Definition at line 808 of file newmat4.cpp.

#### 29.164.3.6 void GeneralMatrix::CleanUp ( ) [virtual]

Reimplemented from [NEWMAT::BaseMatrix](#).

Reimplemented in [NEWMAT::BandLUMatrix](#), [NEWMAT::CroutMatrix](#), [NEWMAT::ColumnVector](#), [NEWMAT::Row↔Vector](#), and [NEWMAT::nricMatrix](#).

Definition at line 817 of file newmat4.cpp.

29.164.3.7 void NEWMAT::GeneralMatrix::Eq ( const BaseMatrix &, MatrixType ) [protected]

29.164.3.8 void NEWMAT::GeneralMatrix::Eq ( const BaseMatrix &, MatrixType, bool ) [protected]

29.164.3.9 void GeneralMatrix::Eq2 ( const BaseMatrix & X, MatrixType mt ) [protected]

Definition at line 654 of file newmat4.cpp.

29.164.3.10 GeneralMatrix \* GeneralMatrix::Evaluate ( MatrixType mt = MatrixTypeUnSp ) [virtual]

Implements [NEWMAT::BaseMatrix](#).

Definition at line 79 of file newmat5.cpp.

29.164.3.11 ReturnMatrixX GeneralMatrix::ForReturn ( ) const

Definition at line 190 of file newmat4.cpp.

29.164.3.12 virtual void NEWMAT::GeneralMatrix::GetCol ( MatrixRowCol & ) [pure virtual]

Implemented in [NEWMAT::IdentityMatrix](#), [NEWMAT::BandLUMatrix](#), [NEWMAT::SymmetricBandMatrix](#), [NEWMAT::BandMatrix](#), [NEWMAT::CroutMatrix](#), [NEWMAT::RowVector](#), [NEWMAT::DiagonalMatrix](#), [NEWMAT::LowerTriangularMatrix](#), [NEWMAT::UpperTriangularMatrix](#), [NEWMAT::SymmetricMatrix](#), and [NEWMAT::Matrix](#).

29.164.3.13 virtual void NEWMAT::GeneralMatrix::GetCol ( MatrixColIX & ) [pure virtual]

Implemented in [NEWMAT::IdentityMatrix](#), [NEWMAT::BandLUMatrix](#), [NEWMAT::SymmetricBandMatrix](#), [NEWMAT::BandMatrix](#), [NEWMAT::CroutMatrix](#), [NEWMAT::RowVector](#), [NEWMAT::DiagonalMatrix](#), [NEWMAT::LowerTriangularMatrix](#), [NEWMAT::UpperTriangularMatrix](#), [NEWMAT::SymmetricMatrix](#), and [NEWMAT::Matrix](#).

29.164.3.14 void GeneralMatrix::GetMatrix ( const GeneralMatrix \* gmx ) [protected]

Definition at line 580 of file newmat4.cpp.

29.164.3.15 virtual void NEWMAT::GeneralMatrix::GetRow ( MatrixRowCol & ) [pure virtual]

Implemented in [NEWMAT::IdentityMatrix](#), [NEWMAT::BandLUMatrix](#), [NEWMAT::SymmetricBandMatrix](#), [NEWMAT::BandMatrix](#), [NEWMAT::CroutMatrix](#), [NEWMAT::DiagonalMatrix](#), [NEWMAT::LowerTriangularMatrix](#), [NEWMAT::UpperTriangularMatrix](#), [NEWMAT::SymmetricMatrix](#), and [NEWMAT::Matrix](#).

29.164.3.16 Real \* GeneralMatrix::GetStore ( ) [protected]

Definition at line 557 of file newmat4.cpp.

29.164.3.17 `void GeneralMatrix::Inject ( const GeneralMatrix & X )`

Definition at line 666 of file `newmat4.cpp`.

29.164.3.18 `bool GeneralMatrix::IsEqual ( const GeneralMatrix & A ) const` [virtual]

Reimplemented in [NEWMAT::BandLUMatrix](#), and [NEWMAT::CroutMatrix](#).

Definition at line 996 of file `newmat7.cpp`.

29.164.3.19 `bool GeneralMatrix::IsZero ( ) const`

Definition at line 969 of file `newmat7.cpp`.

29.164.3.20 `LogAndSign GeneralMatrix::LogDeterminant ( ) const` [virtual]

Reimplemented from [NEWMAT::BaseMatrix](#).

Reimplemented in [NEWMAT::IdentityMatrix](#), [NEWMAT::BandLUMatrix](#), [NEWMAT::SymmetricBandMatrix](#), [NEWMAT::LowerBandMatrix](#), [NEWMAT::UpperBandMatrix](#), [NEWMAT::BandMatrix](#), [NEWMAT::CroutMatrix](#), [NEWMAT::DiagonalMatrix](#), [NEWMAT::LowerTriangularMatrix](#), and [NEWMAT::UpperTriangularMatrix](#).

Definition at line 686 of file `newmat8.cpp`.

29.164.3.21 `GeneralMatrix * GeneralMatrix::MakeSolver ( )` [virtual]

Reimplemented in [NEWMAT::IdentityMatrix](#), [NEWMAT::BandLUMatrix](#), [NEWMAT::SymmetricBandMatrix](#), [NEWMAT::LowerBandMatrix](#), [NEWMAT::UpperBandMatrix](#), [NEWMAT::BandMatrix](#), [NEWMAT::CroutMatrix](#), [NEWMAT::DiagonalMatrix](#), [NEWMAT::LowerTriangularMatrix](#), [NEWMAT::UpperTriangularMatrix](#), and [NEWMAT::Matrix](#).

Definition at line 24 of file `newmat7.cpp`.

29.164.3.22 `Real GeneralMatrix::Maximum ( ) const` [virtual]

Reimplemented from [NEWMAT::BaseMatrix](#).

Reimplemented in [NEWMAT::SymmetricBandMatrix](#), and [NEWMAT::BandMatrix](#).

Definition at line 252 of file `newmat8.cpp`.

29.164.3.23 `Real GeneralMatrix::Maximum1 ( int & i ) const` [virtual]

Reimplemented from [NEWMAT::BaseMatrix](#).

Definition at line 261 of file `newmat8.cpp`.

**29.164.3.24** **Real GeneralMatrix::Maximum2 ( int & i, int & j ) const** [virtual]

Reimplemented from [NEWMAT::BaseMatrix](#).

Reimplemented in [NEWMAT::Matrix](#).

Definition at line 320 of file newmat8.cpp.

**29.164.3.25** **Real GeneralMatrix::MaximumAbsoluteValue ( ) const** [virtual]

Reimplemented from [NEWMAT::BaseMatrix](#).

Reimplemented in [NEWMAT::SymmetricBandMatrix](#), and [NEWMAT::BandMatrix](#).

Definition at line 212 of file newmat8.cpp.

**29.164.3.26** **Real GeneralMatrix::MaximumAbsoluteValue1 ( int & i ) const** [virtual]

Reimplemented from [NEWMAT::BaseMatrix](#).

Definition at line 221 of file newmat8.cpp.

**29.164.3.27** **Real GeneralMatrix::MaximumAbsoluteValue2 ( int & i, int & j ) const** [virtual]

Reimplemented from [NEWMAT::BaseMatrix](#).

Reimplemented in [NEWMAT::Matrix](#).

Definition at line 290 of file newmat8.cpp.

**29.164.3.28** **Real GeneralMatrix::Minimum ( ) const** [virtual]

Reimplemented from [NEWMAT::BaseMatrix](#).

Reimplemented in [NEWMAT::SymmetricBandMatrix](#), and [NEWMAT::BandMatrix](#).

Definition at line 271 of file newmat8.cpp.

**29.164.3.29** **Real GeneralMatrix::Minimum1 ( int & i ) const** [virtual]

Reimplemented from [NEWMAT::BaseMatrix](#).

Definition at line 280 of file newmat8.cpp.



29.164.3.30 **Real** GeneralMatrix::Minimum2 ( int & i, int & j ) const [virtual]

Reimplemented from [NEWMAT::BaseMatrix](#).

Reimplemented in [NEWMAT::Matrix](#).

Definition at line 335 of file newmat8.cpp.

29.164.3.31 **Real** GeneralMatrix::MinimumAbsoluteValue ( ) const [virtual]

Reimplemented from [NEWMAT::BaseMatrix](#).

Reimplemented in [NEWMAT::SymmetricBandMatrix](#), and [NEWMAT::BandMatrix](#).

Definition at line 232 of file newmat8.cpp.

29.164.3.32 **Real** GeneralMatrix::MinimumAbsoluteValue1 ( int & i ) const [virtual]

Reimplemented from [NEWMAT::BaseMatrix](#).

Definition at line 241 of file newmat8.cpp.

29.164.3.33 **Real** GeneralMatrix::MinimumAbsoluteValue2 ( int & i, int & j ) const [virtual]

Reimplemented from [NEWMAT::BaseMatrix](#).

Reimplemented in [NEWMAT::Matrix](#).

Definition at line 305 of file newmat8.cpp.

29.164.3.34 void NEWMAT::GeneralMatrix::Multiply ( **GeneralMatrix** \*, **Real** ) [protected]

29.164.3.35 void NEWMAT::GeneralMatrix::Multiply ( **Real** ) [protected]

29.164.3.36 int NEWMAT::GeneralMatrix::Ncols ( ) const [inline]

Definition at line 453 of file newmat.h.

29.164.3.37 void NEWMAT::GeneralMatrix::NegAdd ( **GeneralMatrix** \*, **Real** ) [protected]

29.164.3.38 void NEWMAT::GeneralMatrix::NegAdd ( **Real** ) [protected]

29.164.3.39 void NEWMAT::GeneralMatrix::Negate ( **GeneralMatrix** \* ) [protected]

29.164.3.40 void GeneralMatrix::Negate ( ) [protected]

Definition at line 434 of file newmat5.cpp.

29.164.3.41 `virtual void NEWMAT::GeneralMatrix::NextCol ( MatrixRowCol & ) [virtual]`

Reimplemented in [NEWMAT::IdentityMatrix](#), [NEWMAT::RowVector](#), [NEWMAT::DiagonalMatrix](#), and [NEWMAT::↵↵  
Matrix](#).

29.164.3.42 `virtual void NEWMAT::GeneralMatrix::NextCol ( MatrixColX & ) [virtual]`

Reimplemented in [NEWMAT::IdentityMatrix](#), [NEWMAT::RowVector](#), [NEWMAT::DiagonalMatrix](#), and [NEWMAT::↵↵  
Matrix](#).

29.164.3.43 `void GeneralMatrix::NextRow ( MatrixRowCol & mrc ) [virtual]`

Reimplemented in [NEWMAT::IdentityMatrix](#), [NEWMAT::BandMatrix](#), [NEWMAT::DiagonalMatrix](#), [NEWMAT::↵↵  
LowerTriangularMatrix](#), [NEWMAT::UpperTriangularMatrix](#), and [NEWMAT::Matrix](#).

Definition at line 81 of file `newmat3.cpp`.

29.164.3.44 `int NEWMAT::GeneralMatrix::Nrows ( ) const [inline]`

Definition at line 452 of file `newmat.h`.

29.164.3.45 `NEWMAT::GeneralMatrix::operator ReturnMatrix ( ) const`

29.164.3.46 `void GeneralMatrix::operator&= ( const BaseMatrix & X )`

Definition at line 693 of file `newmat6.cpp`.

29.164.3.47 `void NEWMAT::GeneralMatrix::operator*=( const BaseMatrix & )`

29.164.3.48 `void NEWMAT::GeneralMatrix::operator*=( Real )`

29.164.3.49 `void NEWMAT::GeneralMatrix::operator+=( const BaseMatrix & )`

29.164.3.50 `void NEWMAT::GeneralMatrix::operator+=( Real )`

29.164.3.51 `void NEWMAT::GeneralMatrix::operator-=( const BaseMatrix & )`

29.164.3.52 `void NEWMAT::GeneralMatrix::operator-=( Real r ) [inline]`

Definition at line 475 of file `newmat.h`.

29.164.3.53 `void NEWMAT::GeneralMatrix::operator/=( Real r ) [inline]`

Definition at line 477 of file `newmat.h`.

29.164.3.54 void NEWMAT::GeneralMatrix::operator<<( const Real \* )

29.164.3.55 void NEWMAT::GeneralMatrix::operator<<( const BaseMatrix & X ) [inline]

Definition at line 466 of file newmat.h.

29.164.3.56 MatrixInput NEWMAT::GeneralMatrix::operator<<( Real )

29.164.3.57 MatrixInput NEWMAT::GeneralMatrix::operator<<( int f ) [inline]

Definition at line 1784 of file newmat.h.

29.164.3.58 void GeneralMatrix::operator=( Real f ) [protected]

Definition at line 511 of file newmat6.cpp.

29.164.3.59 void GeneralMatrix::operator|=( const BaseMatrix & X )

Definition at line 673 of file newmat6.cpp.

29.164.3.60 void NEWMAT::GeneralMatrix::Protect( ) [inline]

Definition at line 459 of file newmat.h.

29.164.3.61 void NEWMAT::GeneralMatrix::Release( ) [inline]

Definition at line 462 of file newmat.h.

29.164.3.62 void NEWMAT::GeneralMatrix::Release( int f ) [inline]

Definition at line 463 of file newmat.h.

29.164.3.63 void NEWMAT::GeneralMatrix::ReleaseAndDelete( ) [inline]

Definition at line 464 of file newmat.h.

29.164.3.64 void NEWMAT::GeneralMatrix::ReSize( int, int, int ) [protected]

29.164.3.65 virtual void NEWMAT::GeneralMatrix::ReSize( const GeneralMatrix & A ) [virtual]

Reimplemented in [NEWMAT::IdentityMatrix](#), [NEWMAT::SymmetricBandMatrix](#), [NEWMAT::LowerBandMatrix](#), [NEWMAT::UpperBandMatrix](#), [NEWMAT::BandMatrix](#), [NEWMAT::ColumnVector](#), [NEWMAT::RowVector](#), [NEWMAT::DiagonalMatrix](#), [NEWMAT::LowerTriangularMatrix](#), [NEWMAT::UpperTriangularMatrix](#), [NEWMAT::SymmetricMatrix](#), [NEWMAT::nricMatrix](#), and [NEWMAT::Matrix](#).

29.164.3.66 `void GeneralMatrix::ReSizeForAdd ( const GeneralMatrix & A, const GeneralMatrix & B )` [virtual]

Reimplemented in [NEWMAT::SymmetricBandMatrix](#), and [NEWMAT::BandMatrix](#).

Definition at line 346 of file `newmat4.cpp`.

29.164.3.67 `void GeneralMatrix::ReSizeForSP ( const GeneralMatrix & A, const GeneralMatrix & B )` [virtual]

Reimplemented in [NEWMAT::SymmetricBandMatrix](#), and [NEWMAT::BandMatrix](#).

Definition at line 349 of file `newmat4.cpp`.

29.164.3.68 `virtual void NEWMAT::GeneralMatrix::RestoreCol ( MatrixRowCol & )` [inline],[virtual]

Reimplemented in [NEWMAT::SymmetricBandMatrix](#), [NEWMAT::BandMatrix](#), [NEWMAT::RowVector](#), [NEWMAT::LowerTriangularMatrix](#), [NEWMAT::UpperTriangularMatrix](#), [NEWMAT::SymmetricMatrix](#), and [NEWMAT::Matrix](#).

Definition at line 485 of file `newmat.h`.

29.164.3.69 `virtual void NEWMAT::GeneralMatrix::RestoreCol ( MatrixColIX & )` [inline],[virtual]

Reimplemented in [NEWMAT::SymmetricBandMatrix](#), [NEWMAT::BandMatrix](#), [NEWMAT::RowVector](#), [NEWMAT::LowerTriangularMatrix](#), [NEWMAT::UpperTriangularMatrix](#), [NEWMAT::SymmetricMatrix](#), and [NEWMAT::Matrix](#).

Definition at line 486 of file `newmat.h`.

29.164.3.70 `virtual void NEWMAT::GeneralMatrix::RestoreRow ( MatrixRowCol & )` [inline],[virtual]

Definition at line 481 of file `newmat.h`.

29.164.3.71 `bool GeneralMatrix::reuse ( )`

Definition at line 538 of file `newmat4.cpp`.

29.164.3.72 `void GeneralMatrix::ReverseElements ( )` [protected]

Definition at line 550 of file `newmat5.cpp`.

29.164.3.73 `void NEWMAT::GeneralMatrix::ReverseElements ( GeneralMatrix * )` [protected]

29.164.3.74 `bool GeneralMatrix::SameStorageType ( const GeneralMatrix & A ) const` [virtual]

Reimplemented in [NEWMAT::SymmetricBandMatrix](#), and [NEWMAT::BandMatrix](#).

Definition at line 358 of file `newmat4.cpp`.

29.164.3.75 `int GeneralMatrix::search ( const BaseMatrix * s ) const` [protected],[virtual]

Implements [NEWMAT::BaseMatrix](#).

Definition at line 367 of file `newmat4.cpp`.

29.164.3.76 `virtual void NEWMAT::GeneralMatrix::SetParameters ( const GeneralMatrix * )` [inline],[virtual]

Reimplemented in [NEWMAT::SymmetricBandMatrix](#), and [NEWMAT::BandMatrix](#).

Definition at line 508 of file `newmat.h`.

29.164.3.77 `virtual short NEWMAT::GeneralMatrix::SimpleAddOK ( const GeneralMatrix * gm )` [inline],[protected],[virtual]

Reimplemented in [NEWMAT::BandMatrix](#).

Definition at line 447 of file `newmat.h`.

29.164.3.78 `virtual void NEWMAT::GeneralMatrix::Solver ( MatrixCoIX & , const MatrixCoIX & )` [inline],[virtual]

Reimplemented in [NEWMAT::IdentityMatrix](#), [NEWMAT::BandLUMatrix](#), [NEWMAT::LowerBandMatrix](#), [NEWMAT::UpperBandMatrix](#), [NEWMAT::CroutMatrix](#), [NEWMAT::DiagonalMatrix](#), [NEWMAT::LowerTriangularMatrix](#), and [NEWMAT::UpperTriangularMatrix](#).

Definition at line 479 of file `newmat.h`.

29.164.3.79 `int NEWMAT::GeneralMatrix::Storage ( ) const` [inline]

Definition at line 454 of file `newmat.h`.

29.164.3.80 `Real* NEWMAT::GeneralMatrix::Store ( ) const` [inline]

Definition at line 455 of file `newmat.h`.

29.164.3.81 `Real GeneralMatrix::Sum ( ) const` [virtual]

Reimplemented from [NEWMAT::BaseMatrix](#).

Reimplemented in [NEWMAT::IdentityMatrix](#), [NEWMAT::SymmetricBandMatrix](#), [NEWMAT::BandMatrix](#), and [NEWMAT::SymmetricMatrix](#).

Definition at line 166 of file `newmat8.cpp`.

29.164.3.82 **Real GeneralMatrix::SumAbsoluteValue ( ) const** [virtual]

Reimplemented from [NEWMAT::BaseMatrix](#).

Reimplemented in [NEWMAT::IdentityMatrix](#), [NEWMAT::SymmetricBandMatrix](#), [NEWMAT::BandMatrix](#), and [NEWMAT::SymmetricMatrix](#).

Definition at line 158 of file newmat8.cpp.

29.164.3.83 **Real GeneralMatrix::SumSquare ( ) const** [virtual]

Reimplemented from [NEWMAT::BaseMatrix](#).

Reimplemented in [NEWMAT::IdentityMatrix](#), [NEWMAT::SymmetricBandMatrix](#), [NEWMAT::BandMatrix](#), and [NEWMAT::SymmetricMatrix](#).

Definition at line 150 of file newmat8.cpp.

29.164.3.84 **int NEWMAT::GeneralMatrix::Tag ( ) const** [inline]

Definition at line 460 of file newmat.h.

29.164.3.85 **void GeneralMatrix::tDelete ( )**

Definition at line 506 of file newmat4.cpp.

29.164.3.86 **GeneralMatrix \* GeneralMatrix::Transpose ( TransposedMatrix \* tm, MatrixType mt )**  
[protected], [virtual]

Reimplemented in [NEWMAT::IdentityMatrix](#), [NEWMAT::SymmetricBandMatrix](#), [NEWMAT::ColumnVector](#), [NEWMAT::RowVector](#), [NEWMAT::DiagonalMatrix](#), and [NEWMAT::SymmetricMatrix](#).

Definition at line 27 of file newmat5.cpp.

29.164.3.87 **virtual MatrixType NEWMAT::GeneralMatrix::Type ( ) const** [pure virtual]

Implemented in [NEWMAT::IdentityMatrix](#), [NEWMAT::BandLUMatrix](#), [NEWMAT::SymmetricBandMatrix](#), [NEWMAT::LowerBandMatrix](#), [NEWMAT::UpperBandMatrix](#), [NEWMAT::BandMatrix](#), [NEWMAT::CroutMatrix](#), [NEWMAT::ColumnVector](#), [NEWMAT::RowVector](#), [NEWMAT::DiagonalMatrix](#), [NEWMAT::LowerTriangularMatrix](#), [NEWMAT::UpperTriangularMatrix](#), [NEWMAT::SymmetricMatrix](#), and [NEWMAT::Matrix](#).

## 29.164.4 Friends And Related Function Documentation

29.164.4.1 **friend class AddedMatrix** [friend]

Definition at line 535 of file newmat.h.

29.164.4.2 friend class **BandMatrix** [friend]

Definition at line 530 of file newmat.h.

29.164.4.3 friend class **BaseMatrix** [friend]

Definition at line 534 of file newmat.h.

29.164.4.4 friend class **ColedMatrix** [friend]

Definition at line 551 of file newmat.h.

29.164.4.5 friend class **ColumnVector** [friend]

Definition at line 529 of file newmat.h.

29.164.4.6 friend class **ConcatenatedMatrix** [friend]

Definition at line 540 of file newmat.h.

29.164.4.7 friend class **CroutMatrix** [friend]

Definition at line 527 of file newmat.h.

29.164.4.8 friend class **DiagedMatrix** [friend]

Definition at line 552 of file newmat.h.

29.164.4.9 friend class **DiagonalMatrix** [friend]

Definition at line 526 of file newmat.h.

29.164.4.10 friend class **GenericMatrix** [friend]

Definition at line 557 of file newmat.h.

29.164.4.11 friend class **GetSubMatrix** [friend]

Definition at line 554 of file newmat.h.

29.164.4.12 friend class **InvertedMatrix** [friend]

Definition at line 549 of file newmat.h.

29.164.4.13 friend class **KPMatrix** [friend]

Definition at line 539 of file newmat.h.

29.164.4.14 friend class **LinearEquationSolver** [friend]

Definition at line 556 of file newmat.h.

29.164.4.15 friend class **LowerBandMatrix** [friend]

Definition at line 531 of file newmat.h.

29.164.4.16 friend class **LowerTriangularMatrix** [friend]

Definition at line 525 of file newmat.h.

29.164.4.17 friend class **MatedMatrix** [friend]

Definition at line 553 of file newmat.h.

29.164.4.18 friend class **Matrix** [friend]

Definition at line 521 of file newmat.h.

29.164.4.19 friend class **MultipliedMatrix** [friend]

Definition at line 536 of file newmat.h.

29.164.4.20 friend class **NegatedMatrix** [friend]

Definition at line 548 of file newmat.h.

29.164.4.21 friend class **NegShiftedMatrix** [friend]

Definition at line 544 of file newmat.h.



29.164.4.22 friend class **nricMatrix** [friend]

Definition at line 522 of file newmat.h.

29.164.4.23 friend class **ReturnMatrixX** [friend]

Definition at line 555 of file newmat.h.

29.164.4.24 friend class **ReversedMatrix** [friend]

Definition at line 547 of file newmat.h.

29.164.4.25 friend class **RowedMatrix** [friend]

Definition at line 550 of file newmat.h.

29.164.4.26 friend class **RowVector** [friend]

Definition at line 528 of file newmat.h.

29.164.4.27 friend class **ScaledMatrix** [friend]

Definition at line 545 of file newmat.h.

29.164.4.28 friend class **ShiftedMatrix** [friend]

Definition at line 543 of file newmat.h.

29.164.4.29 friend class **SolvedMatrix** [friend]

Definition at line 542 of file newmat.h.

29.164.4.30 friend class **SPMatrix** [friend]

Definition at line 538 of file newmat.h.

29.164.4.31 friend class **StackedMatrix** [friend]

Definition at line 541 of file newmat.h.

29.164.4.32 friend class **SubtractedMatrix** [friend]

Definition at line 537 of file newmat.h.

29.164.4.33 friend class **SymmetricBandMatrix** [friend]

Definition at line 533 of file newmat.h.

29.164.4.34 friend class **SymmetricMatrix** [friend]

Definition at line 523 of file newmat.h.

29.164.4.35 friend class **TransposedMatrix** [friend]

Definition at line 546 of file newmat.h.

29.164.4.36 friend class **UpperBandMatrix** [friend]

Definition at line 532 of file newmat.h.

29.164.4.37 friend class **UpperTriangularMatrix** [friend]

Definition at line 524 of file newmat.h.

## 29.164.5 Member Data Documentation

29.164.5.1 int **NEWMAT::GeneralMatrix::ncols** [protected]

Definition at line 420 of file newmat.h.

29.164.5.2 int **NEWMAT::GeneralMatrix::nrows** [protected]

Definition at line 420 of file newmat.h.

29.164.5.3 int **NEWMAT::GeneralMatrix::storage** [protected]

Definition at line 421 of file newmat.h.

29.164.5.4 **Real\*** **NEWMAT::GeneralMatrix::store** [protected]

Definition at line 422 of file newmat.h.

29.164.5.5 int NEWMAT::GeneralMatrix::tag [protected]

Definition at line 419 of file newmat.h.

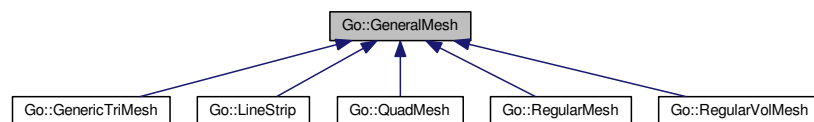
The documentation for this class was generated from the following files:

- [newmat/include/newmat.h](#)
- [newmat/src/newmat3.cpp](#)
- [newmat/src/newmat4.cpp](#)
- [newmat/src/newmat5.cpp](#)
- [newmat/src/newmat6.cpp](#)
- [newmat/src/newmat7.cpp](#)
- [newmat/src/newmat8.cpp](#)

## 29.165 Go::GeneralMesh Class Reference

```
#include <GeneralMesh.h>
```

Inheritance diagram for Go::GeneralMesh:



### Public Member Functions

- [GeneralMesh](#) ()
- virtual [~GeneralMesh](#) ()  
*Destructor.*
- virtual int [numVertices](#) ()=0  
*Number of nodes in mesh.*
- virtual [double](#) \* [vertexArray](#) ()=0  
*Fetch all nodes.*
- virtual [double](#) \* [paramArray](#) ()=0  
*Fetch parameter values corresponding to the nodes.*
- virtual int [atBoundary](#) (int [idx](#))=0  
*Check if a given node lies at the boundary.*
- virtual int [numTriangles](#) ()  
*The number of triangles represented by this mesh.*
- virtual unsigned int \* [triangleIndexArray](#) ()=0  
*Indices of nodes belonging to each triangle.*
- virtual [RegularMesh](#) \* [asRegularMesh](#) ()  
*Casting. Return as regular mesh if possible.*
- virtual [LineStrip](#) \* [asLineStrip](#) ()  
*Casting. Return as line strip if possible.*
- virtual [GenericTriMesh](#) \* [asGenericTriMesh](#) ()  
*Casting. Return as generic tri mesh if possible.*
- void [translate](#) (const std::vector< [double](#) > &vert\_translation)  
*Translate all vertices by vert\_translation, wrt the geometry, discarding any previous translation.*

## Public Attributes

- `std::vector< double > vert_translation_`  
*The 3D-translation wrt the geometry.*

### 29.165.1 Detailed Description

Super class for meshes. Used in transfer of mesh.

Definition at line 58 of file GeneralMesh.h.

### 29.165.2 Constructor & Destructor Documentation

29.165.2.1 `Go::GeneralMesh::GeneralMesh ( )`

29.165.2.2 `virtual Go::GeneralMesh::~~GeneralMesh ( ) [virtual]`

Destructor.

### 29.165.3 Member Function Documentation

29.165.3.1 `virtual GenericTriMesh* Go::GeneralMesh::asGenericTriMesh ( ) [virtual]`

Casting. Return as generic tri mesh if possible.

Reimplemented in [Go::GenericTriMesh](#).

29.165.3.2 `virtual LineStrip* Go::GeneralMesh::asLineStrip ( ) [virtual]`

Casting. Return as line strip if possible.

Reimplemented in [Go::LineStrip](#).

29.165.3.3 `virtual RegularMesh* Go::GeneralMesh::asRegularMesh ( ) [virtual]`

Casting. Return as regular mesh if possible.

Reimplemented in [Go::RegularMesh](#), and [Go::RegularVolMesh](#).

29.165.3.4 `virtual int Go::GeneralMesh::atBoundary ( int idx ) [pure virtual]`

Check if a given node lies at the boundary.

Implemented in [Go::RegularMesh](#), [Go::QuadMesh](#), [Go::RegularVolMesh](#), [Go::GenericTriMesh](#), and [Go::LineStrip](#).

29.165.3.5 `virtual int Go::GeneralMesh::numTriangles ( ) [inline],[virtual]`

The number of triangles represented by this mesh.

Reimplemented in [Go::RegularMesh](#), [Go::RegularVolMesh](#), [Go::GenericTriMesh](#), and [Go::QuadMesh](#).

Definition at line 80 of file `GeneralMesh.h`.

29.165.3.6 `virtual int Go::GeneralMesh::numVertices ( ) [pure virtual]`

Number of nodes in mesh.

Implemented in [Go::RegularMesh](#), [Go::RegularVolMesh](#), [Go::GenericTriMesh](#), [Go::QuadMesh](#), and [Go::LineStrip](#).

29.165.3.7 `virtual double* Go::GeneralMesh::paramArray ( ) [pure virtual]`

Fetch parameter values corresponding to the nodes.

Implemented in [Go::RegularMesh](#), [Go::QuadMesh](#), [Go::RegularVolMesh](#), [Go::GenericTriMesh](#), and [Go::LineStrip](#).

29.165.3.8 `void Go::GeneralMesh::translate ( const std::vector< double > & vert_translation )`

Translate all vertices by `vert_translation`, wrt the geometry, discarding any previous translation.

29.165.3.9 `virtual unsigned int* Go::GeneralMesh::triangleIndexArray ( ) [pure virtual]`

Indices of nodes belonging to each triangle.

Implemented in [Go::RegularMesh](#), [Go::RegularVolMesh](#), [Go::QuadMesh](#), [Go::GenericTriMesh](#), and [Go::LineStrip](#).

29.165.3.10 `virtual double* Go::GeneralMesh::vertexArray ( ) [pure virtual]`

Fetch all nodes.

Implemented in [Go::RegularMesh](#), [Go::RegularVolMesh](#), [Go::QuadMesh](#), [Go::GenericTriMesh](#), and [Go::LineStrip](#).

## 29.165.4 Member Data Documentation

29.165.4.1 `std::vector<double> Go::GeneralMesh::vert_translation_`

The 3D-translation wrt the geometry.

Definition at line 101 of file `GeneralMesh.h`.

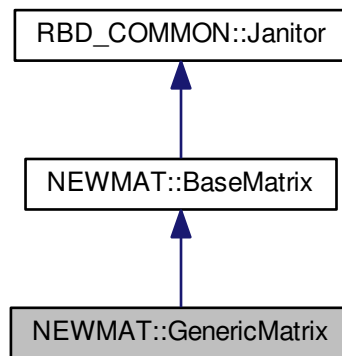
The documentation for this class was generated from the following file:

- `gotools-core/include/GoTools/tesselator/GeneralMesh.h`

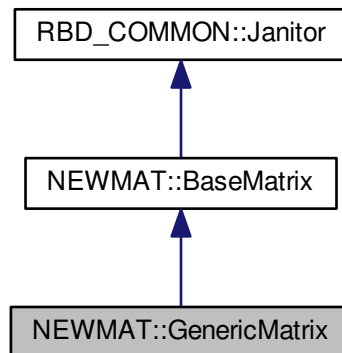
## 29.166 NEWMAT::GenericMatrix Class Reference

```
#include <newmat.h>
```

Inheritance diagram for NEWMAT::GenericMatrix:



Collaboration diagram for NEWMAT::GenericMatrix:



### Public Member Functions

- [GenericMatrix](#) ()
- [GenericMatrix](#) (const [BaseMatrix](#) &bm)
- [GenericMatrix](#) (const [GenericMatrix](#) &bm)
- void [operator=](#) (const [GenericMatrix](#) &)
- void [operator=](#) (const [BaseMatrix](#) &)

- void `operator+=` (`const BaseMatrix &`)
- void `operator-=` (`const BaseMatrix &`)
- void `operator*=` (`const BaseMatrix &`)
- void `operator|=` (`const BaseMatrix &`)
- void `operator&=` (`const BaseMatrix &`)
- void `operator+=` (`Real`)
- void `operator-=` (`Real r`)
- void `operator*=` (`Real`)
- void `operator/=` (`Real r`)
- `~GenericMatrix` ()
- void `CleanUp` ()
- void `Release` ()
- `GeneralMatrix * Evaluate` (`MatrixType=MatrixTypeUnSp`)
- `MatrixBandWidth BandWidth` () `const`

## Friends

- class `BaseMatrix`

## Additional Inherited Members

### 29.166.1 Detailed Description

Definition at line 1141 of file `newmat.h`.

### 29.166.2 Constructor & Destructor Documentation

**29.166.2.1** `NEWMAT::GenericMatrix::GenericMatrix ( )` [`inline`]

Definition at line 1147 of file `newmat.h`.

**29.166.2.2** `NEWMAT::GenericMatrix::GenericMatrix ( const BaseMatrix & bm )` [`inline`]

Definition at line 1148 of file `newmat.h`.

**29.166.2.3** `NEWMAT::GenericMatrix::GenericMatrix ( const GenericMatrix & bm )` [`inline`]

Definition at line 1150 of file `newmat.h`.

**29.166.2.4** `NEWMAT::GenericMatrix::~~GenericMatrix ( )` [`inline`]

Definition at line 1163 of file `newmat.h`.

### 29.166.3 Member Function Documentation

29.166.3.1 `MatrixBandWidth GenericMatrix::BandWidth ( ) const [virtual]`

Reimplemented from [NEWMAT::BaseMatrix](#).

Definition at line 415 of file newmat4.cpp.

29.166.3.2 `void NEWMAT::GenericMatrix::CleanUp ( ) [inline],[virtual]`

Reimplemented from [NEWMAT::BaseMatrix](#).

Definition at line 1164 of file newmat.h.

29.166.3.3 `GeneralMatrix * GenericMatrix::Evaluate ( MatrixType mt = MatrixTypeUnSp ) [virtual]`

Implements [NEWMAT::BaseMatrix](#).

Definition at line 91 of file newmat5.cpp.

29.166.3.4 `void GenericMatrix::operator&= ( const BaseMatrix & X )`

Definition at line 834 of file newmat6.cpp.

29.166.3.5 `void NEWMAT::GenericMatrix::operator*=( const BaseMatrix & )`

29.166.3.6 `void NEWMAT::GenericMatrix::operator*=( Real )`

29.166.3.7 `void NEWMAT::GenericMatrix::operator+=( const BaseMatrix & )`

29.166.3.8 `void NEWMAT::GenericMatrix::operator+=( Real )`

29.166.3.9 `void NEWMAT::GenericMatrix::operator-=( const BaseMatrix & )`

29.166.3.10 `void NEWMAT::GenericMatrix::operator-=( Real r ) [inline]`

Definition at line 1160 of file newmat.h.

29.166.3.11 `void NEWMAT::GenericMatrix::operator/=( Real r ) [inline]`

Definition at line 1162 of file newmat.h.



29.166.3.12 void NEWMAT::GenericMatrix::operator= ( const GenericMatrix & )

29.166.3.13 void NEWMAT::GenericMatrix::operator= ( const BaseMatrix & )

29.166.3.14 void GenericMatrix::operator|= ( const BaseMatrix & X )

Definition at line 812 of file newmat6.cpp.

29.166.3.15 void NEWMAT::GenericMatrix::Release ( ) [inline]

Definition at line 1165 of file newmat.h.

## 29.166.4 Friends And Related Function Documentation

29.166.4.1 friend class BaseMatrix [friend]

Definition at line 1145 of file newmat.h.

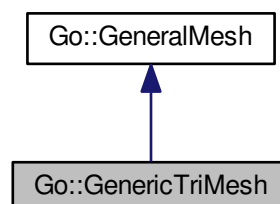
The documentation for this class was generated from the following files:

- [newmat/include/newmat.h](#)
- [newmat/src/newmat4.cpp](#)
- [newmat/src/newmat5.cpp](#)
- [newmat/src/newmat6.cpp](#)

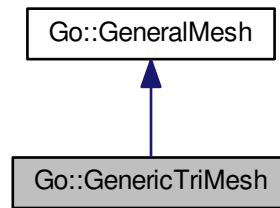
## 29.167 Go::GenericTriMesh Class Reference

```
#include <GenericTriMesh.h>
```

Inheritance diagram for Go::GenericTriMesh:



Collaboration diagram for Go::GenericTriMesh:



## Public Member Functions

- [GenericTriMesh](#) (int num\_vert=0, int num\_tri=0, [bool](#) use\_normals=true, [bool](#) use\_texcoords=false)
- [~GenericTriMesh](#) ()  
*Destructor.*
- [bool useNormals](#) ()  
*Check if normals will be computed.*
- [bool useTexCoords](#) ()  
*Check if texture coordinates will be computed.*
- virtual int [numTriangles](#) ()  
*The total number of triangles.*
- virtual int [numVertices](#) ()  
*Number of nodes in mesh.*
- void [resize](#) (int num\_vert, int num\_tri)  
*Change mesh size.*
- virtual [double \\*](#) [vertexArray](#) ()  
*Fetch all nodes.*
- virtual [double \\*](#) [paramArray](#) ()  
*Fetch parameter values corresponding to the nodes.*
- virtual int [atBoundary](#) (int idx)  
*Check if a given node lies at the boundary.*
- int \* [boundaryArray](#) ()  
*Indices to boundary nodes.*
- [double \\*](#) [normalArray](#) ()  
*Normal information.*
- [double \\*](#) [texcoordArray](#) ()  
*Texture coordinate information.*
- virtual unsigned int \* [triangleIndexArray](#) ()  
*Indices of nodes belonging to each triangle.*
- virtual [GenericTriMesh \\*](#) [asGenericTriMesh](#) ()  
*Casting. Return as generic tri mesh.*

## Additional Inherited Members

### 29.167.1 Detailed Description

Based on [RegularMesh](#), designed to store data for GL visualizing (regular triangles, not triangle strips).

Definition at line 58 of file `GenericTriMesh.h`.

### 29.167.2 Constructor & Destructor Documentation

**29.167.2.1** `Go::GenericTriMesh::GenericTriMesh ( int num_vert = 0, int num_tri = 0, bool use_normals = true, bool use_texcoords = false )`

Constructor. Give mesh size. Set whether or not normals and texture coordinates should be computed

**29.167.2.2** `Go::GenericTriMesh::~~GenericTriMesh ( )`

Destructor.

### 29.167.3 Member Function Documentation

**29.167.3.1** `virtual GenericTriMesh* Go::GenericTriMesh::asGenericTriMesh ( )` `[virtual]`

Casting. Return as generic tri mesh.

Reimplemented from [Go::GeneralMesh](#).

**29.167.3.2** `virtual int Go::GenericTriMesh::atBoundary ( int idx )` `[virtual]`

Check if a given node lies at the boundary.

Implements [Go::GeneralMesh](#).

**29.167.3.3** `int* Go::GenericTriMesh::boundaryArray ( )`

Indices to boundary nodes.

**29.167.3.4** `double* Go::GenericTriMesh::normalArray ( )`

Normal information.

**29.167.3.5** `virtual int Go::GenericTriMesh::numTriangles ( ) [inline],[virtual]`

The total number of triangles.

Reimplemented from [Go::GeneralMesh](#).

Definition at line 79 of file `GenericTriMesh.h`.

**29.167.3.6** `virtual int Go::GenericTriMesh::numVertices ( ) [inline],[virtual]`

Number of nodes in mesh.

Implements [Go::GeneralMesh](#).

Definition at line 82 of file `GenericTriMesh.h`.

**29.167.3.7** `virtual double* Go::GenericTriMesh::paramArray ( ) [virtual]`

Fetch parameter values corresponding to the nodes.

Implements [Go::GeneralMesh](#).

**29.167.3.8** `void Go::GenericTriMesh::resize ( int num_vert, int num_tri )`

Change mesh size.

**29.167.3.9** `double* Go::GenericTriMesh::texcoordArray ( )`

Texture coordinate information.

**29.167.3.10** `virtual unsigned int* Go::GenericTriMesh::triangleIndexArray ( ) [virtual]`

Indices of nodes belonging to each triangle.

Implements [Go::GeneralMesh](#).

**29.167.3.11** `bool Go::GenericTriMesh::useNormals ( ) [inline]`

Check if normals will be computed.

Definition at line 71 of file `GenericTriMesh.h`.

**29.167.3.12** `bool Go::GenericTriMesh::useTexCoords ( ) [inline]`

Check if texture coordinates will be computed.

Definition at line 75 of file `GenericTriMesh.h`.

29.167.3.13 virtual double\* Go::GenericTriMesh::vertexArray ( ) [virtual]

Fetch all nodes.

Implements [Go::GeneralMesh](#).

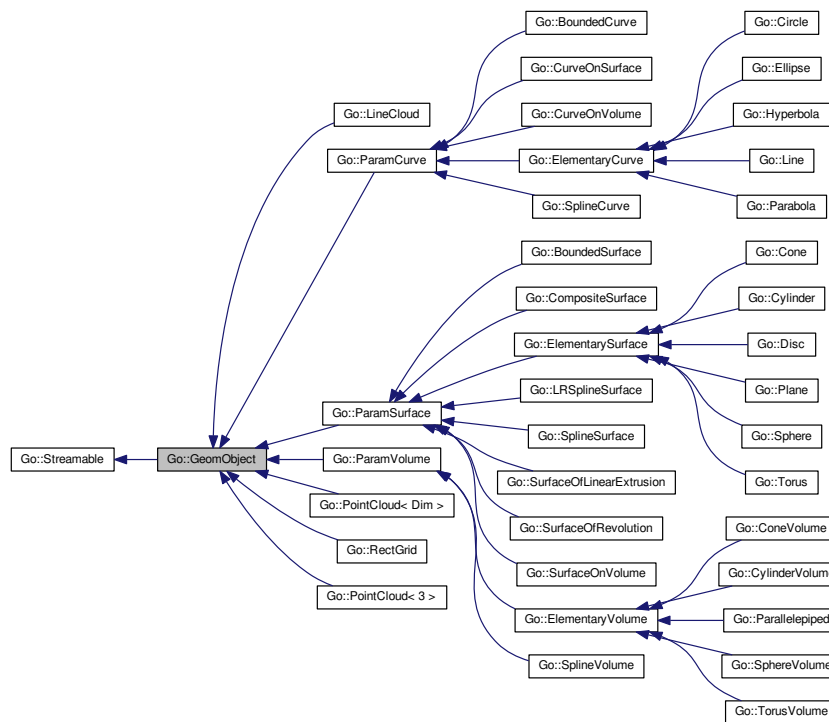
The documentation for this class was generated from the following file:

- [gotools-core/include/GoTools/tesselator/GenericTriMesh.h](#)

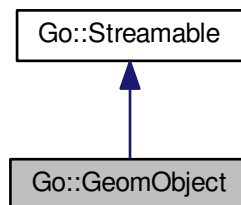
## 29.168 Go::GeomObject Class Reference

```
#include <GeomObject.h>
```

Inheritance diagram for Go::GeomObject:



Collaboration diagram for Go::GeomObject:



## Public Member Functions

- virtual `~GeomObject ()`
- virtual `BoundingBox boundingBox () const =0`  
*Return the object's bounding box.*
- virtual `int dimension () const =0`  
*Return the dimension of the space in which the object lies (usually 2 or 3)*
- virtual `ClassType instanceType () const =0`  
*Return the class type identifier of a given, derived instance of [GeomObject](#).*
- virtual `GeomObject * clone () const =0`  
*Clone the [GeomObject](#) and return a pointer to the clone.*
- void `writeStandardHeader (std::ostream &os) const`

## Static Public Member Functions

- static `ClassType classType ()`  
*Return the class type identifier of a given class derived from [GeomObject](#).*

### 29.168.1 Detailed Description

Base class for geometrical objects (curves, surfaces, etc.) regrouping all the properties that they have in common.

Definition at line 59 of file `GeomObject.h`.

### 29.168.2 Constructor & Destructor Documentation

29.168.2.1 `virtual Go::GeomObject::~~GeomObject ( ) [virtual]`

### 29.168.3 Member Function Documentation

29.168.3.1 `virtual BoundingBox Go::GeomObject::boundingBox ( ) const [pure virtual]`

Return the object's bounding box.

Implemented in [Go::SplineVolume](#), [Go::SplineSurface](#), [Go::LRSplineSurface](#), [Go::SplineCurve](#), [Go::BoundedSurface](#), [Go::CurveOnSurface](#), [Go::SurfaceOnVolume](#), [Go::Disc](#), [Go::Torus](#), [Go::SurfaceOfRevolution](#), [Go::CurveOnVolume](#), [Go::Plane](#), [Go::Cone](#), [Go::Cylinder](#), [Go::Sphere](#), [Go::TorusVolume](#), [Go::ConeVolume](#), [Go::CylinderVolume](#), [Go::SphereVolume](#), [Go::Parallelepiped](#), [Go::PointCloud< Dim >](#), [Go::PointCloud< 3 >](#), [Go::LineCloud](#), [Go::BoundedCurve](#), [Go::CompositeSurface](#), [Go::SurfaceOfLinearExtrusion](#), [Go::Line](#), [Go::Circle](#), [Go::Hyperbola](#), [Go::Parabola](#), [Go::Ellipse](#), and [Go::RectGrid](#).

29.168.3.2 `static ClassType Go::GeomObject::classType ( ) [static]`

Return the class type identifier of a given class derived from [GeomObject](#).

29.168.3.3 `virtual GeomObject* Go::GeomObject::clone ( ) const [pure virtual]`

Clone the [GeomObject](#) and return a pointer to the clone.

Implemented in [Go::SplineVolume](#), [Go::SplineSurface](#), [Go::LRSplineSurface](#), [Go::ParamCurve](#), [Go::SplineCurve](#), [Go::CurveOnSurface](#), [Go::BoundedSurface](#), [Go::SurfaceOnVolume](#), [Go::CurveOnVolume](#), [Go::Disc](#), [Go::LineCloud](#), [Go::PointCloud< Dim >](#), [Go::PointCloud< 3 >](#), [Go::Torus](#), [Go::SurfaceOfRevolution](#), [Go::CompositeSurface](#), [Go::Plane](#), [Go::Cone](#), [Go::Cylinder](#), [Go::Sphere](#), [Go::TorusVolume](#), [Go::BoundedCurve](#), [Go::ConeVolume](#), [Go::CylinderVolume](#), [Go::SphereVolume](#), [Go::Parallelepiped](#), [Go::Line](#), [Go::Circle](#), [Go::Hyperbola](#), [Go::Parabola](#), [Go::SurfaceOfLinearExtrusion](#), [Go::RectGrid](#), [Go::Ellipse](#), [Go::ParamSurface](#), [Go::ElementarySurface](#), [Go::ParamVolume](#), [Go::ElementaryVolume](#), and [Go::ElementaryCurve](#).

29.168.3.4 `virtual int Go::GeomObject::dimension ( ) const [pure virtual]`

Return the dimension of the space in which the object lies (usually 2 or 3)

Implemented in [Go::SplineVolume](#), [Go::SplineSurface](#), [Go::LRSplineSurface](#), [Go::SplineCurve](#), [Go::BoundedSurface](#), [Go::CurveOnSurface](#), [Go::SurfaceOnVolume](#), [Go::CurveOnVolume](#), [Go::Disc](#), [Go::Torus](#), [Go::PointCloud< Dim >](#), [Go::PointCloud< 3 >](#), [Go::SurfaceOfRevolution](#), [Go::LineCloud](#), [Go::BoundedCurve](#), [Go::CompositeSurface](#), [Go::Plane](#), [Go::Cone](#), [Go::Cylinder](#), [Go::Sphere](#), [Go::TorusVolume](#), [Go::Line](#), [Go::ConeVolume](#), [Go::CylinderVolume](#), [Go::SphereVolume](#), [Go::Circle](#), [Go::Hyperbola](#), [Go::Parabola](#), [Go::Parallelepiped](#), [Go::Ellipse](#), [Go::SurfaceOfLinearExtrusion](#), and [Go::RectGrid](#).

29.168.3.5 `virtual ClassType Go::GeomObject::instanceType ( ) const [pure virtual]`

Return the class type identifier of a given, derived instance of [GeomObject](#).

Implemented in [Go::SplineVolume](#), [Go::SplineSurface](#), [Go::LRSplineSurface](#), [Go::SplineCurve](#), [Go::BoundedSurface](#), [Go::CurveOnSurface](#), [Go::SurfaceOnVolume](#), [Go::CurveOnVolume](#), [Go::Disc](#), [Go::PointCloud< Dim >](#), [Go::Torus](#), [Go::PointCloud< 3 >](#), [Go::LineCloud](#), [Go::SurfaceOfRevolution](#), [Go::CompositeSurface](#), [Go::BoundedCurve](#), [Go::Plane](#), [Go::Cone](#), [Go::Cylinder](#), [Go::Sphere](#), [Go::TorusVolume](#), [Go::Line](#), [Go::ConeVolume](#), [Go::CylinderVolume](#), [Go::SphereVolume](#), [Go::Circle](#), [Go::Hyperbola](#), [Go::Parabola](#), [Go::Parallelepiped](#), [Go::Ellipse](#), [Go::SurfaceOfLinearExtrusion](#), and [Go::RectGrid](#).

29.168.3.6 `void Go::GeomObject::writeStandardHeader ( std::ostream & os ) const`

Write header information of the [GeomObject](#) to stream. This typically precedes the act of writing the object itself to a stream, to signal to the receiver what object is streamed.

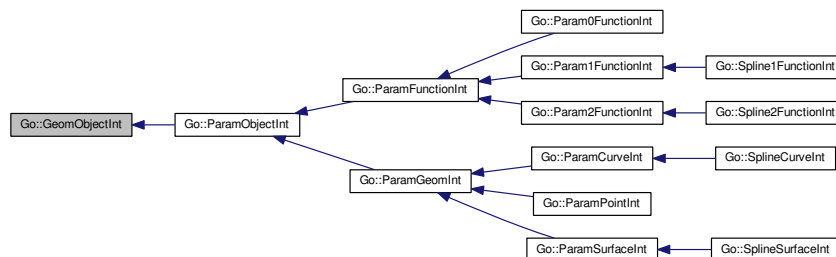
The documentation for this class was generated from the following file:

- [gotools-core/include/GoTools/geometry/GeomObject.h](#)

## 29.169 Go::GeomObjectInt Class Reference

```
#include <GeomObjectInt.h>
```

Inheritance diagram for Go::GeomObjectInt:



### Public Member Functions

- virtual [~GeomObjectInt](#) ()  
*Destructor.*

### 29.169.1 Detailed Description

This class is an abstract base class providing an interface to the "intersection objects". An intersection object is a wrapper around a corresponding geometric object, containing additional functionality, and that is used in the intersection algorithms.

Definition at line 52 of file GeomObjectInt.h.

### 29.169.2 Constructor & Destructor Documentation

**29.169.2.1** virtual [Go::GeomObjectInt::~~GeomObjectInt](#) ( ) [inline], [virtual]

Destructor.

Definition at line 55 of file GeomObjectInt.h.

The documentation for this class was generated from the following file:

- intersections/include/GoTools/intersections/[GeomObjectInt.h](#)

## 29.170 Go::GeoTol Class Reference

Class handling various tolerances.

```
#include <GeoTol.h>
```



## Public Member Functions

- [GeoTol](#) ([double epsg](#), [double rel\\_par\\_res=1.0e-12](#), [double numerical\\_tol=1.0e-16](#))  
*Constructor.*
- [GeoTol](#) ([GeoTol \\*epsg](#))  
*Constructor.*
- [~GeoTol](#) ()  
*Destructor.*
- [const double & getEpsg](#) () [const](#)  
*Get constant tolerance.*
- [const double & getMinEpsg](#) () [const](#)
- [const double & getRelParRes](#) () [const](#)  
*Get relative parameter resolution.*
- [const double & getMaxEpsg](#) () [const](#)
- [const double & getEpsBracket](#) () [const](#)
- [const double & getRefAng](#) () [const](#)
- [const double & getAngleTol](#) () [const](#)  
*Get angular tolerance.*
- [const double & getNumericalTol](#) () [const](#)  
*Get numerical tolerance.*
- [void write](#) ([std::ostream &os](#)) [const](#)  
*Write to output stream.*
- [void read](#) ([std::istream &is](#))  
*Read from input stream.*

### 29.170.1 Detailed Description

Class handling various tolerances.

Definition at line 62 of file GeoTol.h.

### 29.170.2 Constructor & Destructor Documentation

#### 29.170.2.1 [Go::GeoTol::GeoTol](#) ( [double epsg](#), [double rel\\_par\\_res = 1.0e-12](#), [double numerical\\_tol = 1.0e-16](#) )

Constructor.

#### 29.170.2.2 [Go::GeoTol::GeoTol](#) ( [GeoTol \\*epsg](#) )

Constructor.

#### 29.170.2.3 [Go::GeoTol::~~GeoTol](#) ( )

Destructor.

### 29.170.3 Member Function Documentation

29.170.3.1 `const double& Go::GeoTol::getAngleTol ( ) const` [inline]

Get angular tolerance.

Definition at line 98 of file GeoTol.h.

29.170.3.2 `const double& Go::GeoTol::getEpsBracket ( ) const` [inline]

Get "bracket" tolerance used to specify confidence intervals in the parameter domain

Definition at line 91 of file GeoTol.h.

29.170.3.3 `const double& Go::GeoTol::getEpsge ( ) const` [inline]

Get constant tolerance.

Definition at line 75 of file GeoTol.h.

29.170.3.4 `const double& Go::GeoTol::getMaxEpsge ( ) const` [inline]

Definition at line 86 of file GeoTol.h.

29.170.3.5 `const double& Go::GeoTol::getMinEpsge ( ) const` [inline]

Definition at line 78 of file GeoTol.h.

29.170.3.6 `const double& Go::GeoTol::getNumericalTol ( ) const` [inline]

Get numerical tolerance.

Definition at line 102 of file GeoTol.h.

29.170.3.7 `const double& Go::GeoTol::getRefAng ( ) const` [inline]

Definition at line 94 of file GeoTol.h.

29.170.3.8 `const double& Go::GeoTol::getRelParRes ( ) const` [inline]

Get relative parameter resolution.

Definition at line 83 of file GeoTol.h.

29.170.3.9 void Go::GeoTol::read ( std::istream & *is* )

Read from input stream.

29.170.3.10 void Go::GeoTol::write ( std::ostream & *os* ) const

Write to output stream.

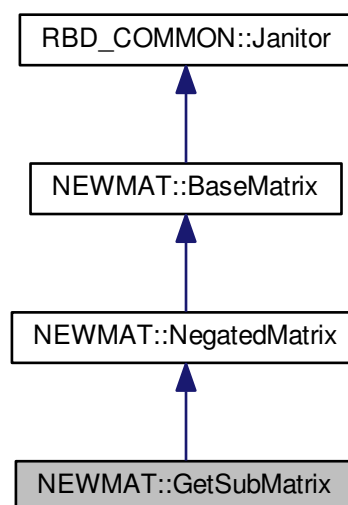
The documentation for this class was generated from the following file:

- [intersections/include/GoTools/intersections/GeoTol.h](#)

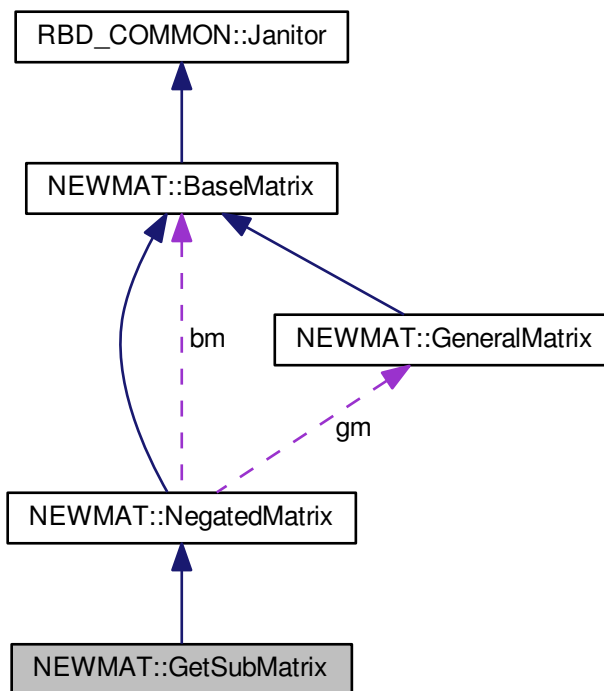
## 29.171 NEWMAT::GetSubMatrix Class Reference

```
#include <newmat.h>
```

Inheritance diagram for NEWMAT::GetSubMatrix:



Collaboration diagram for NEWMAT::GetSubMatrix:



## Public Member Functions

- [GetSubMatrix](#) ([const GetSubMatrix &g](#))
- [~GetSubMatrix](#) ()
- [GeneralMatrix \\* Evaluate](#) ([MatrixType mt=MatrixTypeUnSp](#))
- void [operator=](#) ([const BaseMatrix &](#))
- void [operator+=](#) ([const BaseMatrix &](#))
- void [operator-=](#) ([const BaseMatrix &](#))
- void [operator=](#) ([const GetSubMatrix &m](#))
- void [operator<<](#) ([const BaseMatrix &](#))
- void [operator<<](#) ([const Real \\*](#))
- [MatrixInput operator<<](#) ([Real](#))
- [MatrixInput operator<<](#) ([int f](#))
- void [operator=](#) ([Real](#))
- void [operator+=](#) ([Real](#))
- void [operator-=](#) ([Real r](#))
- void [operator\\*=\[Real\]\(#\)](#)
- void [operator/=](#) ([Real r](#))
- void [Inject](#) ([const GeneralMatrix &](#))
- [MatrixBandWidth BandWidth](#) () [const](#)

## Friends

- class [BaseMatrix](#)

## Additional Inherited Members

### 29.171.1 Detailed Description

Definition at line 1488 of file newmat.h.

### 29.171.2 Constructor & Destructor Documentation

#### 29.171.2.1 NEWMAT::GetSubMatrix::GetSubMatrix ( const GetSubMatrix & g ) [inline]

Definition at line 1503 of file newmat.h.

#### 29.171.2.2 NEWMAT::GetSubMatrix::~GetSubMatrix ( ) [inline]

Definition at line 1506 of file newmat.h.

### 29.171.3 Member Function Documentation

#### 29.171.3.1 MatrixBandWidth GetSubMatrix::BandWidth ( ) const [virtual]

Reimplemented from [NEWMAT::NegatedMatrix](#).

Definition at line 488 of file newmat4.cpp.

#### 29.171.3.2 GeneralMatrix \* GetSubMatrix::Evaluate ( MatrixType mt = MatrixTypeUnSp ) [virtual]

Reimplemented from [NEWMAT::NegatedMatrix](#).

Definition at line 341 of file newmat5.cpp.

#### 29.171.3.3 void GetSubMatrix::Inject ( const GeneralMatrix & gmx )

Definition at line 272 of file submat.cpp.

#### 29.171.3.4 void GetSubMatrix::operator\*= ( Real r )

Definition at line 396 of file submat.cpp.

29.171.3.5 void NEWMAT::GetSubMatrix::operator+=( const BaseMatrix & )

29.171.3.6 void NEWMAT::GetSubMatrix::operator+=( Real )

29.171.3.7 void NEWMAT::GetSubMatrix::operator-= ( const BaseMatrix & )

29.171.3.8 void NEWMAT::GetSubMatrix::operator-= ( Real *r* ) [inline]

Definition at line 1518 of file newmat.h.

29.171.3.9 void NEWMAT::GetSubMatrix::operator/= ( Real *r* ) [inline]

Definition at line 1520 of file newmat.h.

29.171.3.10 void NEWMAT::GetSubMatrix::operator<< ( const BaseMatrix & )

29.171.3.11 void NEWMAT::GetSubMatrix::operator<< ( const Real \* )

29.171.3.12 MatrixInput NEWMAT::GetSubMatrix::operator<< ( Real )

29.171.3.13 MatrixInput NEWMAT::GetSubMatrix::operator<< ( int *f* ) [inline]

Definition at line 1786 of file newmat.h.

29.171.3.14 void NEWMAT::GetSubMatrix::operator= ( const BaseMatrix & )

29.171.3.15 void NEWMAT::GetSubMatrix::operator= ( const GetSubMatrix & *m* ) [inline]

Definition at line 1511 of file newmat.h.

29.171.3.16 void NEWMAT::GetSubMatrix::operator= ( Real )

## 29.171.4 Friends And Related Function Documentation

29.171.4.1 friend class BaseMatrix [friend]

Definition at line 1501 of file newmat.h.

The documentation for this class was generated from the following files:

- [newmat/include/newmat.h](#)
- [newmat/src/newmat4.cpp](#)
- [newmat/src/newmat5.cpp](#)
- [newmat/src/submat.cpp](#)

## 29.172 Go::go\_iterator\_traits< Iterator > Struct Template Reference

```
#include <Utils.h>
```

### Public Types

- typedef Iterator::value\_type [value\\_type](#)
- typedef Iterator::difference\_type [difference\\_type](#)
- typedef Iterator::pointer [pointer](#)
- typedef Iterator::reference [reference](#)

### 29.172.1 Detailed Description

```
template<class Iterator>
struct Go::go_iterator_traits< Iterator >
```

Iterator traits classes are provided for the [Go](#) namespace, so that we won't need to include

```
<iterator>
```

or similar headers.

Definition at line 59 of file [Utils.h](#).

### 29.172.2 Member Typedef Documentation

**29.172.2.1** `template<class Iterator> typedef Iterator::difference_type Go::go_iterator_traits< Iterator >::difference_type`

Definition at line 61 of file [Utils.h](#).

**29.172.2.2** `template<class Iterator> typedef Iterator::pointer Go::go_iterator_traits< Iterator >::pointer`

Definition at line 62 of file [Utils.h](#).

**29.172.2.3** `template<class Iterator> typedef Iterator::reference Go::go_iterator_traits< Iterator >::reference`

Definition at line 63 of file [Utils.h](#).

**29.172.2.4** `template<class Iterator> typedef Iterator::value_type Go::go_iterator_traits< Iterator >::value_type`

Definition at line 60 of file [Utils.h](#).

The documentation for this struct was generated from the following file:

- [gotools-core/include/GoTools/geometry/Utils.h](#)

## 29.173 Go::go\_iterator\_traits< const T \* > Struct Template Reference

```
#include <Utils.h>
```

### Public Types

- typedef [T](#) [value\\_type](#)
- typedef int [difference\\_type](#)
- typedef [const T \\*](#) [pointer](#)
- typedef [const T &](#) [reference](#)

### 29.173.1 Detailed Description

```
template<class T>
struct Go::go_iterator_traits< const T * >
```

Definition at line 76 of file [Utils.h](#).

### 29.173.2 Member Typedef Documentation

29.173.2.1 `template<class T > typedef int Go::go_iterator_traits< const T * >::difference_type`

Definition at line 78 of file [Utils.h](#).

29.173.2.2 `template<class T > typedef const T* Go::go_iterator_traits< const T * >::pointer`

Definition at line 79 of file [Utils.h](#).

29.173.2.3 `template<class T > typedef const T& Go::go_iterator_traits< const T * >::reference`

Definition at line 80 of file [Utils.h](#).

29.173.2.4 `template<class T > typedef T Go::go_iterator_traits< const T * >::value_type`

Definition at line 77 of file [Utils.h](#).

The documentation for this struct was generated from the following file:

- [gtools-core/include/GoTools/geometry/Utils.h](#)

## 29.174 Go::go\_iterator\_traits< T \* > Struct Template Reference

```
#include <Utils.h>
```



## Public Types

- typedef [T](#) [value\\_type](#)
- typedef int [difference\\_type](#)
- typedef [T](#) \* [pointer](#)
- typedef [T](#) & [reference](#)

### 29.174.1 Detailed Description

```
template<class T>
struct Go::go_iterator_traits< T * >
```

Definition at line 67 of file [Utils.h](#).

### 29.174.2 Member Typedef Documentation

29.174.2.1 `template<class T > typedef int Go::go_iterator_traits< T * >::difference_type`

Definition at line 69 of file [Utils.h](#).

29.174.2.2 `template<class T > typedef T* Go::go_iterator_traits< T * >::pointer`

Definition at line 70 of file [Utils.h](#).

29.174.2.3 `template<class T > typedef T& Go::go_iterator_traits< T * >::reference`

Definition at line 71 of file [Utils.h](#).

29.174.2.4 `template<class T > typedef T Go::go_iterator_traits< T * >::value_type`

Definition at line 68 of file [Utils.h](#).

The documentation for this struct was generated from the following file:

- [gotools-core/include/GoTools/geometry/Utils.h](#)

## 29.175 Go::GoTools Class Reference

Class containing some service functions for the [GoTools](#) "system".

```
#include <GoTools.h>
```

## Public Member Functions

- [GoTools](#) ()
- virtual [~GoTools](#) ()

## Static Public Member Functions

- static void [init](#) ()
- static std::string [className](#) ([ClassType](#) class\_type)
- static [double](#) [spaceEpsilon](#) ()  
*Get geometry tolerance.*
- static void [setSpaceEpsilon](#) ([double](#) space\_epsilon)  
*Set geometry tolerance.*
- static [double](#) [parameterEpsilon](#) ()  
*Get parameter tolerance.*
- static void [setParameterEpsilon](#) ([double](#) parameter\_epsilon)  
*Set parameter tolerance.*
- static [double](#) [knotEpsilon](#) ()  
*Get knot tolerance.*
- static void [setKnotEpsilon](#) ([double](#) knot\_epsilon)  
*Set knot tolerance.*

### 29.175.1 Detailed Description

Class containing some service functions for the [GoTools](#) "system".

Definition at line 76 of file [GoTools.h](#).

### 29.175.2 Constructor & Destructor Documentation

29.175.2.1 [Go::GoTools::GoTools](#) ( )

29.175.2.2 virtual [Go::GoTools::~~GoTools](#) ( ) [[virtual](#)]

### 29.175.3 Member Function Documentation

29.175.3.1 static std::string [Go::GoTools::className](#) ( [ClassType](#) class\_type ) [[static](#)]

Conversion function that returns a std::string with the class name corresponding to [ClassType](#) class\_type.

29.175.3.2 static void [Go::GoTools::init](#) ( ) [[static](#)]

This functions sets up the factories for constructing [GeomObjects](#) at run-time (i.e. [Registrar](#) instantiations). If you need to construct [GeomObjects](#) at run-time, such as when reading from a file, then call this function early in the code.

29.175.3.3 `static double Go::GoTools::knotEpsilon ( ) [static]`

Get knot tolerance.

29.175.3.4 `static double Go::GoTools::parameterEpsilon ( ) [static]`

Get parameter tolerance.

29.175.3.5 `static void Go::GoTools::setKnotEpsilon ( double knot_epsilon ) [static]`

Set knot tolerance.

29.175.3.6 `static void Go::GoTools::setParameterEpsilon ( double parameter_epsilon ) [static]`

Set parameter tolerance.

29.175.3.7 `static void Go::GoTools::setSpaceEpsilon ( double space_epsilon ) [static]`

Set geometry tolerance.

29.175.3.8 `static double Go::GoTools::spaceEpsilon ( ) [static]`

Get geometry tolerance.

The documentation for this class was generated from the following file:

- [gotools-core/include/GoTools/geometry/GoTools.h](#)

## 29.176 Go::GPos Struct Reference

```
#include <Mesh2D.h>
```

### Public Member Functions

- [GPos](#) (int i, int m)
- [GPos](#) ()

### Public Attributes

- int [ix](#)
- int [mult](#)

### 29.176.1 Detailed Description

Definition at line 56 of file Mesh2D.h.

### 29.176.2 Constructor & Destructor Documentation

#### 29.176.2.1 `Go::GPos::GPos ( int i, int m ) [inline]`

Definition at line 59 of file Mesh2D.h.

#### 29.176.2.2 `Go::GPos::GPos ( ) [inline]`

Definition at line 60 of file Mesh2D.h.

### 29.176.3 Member Data Documentation

#### 29.176.3.1 `int Go::GPos::ix`

Definition at line 62 of file Mesh2D.h.

#### 29.176.3.2 `int Go::GPos::mult`

Definition at line 63 of file Mesh2D.h.

The documentation for this struct was generated from the following file:

- [Irsplines2D/include/GoTools/Irsplines2D/Mesh2D.h](#)

## 29.177 `Go::GpuMatrix< T >` Class Template Reference

```
#include <GpuMatrix.hpp>
```

### Public Member Functions

- [GpuMatrix](#) (int rows, int cols)
- [GpuMatrix](#) (int rows, int cols, [T](#) \*data)
- [GpuMatrix](#) (int rows, int cols, [T](#) initializer)
- [~GpuMatrix](#) ()
- void [upload](#) ([T](#) \*data)
- void [download](#) ([T](#) \*data)
- int [getRows](#) ()
- int [getCols](#) ()

## Friends

- `template<typename U >`  
`std::ostream & operator<< (std::ostream &os, const GpuMatrix< U > &m)`

### 29.177.1 Detailed Description

```
template<typename T>
class Go::GpuMatrix< T >
```

Definition at line 49 of file GpuMatrix.hpp.

### 29.177.2 Constructor & Destructor Documentation

29.177.2.1 `template<typename T > Go::GpuMatrix< T >::GpuMatrix ( int rows, int cols )`

Constructs a matrix on the GPU

29.177.2.2 `template<typename T > Go::GpuMatrix< T >::GpuMatrix ( int rows, int cols, T * data )`

Constructs a matrix on the GPU using data as initial values

29.177.2.3 `template<typename T > Go::GpuMatrix< T >::GpuMatrix ( int rows, int cols, T initializer )`

Constructs a matrix on the GPU setting all values to initializer.

29.177.2.4 `template<typename T > Go::GpuMatrix< T >::~~GpuMatrix ( )`

Deletes data allocated on the gpu.

### 29.177.3 Member Function Documentation

29.177.3.1 `template<typename T > void Go::GpuMatrix< T >::download ( T * data )`

Downloads data from gpu memory to cpu memory. Using "pinned memory" will give best performance.

29.177.3.2 `template<typename T > int Go::GpuMatrix< T >::getCols ( )`

29.177.3.3 `template<typename T > int Go::GpuMatrix< T >::getRows ( )`

29.177.3.4 `template<typename T > void Go::GpuMatrix< T >::upload ( T * data )`

Uploads data from cpu memory to gpu memory. Using "pinned memory" will give best performance.

### 29.177.4 Friends And Related Function Documentation

29.177.4.1 `template<typename T > template<typename U > std::ostream& operator<< ( std::ostream & os, const GpuMatrix< U > & m ) [friend]`

Prints out the contents of the matrix m.

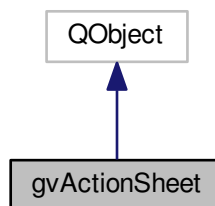
The documentation for this class was generated from the following file:

- [implicitization/include/GoTools/implicitization/GpuMatrix.hpp](#)

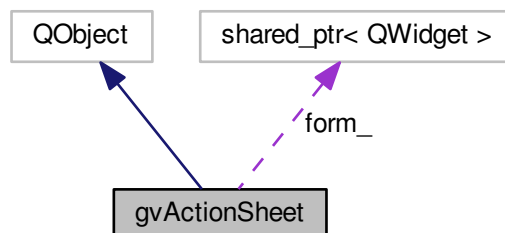
### 29.178 gvActionSheet Class Reference

```
#include <gvActionSheet.h>
```

Inheritance diagram for gvActionSheet:



Collaboration diagram for gvActionSheet:



#### Public Member Functions

- virtual `~gvActionSheet ()`
- virtual void `showDialog (gvObserver *observer)=0`

## Protected Attributes

- `shared_ptr< QWidget > form_`

### 29.178.1 Detailed Description

[gvActionSheet](#):

Definition at line 52 of file `gvActionSheet.h`.

### 29.178.2 Constructor & Destructor Documentation

29.178.2.1 `virtual gvActionSheet::~gvActionSheet ( )` `[inline]`, `[virtual]`

Definition at line 58 of file `gvActionSheet.h`.

### 29.178.3 Member Function Documentation

29.178.3.1 `virtual void gvActionSheet::showDialog ( gvObserver * observer )` `[pure virtual]`

### 29.178.4 Member Data Documentation

29.178.4.1 `shared_ptr<QWidget> gvActionSheet::form_` `[protected]`

Definition at line 63 of file `gvActionSheet.h`.

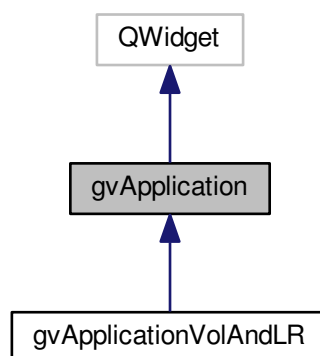
The documentation for this class was generated from the following file:

- `viewlib/include/GoTools/viewlib/gvActionSheet.h`

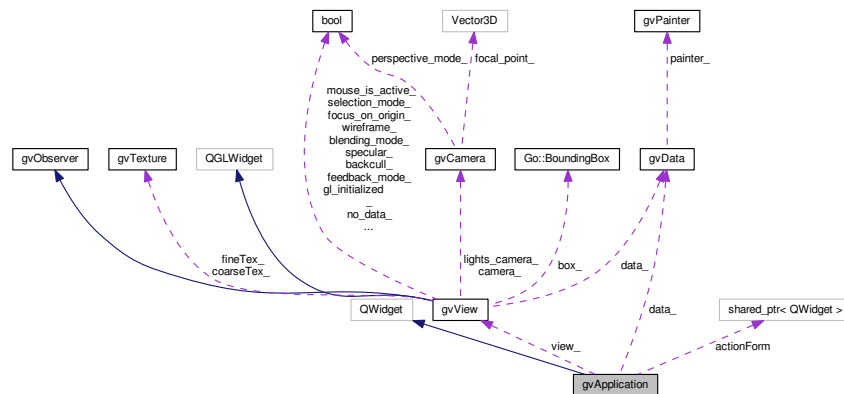
## 29.179 gvApplication Class Reference

```
#include <gvApplication.h>
```

Inheritance diagram for `gvApplication`:



Collaboration diagram for gvApplication:



## Public Slots

- void [open](#) ()
- void [reload\\_last\\_opened\\_file](#) ()
- void [save\\_selection\\_as](#) ()
- void [close\\_document](#) ()
- void [quit](#) ()
- void [about](#) ()
- virtual void [view\\_reset](#) ()
- void [view\\_reset\\_visible](#) ()
- void [view\\_wireframe](#) ()
- void [view\\_axis](#) ()
- void [view\\_cull](#) ()
- void [view\\_specular](#) ()
- void [view\\_orthographic](#) ()
- void [toggle\\_blending\\_mode](#) ()
- void [view\\_focus\\_point](#) ()
- void [view\\_focus\\_point\\_cb](#) (int x, int y)
- void [display\\_object\\_properties](#) ()
- void [assign\\_texture](#) ()
- void [set\\_curve\\_resolutions](#) ()
- virtual void [set\\_surface\\_resolutions](#) ()
- void [enable\\_objects](#) ()
- void [disable\\_objects](#) ()
- void [toggle\\_enable](#) ()
- void [select\\_all](#) ()
- void [select\\_none](#) ()
- void [select\\_inverse](#) ()
- void [select\\_all\\_surfaces](#) ()
- void [select\\_all\\_curves](#) ()
- void [select\\_all\\_visible](#) ()
- void [toggle\\_selection\\_mode](#) ()
- void [toggle\\_select\\_object](#) (unsigned int name)
- void [toggle\\_multiselect\\_object](#) (unsigned int \*names, int numnames)
- void [group\\_selected](#) ()
  - *Put selected objects in a group (user required to name group).*
- virtual void [dismiss\\_selections](#) ()
- void [show\\_control\\_nets](#) ()
- void [set\\_random\\_color](#) ()



## Public Member Functions

- [gvApplication](#) (std::auto\_ptr< [DataHandler](#) > dh, QWidget \*parent=0, const char \*name=0, Qt::WFlags f=0)  
*Constructor.*
- virtual [~gvApplication](#) ()  
*The destructor.*
- virtual QSize [sizeHint](#) () const

## Protected Slots

- void [changeCurveResolutions](#) (int new\_res)
- virtual void [changeSurfaceResolutions](#) (int new\_u\_res, int new\_v\_res)
- virtual void [add\\_group](#) (std::vector< int > &members, QString name)
- void [add\\_objects](#) (ObjContainer &new\_objs, ColContainer &new\_colors)

## Protected Member Functions

- void [buildGUI](#) ()
- Q3ButtonGroup \* [createObjectToggleBox](#) ()
- void [getSelectedObjects](#) (std::vector< shared\_ptr< [Go::GeomObject](#) > > &sel\_objs, std::vector< shared\_ptr< [Go::GeomObject](#) > > &not\_sel\_objs)

## Protected Attributes

- [gvData](#) data\_
- [gvView](#) \* view\_
- QString [curr\\_file\\_type\\_](#)
- QString [last\\_file\\_name\\_](#)
- QString [app\\_name\\_](#)
- QMenuBar \* [menu\\_](#)
- QMenu \* [view\\_menu\\_](#)
- QMenu \* [select\\_menu\\_](#)
- QMenu \* [group\\_menu\\_](#)
- QMenu \* [object\\_menu\\_](#)
- shared\_ptr< QWidget > [actionForm](#)

### 29.179.1 Detailed Description

Definition at line 72 of file gvApplication.h.

### 29.179.2 Constructor & Destructor Documentation

29.179.2.1 [gvApplication::gvApplication](#) ( std::auto\_ptr< [DataHandler](#) > dh, QWidget \* parent = 0, const char \* name = 0, Qt::WFlags f = 0 )

Constructor.

29.179.2.2 `virtual gvApplication::~~gvApplication ( ) [virtual]`

The destructor.

### 29.179.3 Member Function Documentation

29.179.3.1 `void gvApplication::about ( ) [slot]`

29.179.3.2 `virtual void gvApplication::add_group ( std::vector< int > & members, QString name ) [protected], [virtual], [slot]`

29.179.3.3 `void gvApplication::add_objects ( ObjContainer & new_objs, ColContainer & new_colors ) [protected], [slot]`

29.179.3.4 `void gvApplication::assign_texture ( ) [slot]`

29.179.3.5 `void gvApplication::buildGUI ( ) [protected]`

29.179.3.6 `void gvApplication::changeCurveResolutions ( int new_res ) [protected], [slot]`

29.179.3.7 `virtual void gvApplication::changeSurfaceResolutions ( int new_u_res, int new_v_res ) [protected], [virtual], [slot]`

29.179.3.8 `void gvApplication::close_document ( ) [slot]`

29.179.3.9 `Q3ButtonGroup* gvApplication::createObjectToggleBox ( ) [protected]`

29.179.3.10 `void gvApplication::disable_objects ( ) [slot]`

29.179.3.11 `virtual void gvApplication::dismiss_selections ( ) [virtual], [slot]`

29.179.3.12 `void gvApplication::display_object_properties ( ) [slot]`

29.179.3.13 `void gvApplication::enable_objects ( ) [slot]`

29.179.3.14 `void gvApplication::getSelectedObjects ( std::vector< shared_ptr< Go::GeomObject > > & sel_objs, std::vector< shared_ptr< Go::GeomObject > > & not_sel_objs ) [protected]`

29.179.3.15 `void gvApplication::group_selected ( ) [slot]`

Put selected objects in a group (user required to name group).

- 29.179.3.16 void gvApplication::open ( ) [slot]
- 29.179.3.17 void gvApplication::quit ( ) [slot]
- 29.179.3.18 void gvApplication::reload\_last\_opened\_file ( ) [slot]
- 29.179.3.19 void gvApplication::save\_selection\_as ( ) [slot]
- 29.179.3.20 void gvApplication::select\_all ( ) [slot]
- 29.179.3.21 void gvApplication::select\_all\_curves ( ) [slot]
- 29.179.3.22 void gvApplication::select\_all\_surfaces ( ) [slot]
- 29.179.3.23 void gvApplication::select\_all\_visible ( ) [slot]
- 29.179.3.24 void gvApplication::select\_inverse ( ) [slot]
- 29.179.3.25 void gvApplication::select\_none ( ) [slot]
- 29.179.3.26 void gvApplication::set\_curve\_resolutions ( ) [slot]
- 29.179.3.27 void gvApplication::set\_random\_color ( ) [slot]
- 29.179.3.28 virtual void gvApplication::set\_surface\_resolutions ( ) [virtual],[slot]
- 29.179.3.29 void gvApplication::show\_control\_nets ( ) [slot]
- 29.179.3.30 virtual QSize gvApplication::sizeHint ( ) const [virtual]
- 29.179.3.31 void gvApplication::toggle\_blending\_mode ( ) [slot]
- 29.179.3.32 void gvApplication::toggle\_enable ( ) [slot]
- 29.179.3.33 void gvApplication::toggle\_multiselect\_object ( unsigned int \* *names*, int *numnames* ) [slot]
- 29.179.3.34 void gvApplication::toggle\_select\_object ( unsigned int *name* ) [slot]
- 29.179.3.35 void gvApplication::toggle\_selection\_mode ( ) [slot]
- 29.179.3.36 void gvApplication::view\_axis ( ) [slot]
- 29.179.3.37 void gvApplication::view\_cull ( ) [slot]
- 29.179.3.38 void gvApplication::view\_focus\_point ( ) [slot]

29.179.3.39 void gvApplication::view\_focus\_point\_cb ( int x, int y ) [slot]

29.179.3.40 void gvApplication::view\_orthographic ( ) [slot]

29.179.3.41 virtual void gvApplication::view\_reset ( ) [virtual],[slot]

29.179.3.42 void gvApplication::view\_reset\_visible ( ) [slot]

29.179.3.43 void gvApplication::view\_specular ( ) [slot]

29.179.3.44 void gvApplication::view\_wireframe ( ) [slot]

#### 29.179.4 Member Data Documentation

29.179.4.1 shared\_ptr<QWidget> gvApplication::actionForm [protected]

Definition at line 164 of file gvApplication.h.

29.179.4.2 QString gvApplication::app\_name\_ [protected]

Definition at line 153 of file gvApplication.h.

29.179.4.3 QString gvApplication::curr\_file\_type\_ [protected]

Definition at line 151 of file gvApplication.h.

29.179.4.4 gvData gvApplication::data\_ [protected]

Definition at line 149 of file gvApplication.h.

29.179.4.5 QMenu\* gvApplication::group\_menu\_ [protected]

Definition at line 159 of file gvApplication.h.

29.179.4.6 QString gvApplication::last\_file\_name\_ [protected]

Definition at line 152 of file gvApplication.h.

29.179.4.7 QMenuBar\* gvApplication::menu\_ [protected]

Definition at line 155 of file gvApplication.h.

29.179.4.8 QMenu\* gvApplication::object\_menu\_ [protected]

Definition at line 160 of file gvApplication.h.

29.179.4.9 QMenu\* gvApplication::select\_menu\_ [protected]

Definition at line 158 of file gvApplication.h.

29.179.4.10 gvView\* gvApplication::view\_ [protected]

Definition at line 150 of file gvApplication.h.

29.179.4.11 QMenu\* gvApplication::view\_menu\_ [protected]

Definition at line 157 of file gvApplication.h.

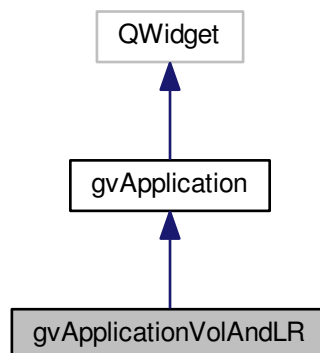
The documentation for this class was generated from the following file:

- [viewlib/include/GoTools/viewlib/gvApplication.h](#)

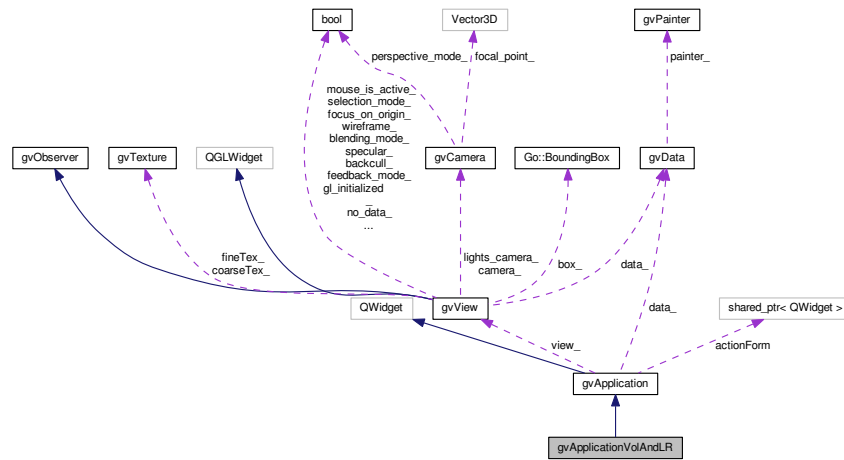
## 29.180 gvApplicationVolAndLR Class Reference

```
#include <gvApplicationVolAndLR.h>
```

Inheritance diagram for gvApplicationVolAndLR:



Collaboration diagram for gvApplicationVolAndLR:



## Public Slots

- virtual void [view\\_reset](#) ()
- void [translate\\_to\\_origin](#) ()
- void [move\\_vertices\\_to\\_origin](#) ()

## Public Member Functions

- [gvApplicationVolAndLR](#) (std::auto\_ptr< [DataHandler](#) > dh, QWidget \*parent=0, const char \*name=0, Qt::WFlags f=0)
- virtual [~gvApplicationVolAndLR](#) ()

## Protected Member Functions

- void [buildExtraGUI](#) ()

## Additional Inherited Members

### 29.180.1 Detailed Description

Definition at line 48 of file gvApplicationVolAndLR.h.

## 29.180.2 Constructor & Destructor Documentation

29.180.2.1 `gvApplicationVolAndLR::gvApplicationVolAndLR ( std::auto_ptr< DataHandler > dh, QWidget * parent = 0, const char * name = 0, Qt::WFlags f = 0 )`

29.180.2.2 `virtual gvApplicationVolAndLR::~gvApplicationVolAndLR ( ) [virtual]`

## 29.180.3 Member Function Documentation

29.180.3.1 `void gvApplicationVolAndLR::buildExtraGUI ( ) [protected]`

29.180.3.2 `void gvApplicationVolAndLR::move_vertices_to_origin ( ) [slot]`

29.180.3.3 `void gvApplicationVolAndLR::translate_to_origin ( ) [slot]`

29.180.3.4 `virtual void gvApplicationVolAndLR::view_reset ( ) [virtual],[slot]`

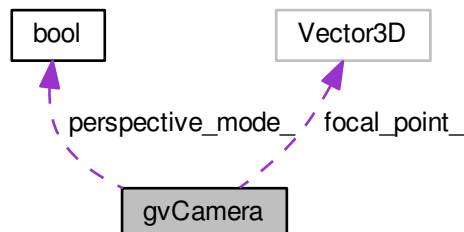
The documentation for this class was generated from the following file:

- [viewlib/include/GoTools/viewlib/vol\\_and\\_lr/gvApplicationVolAndLR.h](#)

## 29.181 gvCamera Class Reference

```
#include <gvCamera.h>
```

Collaboration diagram for gvCamera:



## Public Member Functions

- [gvCamera](#) ([const](#) Vector3D &fpoint, int x, int y, int w, int h)
- [~gvCamera](#) ()
- void [initializeGL](#) ()
- void [use](#) () [const](#)
- void [useModelView](#) () [const](#)  
*Sets up the modelview only.*
- void [pick](#) (int x, int y, int w, int h) [const](#)  
*Sets up the projection, viewport and modelview, for picking.*
- void [setFocalPoint](#) ([const](#) Vector3D &fpoint)
- void [getFocalPoint](#) (Vector3D &fpoint) [const](#)  
*Gets the current focal point.*
- void [setDistance](#) ([double](#) d)
- void [getDistance](#) ([double](#) &d) [const](#)  
*Gets the current distance.*
- void [setAxisSize](#) ([double](#) size)
- void [getAxisSize](#) ([double](#) &size) [const](#)  
*Gets the current axissize.*
- Vector3D [getWorldRay](#) () [const](#)
- Vector3D [getWorldUp](#) () [const](#)
- Vector3D [getWorldSide](#) () [const](#)
- void [rotate](#) ([double](#) axisx, [double](#) axisy, [double](#) angle)
- void [rotateTransversal](#) ([double](#) angle)
- void [resetRotation](#) ()
- void [moveFocalPointRelative](#) ([const](#) Vector3D &eyeptrel)  
*Moves the focal point relative to screen coordinates.*
- Vector3D [eyeCoordsToObjectCoords](#) ([const](#) Vector3D &eyept) [const](#)
- Vector3D [objectCoordsToWindowCoords](#) ([const](#) Vector3D &pt) [const](#)
- void [setViewPort](#) (int x, int y, int w, int h)
- void [getViewPort](#) (int &x, int &y, int &w, int &h) [const](#)  
*Gets the viewport.*
- void [setPerspectiveMode](#) ([bool](#) perspective\_mode)
- [bool](#) [getPerspectiveMode](#) () [const](#)  
*Gets the current perspective mode.*

## Protected Member Functions

- void [computeWinz](#) ()

## Protected Attributes

- Vector3D [focal\\_point\\_](#)
- [double](#) [distance\\_](#)
- [double](#) [mv\\_](#) [16]  
*mv\_ is the rotation matrix*
- [double](#) [axissize\\_](#)
- int [viewx\\_](#)
- int [viewy\\_](#)
- int [viewwidth\\_](#)
- int [viewheight\\_](#)
- [double](#) [viewfield\\_](#)
- [double](#) [winz\\_](#)
- [bool](#) [perspective\\_mode\\_](#)



### 29.181.1 Detailed Description

[gvCamera](#) deals with perspective and modelview transformations, with GL viewports, and with drawing 3D axes.

Definition at line 52 of file gvCamera.h.

### 29.181.2 Constructor & Destructor Documentation

29.181.2.1 `gvCamera::gvCamera ( const Vector3D & fpoint, int x, int y, int w, int h )`

This constructor takes the initial focus point and the viewport as parameters. Default distance (from camera to focal point) is 10.0. Default axissize is zero.

29.181.2.2 `gvCamera::~gvCamera ( )`

### 29.181.3 Member Function Documentation

29.181.3.1 `void gvCamera::computeWinz ( )` `[protected]`

29.181.3.2 `Vector3D gvCamera::eyeCoordsToObjectCoords ( const Vector3D & eyept ) const`

Computes a ray in the model space corresponding to the screen point indicated.

29.181.3.3 `void gvCamera::getAxisSize ( double & size ) const` `[inline]`

Gets the current axissize.

Definition at line 99 of file gvCamera.h.

29.181.3.4 `void gvCamera::getDistance ( double & d ) const` `[inline]`

Gets the current distance.

Definition at line 91 of file gvCamera.h.

29.181.3.5 `void gvCamera::getFocalPoint ( Vector3D & fpoint ) const` `[inline]`

Gets the current focal point.

Definition at line 82 of file gvCamera.h.

29.181.3.6 `bool gvCamera::getPerspectiveMode ( ) const` `[inline]`

Gets the current perspective mode.

Definition at line 140 of file gvCamera.h.

29.181.3.7 `void gvCamera::getViewPort ( int & x, int & y, int & w, int & h ) const` `[inline]`

Gets the viewport.

Definition at line 132 of file gvCamera.h.

29.181.3.8 `Vector3D gvCamera::getWorldRay ( ) const`

29.181.3.9 `Vector3D gvCamera::getWorldSide ( ) const`

29.181.3.10 `Vector3D gvCamera::getWorldUp ( ) const`

29.181.3.11 `void gvCamera::initializeGL ( )`

This is called to initialize the camera. You need a valid OpenGL context to call this.

29.181.3.12 `void gvCamera::moveFocalPointRelative ( const Vector3D & eyeptr )`

Moves the focal point relative to screen coordinates.

29.181.3.13 `Vector3D gvCamera::objectCoordsToWindowCoords ( const Vector3D & pt ) const`

29.181.3.14 `void gvCamera::pick ( int x, int y, int w, int h ) const`

Sets up the projection, viewport and modelview, for picking.

29.181.3.15 `void gvCamera::resetRotation ( )`

Sets the internal modelview matrix to identity. The effect will be visible the next time `use()` is called.

29.181.3.16 `void gvCamera::rotate ( double axisx, double axisy, double angle )`

The axis of rotation is in screen coordinates, and will be translated to the correct coordinate system by the function. To give an example, if you detect a mouse drag in the direction (mx, my) you may want to rotate like this: `camera.↵ rotate(-my, mx, 0.1*sqrt(mx*mx + my*my));`

29.181.3.17 `void gvCamera::rotateTransversal ( double angle )`

Rotates the camera about the current (screen coordinate) z-axis.

29.181.3.18 `void gvCamera::setAxisSize ( double size )` `[inline]`

Sets the size (in modelspace units) of the axis. Setting the size to zero turns off the axis.

Definition at line 96 of file gvCamera.h.

29.181.3.19 `void gvCamera::setDistance ( double d ) [inline]`

Sets the distance. The distance is the modelspace distance from the camera to the focal point.

Definition at line 88 of file gvCamera.h.

29.181.3.20 `void gvCamera::setFocalPoint ( const Vector3D & fpoint ) [inline]`

Sets the focal point. The focal point is the point towards which the camera is looking. Also, the [rotate\(\)](#) call rotates the camera about this point.

Definition at line 79 of file gvCamera.h.

29.181.3.21 `void gvCamera::setPerspectiveMode ( bool perspective_mode ) [inline]`

Sets perspective mode. True means perspective, false means orthographic projection mode.

Definition at line 137 of file gvCamera.h.

29.181.3.22 `void gvCamera::setViewPort ( int x, int y, int w, int h ) [inline]`

Sets the viewport. The viewport should be set whenever the window size changes.

Definition at line 129 of file gvCamera.h.

29.181.3.23 `void gvCamera::use ( ) const`

Sets up the projection, viewport and modelview. Draws axis if axissize is nonzero.

29.181.3.24 `void gvCamera::useModelView ( ) const`

Sets up the modelview only.

## 29.181.4 Member Data Documentation

29.181.4.1 `double gvCamera::axissize_ [protected]`

Definition at line 152 of file gvCamera.h.

29.181.4.2 `double gvCamera::distance_ [protected]`

Definition at line 147 of file gvCamera.h.

29.181.4.3 `Vector3D gvCamera::focal_point_` [protected]

Definition at line 146 of file gvCamera.h.

29.181.4.4 `double gvCamera::mv_[16]` [protected]

mv\_ is the rotation matrix

Definition at line 150 of file gvCamera.h.

29.181.4.5 `bool gvCamera::perspective_mode_` [protected]

Definition at line 163 of file gvCamera.h.

29.181.4.6 `double gvCamera::viewfield_` [protected]

Definition at line 159 of file gvCamera.h.

29.181.4.7 `int gvCamera::viewheight_` [protected]

Definition at line 157 of file gvCamera.h.

29.181.4.8 `int gvCamera::viewwidth_` [protected]

Definition at line 156 of file gvCamera.h.

29.181.4.9 `int gvCamera::viewx_` [protected]

Definition at line 154 of file gvCamera.h.

29.181.4.10 `int gvCamera::viewy_` [protected]

Definition at line 155 of file gvCamera.h.

29.181.4.11 `double gvCamera::winz_` [protected]

Definition at line 161 of file gvCamera.h.

The documentation for this class was generated from the following file:

- [viewlib/include/GoTools/viewlib/gvCamera.h](#)

## 29.182 gvColor Struct Reference

Represents a color by four floats, like in OpenGL.

```
#include <gvColor.h>
```

### Public Member Functions

- [gvColor](#) ()  
*Default color is white.*
- `template<typename NumericType >`  
[gvColor](#) (NumericType r, NumericType g, NumericType b)
- `template<typename NumericType >`  
[gvColor](#) (NumericType r, NumericType g, NumericType b, NumericType alpha)
- [gvColor](#) (const [gvColor](#) &other)
- [gvColor](#) & `operator=` (const [gvColor](#) &other)

### Static Public Member Functions

- `template<typename NumericType >`  
static [gvColor](#) [hsva](#) (NumericType h, NumericType s, NumericType v, NumericType alpha)

### Public Attributes

- float [rgba](#) [4]

#### 29.182.1 Detailed Description

Represents a color by four floats, like in OpenGL.

Definition at line 49 of file [gvColor.h](#).

#### 29.182.2 Constructor & Destructor Documentation

##### 29.182.2.1 [gvColor::gvColor](#) ( ) [[inline](#)]

Default color is white.

Definition at line 53 of file [gvColor.h](#).

##### 29.182.2.2 `template<typename NumericType >` [gvColor::gvColor](#) ( NumericType r, NumericType g, NumericType b ) [[inline](#)]

Definition at line 58 of file [gvColor.h](#).

29.182.2.3 `template<typename NumericType > gvColor::gvColor ( NumericType r, NumericType g, NumericType b,  
NumericType alpha ) [inline]`

Definition at line 64 of file gvColor.h.

29.182.2.4 `gvColor::gvColor ( const gvColor & other ) [inline]`

Definition at line 103 of file gvColor.h.

### 29.182.3 Member Function Documentation

29.182.3.1 `template<typename NumericType > static gvColor gvColor::hsva ( NumericType h, NumericType s, NumericType  
v, NumericType alpha ) [inline],[static]`

Definition at line 72 of file gvColor.h.

29.182.3.2 `gvColor& gvColor::operator= ( const gvColor & other ) [inline]`

Definition at line 110 of file gvColor.h.

### 29.182.4 Member Data Documentation

29.182.4.1 `float gvColor::rgba[4]`

Definition at line 51 of file gvColor.h.

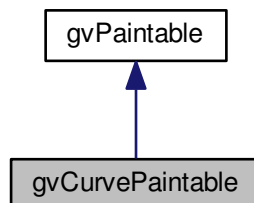
The documentation for this struct was generated from the following file:

- [viewlib/include/GoTools/viewlib/gvColor.h](#)

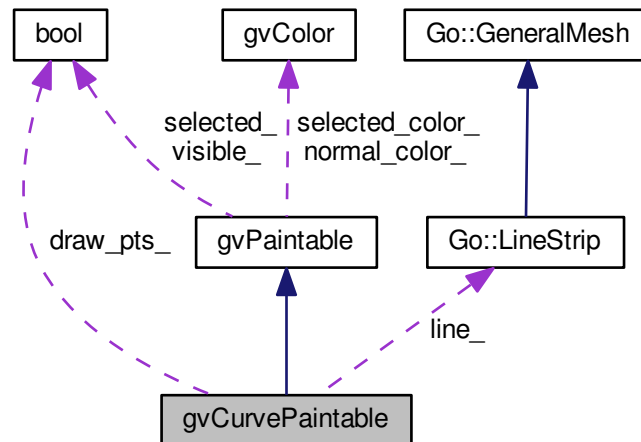
## 29.183 gvCurvePaintable Class Reference

```
#include <gvCurvePaintable.h>
```

Inheritance diagram for gvCurvePaintable:



Collaboration diagram for gvCurvePaintable:



## Public Member Functions

- [gvCurvePaintable](#) ([Go::LineStrip](#) &line, [const gvColor](#) &ncolor, [const gvColor](#) &scolor, int id)
- [gvCurvePaintable](#) ([Go::LineStrip](#) &line, [const gvColor](#) &ncolor, int id)
- virtual [~gvCurvePaintable](#) ()
- virtual void [paint](#) ([gvTexture](#) \*texture)
- void [setDrawPoints](#) ([bool](#) dp)
- [bool](#) [drawPoints](#) ()

## Protected Attributes

- [Go::LineStrip](#) & [line\\_](#)
- [bool](#) [draw\\_pts\\_](#)

### 29.183.1 Detailed Description

OpenGL calls for a parametric curve (represented by a LineStrip).

Definition at line 53 of file gvCurvePaintable.h.

### 29.183.2 Constructor & Destructor Documentation

29.183.2.1 [gvCurvePaintable::gvCurvePaintable](#) ([Go::LineStrip](#) & *line*, [const gvColor](#) & *ncolor*, [const gvColor](#) & *scolor*, int *id*) [[inline](#)]

Definition at line 56 of file gvCurvePaintable.h.

29.183.2.2 `gvCurvePaintable::gvCurvePaintable ( Go::LineStrip & line, const gvColor & ncolor, int id )` `[inline]`

Definition at line 63 of file `gvCurvePaintable.h`.

29.183.2.3 `virtual gvCurvePaintable::~~gvCurvePaintable ( )` `[virtual]`

### 29.183.3 Member Function Documentation

29.183.3.1 `bool gvCurvePaintable::drawPoints ( )` `[inline]`

Definition at line 77 of file `gvCurvePaintable.h`.

29.183.3.2 `virtual void gvCurvePaintable::paint ( gvTexture * texture )` `[virtual]`

The function that does the painting. Must be overridden by all subclasses. If your subclass does not use textures, just ignore the texture argument.

Implements [gvPaintable](#).

29.183.3.3 `void gvCurvePaintable::setDrawPoints ( bool dp )` `[inline]`

Definition at line 73 of file `gvCurvePaintable.h`.

### 29.183.4 Member Data Documentation

29.183.4.1 `bool gvCurvePaintable::draw_pts_` `[protected]`

Definition at line 85 of file `gvCurvePaintable.h`.

29.183.4.2 `Go::LineStrip& gvCurvePaintable::line_` `[protected]`

Definition at line 84 of file `gvCurvePaintable.h`.

The documentation for this class was generated from the following file:

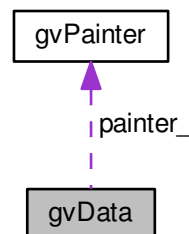
- [viewlib/include/GoTools/viewlib/gvCurvePaintable.h](#)



## 29.184 gvData Class Reference

```
#include <gvData.h>
```

Collaboration diagram for gvData:



### Public Member Functions

- [gvData](#) (std::auto\_ptr< [DataHandler](#) > datahandler)
- virtual [~gvData](#) ()
- void [readIges](#) (std::istream &is)
- void [readSisISrfs](#) (std::istream &is)
- void [readSisCrvs](#) (std::istream &is)
- void [readGo](#) (std::istream &is)
- void [readGo](#) (const std::vector< shared\_ptr< [Go::GeomObject](#) > > &new\_objects, std::vector< shared\_ptr< [gvColor](#) > > &new\_colors)
- void [writeSelectedGo](#) (std::ostream &os)
- void [writeSelectedIges](#) (std::ostream &os)
- void [setTexture](#) (int index, shared\_ptr< [gvTexture](#) > tex)
- shared\_ptr< [gvTexture](#) > [getTexture](#) (int index)
- int [numObjects](#) () const
- shared\_ptr< [Go::GeomObject](#) > [object](#) (int i)
- shared\_ptr< const [Go::GeomObject](#) > [object](#) (int i) const
- shared\_ptr< [gvColor](#) > [color](#) (int i)
- shared\_ptr< [gvPropertySheet](#) > [propertySheet](#) (int i)
- shared\_ptr< [Go::Tesselator](#) > [tesselator](#) (int index)
- void [addGroup](#) ([gvGroup](#) group)
- int [nmbGroups](#) () const
- std::vector< [gvGroup](#) > [objectGroups](#) () const
- [gvGroup](#) [getGroup](#) (int index) const
- void [clearGroup](#) ()
- [gvPainter](#) & [painter](#) ()
- void [registerObserver](#) ([gvObserver](#) \*obs)
- const [Go::BoundingBox](#) & [boundingBox](#) ()
- [Go::BoundingBox](#) [boundingBox](#) (const std::vector< int > &objs) const
- void [extractSelectedObjects](#) (std::vector< shared\_ptr< [Go::GeomObject](#) > > &sel\_objs)
- void [setSelectedStateObject](#) (int id, bool state)
- void [setVisibleStateObject](#) (int id, bool state)

- [bool getSelectedStateObject](#) (int id)
- [bool getVisibleStateObject](#) (int id)
- void [disableUpdates](#) ()
- void [enableUpdates](#) ()
- void [updateObservers](#) ()
- void [clear](#) ()
- void [deleteObj](#) (int obj)
- void [clearLast](#) ()
  - Removes pointers for last object only.*
- void [recreateDataStructure](#) (int nmb\_new\_objs)
  - Not always any need to retesselate all objects.*

### Protected Attributes

- `std::vector< shared_ptr< Go::GeomObject > >` [objects\\_](#)
- `std::vector< shared_ptr< gvColor > >` [object\\_colors\\_](#)
- `gvPainter` [painter\\_](#)

#### 29.184.1 Detailed Description

[gvData](#) reads and stores the geometric data.

Definition at line 65 of file [gvData.h](#).

#### 29.184.2 Constructor & Destructor Documentation

**29.184.2.1** `gvData::gvData ( std::auto\_ptr< DataHandler > datahandler )` `[inline]`

Definition at line 68 of file [gvData.h](#).

**29.184.2.2** `virtual gvData::~gvData ( )` `[inline],[virtual]`

Definition at line 73 of file [gvData.h](#).

#### 29.184.3 Member Function Documentation

**29.184.3.1** `void gvData::addGroup ( gvGroup group )` `[inline]`

Definition at line 117 of file [gvData.h](#).

**29.184.3.2** `const Go::BoundingBox& gvData::boundingBox ( )` `[inline]`

Definition at line 134 of file [gvData.h](#).

29.184.3.3 `Go::BoundingBox gvData::boundingBox ( const std::vector< int > & objs ) const`

29.184.3.4 `void gvData::clear ( )`

29.184.3.5 `void gvData::clearGroup ( ) [inline]`

Definition at line 125 of file gvData.h.

29.184.3.6 `void gvData::clearLast ( )`

Removes pointers for last object only.

29.184.3.7 `shared_ptr<gvColor> gvData::color ( int i ) [inline]`

Definition at line 109 of file gvData.h.

29.184.3.8 `void gvData::deleteObj ( int obj )`

29.184.3.9 `void gvData::disableUpdates ( ) [inline]`

Definition at line 148 of file gvData.h.

29.184.3.10 `void gvData::enableUpdates ( ) [inline]`

Definition at line 150 of file gvData.h.

29.184.3.11 `void gvData::extractSelectedObjects ( std::vector< shared_ptr< Go::GeomObject > > & sel_objs )`

29.184.3.12 `gvGroup gvData::getGroup ( int index ) const [inline]`

Definition at line 123 of file gvData.h.

29.184.3.13 `bool gvData::getSelectedStateObject ( int id )`

29.184.3.14 `shared_ptr<gvTexture> gvData::getTexture ( int index ) [inline]`

Definition at line 94 of file gvData.h.

29.184.3.15 `bool gvData::getVisibleStateObject ( int id )`

29.184.3.16 `int gvData::nmbGroups ( ) const [inline]`

Definition at line 119 of file gvData.h.

29.184.3.17 `int gvData::numObjects ( ) const [inline]`

Definition at line 97 of file gvData.h.

29.184.3.18 `shared_ptr<Go::GeomObject> gvData::object ( int i ) [inline]`

Definition at line 99 of file gvData.h.

29.184.3.19 `shared_ptr<const Go::GeomObject> gvData::object ( int i ) const [inline]`

Definition at line 107 of file gvData.h.

29.184.3.20 `std::vector<gvGroup> gvData::objectGroups ( ) const [inline]`

Definition at line 121 of file gvData.h.

29.184.3.21 `gvPainter& gvData::painter ( ) [inline]`

Definition at line 128 of file gvData.h.

29.184.3.22 `shared_ptr<gvPropertySheet> gvData::propertySheet ( int i ) [inline]`

Definition at line 111 of file gvData.h.

29.184.3.23 `void gvData::readGo ( std::istream & is )`

29.184.3.24 `void gvData::readGo ( const std::vector< shared_ptr< Go::GeomObject > > & new_objects, std::vector< shared_ptr< gvColor > > & new_colors )`

29.184.3.25 `void gvData::readIges ( std::istream & is )`

29.184.3.26 `void gvData::readSisIcrvs ( std::istream & is )`

29.184.3.27 `void gvData::readSisIsrcs ( std::istream & is )`

29.184.3.28 `void gvData::recreateDataStructure ( int nmb_new_objs )`

Not always any need to retesselate all objects.

29.184.3.29 `void gvData::registerObserver ( gvObserver * obs ) [inline]`

Definition at line 131 of file gvData.h.

29.184.3.30 void gvData::setSelectedStateObject ( int *id*, bool *state* )

29.184.3.31 void gvData::setTexture ( int *index*, shared\_ptr< gvTexture > *tex* ) [inline]

Definition at line 91 of file gvData.h.

29.184.3.32 void gvData::setVisibleStateObject ( int *id*, bool *state* )

29.184.3.33 shared\_ptr<Go::Tesselator> gvData::tesselator ( int *index* ) [inline]

Definition at line 113 of file gvData.h.

29.184.3.34 void gvData::updateObservers ( )

29.184.3.35 void gvData::writeSelectedGo ( std::ostream & *os* )

29.184.3.36 void gvData::writeSelectedIges ( std::ostream & *os* )

## 29.184.4 Member Data Documentation

29.184.4.1 std::vector< shared\_ptr<gvColor> > gvData::object\_colors\_ [protected]

Definition at line 165 of file gvData.h.

29.184.4.2 std::vector< shared\_ptr<Go::GeomObject> > gvData::objects\_ [protected]

Definition at line 164 of file gvData.h.

29.184.4.3 gvPainter gvData::painter\_ [protected]

Definition at line 166 of file gvData.h.

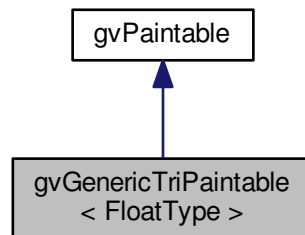
The documentation for this class was generated from the following file:

- [viewlib/include/GoTools/viewlib/gvData.h](#)

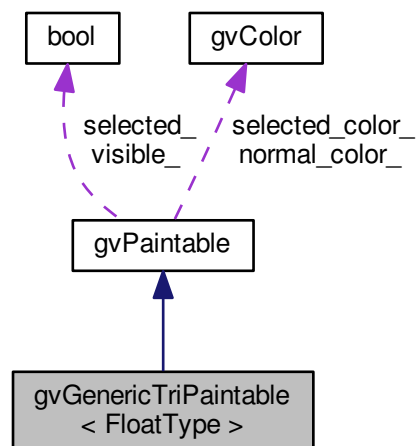
## 29.185 gvGenericTriPaintable< FloatType > Class Template Reference

```
#include <gvGenericTriPaintable.h>
```

Inheritance diagram for gvGenericTriPaintable< FloatType >:



Collaboration diagram for gvGenericTriPaintable< FloatType >:



### Public Member Functions

- [gvGenericTriPaintable](#) (GenericTriMesh< FloatType > &tri, [const gvColor](#) &ncolor, [const gvColor](#) &scolor, int id)
- [gvGenericTriPaintable](#) (GenericTriMesh< FloatType > &tri, [const gvColor](#) &ncolor, int id)
- virtual [~gvGenericTriPaintable](#) ()
- virtual void [paint](#) ([gvTexture](#) \*)

## Protected Attributes

- GenericTriMesh< FloatType > & [tri\\_](#)

### 29.185.1 Detailed Description

```
template<typename FloatType>
class gvGenericTriPaintable< FloatType >
```

Documentation ... etc

Definition at line 70 of file gvGenericTriPaintable.h.

### 29.185.2 Constructor & Destructor Documentation

29.185.2.1 `template<typename FloatType > gvGenericTriPaintable< FloatType >::gvGenericTriPaintable ( GenericTriMesh< FloatType > & tri, const gvColor & ncolor, const gvColor & scolor, int id )` [\[inline\]](#)

Definition at line 73 of file gvGenericTriPaintable.h.

29.185.2.2 `template<typename FloatType > gvGenericTriPaintable< FloatType >::gvGenericTriPaintable ( GenericTriMesh< FloatType > & tri, const gvColor & ncolor, int id )` [\[inline\]](#)

Definition at line 80 of file gvGenericTriPaintable.h.

29.185.2.3 `template<typename FloatType > virtual gvGenericTriPaintable< FloatType >::~~gvGenericTriPaintable ( )` [\[inline\]](#), [\[virtual\]](#)

Definition at line 86 of file gvGenericTriPaintable.h.

### 29.185.3 Member Function Documentation

29.185.3.1 `template<typename FloatType > virtual void gvGenericTriPaintable< FloatType >::paint ( gvTexture * texture )` [\[inline\]](#), [\[virtual\]](#)

The function that does the painting. Must be overridden by all subclasses. If your subclass does not use textures, just ignore the texture argument. Set up the vertex arrays

Set up the normal arrays

Draw the triangles

Implements [gvPaintable](#).

Definition at line 89 of file gvGenericTriPaintable.h.

### 29.185.4 Member Data Documentation

29.185.4.1 `template<typename FloatType > GenericTriMesh<FloatType>& gvGenericTriPaintable< FloatType >::tri_`  
`[protected]`

Definition at line 133 of file gvGenericTriPaintable.h.

The documentation for this class was generated from the following file:

- [viewlib/include/GoTools/viewlib/gvGenericTriPaintable.h](#)

## 29.186 gvGenericTriQuadMesh< FloatType > Class Template Reference

```
#include <gvGenericTriQuadMesh.h>
```

### Public Member Functions

- `gvGenericTriQuadMesh` (int num\_vert, int num\_tri, int num\_quads, `bool` use\_normals=true, `bool` use\_↔  
texcoords=false)
- `bool useNormals` ()
- `bool useTexCoords` ()
- int `numTriangles` ()  
*The total number of triangles.*
- int `numQuads` ()  
*The total number of quads.*
- int `numVertices` ()
- void `resize` (int num\_vert, int num\_tri, int num\_quads)
- `FloatType * vertexArray` ()
- `FloatType * normalArray` ()
- `FloatType * texcoordArray` ()
- int \* `triangleIndexArray` ()
- int \* `quadIndexArray` ()

### 29.186.1 Detailed Description

```
template<typename FloatType>
class gvGenericTriQuadMesh< FloatType >
```

Structure for ...

Definition at line 51 of file gvGenericTriQuadMesh.h.

### 29.186.2 Constructor & Destructor Documentation

29.186.2.1 `template<typename FloatType> gvGenericTriQuadMesh< FloatType >::gvGenericTriQuadMesh ( int`  
`num_vert, int num_tri, int num_quads, bool use_normals = true, bool use_texcoords = false )` `[inline]`

Definition at line 54 of file gvGenericTriQuadMesh.h.



### 29.186.3 Member Function Documentation

29.186.3.1 `template<typename FloatType> FloatType* gvGenericTriQuadMesh< FloatType >::normalArray ( )`  
[inline]

Definition at line 91 of file gvGenericTriQuadMesh.h.

29.186.3.2 `template<typename FloatType> int gvGenericTriQuadMesh< FloatType >::numQuads ( )` [inline]

The total number of quads.

Definition at line 73 of file gvGenericTriQuadMesh.h.

29.186.3.3 `template<typename FloatType> int gvGenericTriQuadMesh< FloatType >::numTriangles ( )` [inline]

The total number of triangles.

Definition at line 70 of file gvGenericTriQuadMesh.h.

29.186.3.4 `template<typename FloatType> int gvGenericTriQuadMesh< FloatType >::numVertices ( )` [inline]

Definition at line 76 of file gvGenericTriQuadMesh.h.

29.186.3.5 `template<typename FloatType> int* gvGenericTriQuadMesh< FloatType >::quadIndexArray ( )`  
[inline]

Definition at line 94 of file gvGenericTriQuadMesh.h.

29.186.3.6 `template<typename FloatType> void gvGenericTriQuadMesh< FloatType >::resize ( int num_vert, int num_tri, int num_quads )` [inline]

Definition at line 79 of file gvGenericTriQuadMesh.h.

29.186.3.7 `template<typename FloatType> FloatType* gvGenericTriQuadMesh< FloatType >::texcoordArray ( )`  
[inline]

Definition at line 92 of file gvGenericTriQuadMesh.h.

29.186.3.8 `template<typename FloatType> int* gvGenericTriQuadMesh< FloatType >::triangleIndexArray ( )`  
[inline]

Definition at line 93 of file gvGenericTriQuadMesh.h.

```
29.186.3.9 template<typename FloatType> bool gvGenericTriQuadMesh< FloatType >::useNormals ()
 [inline]
```

Definition at line 63 of file gvGenericTriQuadMesh.h.

```
29.186.3.10 template<typename FloatType> bool gvGenericTriQuadMesh< FloatType >::useTexCoords ()
 [inline]
```

Definition at line 66 of file gvGenericTriQuadMesh.h.

```
29.186.3.11 template<typename FloatType> FloatType* gvGenericTriQuadMesh< FloatType >::vertexArray ()
 [inline]
```

Definition at line 90 of file gvGenericTriQuadMesh.h.

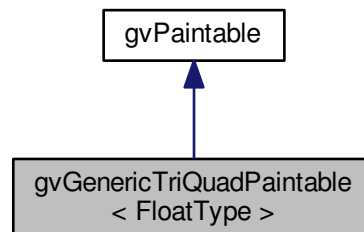
The documentation for this class was generated from the following file:

- [viewlib/include/GoTools/viewlib/gvGenericTriQuadMesh.h](#)

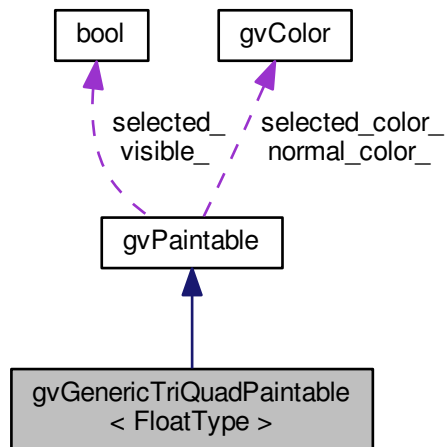
## 29.187 gvGenericTriQuadPaintable< FloatType > Class Template Reference

```
#include <gvGenericTriQuadPaintable.h>
```

Inheritance diagram for gvGenericTriQuadPaintable< FloatType >:



Collaboration diagram for gvGenericTriQuadPaintable< FloatType >:



## Public Member Functions

- [gvGenericTriQuadPaintable](#) ([gvGenericTriQuadMesh](#)< FloatType > &mesh, [const gvColor](#) &ncolor, [const gvColor](#) &scolor, int id)
- [gvGenericTriQuadPaintable](#) ([gvGenericTriQuadMesh](#)< FloatType > &mesh, [const gvColor](#) &ncolor, int id)
- virtual [~gvGenericTriQuadPaintable](#) ()
- virtual void [paint](#) ()

## Protected Attributes

- [gvGenericTriQuadMesh](#)< FloatType > & [mesh\\_](#)

### 29.187.1 Detailed Description

```
template<typename FloatType>
class gvGenericTriQuadPaintable< FloatType >
```

Documentation ... etc

Definition at line 58 of file gvGenericTriQuadPaintable.h.

### 29.187.2 Constructor & Destructor Documentation

29.187.2.1 `template<typename FloatType > gvGenericTriQuadPaintable< FloatType >::gvGenericTriQuadPaintable ( gvGenericTriQuadMesh< FloatType > & mesh, const gvColor & ncolor, const gvColor & scolor, int id ) [inline]`

Definition at line 61 of file gvGenericTriQuadPaintable.h.

29.187.2.2 `template<typename FloatType > gvGenericTriQuadPaintable< FloatType >::gvGenericTriQuadPaintable ( gvGenericTriQuadMesh< FloatType > & mesh, const gvColor & ncolor, int id )`  
`[inline]`

Definition at line 68 of file gvGenericTriQuadPaintable.h.

29.187.2.3 `template<typename FloatType > virtual gvGenericTriQuadPaintable< FloatType >::~~gvGenericTriQuadPaintable ( )` `[inline],[virtual]`

Definition at line 74 of file gvGenericTriQuadPaintable.h.

### 29.187.3 Member Function Documentation

29.187.3.1 `template<typename FloatType > virtual void gvGenericTriQuadPaintable< FloatType >::paint ( )`  
`[inline],[virtual]`

Definition at line 77 of file gvGenericTriQuadPaintable.h.

### 29.187.4 Member Data Documentation

29.187.4.1 `template<typename FloatType > gvGenericTriQuadMesh<FloatType>& gvGenericTriQuadPaintable< FloatType >::mesh_` `[protected]`

Definition at line 122 of file gvGenericTriQuadPaintable.h.

The documentation for this class was generated from the following file:

- [viewlib/include/GoTools/viewlib/gvGenericTriQuadPaintable.h](#)

## 29.188 gvGroup Class Reference

Structure used to store name and index of members of a group of objects.

```
#include <gvGroup.h>
```

### Public Member Functions

- [gvGroup](#) (std::vector< int > &members, QString name)
- [~gvGroup](#) ()
- [QString name \(\) const](#)
- [int size \(\) const](#)
- [int operator\[\]](#) (int index) [const](#)

### 29.188.1 Detailed Description

Structure used to store name and index of members of a group of objects.

Definition at line 47 of file gvGroup.h.

### 29.188.2 Constructor & Destructor Documentation

29.188.2.1 `gvGroup::gvGroup ( std::vector< int > & members, QString name )`

29.188.2.2 `gvGroup::~gvGroup ( )`

### 29.188.3 Member Function Documentation

29.188.3.1 `QString gvGroup::name ( ) const`

29.188.3.2 `int gvGroup::operator[] ( int index ) const`

29.188.3.3 `int gvGroup::size ( ) const`

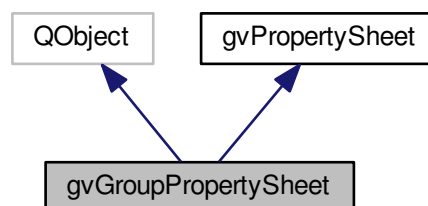
The documentation for this class was generated from the following file:

- [viewlib/include/GoTools/viewlib/gvGroup.h](#)

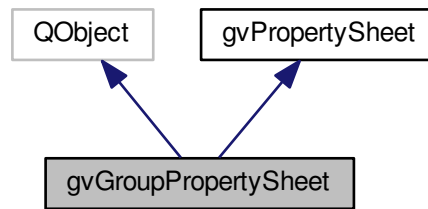
## 29.189 gvGroupPropertySheet Class Reference

```
#include <gvGroupPropertySheet.h>
```

Inheritance diagram for gvGroupPropertySheet:



Collaboration diagram for gvGroupPropertySheet:



### Public Slots

- void [accept](#) ()

### Signals

- void [value\\_changed](#) (std::vector< int > &members, QString name)

### Public Member Functions

- [gvGroupPropertySheet](#) (std::vector< int > &members, QString &def\_name)
- virtual void [createSheet](#) (QWidget \*parent, [gvObserver](#) \*obs)
- std::vector< int > [getMembers](#) ()

#### 29.189.1 Detailed Description

Representation of several objects connected into one logical group.

Definition at line 63 of file gvGroupPropertySheet.h.

#### 29.189.2 Constructor & Destructor Documentation

29.189.2.1 `gvGroupPropertySheet::gvGroupPropertySheet ( std::vector< int > & members, QString & def_name )`

#### 29.189.3 Member Function Documentation

29.189.3.1 `void gvGroupPropertySheet::accept ( ) [slot]`

29.189.3.2 `virtual void gvGroupPropertySheet::createSheet ( QWidget * parent, gvObserver * obs ) [virtual]`

Implements [gvPropertySheet](#).

29.189.3.3 `std::vector<int> gvGroupPropertySheet::getMembers ( )`

29.189.3.4 `void gvGroupPropertySheet::value_changed ( std::vector< int > & members, QString name ) [signal]`

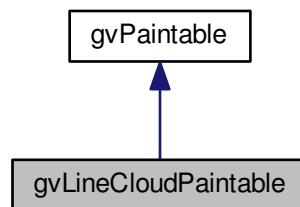
The documentation for this class was generated from the following file:

- [viewlib/include/GoTools/viewlib/gvGroupPropertySheet.h](#)

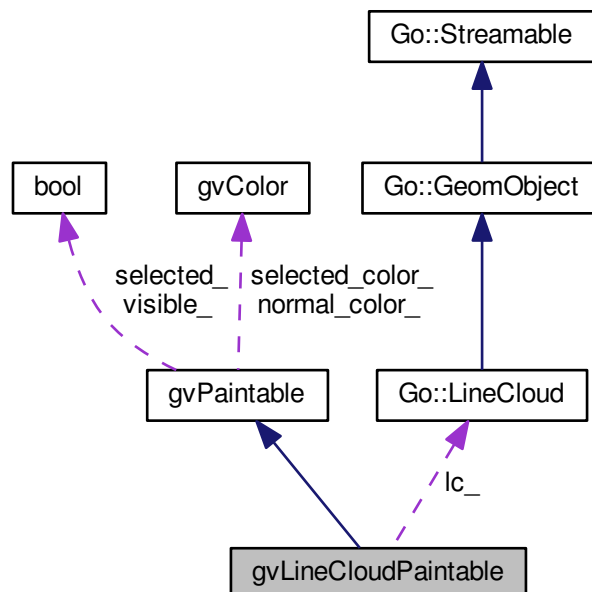
## 29.190 gvLineCloudPaintable Class Reference

```
#include <gvLineCloudPaintable.h>
```

Inheritance diagram for gvLineCloudPaintable:



Collaboration diagram for gvLineCloudPaintable:



## Public Member Functions

- [gvLineCloudPaintable](#) ([const Go::LineCloud](#) &lc, [const gvColor](#) &ncolor, [const gvColor](#) &scolor, int id)
- [gvLineCloudPaintable](#) ([const Go::LineCloud](#) &lc, [const gvColor](#) &ncolor, int id)
- virtual [~gvLineCloudPaintable](#) ()
- virtual void [paint](#) ([gvTexture](#) \*)
- void [setFractionRendered](#) (double f)
- [double fractionRendered](#) ()

## Protected Attributes

- [const Go::LineCloud](#) &lc\_
- [double fractionrendered\\_](#)

### 29.190.1 Detailed Description

Documentation ... etc

Definition at line 64 of file gvLineCloudPaintable.h.

### 29.190.2 Constructor & Destructor Documentation

29.190.2.1 [gvLineCloudPaintable::gvLineCloudPaintable](#) ( [const Go::LineCloud](#) &lc, [const gvColor](#) &ncolor, [const gvColor](#) &scolor, int id ) [[inline](#)]

Definition at line 67 of file gvLineCloudPaintable.h.

29.190.2.2 [gvLineCloudPaintable::gvLineCloudPaintable](#) ( [const Go::LineCloud](#) &lc, [const gvColor](#) &ncolor, int id ) [[inline](#)]

Definition at line 74 of file gvLineCloudPaintable.h.

29.190.2.3 [virtual gvLineCloudPaintable::~~gvLineCloudPaintable](#) ( ) [[inline](#)],[[virtual](#)]

Definition at line 80 of file gvLineCloudPaintable.h.

### 29.190.3 Member Function Documentation

29.190.3.1 [double gvLineCloudPaintable::fractionRendered](#) ( ) [[inline](#)]

Definition at line 105 of file gvLineCloudPaintable.h.



29.190.3.2 `virtual void gvLineCloudPaintable::paint ( gvTexture * texture ) [inline],[virtual]`

The function that does the painting. Must be overridden by all subclasses. If your subclass does not use textures, just ignore the texture argument.

Implements [gvPaintable](#).

Definition at line 83 of file `gvLineCloudPaintable.h`.

29.190.3.3 `void gvLineCloudPaintable::setFractionRendered ( double f ) [inline]`

Definition at line 100 of file `gvLineCloudPaintable.h`.

## 29.190.4 Member Data Documentation

29.190.4.1 `double gvLineCloudPaintable::fractionrendered_ [protected]`

Definition at line 112 of file `gvLineCloudPaintable.h`.

29.190.4.2 `const Go::LineCloud& gvLineCloudPaintable::lc_ [protected]`

Definition at line 111 of file `gvLineCloudPaintable.h`.

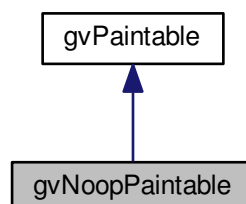
The documentation for this class was generated from the following file:

- [viewlib/include/GoTools/viewlib/gvLineCloudPaintable.h](#)

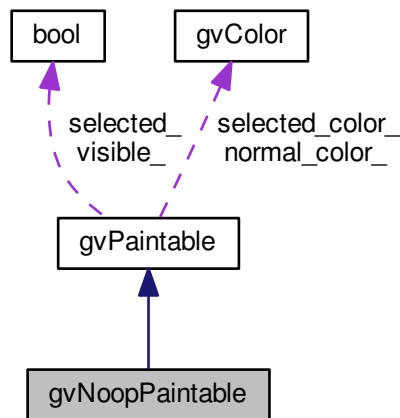
## 29.191 gvNoopPaintable Class Reference

```
#include <gvNoopPaintable.h>
```

Inheritance diagram for `gvNoopPaintable`:



Collaboration diagram for gvNoopPaintable:



## Public Member Functions

- `gvNoopPaintable` (`const gvColor` &ncolor, `const gvColor` &scolor, int id)
- `gvNoopPaintable` (`const gvColor` &ncolor, int id)
- virtual `~gvNoopPaintable` ()
- virtual void `paint` (`gvTexture` \*texture)

## Additional Inherited Members

### 29.191.1 Detailed Description

Documentation ...

Definition at line 51 of file gvNoopPaintable.h.

### 29.191.2 Constructor & Destructor Documentation

29.191.2.1 `gvNoopPaintable::gvNoopPaintable ( const gvColor & ncolor, const gvColor & scolor, int id )` [`inline`]

Definition at line 54 of file gvNoopPaintable.h.

29.191.2.2 `gvNoopPaintable::gvNoopPaintable ( const gvColor & ncolor, int id )` [`inline`]

Definition at line 59 of file gvNoopPaintable.h.

29.191.2.3 virtual gvNoopPaintable::~~gvNoopPaintable ( ) [virtual]

### 29.191.3 Member Function Documentation

29.191.3.1 virtual void gvNoopPaintable::paint ( gvTexture \* texture ) [virtual]

The function that does the painting. Must be overridden by all subclasses. If your subclass does not use textures, just ignore the texture argument.

Implements [gvPaintable](#).

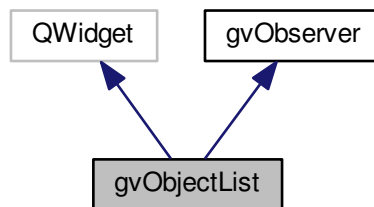
The documentation for this class was generated from the following file:

- [viewlib/include/GoTools/viewlib/gvNoopPaintable.h](#)

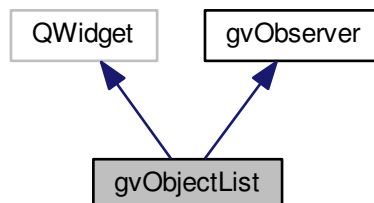
## 29.192 gvObjectList Class Reference

```
#include <gvObjectList.h>
```

Inheritance diagram for gvObjectList:



Collaboration diagram for gvObjectList:



## Public Member Functions

- [gvObjectList](#) ([gvData](#) &data, [QWidget](#) \*parent=0, [const](#) char \*name=0, [Qt::WFlags](#) f=0)
- virtual [~gvObjectList](#) ()
- virtual void [observedChanged](#) ()
- void [buildGUI](#) ()

## Protected Slots

- void [clicked](#) (int id)

### 29.192.1 Detailed Description

Documentation ... etc

Definition at line 60 of file [gvObjectList.h](#).

### 29.192.2 Constructor & Destructor Documentation

29.192.2.1 [gvObjectList::gvObjectList](#) ( [gvData](#) & data, [QWidget](#) \* parent = 0, [const](#) char \* name = 0, [Qt::WFlags](#) f = 0 )

29.192.2.2 virtual [gvObjectList::~~gvObjectList](#) ( ) [[virtual](#)]

### 29.192.3 Member Function Documentation

29.192.3.1 void [gvObjectList::buildGUI](#) ( )

29.192.3.2 void [gvObjectList::clicked](#) ( int id ) [[protected](#)],[[slot](#)]

29.192.3.3 virtual void [gvObjectList::observedChanged](#) ( ) [[virtual](#)]

Implements [gvObserver](#).

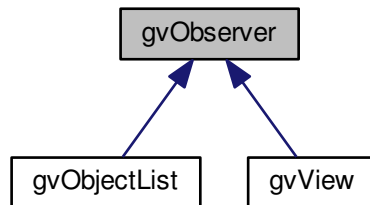
The documentation for this class was generated from the following file:

- [viewlib/include/GoTools/viewlib/gvObjectList.h](#)

## 29.193 gvObserver Class Reference

```
#include <gvObserver.h>
```

Inheritance diagram for gvObserver:



### Public Member Functions

- virtual [~gvObserver](#) ()
- virtual void [observedChanged](#) ()=0

#### 29.193.1 Detailed Description

Documentation ... etc

Definition at line 47 of file gvObserver.h.

#### 29.193.2 Constructor & Destructor Documentation

29.193.2.1 virtual gvObserver::~gvObserver ( ) [virtual]

#### 29.193.3 Member Function Documentation

29.193.3.1 virtual void gvObserver::observedChanged ( ) [pure virtual]

Implemented in [gvView](#), and [gvObjectList](#).

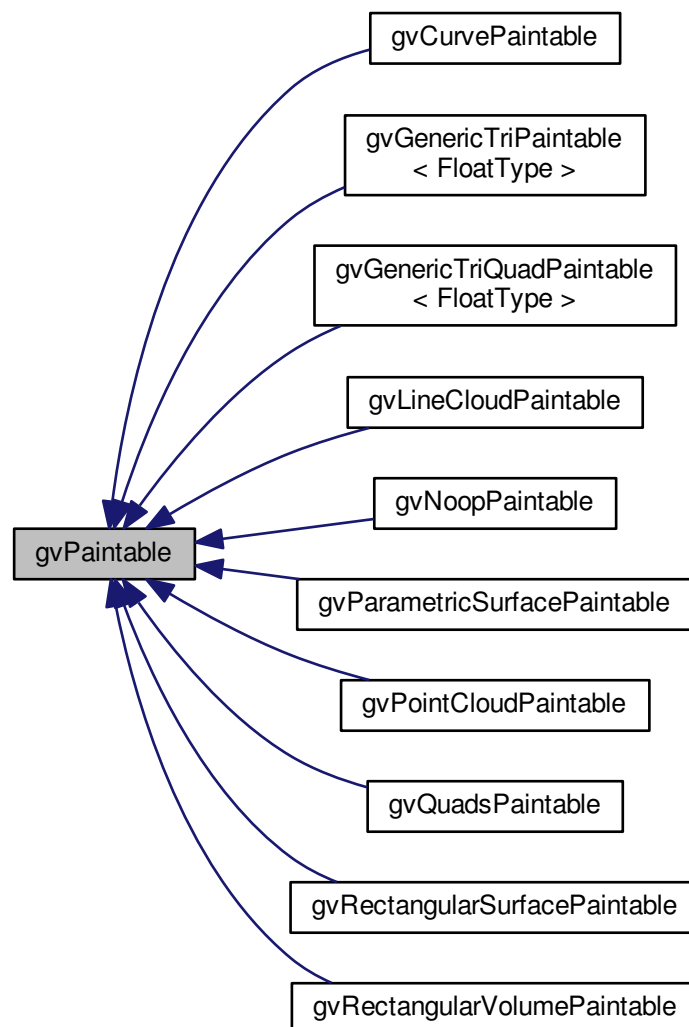
The documentation for this class was generated from the following file:

- [viewlib/include/GoTools/viewlib/gvObserver.h](#)

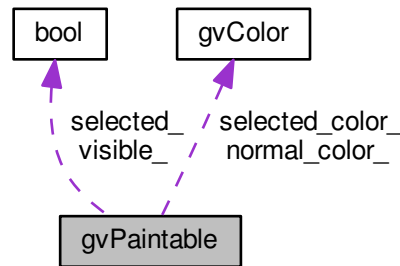
## 29.194 gvPaintable Class Reference

```
#include <gvPaintable.h>
```

Inheritance diagram for gvPaintable:



Collaboration diagram for gvPaintable:



## Public Member Functions

- [gvPaintable](#) ([const gvColor](#) &ncolor, [const gvColor](#) &scolor, int id)
- [gvPaintable](#) ([const gvColor](#) &ncolor, int id)
- virtual [~gvPaintable](#) ()
- virtual void [paint](#) ([gvTexture](#) \*texture)=0
- virtual void [paintGL](#) (QGLWidget \*w, [gvTexture](#) \*texture)
- void [setVisible](#) (bool state)
- bool [visible](#) () const
- void [setSelected](#) (bool state)
- bool [selected](#) () const
- void [setColor](#) ([const gvColor](#) &ncolor, [const gvColor](#) &scolor)
- void [setColor](#) ([const gvColor](#) &ncolor)
- [gvColor](#) [getNormalColor](#) ()
- int [id](#) () const
- void [setId](#) (int id)

## Protected Attributes

- bool [visible\\_](#)
- bool [selected\\_](#)
- [gvColor](#) [normal\\_color\\_](#)
- [gvColor](#) [selected\\_color\\_](#)
- int [id\\_](#)

### 29.194.1 Detailed Description

[gvPaintable](#): Super class for OpenGL calls to geometric objects.

Definition at line 52 of file [gvPaintable.h](#).

## 29.194.2 Constructor & Destructor Documentation

29.194.2.1 `gvPaintable::gvPaintable ( const gvColor & ncolor, const gvColor & scolor, int id )` `[inline]`

The arguments of this constructor (which will be called only by subclasses) are the colors to use in normal and selected states.

Definition at line 58 of file `gvPaintable.h`.

29.194.2.2 `gvPaintable::gvPaintable ( const gvColor & ncolor, int id )` `[inline]`

If the selected-mode color is not specified, a lighter shade of the normal color will be used.

Definition at line 68 of file `gvPaintable.h`.

29.194.2.3 `virtual gvPaintable::~~gvPaintable ( )` `[virtual]`

## 29.194.3 Member Function Documentation

29.194.3.1 `gvColor gvPaintable::getNormalColor ( )` `[inline]`

Definition at line 125 of file `gvPaintable.h`.

29.194.3.2 `int gvPaintable::id ( ) const` `[inline]`

Definition at line 129 of file `gvPaintable.h`.

29.194.3.3 `virtual void gvPaintable::paint ( gvTexture * texture )` `[pure virtual]`

The function that does the painting. Must be overridden by all subclasses. If your subclass does not use textures, just ignore the texture argument.

Implemented in [gvParametricSurfacePaintable](#), [gvGenericTriPaintable< FloatType >](#), [gvPointCloudPaintable](#), [gvLineCloudPaintable](#), [gvCurvePaintable](#), [gvRectangularVolumePaintable](#), [gvQuadsPaintable](#), [gvRectangularSurfacePaintable](#), and [gvNoopPaintable](#).

29.194.3.4 `virtual void gvPaintable::paintGL ( QGLWidget * w, gvTexture * texture )` `[inline],[virtual]`

Reimplemented in [gvPointCloudPaintable](#).

Definition at line 88 of file `gvPaintable.h`.

29.194.3.5 `bool gvPaintable::selected ( ) const` `[inline]`

Definition at line 105 of file `gvPaintable.h`.



29.194.3.6 void gvPaintable::setColor ( const gvColor & ncolor, const gvColor & scolor ) [inline]

Definition at line 109 of file gvPaintable.h.

29.194.3.7 void gvPaintable::setColor ( const gvColor & ncolor ) [inline]

Definition at line 115 of file gvPaintable.h.

29.194.3.8 void gvPaintable::setId ( int id ) [inline]

Definition at line 133 of file gvPaintable.h.

29.194.3.9 void gvPaintable::setSelected ( bool state ) [inline]

Definition at line 101 of file gvPaintable.h.

29.194.3.10 void gvPaintable::setVisible ( bool state ) [inline]

Definition at line 93 of file gvPaintable.h.

29.194.3.11 bool gvPaintable::visible ( ) const [inline]

Definition at line 97 of file gvPaintable.h.

## 29.194.4 Member Data Documentation

29.194.4.1 int gvPaintable::id\_ [protected]

Definition at line 143 of file gvPaintable.h.

29.194.4.2 gvColor gvPaintable::normal\_color\_ [protected]

Definition at line 141 of file gvPaintable.h.

29.194.4.3 bool gvPaintable::selected\_ [protected]

Definition at line 140 of file gvPaintable.h.

29.194.4.4 gvColor gvPaintable::selected\_color\_ [protected]

Definition at line 142 of file gvPaintable.h.

29.194.4.5 `bool gvPaintable::visible_` [`protected`]

Definition at line 139 of file `gvPaintable.h`.

The documentation for this class was generated from the following file:

- [viewlib/include/GoTools/viewlib/gvPaintable.h](#)

## 29.195 gvPainter Class Reference

The `gvPainter` is responsible for painting the 3D scene.

```
#include <gvPainter.h>
```

### Public Member Functions

- `gvPainter` ()
- virtual `~gvPainter` ()
- virtual void `drawScene` (QGLWidget \*w=NULL)
- void `addPaintable` (shared\_ptr< `gvPaintable` > pa)
- void `removeAllPaintables` ()
- void `removePaintable` (int id)
- void `removeLastPaintable` ()
- `gvPaintable` & `getPaintable` (int index)
- void `setTexture` (int index, shared\_ptr< `gvTexture` > tex)
- shared\_ptr< `gvTexture` > `getTexture` (int index)
- shared\_ptr< const `gvTexture` > `setTexture` (int index)

### Protected Attributes

- std::vector< shared\_ptr< `gvPaintable` > > `paintables_`
- std::vector< shared\_ptr< `gvTexture` > > `textures_`

#### 29.195.1 Detailed Description

The `gvPainter` is responsible for painting the 3D scene.

Definition at line 53 of file `gvPainter.h`.

#### 29.195.2 Constructor & Destructor Documentation

29.195.2.1 `gvPainter::gvPainter` ( ) [`inline`]

Definition at line 56 of file `gvPainter.h`.

29.195.2.2 `virtual gvPainter::~gvPainter ( ) [virtual]`

### 29.195.3 Member Function Documentation

29.195.3.1 `void gvPainter::addPaintable ( shared_ptr< gvPaintable > pa ) [inline]`

Add a new paintable. [gvPaintable](#) is the abstract bas class of all paintables.

Definition at line 66 of file gvPainter.h.

29.195.3.2 `virtual void gvPainter::drawScene ( QGLWidget * w =NULL ) [virtual]`

Draws the 3D scene by calling the paint() virtual method of all its contained gvPaintables.

29.195.3.3 `gvPaintable& gvPainter::getPaintable ( int index ) [inline]`

Call this to access the individual paintables, in order to manipulate selection state, visibility state or color.

Definition at line 98 of file gvPainter.h.

29.195.3.4 `shared_ptr<gvTexture> gvPainter::getTexture ( int index ) [inline]`

Definition at line 109 of file gvPainter.h.

29.195.3.5 `void gvPainter::removeAllPaintables ( ) [inline]`

Removes all paintables, if you want to paint something you have to [addPaintable\(\)](#) again.

Definition at line 79 of file gvPainter.h.

29.195.3.6 `void gvPainter::removeLastPaintable ( ) [inline]`

Definition at line 91 of file gvPainter.h.

29.195.3.7 `void gvPainter::removePaintable ( int id ) [inline]`

Definition at line 85 of file gvPainter.h.

29.195.3.8 `void gvPainter::setTexture ( int index, shared_ptr< gvTexture > tex ) [inline]`

Definition at line 103 of file gvPainter.h.

29.195.3.9 `shared_ptr<const gvTexture> gvPainter::setTexture ( int index )` [inline]

Definition at line 115 of file gvPainter.h.

#### 29.195.4 Member Data Documentation

29.195.4.1 `std::vector< shared_ptr<gvPaintable> > gvPainter::paintables_` [protected]

Definition at line 122 of file gvPainter.h.

29.195.4.2 `std::vector< shared_ptr<gvTexture> > gvPainter::textures_` [protected]

Definition at line 123 of file gvPainter.h.

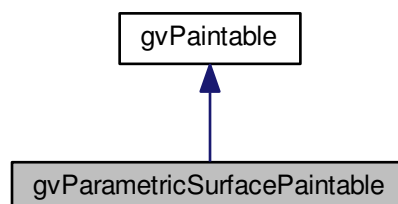
The documentation for this class was generated from the following file:

- [viewlib/include/GoTools/viewlib/gvPainter.h](#)

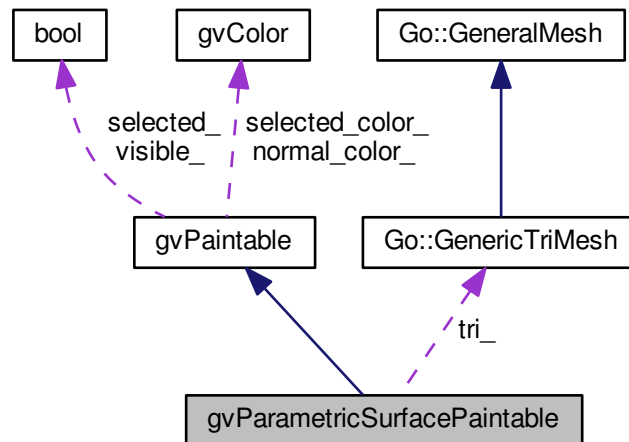
#### 29.196 gvParametricSurfacePaintable Class Reference

```
#include <gvParametricSurfacePaintable.h>
```

Inheritance diagram for gvParametricSurfacePaintable:



Collaboration diagram for gvParametricSurfacePaintable:



## Public Member Functions

- `gvParametricSurfacePaintable` (`genMesh` &tri, `const gvColor` &ncolor, `const gvColor` &scolor, int id)
- `gvParametricSurfacePaintable` (`genMesh` &tri, `const gvColor` &ncolor, int id)
- virtual `~gvParametricSurfacePaintable` ()
- virtual void `paint` (`gvTexture` \*texture)

## Protected Member Functions

- void `createSurface` ()
- void `drawSurface` ()

## Protected Attributes

- `genMesh` & tri\_

### 29.196.1 Detailed Description

`gvParametricSurfacePaintable`: OpenGL calls for a parametric surface.

Definition at line 63 of file `gvParametricSurfacePaintable.h`.

### 29.196.2 Constructor & Destructor Documentation

29.196.2.1 `gvParametricSurfacePaintable::gvParametricSurfacePaintable` (`genMesh` & tri, `const gvColor` & ncolor, `const gvColor` & scolor, int id ) `[inline]`

Definition at line 66 of file `gvParametricSurfacePaintable.h`.

29.196.2.2 `gvParametricSurfacePaintable::gvParametricSurfacePaintable ( genMesh & tri, const gvColor & ncolor, int id )`  
`[inline]`

Definition at line 73 of file `gvParametricSurfacePaintable.h`.

29.196.2.3 `virtual gvParametricSurfacePaintable::~~gvParametricSurfacePaintable ( )` `[virtual]`

### 29.196.3 Member Function Documentation

29.196.3.1 `void gvParametricSurfacePaintable::createSurface ( )` `[protected]`

29.196.3.2 `void gvParametricSurfacePaintable::drawSurface ( )` `[protected]`

29.196.3.3 `virtual void gvParametricSurfacePaintable::paint ( gvTexture * texture )` `[virtual]`

The function that does the painting. Must be overridden by all subclasses. If your subclass does not use textures, just ignore the texture argument.

Implements [gvPaintable](#).

### 29.196.4 Member Data Documentation

29.196.4.1 `genMesh& gvParametricSurfacePaintable::tri_` `[protected]`

Definition at line 94 of file `gvParametricSurfacePaintable.h`.

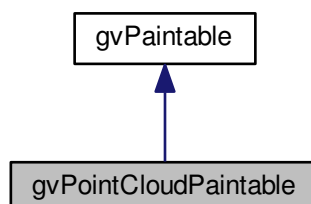
The documentation for this class was generated from the following file:

- [viewlib/include/GoTools/viewlib/gvParametricSurfacePaintable.h](#)

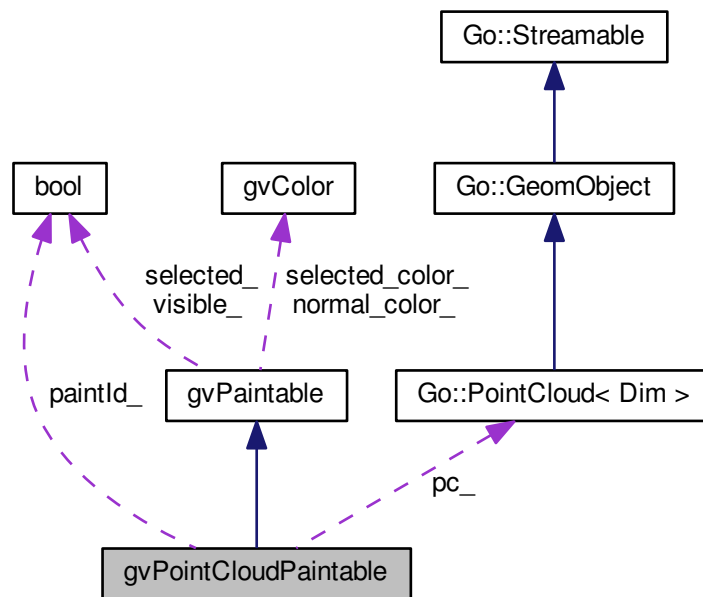
## 29.197 gvPointCloudPaintable Class Reference

```
#include <gvPointCloudPaintable.h>
```

Inheritance diagram for `gvPointCloudPaintable`:



Collaboration diagram for gvPointCloudPaintable:



## Public Member Functions

- `gvPointCloudPaintable` (`const Go::PointCloud3D &pc`, `const gvColor &ncolor`, `const gvColor &scolor`, `int id`, `bool paintId=false`)
- `gvPointCloudPaintable` (`const Go::PointCloud3D &pc`, `const gvColor &ncolor`, `int id`, `bool paintId=false`)
- virtual `~gvPointCloudPaintable` ()
- virtual void `paint` (`gvTexture *`)
- virtual void `paintGL` (`QGLWidget *w`, `gvTexture *texture`)
- void `setFractionRendered` (`double f`)
- `double fractionRendered` ()
- void `setPointSize` (`double sz`)
- `double pointSize` ()
- `bool getPaintId` () `const`
- void `setPaintId` (`bool paintId`)
- void `translate` (`const std::vector< double > &vert_translation`)

*Translate all vertices by `vert_translation`, wrt the geometry, discarding any previous translation.*

## Protected Attributes

- `const Go::PointCloud3D & pc_`
- `double fractionrendered_`
- `double pointsize_`
- `bool paintId_`
- `std::vector< double > vert_translation_`

*The 3D-translation wrt the geometry.*

### 29.197.1 Detailed Description

Documentation ... etc

Definition at line 68 of file gvPointCloudPaintable.h.

### 29.197.2 Constructor & Destructor Documentation

29.197.2.1 `gvPointCloudPaintable::gvPointCloudPaintable ( const Go::PointCloud3D & pc, const gvColor & ncolor, const gvColor & scolor, int id, bool paintId = false ) [inline]`

Definition at line 71 of file gvPointCloudPaintable.h.

29.197.2.2 `gvPointCloudPaintable::gvPointCloudPaintable ( const Go::PointCloud3D & pc, const gvColor & ncolor, int id, bool paintId = false ) [inline]`

Definition at line 79 of file gvPointCloudPaintable.h.

29.197.2.3 `virtual gvPointCloudPaintable::~gvPointCloudPaintable ( ) [inline],[virtual]`

Definition at line 86 of file gvPointCloudPaintable.h.

### 29.197.3 Member Function Documentation

29.197.3.1 `double gvPointCloudPaintable::fractionRendered ( ) [inline]`

Definition at line 142 of file gvPointCloudPaintable.h.

29.197.3.2 `bool gvPointCloudPaintable::getPaintId ( ) const [inline]`

Definition at line 157 of file gvPointCloudPaintable.h.

29.197.3.3 `virtual void gvPointCloudPaintable::paint ( gvTexture * texture ) [inline],[virtual]`

The function that does the painting. Must be overridden by all subclasses. If your subclass does not use textures, just ignore the texture argument.

Implements [gvPaintable](#).

Definition at line 89 of file gvPointCloudPaintable.h.



29.197.3.4 `virtual void gvPointCloudPaintable::paintGL ( QGLWidget * w, gvTexture * texture ) [inline], [virtual]`

Reimplemented from [gvPaintable](#).

Definition at line 112 of file `gvPointCloudPaintable.h`.

29.197.3.5 `double gvPointCloudPaintable::pointSize ( ) [inline]`

Definition at line 152 of file `gvPointCloudPaintable.h`.

29.197.3.6 `void gvPointCloudPaintable::setFractionRendered ( double f ) [inline]`

Definition at line 137 of file `gvPointCloudPaintable.h`.

29.197.3.7 `void gvPointCloudPaintable::setPaintId ( bool paintId ) [inline]`

Definition at line 162 of file `gvPointCloudPaintable.h`.

29.197.3.8 `void gvPointCloudPaintable::setPointSize ( double sz ) [inline]`

Definition at line 147 of file `gvPointCloudPaintable.h`.

29.197.3.9 `void gvPointCloudPaintable::translate ( const std::vector< double > & vert_translation ) [inline]`

Translate all vertices by `vert_translation`, wrt the geometry, discarding any previous translation.

Definition at line 168 of file `gvPointCloudPaintable.h`.

## 29.197.4 Member Data Documentation

29.197.4.1 `double gvPointCloudPaintable::fractionrendered_ [protected]`

Definition at line 175 of file `gvPointCloudPaintable.h`.

29.197.4.2 `bool gvPointCloudPaintable::paintId_ [protected]`

Definition at line 177 of file `gvPointCloudPaintable.h`.

29.197.4.3 `const Go::PointCloud3D& gvPointCloudPaintable::pc_ [protected]`

Definition at line 174 of file `gvPointCloudPaintable.h`.

29.197.4.4 `double gvPointCloudPaintable::pointsize_` [protected]

Definition at line 176 of file `gvPointCloudPaintable.h`.

29.197.4.5 `std::vector<double> gvPointCloudPaintable::vert_translation_` [protected]

The 3D-translation wrt the geometry.

Definition at line 180 of file `gvPointCloudPaintable.h`.

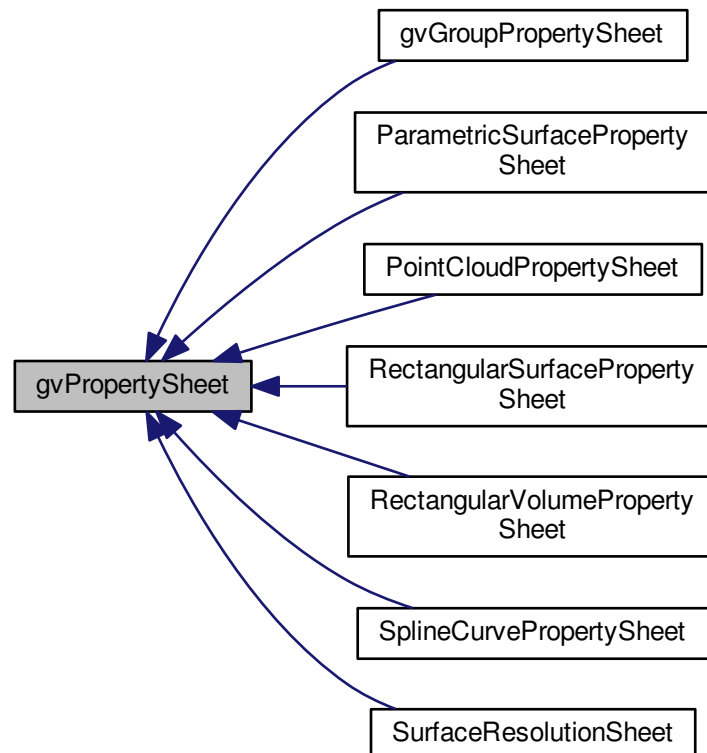
The documentation for this class was generated from the following file:

- [viewlib/include/GoTools/viewlib/gvPointCloudPaintable.h](#)

## 29.198 gvPropertySheet Class Reference

```
#include <gvPropertySheet.h>
```

Inheritance diagram for `gvPropertySheet`:



## Public Member Functions

- virtual [~gvPropertySheet](#) ()
- virtual void [createSheet](#) (QWidget \*parent, [gvObserver](#) \*observer)=0

### 29.198.1 Detailed Description

Documentation ... etc

Definition at line 50 of file gvPropertySheet.h.

### 29.198.2 Constructor & Destructor Documentation

29.198.2.1 virtual gvPropertySheet::~gvPropertySheet ( ) [virtual]

### 29.198.3 Member Function Documentation

29.198.3.1 virtual void gvPropertySheet::createSheet ( QWidget \* *parent*, [gvObserver](#) \* *observer* ) [pure virtual]

Implemented in [ParametricSurfacePropertySheet](#), [RectangularVolumePropertySheet](#), [RectangularSurfacePropertySheet](#), [gvGroupPropertySheet](#), [SurfaceResolutionSheet](#), [SplineCurvePropertySheet](#), and [PointCloudPropertySheet](#).

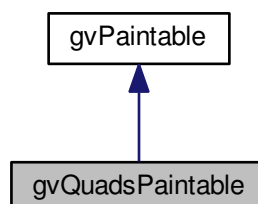
The documentation for this class was generated from the following file:

- viewlib/include/GoTools/viewlib/[gvPropertySheet.h](#)

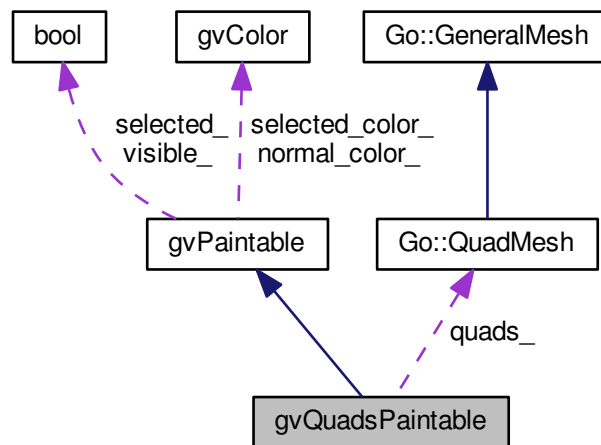
## 29.199 gvQuadsPaintable Class Reference

```
#include <gvQuadsPaintable.h>
```

Inheritance diagram for gvQuadsPaintable:



Collaboration diagram for gvQuadsPaintable:



## Public Member Functions

- [gvQuadsPaintable](#) ([Go::QuadMesh](#) &quads, [const gvColor](#) &ncolor, [const gvColor](#) &scolor, int id)
- [gvQuadsPaintable](#) ([Go::QuadMesh](#) &quads, [const gvColor](#) &ncolor, int id)
- virtual [~gvQuadsPaintable](#) ()
- virtual void [paint](#) ([gvTexture](#) \*texture)

## Protected Attributes

- [Go::QuadMesh](#) & [quads\\_](#)

### 29.199.1 Detailed Description

Brief description. Detailed description.

Definition at line 52 of file gvQuadsPaintable.h.

### 29.199.2 Constructor & Destructor Documentation

29.199.2.1 [gvQuadsPaintable::gvQuadsPaintable](#) ( [Go::QuadMesh](#) & *quads*, [const gvColor](#) & *ncolor*, [const gvColor](#) & *scolor*, int *id* ) `[inline]`

Definition at line 55 of file gvQuadsPaintable.h.

```
29.199.2.2 gvQuadsPaintable::gvQuadsPaintable (Go::QuadMesh & quads, const gvColor & ncolor, int id)
 [inline]
```

Definition at line 62 of file gvQuadsPaintable.h.

```
29.199.2.3 virtual gvQuadsPaintable::~~gvQuadsPaintable () [virtual]
```

### 29.199.3 Member Function Documentation

```
29.199.3.1 virtual void gvQuadsPaintable::paint (gvTexture * texture) [virtual]
```

The function that does the painting. Must be overridden by all subclasses. If your subclass does not use textures, just ignore the texture argument.

Implements [gvPaintable](#).

### 29.199.4 Member Data Documentation

```
29.199.4.1 Go::QuadMesh& gvQuadsPaintable::quads_ [protected]
```

Definition at line 74 of file gvQuadsPaintable.h.

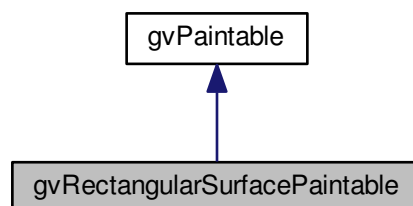
The documentation for this class was generated from the following file:

- [viewlib/include/GoTools/viewlib/gvQuadsPaintable.h](#)

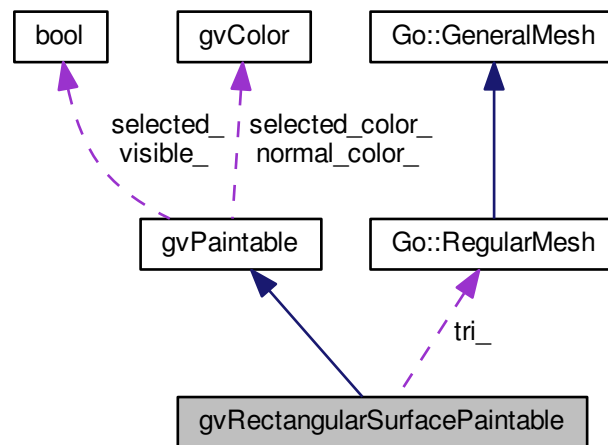
## 29.200 gvRectangularSurfacePaintable Class Reference

```
#include <gvRectangularSurfacePaintable.h>
```

Inheritance diagram for gvRectangularSurfacePaintable:



Collaboration diagram for `gvRectangularSurfacePaintable`:



## Public Member Functions

- `gvRectangularSurfacePaintable` (`Go::RegularMesh` &tri, `const gvColor` &ncolor, `const gvColor` &scolor, int id)
- `gvRectangularSurfacePaintable` (`Go::RegularMesh` &tri, `const gvColor` &ncolor, int id)
- virtual `~gvRectangularSurfacePaintable` ()
- virtual void `paint` (`gvTexture` \*texture)

## Protected Attributes

- `Go::RegularMesh` & tri\_

### 29.200.1 Detailed Description

`gvRectangularSurfacePaintable`: OpenGL calls for a parametric surface on a rectangular domain.

Definition at line 52 of file `gvRectangularSurfacePaintable.h`.

### 29.200.2 Constructor & Destructor Documentation

29.200.2.1 `gvRectangularSurfacePaintable::gvRectangularSurfacePaintable` (`Go::RegularMesh` & tri, `const gvColor` & ncolor, `const gvColor` & scolor, int id ) [inline]

Definition at line 55 of file `gvRectangularSurfacePaintable.h`.

29.200.2.2 `gvRectangularSurfacePaintable::gvRectangularSurfacePaintable ( Go::RegularMesh & tri, const gvColor & ncolor, int id ) [inline]`

Definition at line 62 of file `gvRectangularSurfacePaintable.h`.

29.200.2.3 `virtual gvRectangularSurfacePaintable::~~gvRectangularSurfacePaintable ( ) [virtual]`

### 29.200.3 Member Function Documentation

29.200.3.1 `virtual void gvRectangularSurfacePaintable::paint ( gvTexture * texture ) [virtual]`

The function that does the painting. Must be overridden by all subclasses. If your subclass does not use textures, just ignore the texture argument.

Implements [gvPaintable](#).

### 29.200.4 Member Data Documentation

29.200.4.1 `Go::RegularMesh& gvRectangularSurfacePaintable::tri_ [protected]`

Definition at line 74 of file `gvRectangularSurfacePaintable.h`.

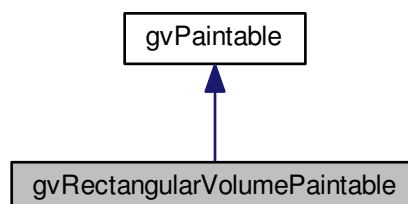
The documentation for this class was generated from the following file:

- [viewlib/include/GoTools/viewlib/gvRectangularSurfacePaintable.h](#)

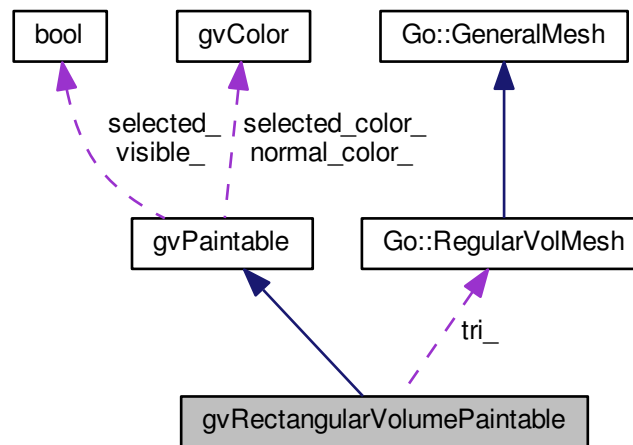
## 29.201 gvRectangularVolumePaintable Class Reference

```
#include <gvRectangularVolumePaintable.h>
```

Inheritance diagram for `gvRectangularVolumePaintable`:



Collaboration diagram for gvRectangularVolumePaintable:



## Public Member Functions

- `gvRectangularVolumePaintable` (`Go::RegularVolMesh &tri`, `const gvColor &ncolor`, `const gvColor &scolor`, `int id`)
- `gvRectangularVolumePaintable` (`Go::RegularVolMesh &tri`, `const gvColor &ncolor`, `int id`)
- virtual `~gvRectangularVolumePaintable` ()
- virtual void `paint` (`gvTexture *texture`)

## Protected Attributes

- `Go::RegularVolMesh & tri_`

### 29.201.1 Detailed Description

`gvRectangularSurfacePaintable`: OpenGL calls for a parametric surface on a rectangular domain.

Definition at line 53 of file `gvRectangularVolumePaintable.h`.

### 29.201.2 Constructor & Destructor Documentation

29.201.2.1 `gvRectangularVolumePaintable::gvRectangularVolumePaintable` (`Go::RegularVolMesh & tri`, `const gvColor & ncolor`, `const gvColor & scolor`, `int id`) [`inline`]

Definition at line 56 of file `gvRectangularVolumePaintable.h`.



29.201.2.2 `gvRectangularVolumePaintable::gvRectangularVolumePaintable ( Go::RegularVolMesh & tri, const gvColor & ncolor, int id ) [inline]`

Definition at line 63 of file `gvRectangularVolumePaintable.h`.

29.201.2.3 `virtual gvRectangularVolumePaintable::~~gvRectangularVolumePaintable ( ) [virtual]`

### 29.201.3 Member Function Documentation

29.201.3.1 `virtual void gvRectangularVolumePaintable::paint ( gvTexture * texture ) [virtual]`

The function that does the painting. Must be overridden by all subclasses. If your subclass does not use textures, just ignore the texture argument.

Implements [gvPaintable](#).

### 29.201.4 Member Data Documentation

29.201.4.1 `Go::RegularVolMesh& gvRectangularVolumePaintable::tri_ [protected]`

Definition at line 75 of file `gvRectangularVolumePaintable.h`.

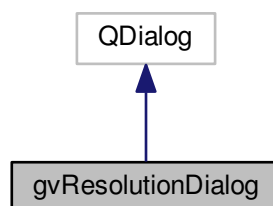
The documentation for this class was generated from the following file:

- [viewlib/include/GoTools/viewlib/vol\\_and\\_lr/gvRectangularVolumePaintable.h](#)

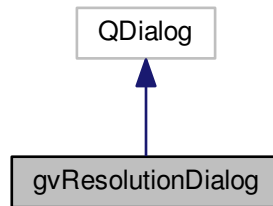
## 29.202 gvResolutionDialog Class Reference

```
#include <gvResolutionDialog.h>
```

Inheritance diagram for `gvResolutionDialog`:



Collaboration diagram for gvResolutionDialog:



## Signals

- void `valuesChanged` (int, int)

## Public Member Functions

- `gvResolutionDialog` (int `current_res_u`, int `current_res_v`, int `minimum_res`, int `maximum_res`, `QWidget` `*parent=0`, `const char` `*name=0`, `bool` `modal=FALSE`, `Qt::WFlags` `f=0`)
- virtual `~gvResolutionDialog` ()

## Protected Slots

- void `apply` ()
- virtual void `accept` ()

## Protected Attributes

- int `ures_`
- int `vres_`
- `QSlider` `* uslide_`
- `QSlider` `* vslide_`

### 29.202.1 Detailed Description

Documentation ... etc

Definition at line 51 of file `gvResolutionDialog.h`.

## 29.202.2 Constructor & Destructor Documentation

29.202.2.1 `gvResolutionDialog::gvResolutionDialog ( int current_res_u, int current_res_v, int minimum_res, int maximum_res, QWidget * parent = 0, const char * name = 0, bool modal = FALSE, Qt::WFlags f = 0 )`

29.202.2.2 `virtual gvResolutionDialog::~gvResolutionDialog ( )` [virtual]

## 29.202.3 Member Function Documentation

29.202.3.1 `virtual void gvResolutionDialog::accept ( )` [protected],[virtual],[slot]

29.202.3.2 `void gvResolutionDialog::apply ( )` [protected],[slot]

29.202.3.3 `void gvResolutionDialog::valuesChanged ( int, int )` [signal]

## 29.202.4 Member Data Documentation

29.202.4.1 `int gvResolutionDialog::ures_` [protected]

Definition at line 74 of file gvResolutionDialog.h.

29.202.4.2 `QSlider* gvResolutionDialog::uslide_` [protected]

Definition at line 76 of file gvResolutionDialog.h.

29.202.4.3 `int gvResolutionDialog::vres_` [protected]

Definition at line 75 of file gvResolutionDialog.h.

29.202.4.4 `QSlider* gvResolutionDialog::vslide_` [protected]

Definition at line 77 of file gvResolutionDialog.h.

The documentation for this class was generated from the following file:

- [viewlib/include/GoTools/viewlib/gvResolutionDialog.h](#)

## 29.203 gvStandardMouseHandler Class Reference

```
#include <gvStandardMouseHandler.h>
```

## Public Member Functions

- [gvStandardMouseHandler](#) ()
- virtual [~gvStandardMouseHandler](#) ()

### 29.203.1 Detailed Description

Documentation ... etc

Definition at line 66 of file gvStandardMouseHandler.h.

### 29.203.2 Constructor & Destructor Documentation

29.203.2.1 [gvStandardMouseHandler::gvStandardMouseHandler](#) ( )

29.203.2.2 virtual [gvStandardMouseHandler::~~gvStandardMouseHandler](#) ( ) [virtual]

The documentation for this class was generated from the following file:

- [viewlib/include/GoTools/viewlib/gvStandardMouseHandler.h](#)

## 29.204 gvTexture Class Reference

```
#include <gvTexture.h>
```

### Public Member Functions

- [gvTexture](#) (bool mipmapped=false)
- [gvTexture](#) (const [gvTexture](#) &other)
- [gvTexture](#) & [operator=](#) (const [gvTexture](#) &other)
- [gvTexture](#) (int height, int width, unsigned char \*pixels, bool mipmapped=false)
- [gvTexture](#) (std::string filename, bool mipmapped=false)
- virtual void [readFile](#) (std::string filename)
- virtual [~gvTexture](#) ()
- bool [genTexture](#) () const
- bool [bind](#) () const
- void [getQImage](#) (QImage &res, bool withAlpha=false) const
- int [height](#) () const
- int [width](#) () const
- void [setAlphaValue](#) (int val)
- void [setTextureMatrix](#) (const double tm[])
- void [setIdentMatrix](#) ()
- void [setMinFilter](#) ([MinFilterSet](#) filter, bool flush=true)
- void [setMagFilter](#) ([MagFilterSet](#) filter, bool flush=true)
- void [setEnvMode](#) ([EnvModeSet](#) mode, bool flush=true)
- void [setWrapMode](#) ([WrapModeSet](#) mode, bool flush=true)
- [MinFilterSet](#) [getMinFilter](#) ()
- [MagFilterSet](#) [getMagFilter](#) ()
- [EnvModeSet](#) [getEnvMode](#) ()
- void [setCenterEdgeTexels](#) (bool enable)

### 29.204.1 Detailed Description

Documentation ... etc

Definition at line 66 of file gvTexture.h.

### 29.204.2 Constructor & Destructor Documentation

29.204.2.1 `gvTexture::gvTexture ( bool mipmapped = false )`

29.204.2.2 `gvTexture::gvTexture ( const gvTexture & other )`

29.204.2.3 `gvTexture::gvTexture ( int height, int width, unsigned char * pixels, bool mipmapped = false )`

the size of the pixels array should be height\*width\*4 in RGBA format

29.204.2.4 `gvTexture::gvTexture ( std::string filename, bool mipmapped = false )`

29.204.2.5 `virtual gvTexture::~~gvTexture ( )` [virtual]

### 29.204.3 Member Function Documentation

29.204.3.1 `bool gvTexture::bind ( ) const`

29.204.3.2 `bool gvTexture::genTexture ( ) const`

29.204.3.3 `EnvModeSet gvTexture::getEnvMode ( )`

29.204.3.4 `MagFilterSet gvTexture::getMagFilter ( )`

29.204.3.5 `MinFilterSet gvTexture::getMinFilter ( )`

29.204.3.6 `void gvTexture::getQImage ( QImage & res, bool withAlpha = false ) const`

29.204.3.7 `int gvTexture::height ( ) const` [inline]

Definition at line 85 of file gvTexture.h.

29.204.3.8 `gvTexture& gvTexture::operator= ( const gvTexture & other )`

29.204.3.9 `virtual void gvTexture::readFile ( std::string filename )` [virtual]

29.204.3.10 `void gvTexture::setAlphaValue ( int val )`

29.204.3.11 `void gvTexture::setCenterEdgeTexels ( bool enable )` [inline]

Definition at line 107 of file gvTexture.h.

29.204.3.12 void gvTexture::setEnvMode ( EnvModeSet mode, bool flush = true )

29.204.3.13 void gvTexture::setIdentMatrix ( )

29.204.3.14 void gvTexture::setMagFilter ( MagFilterSet filter, bool flush = true )

29.204.3.15 void gvTexture::setMinFilter ( MinFilterSet filter, bool flush = true )

29.204.3.16 void gvTexture::setTextureMatrix ( const double tm[ ] )

29.204.3.17 void gvTexture::setWrapMode ( WrapModeSet mode, bool flush = true )

29.204.3.18 int gvTexture::width ( ) const [inline]

Definition at line 86 of file gvTexture.h.

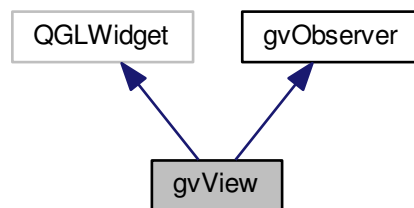
The documentation for this class was generated from the following file:

- [viewlib/include/GoTools/viewlib/gvTexture.h](#)

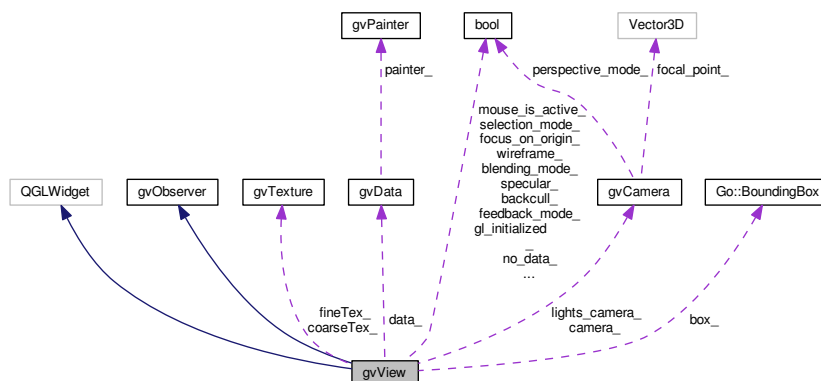
## 29.205 gvView Class Reference

```
#include <gvView.h>
```

Inheritance diagram for gvView:



Collaboration diagram for gvView:



## Public Slots

- void [setWireframe](#) (bool mode)  
*Set wireframe state.*
- void [setSelectionMode](#) (bool mode)  
*Set selection mode.*
- void [setAxis](#) (bool mode)  
*Set paint-axis state.*
- void [setBackCull](#) (bool mode)  
*Set backface-culling state.*
- void [setSpecular](#) (bool mode)  
*Set specular highlighting state.*
- void [setPerspective](#) (bool mode)  
*Set perspective projection state.*
- void [setFeedbackmode](#) (bool mode)  
*Set feedback state.*
- void [setBlendingmode](#) (bool mode)  
*Set blending state.*
- void [setGetClickmode](#) (bool mode)  
*Set get click mode.*
- void [setCenter](#) (int x, int y)  
*Centers view on point at x,y in back buffer.*

## Signals

- void [objectPicked](#) (unsigned int name)  
*An object has been picked.*
- void [objectsPicked](#) (unsigned int \*names, int numnames)  
*An object has been picked.*
- void [feedback](#) (int xrel, int yrel)

## Public Member Functions

- [gvView](#) (gvData &data, QWidget \*parent=0, const char \*name=0, const QGLWidget \*shareWidget=0, Qt::WFlags f=0)
- virtual [~gvView](#) ()
- virtual void [paintGL](#) ()
- virtual void [saveSnapshot](#) (int w, int h, const QString &filename)
- virtual void [initializeGL](#) ()  
*Initialize GL state (shademodel, lights, etc.)*
- virtual void [resizeGL](#) (int w, int h)
- virtual void [getObjAndParam](#) (int mousex, int mousey, shared\_ptr< const Go::ParamSurface > &obj, double &tex\_u, double &tex\_v)  
*Searching among visible and selected objects.*
- void [pick](#) (int mousex, int mousey)  
*Pick the geometrical object at a mouse coordinate.*
- void [pickRegion](#) (int mousex, int mousey, int w, int h)  
*Pick the geometrical objects in a region of the window.*
- void [getWindowCoords](#) (const Vector3D &pt, int &mousex, int &mousey) const
- virtual QSize [sizeHint](#) () const

*@ Not sure if I need this*

- virtual void [observedChanged](#) ()
- void [focusOnBox](#) ()
- void [focusOnVisible](#) ()
- [bool wireframe](#) ()
  - Are we in wireframe mode?*
- [bool selectionmode](#) ()
  - Are we in selection mode?*
- [bool axis](#) ()
  - Are we in paint-axis mode?*
- [bool backCull](#) ()
  - Are we in bacface-culling mode?*
- [bool specular](#) ()
  - Are we in specular highlight mode?*
- [bool perspective](#) ()
  - Are we in perspective projection mode?*
- [bool feedbackmode](#) () [const](#)
  - Are we in feedback mode?*
- [bool blendingmode](#) () [const](#)
  - Should the alpha value be used for blending?*
- void [setOriginFocalPoint](#) ([bool](#) origin)
  - Let the camera focus on the origin. Vertices are then translated towards the origin.*
- virtual QSize [minimumSizeHint](#) () [const](#)

### Protected Member Functions

- [gvView](#) ([const](#) QGLFormat &format, [gvData](#) &data, QWidget \*parent, [const](#) char \*name, [const](#) QGLWidget \*shareWidget, Qt::WFlags f)
- virtual void [mousePressEvent](#) (QMouseEvent \*e)
- virtual void [mouseReleaseEvent](#) (QMouseEvent \*e)
- virtual void [mouseMoveEvent](#) (QMouseEvent \*e)
- virtual [bool get3Dpoint](#) (int mousex, int mousey, Vector3D &objpt)
- void [drawOverlay](#) ()
- [gvTexture](#) \* [makeFineCheckImage](#) ()
- [gvTexture](#) \* [makeCoarseCheckImage](#) ()

### Static Protected Member Functions

- static void [texCoordInt](#) (int num\_objs, int ind, [double](#) &min\_s, [double](#) &max\_s, [double](#) &min\_t, [double](#) &max\_↵  
\_t)

### Protected Attributes

- [bool gl\\_initialized\\_](#)
- [bool no\\_data\\_](#)
- [bool selection\\_mode\\_](#)
- [bool selecting\\_](#)
- [bool feedback\\_mode\\_](#)
- [bool get\\_click\\_mode\\_](#)
- [bool mouse\\_is\\_active\\_](#)



*True if mouse was pressed in widget area and is not yet released.*

- [bool blending\\_mode\\_](#)
- [gvData & data\\_](#)
- [Go::BoundingBox box\\_](#)
- [gvCamera camera\\_](#)
- [gvCamera lights\\_camera\\_](#)
- [double base\\_axis\\_size\\_](#)
- [Qt::MouseButton mouse\\_button\\_](#)
- [QPoint last\\_mouse\\_pos\\_](#)
- [QPoint starting\\_mouse\\_pos\\_](#)
- [double draglength\\_](#)
- [double unitx\\_](#)
- [double unity\\_](#)
- [bool wireframe\\_](#)
- [bool axis\\_](#)
- [bool backcull\\_](#)
- [bool specular\\_](#)
- [gvTexture \\* coarseTex\\_](#)
- [gvTexture \\* fineTex\\_](#)
- [QPainter \\* painter\\_](#)
- [bool focus\\_on\\_origin\\_](#)

### 29.205.1 Detailed Description

Documentation ... etc

Definition at line 68 of file gvView.h.

### 29.205.2 Constructor & Destructor Documentation

29.205.2.1 `gvView::gvView ( gvData & data, QWidget * parent = 0, const char * name = 0, const QGLWidget * shareWidget = 0, Qt::WFlags f = 0 )`

The [gvData&](#) argument is the [gvData](#) object this object should observe. The other arguments are standard arguments for the [QGLWidget](#) superclass.

29.205.2.2 `virtual gvView::~~gvView ( ) [virtual]`

29.205.2.3 `gvView::gvView ( const QGLFormat & format, gvData & data, QWidget * parent, const char * name, const QGLWidget * shareWidget, Qt::WFlags f ) [protected]`

### 29.205.3 Member Function Documentation

29.205.3.1 `bool gvView::axis ( ) [inline]`

Are we in paint-axis mode?

Definition at line 127 of file gvView.h.

29.205.3.2 `bool gvView::backCull ( ) [inline]`

Are we in bacface-culling mode?

Definition at line 129 of file gvView.h.

29.205.3.3 `bool gvView::blendingmode ( ) const [inline]`

Should the alpha value be used for blending?

Definition at line 137 of file gvView.h.

29.205.3.4 `void gvView::drawOverlay ( ) [protected]`

29.205.3.5 `void gvView::feedback ( int xrel, int yrel ) [signal]`

In feedback mode, indicates mouse position relative to start of drag.

29.205.3.6 `bool gvView::feedbackmode ( ) const [inline]`

Are we in feedback mode?

Definition at line 135 of file gvView.h.

29.205.3.7 `void gvView::focusOnBox ( )`

Put the center of the bounding box of the model in the middle of the screen, at a (hopefully) useful distance from the camera.

29.205.3.8 `void gvView::focusOnVisible ( )`

29.205.3.9 `virtual bool gvView::get3Dpoint ( int mousex, int mousey, Vector3D & objpt ) [protected],[virtual]`

29.205.3.10 `virtual void gvView::getObjAndParam ( int mousex, int mousey, shared_ptr< const Go::ParamSurface > & obj, double & tex_u, double & tex_v ) [virtual]`

Searching among visible and selected objects.

29.205.3.11 `void gvView::getWindowCoords ( const Vector3D & pt, int & mousex, int & mousey ) const`

29.205.3.12 `virtual void gvView::initializeGL ( ) [virtual]`

Initialize GL state (shademodel, lights, etc.)

29.205.3.13 `gvTexture* gvView::makeCoarseCheckImage ( )` [protected]

29.205.3.14 `gvTexture* gvView::makeFineCheckImage ( )` [protected]

29.205.3.15 `virtual QSize gvView::minimumSizeHint ( ) const` [inline],[virtual]

Definition at line 142 of file gvView.h.

29.205.3.16 `virtual void gvView::mouseMoveEvent ( QMouseEvent * e )` [protected],[virtual]

Overridden in order to make mouse moves translate, zoom and rotate the camera.

29.205.3.17 `virtual void gvView::mousePressEvent ( QMouseEvent * e )` [protected],[virtual]

Overridden in order to make mouse moves translate, zoom and rotate the camera.

29.205.3.18 `virtual void gvView::mouseReleaseEvent ( QMouseEvent * e )` [protected],[virtual]

Overridden in order to make mouse moves translate, zoom and rotate the camera.

29.205.3.19 `void gvView::objectPicked ( unsigned int name )` [signal]

An object has been picked.

29.205.3.20 `void gvView::objectsPicked ( unsigned int * names, int numnames )` [signal]

An object has been picked.

29.205.3.21 `virtual void gvView::observedChanged ( )` [virtual]

This function is called whenever the observed [gvData](#) object is changed.

Implements [gvObserver](#).

29.205.3.22 `virtual void gvView::paintGL ( )` [virtual]

This function does the actual drawing into the widget's area. After clearing the area, it will call `use()` on the camera, and ask the [gvPainter](#) owned by the observed [gvData](#) object to do a repaint.

29.205.3.23 `bool gvView::perspective ( )` [inline]

Are we in perspective projection mode?

Definition at line 133 of file gvView.h.

29.205.3.24 `void gvView::pick ( int mousex, int mousey )`

Pick the geometrical object at a mouse coordinate.

29.205.3.25 `void gvView::pickRegion ( int mousex, int mousey, int w, int h )`

Pick the geometrical objects in a region of the window.

29.205.3.26 `virtual void gvView::resizeGL ( int w, int h )` [virtual]

Called when the widget is resized. It will set the viewport of the camera and then do a repaint.

29.205.3.27 `virtual void gvView::saveSnapshot ( int w, int h, const QString & filename )` [virtual]

29.205.3.28 `bool gvView::selectionmode ( )` [inline]

Are we in selection mode?

Definition at line 125 of file gvView.h.

29.205.3.29 `void gvView::setAxis ( bool mode )` [slot]

Set paint-axis state.

29.205.3.30 `void gvView::setBackCull ( bool mode )` [slot]

Set backface-culling state.

29.205.3.31 `void gvView::setBlendingmode ( bool mode )` [slot]

Set blending state.

29.205.3.32 `void gvView::setCenter ( int x, int y )` [slot]

Centers view on point at x,y in back buffer.

29.205.3.33 `void gvView::setFeedbackmode ( bool mode )` [slot]

Set feedback state.

29.205.3.34 `void gvView::setGetClickmode ( bool mode ) [slot]`

Set get click mode.

29.205.3.35 `void gvView::setOriginFocalPoint ( bool origin )`

Let the camera focus on the origin. Vertices are then translated towards the origin.

29.205.3.36 `void gvView::setPerspective ( bool mode ) [slot]`

Set perspective projection state.

29.205.3.37 `void gvView::setSelectionMode ( bool mode ) [slot]`

Set selection mode.

29.205.3.38 `void gvView::setSpecular ( bool mode ) [slot]`

Set specular highlighting state.

29.205.3.39 `void gvView::setWireframe ( bool mode ) [slot]`

Set wireframe state.

29.205.3.40 `virtual QSize gvView::sizeHint ( ) const [virtual]`

@ Not sure if I need this

29.205.3.41 `bool gvView::specular ( ) [inline]`

Are we in specular highlight mode?

Definition at line 131 of file gvView.h.

29.205.3.42 `static void gvView::texCoordInt ( int num_objs, int ind, double & min_s, double & max_s, double & min_t, double & max_t ) [static],[protected]`

29.205.3.43 `bool gvView::wireframe ( ) [inline]`

Are we in wireframe mode?

Definition at line 123 of file gvView.h.

## 29.205.4 Member Data Documentation

### 29.205.4.1 `bool gvView::axis_` [protected]

Definition at line 238 of file gvView.h.

### 29.205.4.2 `bool gvView::backcull_` [protected]

Definition at line 239 of file gvView.h.

### 29.205.4.3 `double gvView::base_axis_size_` [protected]

Definition at line 225 of file gvView.h.

### 29.205.4.4 `bool gvView::blending_mode_` [protected]

Definition at line 218 of file gvView.h.

### 29.205.4.5 `Go::BoundingBox gvView::box_` [protected]

Definition at line 222 of file gvView.h.

### 29.205.4.6 `gvCamera gvView::camera_` [protected]

Definition at line 223 of file gvView.h.

### 29.205.4.7 `gvTexture* gvView::coarseTex_` [protected]

Definition at line 241 of file gvView.h.

### 29.205.4.8 `gvData& gvView::data_` [protected]

Definition at line 220 of file gvView.h.

### 29.205.4.9 `double gvView::draglength_` [protected]

Definition at line 233 of file gvView.h.

### 29.205.4.10 `bool gvView::feedback_mode_` [protected]

Definition at line 213 of file gvView.h.

29.205.4.11 **gvTexture\*** gvView::fineTex\_ [protected]

Definition at line 242 of file gvView.h.

29.205.4.12 **bool** gvView::focus\_on\_origin\_ [protected]

Definition at line 246 of file gvView.h.

29.205.4.13 **bool** gvView::get\_click\_mode\_ [protected]

Definition at line 214 of file gvView.h.

29.205.4.14 **bool** gvView::gl\_initialized\_ [protected]

Definition at line 209 of file gvView.h.

29.205.4.15 **QPoint** gvView::last\_mouse\_pos\_ [protected]

Definition at line 231 of file gvView.h.

29.205.4.16 **gvCamera** gvView::lights\_camera\_ [protected]

Definition at line 224 of file gvView.h.

29.205.4.17 **Qt::MouseButton** gvView::mouse\_button\_ [protected]

Definition at line 228 of file gvView.h.

29.205.4.18 **bool** gvView::mouse\_is\_active\_ [protected]

True if mouse was pressed in widget area and is not yet released.

Definition at line 216 of file gvView.h.

29.205.4.19 **bool** gvView::no\_data\_ [protected]

Definition at line 210 of file gvView.h.

29.205.4.20 **QPainter\*** gvView::painter\_ [protected]

Definition at line 244 of file gvView.h.

29.205.4.21 **bool gvView::selecting\_** [protected]

Definition at line 212 of file gvView.h.

29.205.4.22 **bool gvView::selection\_mode\_** [protected]

Definition at line 211 of file gvView.h.

29.205.4.23 **bool gvView::specular\_** [protected]

Definition at line 240 of file gvView.h.

29.205.4.24 **QPoint gvView::starting\_mouse\_pos\_** [protected]

Definition at line 232 of file gvView.h.

29.205.4.25 **double gvView::unitx\_** [protected]

Definition at line 234 of file gvView.h.

29.205.4.26 **double gvView::unity\_** [protected]

Definition at line 235 of file gvView.h.

29.205.4.27 **bool gvView::wireframe\_** [protected]

Definition at line 237 of file gvView.h.

The documentation for this class was generated from the following file:

- [viewlib/include/GoTools/viewlib/gvView.h](#)

## 29.206 Handle< T > Class Template Reference

Template class for smart pointers. The actual class must inherit from [HandleId](#).

```
#include <Handle.h>
```



## Public Member Functions

- [Handle \(\)](#)
- [Handle \(const T &ref\)](#)
- [Handle \(T \\*p\)](#)
- [Handle \(const Handle< T > &ref\)](#)
- [~Handle \(\)](#)
- [void rebind \(const T \\*pc\)](#)
- [void rebind \(const T &p\)](#)
- [const T \\* operator-> \(\) const](#)
- [T \\* operator-> \(\)](#)
- [const T & operator\(\) \(\) const](#)
- [T & operator\(\) \(\)](#)
- [const T & operator\\* \(\) const](#)
- [T & operator\\* \(\)](#)
- [const T \\* getPtr \(\) const](#)
- [T \\* getPtr \(\)](#)
- [const T & getRef \(\) const](#)
- [T & getRef \(\)](#)
- [void operator= \(const Handle< T > &h\)](#)
- [void operator= \(const T \\*p\)](#)
- [void operator= \(const T &p\)](#)
- [bool operator== \(const Handle< T > &h\) const](#)
- [bool operator!= \(const Handle< T > &h\) const](#)
- [bool operator< \(const Handle< T > &h\) const](#)
- [bool operator> \(const Handle< T > &h\) const](#)

## Protected Attributes

- [T \\* classptr](#)

### 29.206.1 Detailed Description

```
template<class T>
class Handle< T >
```

Template class for smart pointers. The actual class must inherit from [HandleId](#).

Definition at line 57 of file Handle.h.

### 29.206.2 Constructor & Destructor Documentation

29.206.2.1 `template<class T> Handle< T >::Handle ( )` [[inline](#)]

Definition at line 63 of file Handle.h.

29.206.2.2 `template<class T> Handle< T >::Handle ( const T & ref )` [[inline](#)]

Definition at line 65 of file Handle.h.

29.206.2.3 `template<class T> Handle< T >::Handle ( T * p ) [inline]`

Definition at line 70 of file Handle.h.

29.206.2.4 `template<class T> Handle< T >::Handle ( const Handle< T > & ref ) [inline]`

Definition at line 75 of file Handle.h.

29.206.2.5 `template<class T> Handle< T >::~~Handle ( ) [inline]`

Definition at line 81 of file Handle.h.

### 29.206.3 Member Function Documentation

29.206.3.1 `template<class T> const T* Handle< T >::getPtr ( ) const [inline]`

Definition at line 115 of file Handle.h.

29.206.3.2 `template<class T> T* Handle< T >::getPtr ( ) [inline]`

Definition at line 116 of file Handle.h.

29.206.3.3 `template<class T> const T& Handle< T >::getRef ( ) const [inline]`

Definition at line 117 of file Handle.h.

29.206.3.4 `template<class T> T& Handle< T >::getRef ( ) [inline]`

Definition at line 118 of file Handle.h.

29.206.3.5 `template<class T> bool Handle< T >::operator!=( const Handle< T > & h ) const [inline]`

Definition at line 125 of file Handle.h.

29.206.3.6 `template<class T> const T& Handle< T >::operator()( ) const [inline]`

Definition at line 111 of file Handle.h.

29.206.3.7 `template<class T> T& Handle< T >::operator()( ) [inline]`

Definition at line 112 of file Handle.h.

29.206.3.8 `template<class T> const T& Handle< T >::operator*( ) const [inline]`

Definition at line 113 of file Handle.h.

29.206.3.9 `template<class T> T& Handle< T >::operator*( ) [inline]`

Definition at line 114 of file Handle.h.

29.206.3.10 `template<class T> const T* Handle< T >::operator->( ) const [inline]`

Definition at line 109 of file Handle.h.

29.206.3.11 `template<class T> T* Handle< T >::operator->( ) [inline]`

Definition at line 110 of file Handle.h.

29.206.3.12 `template<class T> bool Handle< T >::operator<( const Handle< T > & h ) const [inline]`

Definition at line 126 of file Handle.h.

29.206.3.13 `template<class T> void Handle< T >::operator=( const Handle< T > & h ) [inline]`

Definition at line 120 of file Handle.h.

29.206.3.14 `template<class T> void Handle< T >::operator=( const T * p ) [inline]`

Definition at line 121 of file Handle.h.

29.206.3.15 `template<class T> void Handle< T >::operator=( const T & p ) [inline]`

Definition at line 122 of file Handle.h.

29.206.3.16 `template<class T> bool Handle< T >::operator==( const Handle< T > & h ) const [inline]`

Definition at line 124 of file Handle.h.

29.206.3.17 `template<class T> bool Handle< T >::operator>( const Handle< T > & h ) const [inline]`

Definition at line 127 of file Handle.h.

29.206.3.18 `template<class T> void Handle< T >::rebind ( const T * p ) [inline]`

Definition at line 92 of file Handle.h.

29.206.3.19 `template<class T> void Handle< T >::rebind ( const T & p ) [inline]`

Definition at line 107 of file Handle.h.

## 29.206.4 Member Data Documentation

29.206.4.1 `template<class T> T* Handle< T >::classptr [protected]`

Definition at line 60 of file Handle.h.

The documentation for this class was generated from the following file:

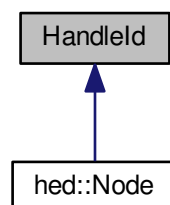
- [ttl/include/ttl/utis/Handle.h](#)

## 29.207 HandleId Class Reference

Base class with reference counting for smart pointers.

```
#include <HandleId.h>
```

Inheritance diagram for HandleId:



### Public Member Functions

- [HandleId \(\)](#)
- [virtual ~HandleId \(\)](#)
- [bool isReferenced \(\) const](#)
- [int getNoRefs \(\) const](#)
- [bool dynamicObj \(\) const](#)
- [void increment \(\)](#)
- [void decrement \(\)](#)
- [void \\* operator new \(size\\_t t\)](#)
- [void \\* operator new \(size\\_t t, int, const char \\*file, int line\)](#)
- [void operator delete \(void \\*v\)](#)

## Protected Attributes

- int [refcount](#)
- char [dynamic\\_object](#)

### 29.207.1 Detailed Description

Base class with reference counting for smart pointers.

Definition at line 55 of file HandleId.h.

### 29.207.2 Constructor & Destructor Documentation

#### 29.207.2.1 HandleId::HandleId ( )

Definition at line 54 of file HandleId.cpp.

#### 29.207.2.2 virtual HandleId::~~HandleId ( ) `[inline]`, `[virtual]`

Definition at line 63 of file HandleId.h.

### 29.207.3 Member Function Documentation

#### 29.207.3.1 void HandleId::decrement ( ) `[inline]`

Definition at line 70 of file HandleId.h.

#### 29.207.3.2 bool HandleId::dynamicObj ( ) const `[inline]`

Definition at line 67 of file HandleId.h.

#### 29.207.3.3 int HandleId::getNoRefs ( ) const `[inline]`

Definition at line 66 of file HandleId.h.

#### 29.207.3.4 void HandleId::increment ( ) `[inline]`

Definition at line 69 of file HandleId.h.

#### 29.207.3.5 bool HandleId::isReferenced ( ) const `[inline]`

Definition at line 65 of file HandleId.h.

29.207.3.6 void HandleId::operator delete ( void \* v )

Definition at line 87 of file HandleId.cpp.

29.207.3.7 void \* HandleId::operator new ( size\_t t )

Definition at line 62 of file HandleId.cpp.

29.207.3.8 void \* HandleId::operator new ( size\_t t, int n, const char \* file, int line )

Definition at line 68 of file HandleId.cpp.

## 29.207.4 Member Data Documentation

29.207.4.1 char HandleId::dynamic\_object [protected]

Definition at line 59 of file HandleId.h.

29.207.4.2 int HandleId::refcount [protected]

Definition at line 58 of file HandleId.h.

The documentation for this class was generated from the following files:

- [ttl/include/ttl/utis/HandleId.h](#)
- [ttl/src/utis/HandleId.cpp](#)

## 29.208 HeapNode Class Reference

```
#include <PrDijkstra.h>
```

### Public Member Functions

- [HeapNode \(\)](#)  
*Constructor.*
- [HeapNode \(int idx, double key\)](#)  
*Constructor.*
- [~HeapNode \(\)](#)  
*Destructor.*
- [bool operator> \(const HeapNode &x\) const](#)
- [bool operator>= \(const HeapNode &x\) const](#)

## Public Attributes

- [int idx\\_](#)
- [double key\\_](#)

### 29.208.1 Detailed Description

Definition at line 53 of file PrDijkstra.h.

### 29.208.2 Constructor & Destructor Documentation

#### 29.208.2.1 HeapNode::HeapNode ( ) `[inline]`

Constructor.

Definition at line 60 of file PrDijkstra.h.

#### 29.208.2.2 HeapNode::HeapNode ( int *idx*, double *key* ) `[inline]`

Constructor.

Definition at line 62 of file PrDijkstra.h.

#### 29.208.2.3 HeapNode::~HeapNode ( ) `[inline]`

Destructor.

Definition at line 64 of file PrDijkstra.h.

### 29.208.3 Member Function Documentation

#### 29.208.3.1 bool HeapNode::operator> ( const HeapNode & x ) const `[inline]`

Definition at line 66 of file PrDijkstra.h.

#### 29.208.3.2 bool HeapNode::operator>= ( const HeapNode & x ) const `[inline]`

Definition at line 67 of file PrDijkstra.h.

### 29.208.4 Member Data Documentation

#### 29.208.4.1 int HeapNode::idx\_

Definition at line 56 of file PrDijkstra.h.

#### 29.208.4.2 double HeapNode::key\_

Definition at line 57 of file PrDijkstra.h.

The documentation for this class was generated from the following file:

- parametrization/include/GoTools/parametrization/PrDijkstra.h

## 29.209 HeapNode2 Class Reference

```
#include <PrMultiDijkstra.h>
```

### Public Member Functions

- [HeapNode2](#) ()
- [HeapNode2](#) (int *idx*, double *key*)
- [~HeapNode2](#) ()
- [bool operator>](#) (const [HeapNode2](#) &x) const
- [bool operator>=](#) (const [HeapNode2](#) &x) const

### Public Attributes

- int [idx\\_](#)
- double [key\\_](#)

### 29.209.1 Detailed Description

[HeapNode2](#) - Short description. Detailed description.

Definition at line 56 of file PrMultiDijkstra.h.

### 29.209.2 Constructor & Destructor Documentation

#### 29.209.2.1 [HeapNode2::HeapNode2](#) ( ) [inline]

Definition at line 62 of file PrMultiDijkstra.h.

#### 29.209.2.2 [HeapNode2::HeapNode2](#) ( int *idx*, double *key* ) [inline]

Definition at line 63 of file PrMultiDijkstra.h.

#### 29.209.2.3 [HeapNode2::~~HeapNode2](#) ( ) [inline]

Definition at line 64 of file PrMultiDijkstra.h.



### 29.209.3 Member Function Documentation

29.209.3.1 `bool HeapNode2::operator> ( const HeapNode2 & x ) const` `[inline]`

Definition at line 66 of file PrMultiDijkstra.h.

29.209.3.2 `bool HeapNode2::operator>= ( const HeapNode2 & x ) const` `[inline]`

Definition at line 67 of file PrMultiDijkstra.h.

### 29.209.4 Member Data Documentation

29.209.4.1 `int HeapNode2::idx_`

Definition at line 59 of file PrMultiDijkstra.h.

29.209.4.2 `double HeapNode2::key_`

Definition at line 60 of file PrMultiDijkstra.h.

The documentation for this class was generated from the following file:

- parametrization/include/GoTools/parametrization/PrMultiDijkstra.h

## 29.210 Go::HermiteAppC Class Reference

```
#include <HermiteAppC.h>
```

### Public Member Functions

- [HermiteAppC](#) ([EvalCurve](#) \*crv, [double tolerance1](#), [double tolerance2](#))
- [HermiteAppC](#) ([EvalCurve](#) \*crv, [double](#) initpars[], [int](#) n, [double tolerance1](#), [double tolerance2](#))
- [~HermiteAppC](#) ()  
*Empty destructor.*
- void [refineApproximation](#) ()
- [shared\\_ptr](#)< [SplineCurve](#) > [getCurve](#) ()

### 29.210.1 Detailed Description

This class is used to generate a [SplineCurve](#) from a [EvalCurve](#) using Hermite interpolation. The generated curve will approximate the [EvalCurve](#) within specified tolerances.

Definition at line 55 of file HermiteAppC.h.

### 29.210.2 Constructor & Destructor Documentation

29.210.2.1 `Go::HermiteAppC::HermiteAppC ( EvalCurve * crv, double tolerance1, double tolerance2 )`

Constructor where the tolerances and the curve to approximate are specified.

## Parameters

<i>crv</i>	the curve that we want to generate a Hermite approximation of. The curve is <i>not</i> copied, only pointed to by the <a href="#">HermiteAppC</a> .
<i>tolerance1</i>	the required geometrical accuracy of approximation
<i>tolerance2</i>	another tolerance, used for some kinds of EvalCurves

**29.210.2.2** `Go::HermiteAppC::HermiteAppC ( EvalCurve * crv, double initpars[], int n, double tolerance1, double tolerance2 )`

Constructor where the tolerances and the curve to approximate are specified, as well as the parameters for which we will sample the input curve before starting the approximating process.

## Parameters

<i>crv</i>	the curve that we want to generate a Hermite approximation of. The curve is <i>not</i> copied, only pointed to by the <a href="#">HermiteAppC</a> .
<i>initpars</i>	pointer to the array of parameter values for which we will sample the input curve.
<i>n</i>	number of parameter values in the array 'initpars'.
<i>tolerance1</i>	the required geometrical accuracy of approximation
<i>tolerance2</i>	another tolerance, used for some kinds of EvalCurves

**29.210.2.3** `Go::HermiteAppC::~~HermiteAppC ( ) [inline]`

Empty destructor.

Definition at line 80 of file HermiteAppC.h.

**29.210.3 Member Function Documentation**

**29.210.3.1** `shared_ptr<SplineCurve> Go::HermiteAppC::getCurve ( )`

Return the cubic spline curve Hermite interpolating the grid.

## Returns

the cubic spline curve that Hermite interpolates the sampled points of the [EvalCurve](#) specified in the constructor.

**29.210.3.2** `void Go::HermiteAppC::refineApproximation ( )`

Refine the internal sampling of the curve to approximate such that the Hermite interpolated curve of this sampling approximates parametrically the original curve within the specified tolerance.

The documentation for this class was generated from the following file:

- [gotools-core/include/GoTools/creators/HermiteAppC.h](#)

## 29.211 Go::HermiteApprEvalSurf Class Reference

```
#include <HermiteApprEvalSurf.h>
```

### Public Member Functions

- [HermiteApprEvalSurf](#) ([EvalSurface](#) \*sf, [double](#) tolerance1, [double](#) tolerance2)
- [HermiteApprEvalSurf](#) ([EvalSurface](#) \*sf, [double](#) initpars\_u[], [int](#) mm, [double](#) initpars\_v[], [int](#) nn, [double](#) tolerance1, [double](#) tolerance2)
- [~HermiteApprEvalSurf](#) ()  
*Empty destructor.*
- void [refineApproximation](#) ()
- [shared\\_ptr](#)< [SplineSurface](#) > [getSurface](#) ([bool](#) &method\_failed)
- [const](#) [HermiteGrid2D](#) & [getGrid](#) () [const](#)
- void [removeGridLines](#) ([const](#) [std::vector](#)< [double](#) > &grid\_lines\_u, [const](#) [std::vector](#)< [double](#) > &grid\_lines\_v)
- void [setNoSplit](#) ([const](#) [std::vector](#)< [shared\\_ptr](#)< [SplineCurve](#) > > &no\_split\_cvs\_2d, [double](#) no\_split\_dist\_2d)

### 29.211.1 Detailed Description

This class is used to generate a [SplineSurface](#) from a [EvalSurface](#) using Hermite interpolation. The generated surface will approximate the [EvalSurface](#) within specified tolerances.

Definition at line 57 of file [HermiteApprEvalSurf.h](#).

### 29.211.2 Constructor & Destructor Documentation

#### 29.211.2.1 Go::HermiteApprEvalSurf::HermiteApprEvalSurf ( [EvalSurface](#) \* sf, [double](#) tolerance1, [double](#) tolerance2 )

Constructor where the tolerances and the surface to approximate are specified.

#### Parameters

<i>sf</i>	the surface that we want to generate a Hermite approximation of. The surface is <i>not</i> copied, only pointed to by the <a href="#">HermiteApprEvalSurf</a> .
<i>tolerance1</i>	the required geometrical accuracy of approximation
<i>tolerance2</i>	another tolerance, used for some kinds of <a href="#">EvalSurfaces</a>

#### 29.211.2.2 Go::HermiteApprEvalSurf::HermiteApprEvalSurf ( [EvalSurface](#) \* sf, [double](#) initpars\_u[], [int](#) mm, [double](#) initpars\_v[], [int](#) nn, [double](#) tolerance1, [double](#) tolerance2 )

Constructor where the tolerances and the surface to approximate are specified, as well as the parameters for which we will sample the input surface before starting the approximating process.

## Parameters

<i>sf</i>	the surface that we want to generate a Hermite approximation of. The surface is <i>not</i> copied, only pointed to by the <a href="#">HermiteApprEvalSurf</a> .
<i>initpars</i>	pointer to the array of parameter values for which we will sample the input surface.
<i>n</i>	number of parameter values in the array 'initpars'.
<i>tolerance1</i>	the required geometrical accuracy of approximation
<i>tolerance2</i>	another tolerance, used for some kinds of EvalSurfaces

29.211.2.3 `Go::HermiteApprEvalSurf::~~HermiteApprEvalSurf ( ) [inline]`

Empty destructor.

Definition at line 86 of file `HermiteApprEvalSurf.h`.

### 29.211.3 Member Function Documentation

29.211.3.1 `const HermiteGrid2D& Go::HermiteApprEvalSurf::getGrid ( ) const`

29.211.3.2 `shared_ptr<SplineSurface> Go::HermiteApprEvalSurf::getSurface ( bool & method_failed )`

Return the cubic spline surface Hermite interpolating the grid.

#### Returns

the cubic spline surface that Hermite interpolates the sampled points of the [EvalSurface](#) specified in the constructor.

29.211.3.3 `void Go::HermiteApprEvalSurf::refineApproximation ( )`

Refine the internal sampling of the surface to approximate such that the Hermite interpolated surface of this sampling approximates parametrically the original surface within the specified tolerance.

29.211.3.4 `void Go::HermiteApprEvalSurf::removeGridLines ( const std::vector< double > & grid_lines_u, const std::vector< double > & grid_lines_v )`

29.211.3.5 `void Go::HermiteApprEvalSurf::setNoSplit ( const std::vector< shared_ptr< SplineCurve > > & no_split_cvs_2d, double no_split_dist_2d )`

The documentation for this class was generated from the following file:

- `compositemodel/include/GoTools/compositemodel/HermiteApprEvalSurf.h`

## 29.212 Go::HermiteAppS Class Reference

```
#include <HermiteAppS.h>
```

### Public Member Functions

- [HermiteAppS](#) ([EvalCurveSet](#) \**surface*, *double tolerance1*, *double tolerance2*, [std::vector](#)< int > *dims*)
- [HermiteAppS](#) ([EvalCurveSet](#) \**surface*, *double* *initpars*[], *int n*, *double tolerance1*, *double tolerance2*, [std::vector](#)< int > *dims*)
- [~HermiteAppS](#) ()  
*Empty destructor.*
- [void refineApproximation](#) ()
- [std::vector](#)< [shared\\_ptr](#)< [SplineCurve](#) > > [getCurves](#) ()

### 29.212.1 Detailed Description

This class is used to generate a set of [SplineCurves](#) from a [EvalCurveSet](#) (which itself represents a set of related curves) using Hermite interpolation. The generated curves will approximate those defined by the [EvalCurveSet](#) within specified tolerances. This class is really a generalization of [HermiteAppC](#).

Definition at line 56 of file [HermiteAppS.h](#).

### 29.212.2 Constructor & Destructor Documentation

**29.212.2.1** [Go::HermiteAppS::HermiteAppS](#) ( [EvalCurveSet](#) \* *surface*, *double tolerance1*, *double tolerance2*, [std::vector](#)< int > *dims* )

Constructor where the tolerances and the curves to approximate are specified.

#### Parameters

<i>surface</i>	the curve set that we want to approximate by Hermite interpolation of sampled values.
<i>tolerance1</i>	the required geometrical accuracy of approximation
<i>tolerance2</i>	another tolerance, used for some kinds of <a href="#">EvalCurveSets</a> .
<i>dims</i>	vector that specifies the spatial dimensions of the curves contained in the <a href="#">EvalCurveSet</a> . The size of the vector should be equal to the total number of curves in 'surf', ie. the return value of its <a href="#">EvalCurveSet::nmbCvs()</a> function.

**29.212.2.2** [Go::HermiteAppS::HermiteAppS](#) ( [EvalCurveSet](#) \* *surface*, *double* *initpars*[], *int n*, *double tolerance1*, *double tolerance2*, [std::vector](#)< int > *dims* )

Constructor where the tolerances and the curve set to approximate are specified, as well as the parameters where they should be sampled prior to the Hermite interpolation.

#### Parameters

<i>surface</i>	the curve set that we want to approximate by Hermite interpolation of sampled values.
----------------	---------------------------------------------------------------------------------------

## Parameters

<i>initpars</i>	pointer to the array of parameter values for which we will sample the input curves.
<i>n</i>	number of parameter values in the array 'initpars[]'.
<i>tolerance1</i>	the required geometrical accuracy of approximation
<i>tolerance2</i>	another tolerance, used for some kinds of EvalCurveSets.
<i>dims</i>	vector that specifies the spatial dimensions of the curves contained in the <a href="#">EvalCurveSet</a> . The size of the vector should be equal to the total number of curves in 'surf', ie. the return value of its <a href="#">EvalCurveSet::nmbCvs()</a> function.

### 29.212.2.3 Go::HermiteAppS::~HermiteAppS ( ) [inline]

Empty destructor.

Definition at line 92 of file HermiteAppS.h.

### 29.212.3 Member Function Documentation

#### 29.212.3.1 std::vector<shared\_ptr<SplineCurve>> Go::HermiteAppS::getCurves ( )

Return the cubic spline curves interpolating the grid (ie. approximating the original curve set).

#### Returns

a vector containing shared pointers to the newly created spline curves that Hermite interpolate the sampled points of the [EvalCurveSet](#) (curve set) specified in the constructor.

#### 29.212.3.2 void Go::HermiteAppS::refineApproximation ( )

Refine the internal sampling of the curves such that the Hermite interpolated curves parametrically approximates the original curve within a specified tolerance.

The documentation for this class was generated from the following file:

- [gtools-core/include/GoTools/creators/HermiteAppS.h](#)

### 29.213 Go::HermiteGrid1D Class Reference

```
#include <HermiteGrid1D.h>
```

## Public Member Functions

- [HermiteGrid1D](#) ([const EvalCurve](#) &crv, [double](#) start, [double](#) end)
- [HermiteGrid1D](#) ([const EvalCurve](#) &crv, [double](#) param[], [int](#) n)
- [~HermiteGrid1D](#) ()  
*Default destructor.*
- [int addKnot](#) ([const EvalCurve](#) &crv, [double](#) knot)
- [void getSegment](#) ([int](#) left, [int](#) right, [double](#) &spar, [double](#) &epar, [Point](#) bezcoef[4])
- [std::vector< double > getKnots](#) ()  
*Return the grid parameters.*
- [std::vector< Point > getData](#) ()  
*Return the sample values (positions and first derivatives)*
- [int dim](#) ()  
*Return the spatial dimension.*
- [int size](#) ()  
*Return the number of samples in the grid.*

### 29.213.1 Detailed Description

The type "GoHemiteGrid1D" holds a one dimensional grid containing sampled points and derivatives from a curve. It can be used to generate bezier curve segments obtained by Hermite interpolation of intervals between the sampled parameter values.

Definition at line 56 of file HermiteGrid1D.h.

### 29.213.2 Constructor & Destructor Documentation

#### 29.213.2.1 Go::HermiteGrid1D::HermiteGrid1D ( [const EvalCurve](#) & crv, [double](#) start, [double](#) end )

Construct a [HermiteGrid1D](#) from a curve and a given interval. The grid will only contain the sampled values (position, derivative) for the first and last value of the interval.

##### Parameters

<i>crv</i>	curve to sample
<i>start</i>	start of interval
<i>end</i>	end of interval

#### 29.213.2.2 Go::HermiteGrid1D::HermiteGrid1D ( [const EvalCurve](#) & crv, [double](#) param[], [int](#) n )

Construct a [HermiteGrid1D](#) from a curve and a set of parameter values.

##### Parameters

<i>crv</i>	curve to sample
<i>param</i>	array of strictly increasing parameters contained in the parameter domain of 'crv'.
<i>n</i>	number of elements in 'param[]'.

### 29.213.2.3 Go::HermiteGrid1D::~~HermiteGrid1D ( )

Default destructor.

## 29.213.3 Member Function Documentation

### 29.213.3.1 int Go::HermiteGrid1D::addKnot ( const EvalCurve & *crv*, double *knot* )

Add another sample (parameter, position, tangent) to the grid. Returns the index of the new knot (parameter value) in the sorted knot vector after insertion.

#### Parameters

<i>crv</i>	curve to evaluate
<i>knot</i>	the new sample value (parameter value, knot)

### 29.213.3.2 int Go::HermiteGrid1D::dim ( ) [inline]

Return the spatial dimension.

Definition at line 104 of file HermiteGrid1D.h.

### 29.213.3.3 std::vector<Point> Go::HermiteGrid1D::getData ( ) [inline]

Return the sample values (positions and first derivatives)

Definition at line 101 of file HermiteGrid1D.h.

### 29.213.3.4 std::vector<double> Go::HermiteGrid1D::getKnots ( ) [inline]

Return the grid parameters.

Definition at line 98 of file HermiteGrid1D.h.

### 29.213.3.5 void Go::HermiteGrid1D::getSegment ( int *left*, int *right*, double & *spar*, double & *epar*, Point *bezcoef*[4] )

Calculate Bezier coefficients of the cubic curve interpolating the point and tangent values at grid nodes with indices "left" and "right".

#### Parameters

<i>left</i>	indicating grid node for start of curve segment
<i>right</i>	indicating grid node for end of curve segment
<i>spar</i>	start parameter of segment
<i>epar</i>	end parameter of segment
<i>bezcoef</i>	array of cubic Bezier coefficients



29.213.3.6 `int Go::HermiteGrid1D::size ( ) [inline]`

Return the number of samples in the grid.

Definition at line 107 of file HermiteGrid1D.h.

The documentation for this class was generated from the following file:

- [gotools-core/include/GoTools/creators/HermiteGrid1D.h](#)

## 29.214 Go::HermiteGrid1DMulti Class Reference

```
#include <HermiteGrid1DMulti.h>
```

### Public Member Functions

- [HermiteGrid1DMulti](#) ([EvalCurveSet](#) &surf, [double](#) start, [double](#) end, [std::vector< int >](#) dims)
- [HermiteGrid1DMulti](#) ([EvalCurveSet](#) &surf, [double](#) param[], [int](#) n, [std::vector< int >](#) dims)
- [~HermiteGrid1DMulti](#) ()  
*Default destructor.*
- [int](#) [addKnot](#) ([EvalCurveSet](#) &surf, [double](#) knot)
- [void](#) [getSegment](#) ([int](#) left, [int](#) right, [double](#) &spar, [double](#) &epar, [std::vector< std::vector< Point > >](#) &bezcoef)
- [std::vector< double >](#) [getKnots](#) ()  
*Return the grid parameters.*
- [std::vector< std::vector< Point > >](#) [getData](#) ()  
*Return the sample values (positions and first derivatives)*
- [int](#) [dims](#) ([int](#) ki)  
*Return the spatial dimension of each of the curves.*
- [int](#) [size](#) ()  
*Return the number of samples in the grid.*

### 29.214.1 Detailed Description

The type "GoHemiteGrid1DMulti" holds a one dimensional grid containing sampled points and derivatives from a set of curves, as represented by a [EvalCurveSet](#). It can be used to generate bezier curve segments obtained by Hermite interpolation of intervals between the sampled parameter values.

Definition at line 56 of file HermiteGrid1DMulti.h.

### 29.214.2 Constructor & Destructor Documentation

29.214.2.1 `Go::HermiteGrid1DMulti::HermiteGrid1DMulti ( EvalCurveSet & surf, double start, double end, std::vector< int > dims )`

Construct a [HermiteGrid1DMulti](#) from a set of related curves (represented by a [EvalCurveSet](#)) and a given interval.

## Parameters

<i>surf</i>	the curve collection to sample from
<i>start</i>	start of parameter interval
<i>end</i>	end of parameter interval
<i>dims</i>	vector that specifies the dimensions of the curves contained in the <a href="#">EvalCurveSet</a> . The size of the vector should be equal to the total number of curves in 'surf', ie. the return value of its <a href="#">EvalCurveSet::nmbCvs()</a> function.

29.214.2.2 `Go::HermiteGrid1DMulti::HermiteGrid1DMulti ( EvalCurveSet & surf, double param[], int n, std::vector< int > dims )`

Construct a [HermiteGrid1DMulti](#) from a set of related curves (represented by a [EvalCurveSet](#)) and a set of parameter values.

## Parameters

<i>surf</i>	the curve collection to sample from
<i>param</i>	array of strictly increasing parameters contained in the parameter domain of 'surf'.
<i>n</i>	number of elements in 'param[]'
<i>dims</i>	vector that specifies the dimensions of the curves contained in the <a href="#">EvalCurveSet</a> . The size of the vector should be equal to the total number of curves in 'surf', ie. the return value of its <a href="#">EvalCurveSet::nmbCvs()</a> function.

29.214.2.3 `Go::HermiteGrid1DMulti::~~HermiteGrid1DMulti ( )`

Default destructor.

## 29.214.3 Member Function Documentation

29.214.3.1 `int Go::HermiteGrid1DMulti::addKnot ( EvalCurveSet & surf, double knot )`

Add another sample (parameter, positions, tangents) to the grid. Returns the index of the new knot (parameter value) in the sorted knot vector after insertion.

## Parameters

<i>surf</i>	the curve collection to sample from
<i>knot</i>	the new sample value (parameter value, knot)

29.214.3.2 `int Go::HermiteGrid1DMulti::dims ( int ki ) [inline]`

Return the spatial dimension of each of the curves.

Definition at line 114 of file [HermiteGrid1DMulti.h](#).

29.214.3.3 `std::vector<std::vector<Point>>` `Go::HermiteGrid1DMulti::getData ( )` `[inline]`

Return the sample values (positions and first derivatives)

Definition at line 111 of file `HermiteGrid1DMulti.h`.

29.214.3.4 `std::vector<double>` `Go::HermiteGrid1DMulti::getKnots ( )` `[inline]`

Return the grid parameters.

Definition at line 107 of file `HermiteGrid1DMulti.h`.

29.214.3.5 `void Go::HermiteGrid1DMulti::getSegment ( int left, int right, double & spar, double & epar, std::vector<std::vector< Point >> & bezcoef )`

Calculate Bezier coefficients of the cubic curves interpolating the points and tangents at the grid nodes with indices 'left' and 'right'.

#### Parameters

<i>left</i>	indicating grid node for start of curve segment
<i>right</i>	indicating grid node for end of curve segment
<i>spar</i>	start parameter of segment
<i>epar</i>	end parameter of segment
<i>bezcoef</i>	a vector containing arrays of cubic Bezier coefficients. There are as many vector entries as there are curves, and each entry contains the Bezier coefficients for that curve.

29.214.3.6 `int Go::HermiteGrid1DMulti::size ( )` `[inline]`

Return the number of samples in the grid.

Definition at line 117 of file `HermiteGrid1DMulti.h`.

The documentation for this class was generated from the following file:

- `gtools-core/include/GoTools/creators/HermiteGrid1DMulti.h`

## 29.215 Go::HermiteGrid2D Class Reference

```
#include <HermiteGrid2D.h>
```

## Public Member Functions

- [HermiteGrid2D](#) ([const EvalSurface](#) &sf, [double](#) u1, [double](#) u2, [double](#) v1, [double](#) v2)
- [HermiteGrid2D](#) ([const EvalSurface](#) &sf, [double](#) param\_u[], [double](#) param\_v[], [int](#) mm, [int](#) nn)
- [~HermiteGrid2D](#) ()
  - Default destructor.*
- [int addKnot](#) ([const EvalSurface](#) &sf, [double](#) knot, [bool](#) dir\_is\_u)
- [void getSegment](#) ([int](#) left1, [int](#) right1, [int](#) left2, [int](#) right2, [double](#) &spar1, [double](#) &epar1, [double](#) &spar2, [double](#) &epar2, [Point](#) bezcoef[4])
- [std::vector< double > getKnots](#) ([bool](#) dir\_is\_u) [const](#)
  - Return the grid parameters.*
- [std::vector< Point > getData](#) () [const](#)
  - Return the sample values (positions and first derivatives)*
- [int dim](#) () [const](#)
  - Return the spatial dimension.*
- [int size1](#) () [const](#)
  - Return the number of samples in the grid.*
- [int size2](#) () [const](#)
- [void removeGridLines](#) ([const](#) [std::vector< int >](#) &grid\_lines\_u, [const](#) [std::vector< int >](#) &grid\_lines\_v)
- [int getNoSplitStatus](#) ([int](#) ind\_u, [int](#) ind\_v)
- [void setNoSplitStatus](#) ([int](#) ind\_u, [int](#) ind\_v, [int](#) no\_split\_status)

### 29.215.1 Detailed Description

The type "GoHermiteGrid1D" holds a one dimensional grid containing sampled points and derivatives from a curve. It can be used to generate bezier curve segments obtained by Hermite interpolation of intervals between the sampled parameter values.

Definition at line 57 of file HermiteGrid2D.h.

### 29.215.2 Constructor & Destructor Documentation

29.215.2.1 [Go::HermiteGrid2D::HermiteGrid2D](#) ( [const EvalSurface](#) &sf, [double](#) u1, [double](#) u2, [double](#) v1, [double](#) v2 )

Construct a [HermiteGrid1D](#) from a curve and a given interval. The grid will only contain the sampled values (position, derivative) for the first and last value of the interval.

#### Parameters

<i>crv</i>	curve to sample
<i>start</i>	start of interval
<i>end</i>	end of interval

29.215.2.2 [Go::HermiteGrid2D::HermiteGrid2D](#) ( [const EvalSurface](#) &sf, [double](#) param\_u[], [double](#) param\_v[], [int](#) mm, [int](#) nn )

Construct a [HermiteGrid1D](#) from a curve and a set of parameter values.

## Parameters

<i>crv</i>	curve to sample
<i>param</i>	array of strictly increasing parameters contained in the parameter domain of 'crv'.
<i>n</i>	number of elements in 'param[]'.

## 29.215.2.3 Go::HermiteGrid2D::~~HermiteGrid2D ( )

Default destructor.

## 29.215.3 Member Function Documentation

## 29.215.3.1 int Go::HermiteGrid2D::addKnot ( const EvalSurface &amp; sf, double knot, bool dir\_is\_u )

Add another sample (parameter, position, tangent) to the grid. Returns the index of the new knot (parameter value) in the sorted knot vector after insertion.

## Parameters

<i>crv</i>	curve to evaluate
<i>knot</i>	the new sample value (parameter value, knot)

## 29.215.3.2 int Go::HermiteGrid2D::dim ( ) const [inline]

Return the spatial dimension.

Definition at line 110 of file HermiteGrid2D.h.

## 29.215.3.3 std::vector&lt;Point&gt; Go::HermiteGrid2D::getData ( ) const [inline]

Return the sample values (positions and first derivatives)

Definition at line 107 of file HermiteGrid2D.h.

## 29.215.3.4 std::vector&lt;double&gt; Go::HermiteGrid2D::getKnots ( bool dir\_is\_u ) const [inline]

Return the grid parameters.

Definition at line 104 of file HermiteGrid2D.h.

## 29.215.3.5 int Go::HermiteGrid2D::getNoSplitStatus ( int ind\_u, int ind\_v )

## 29.215.3.6 void Go::HermiteGrid2D::getSegment ( int left1, int right1, int left2, int right2, double &amp; spar1, double &amp; epar1, double &amp; spar2, double &amp; epar2, Point bezcoef[4] )

Calculate Bezier coefficients of the cubic curve interpolating the point and tangent values at grid nodes with indices "left" and "right".

## Parameters

<i>left</i>	indicating grid node for start of curve segment
<i>right</i>	indicating grid node for end of curve segment
<i>spar</i>	start parameter of segment
<i>epar</i>	end parameter of segment
<i>bezcoef</i>	array of cubic Bezier coefficients

29.215.3.7 void Go::HermiteGrid2D::removeGridLines ( const std::vector< int > & *grid\_lines\_u*, const std::vector< int > & *grid\_lines\_v* )

29.215.3.8 void Go::HermiteGrid2D::setNoSplitStatus ( int *ind\_u*, int *ind\_v*, int *no\_split\_status* )

29.215.3.9 int Go::HermiteGrid2D::size1 ( ) const [inline]

Return the number of samples in the grid.

Definition at line 113 of file HermiteGrid2D.h.

29.215.3.10 int Go::HermiteGrid2D::size2 ( ) const [inline]

Definition at line 115 of file HermiteGrid2D.h.

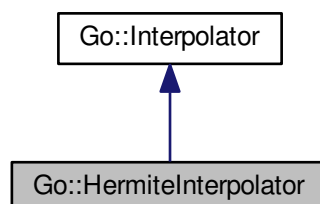
The documentation for this class was generated from the following file:

- [gotools-core/include/GoTools/creators/HermiteGrid2D.h](#)

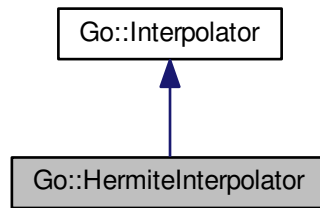
## 29.216 Go::HermiteInterpolator Class Reference

```
#include <HermiteInterpolator.h>
```

Inheritance diagram for Go::HermiteInterpolator:



Collaboration diagram for Go::HermitInterpolator:



## Public Member Functions

- [HermitInterpolator](#) ()  
*Constructor takes no arguments.*
- virtual [~HermitInterpolator](#) ()  
*Virtual destructor enables safe inheritance.*
- virtual [const BsplineBasis & basis](#) ()
- virtual void [interpolate](#) (int num\_points, int dimension, [const double](#) \*param\_start, [const double](#) \*data\_start, [std::vector< double >](#) &coefs)
- void [interpolate](#) ([const std::vector< Point >](#) &data, [const std::vector< double >](#) &param, [std::vector< double >](#) &coefs)

### 29.216.1 Detailed Description

An [Interpolator](#) that generates a hermite spline curve through given points and tangents.

Definition at line 55 of file `HermitInterpolator.h`.

### 29.216.2 Constructor & Destructor Documentation

#### 29.216.2.1 `Go::HermitInterpolator::HermitInterpolator ( )` [`inline`]

Constructor takes no arguments.

Definition at line 59 of file `HermitInterpolator.h`.

#### 29.216.2.2 `virtual Go::HermitInterpolator::~~HermitInterpolator ( )` [`virtual`]

Virtual destructor enables safe inheritance.

### 29.216.3 Member Function Documentation

#### 29.216.3.1 virtual const BsplineBasis& Go::HermitelInterpolator::basis ( ) [virtual]

after the function [interpolate\(\)](#) has been successfully run, this function can be called to get the [BsplineBasis](#) of the generated curve.

##### Returns

a constant reference to the [BsplineBasis](#) of the curve previously generated by [interpolate\(\)](#).

Implements [Go::Interpolator](#).

#### 29.216.3.2 virtual void Go::HermitelInterpolator::interpolate ( int num\_points, int dimension, const double \* param\_start, const double \* data\_start, std::vector< double > & coefs ) [virtual]

Hermite interpolation of a sequence of points with associated tangents and parameter values

##### Parameters

<i>num_points</i>	number of points to interpolate
<i>dimension</i>	dimension of points to interpolate (2D, 3D, etc..)
<i>param_start</i>	pointer to the start of the array where the parameter values of the points are stored. This should be a strictly increasing sequence of 'num_points' values.
<i>data_start</i>	pointer to the start of the array where the points and tangents to be interpolated are stored. Each point and tangent consist of 'dimension' coordinates, and each tangent is stored immediately after its corresponding point.

##### Return values

<i>coefs</i>	The control points of the computed hermite interpolation curve will be returned in this vector. (Use the <a href="#">basis()</a> function to get the associated b-spline basis).
--------------	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Implements [Go::Interpolator](#).

#### 29.216.3.3 void Go::HermitelInterpolator::interpolate ( const std::vector< Point > & data, const std::vector< double > & param, std::vector< double > & coefs )

Hermite interpolation of a sequence of points with associated tangents and parameter values.

##### Parameters

<i>data</i>	This vector contains the set of points and tangents to be interpolated. The size of the vector is twice the total number of points. Each entry on the form [2i] represents a point, and the entry [2i+1] represents the associated tangent.
<i>param</i>	This vector represent the parameterization of the points, and should have one entry per point. (Making it half the size of the 'data' vector). The parameter sequence must be strictly increasing.



## Return values

<i>coefs</i>	The control points of the computed hermite interpolation curve will be returned in this vector. (Use the <a href="#">basis()</a> function to get the associated b-spline basis.
--------------	---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

The documentation for this class was generated from the following file:

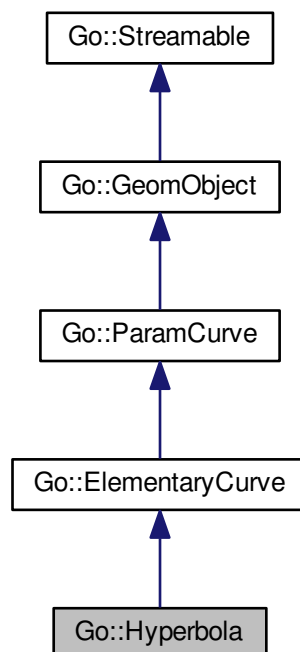
- [gotools-core/include/GoTools/geometry/HermiteInterpolator.h](#)

## 29.217 Go::Hyperbola Class Reference

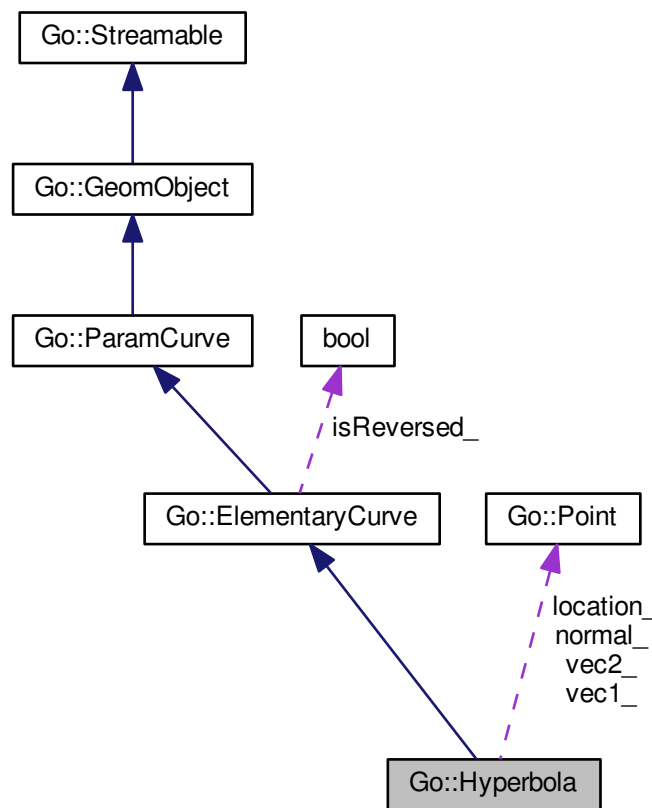
Class that represents a hyperbola. It is a subclass of [ElementaryCurve](#) and thus has a parametrization.

```
#include <Hyperbola.h>
```

Inheritance diagram for Go::Hyperbola:



Collaboration diagram for Go::Hyperbola:



## Public Member Functions

- [Hyperbola](#) ()
- [Hyperbola](#) ([Point](#) location, [Point](#) direction, [Point](#) normal, [double](#) r1, [double](#) r2, [bool](#) isReversed=false)
- virtual [~Hyperbola](#) ()
  - virtual destructor - ensures safe inheritance*
- virtual void [read](#) (std::istream &is)
- virtual void [write](#) (std::ostream &os) **const**
- virtual [BoundingBox](#) boundingBox () **const**
  - Return the object's bounding box.*
- virtual int [dimension](#) () **const**
  - Return the dimension of the space in which the object lies (usually 2 or 3)*
- virtual [ClassType](#) instanceType () **const**
  - Return the class type identifier of a given, derived instance of [GeomObject](#).*
- virtual [Hyperbola](#) \* [clone](#) () **const**
- virtual void [point](#) ([Point](#) &pt, [double](#) tpar) **const**
- virtual void [point](#) (std::vector< [Point](#) > &pts, [double](#) tpar, int derivs, [bool](#) from\_right=true) **const**
- virtual [double](#) startparam () **const**
- virtual [double](#) endparam () **const**

- virtual void [setParameterInterval](#) (double t1, double t2)  
*Limit the curve by limiting the parameter interval.*
- virtual [SplineCurve](#) \* [geometryCurve](#) ()
- virtual [SplineCurve](#) \* [createSplineCurve](#) () const  
*Fetch spline representation of curve.*
- virtual [bool](#) [isDegenerate](#) (double degenerate\_epsilon)
- virtual [Hyperbola](#) \* [subCurve](#) (double from\_par, double to\_par, double fuzzy=DEFAULT\_PARAMETER\_EPSILON) const
- virtual [DirectionCone](#) [directionCone](#) () const
- virtual void [appendCurve](#) ([ParamCurve](#) \*cv, [bool](#) reparam=true)
- virtual void [appendCurve](#) ([ParamCurve](#) \*cv, int continuity, [double](#) &dist, [bool](#) reparam=true)
- virtual void [closestPoint](#) (const [Point](#) &pt, [double](#) tmin, [double](#) tmax, [double](#) &clo\_t, [Point](#) &clo\_pt, [double](#) &clo\_dist, [double](#) const \*seed=0) const
- virtual [double](#) [length](#) ([double](#) tol)
- virtual void [setParamBounds](#) ([double](#) startpar, [double](#) endpar)
- virtual void [translateCurve](#) (const [Point](#) &dir)
- [bool](#) [isBounded](#) () const
- virtual void [swapParameters2D](#) ()
- virtual [bool](#) [isInPlane](#) (const [Point](#) &norm, [double](#) eps, [Point](#) &pos) const  
*Check if the hyperbola lies in a plane with a given normal.*

### Static Public Member Functions

- static [ClassType](#) [classType](#) ()

### Protected Member Functions

- void [setSpanningVectors](#) ()

### Protected Attributes

- [Point](#) [location\\_](#)
- [Point](#) [vec1\\_](#)
- [Point](#) [vec2\\_](#)
- [Point](#) [normal\\_](#)
- [double](#) [r1\\_](#)
- [double](#) [r2\\_](#)
- [double](#) [startparam\\_](#)
- [double](#) [endparam\\_](#)

#### 29.217.1 Detailed Description

Class that represents a hyperbola. It is a subclass of [ElementaryCurve](#) and thus has a parametrization.

A hyperbola has a natural parametrization in terms of its values:  $f(t) = C + (R_1 * \cosh(t))*x + (R_2 * \sinh(t)) * y$ . This parametrization is unbounded:  $-\infty < t < \infty$ .

Definition at line 59 of file [Hyperbola.h](#).

## 29.217.2 Constructor & Destructor Documentation

### 29.217.2.1 `Go::Hyperbola::Hyperbola ( )` `[inline]`

Default constructor. Constructs an uninitialized [Hyperbola](#) which can only be assigned to or read into.

Definition at line 64 of file `Hyperbola.h`.

### 29.217.2.2 `Go::Hyperbola::Hyperbola ( Point location, Point direction, Point normal, double r1, double r2, bool isReversed = false )`

Constructor. Input is location, direction (x-axis), normal and the two semi-axes. The default hyperbola is unbounded in its parametrization. To bound it, use [setParamBounds\(\)](#).

### 29.217.2.3 `virtual Go::Hyperbola::~~Hyperbola ( )` `[virtual]`

virtual destructor - ensures safe inheritance

## 29.217.3 Member Function Documentation

### 29.217.3.1 `virtual void Go::Hyperbola::appendCurve ( ParamCurve * cv, bool reparam = true )` `[virtual]`

append a curve to this curve, with eventual reparametrization NB: This virtual member function currently only works for `SplineCurves` and `CurveOnSurfaces`. Moreover, 'this' curve and the 'cv' curve must be of the same type.

#### Parameters

<i>cv</i>	the curve to append to 'this' curve.
<i>reparam</i>	specify whether or not there should be reparametrization

Implements [Go::ParamCurve](#).

### 29.217.3.2 `virtual void Go::Hyperbola::appendCurve ( ParamCurve * cv, int continuity, double & dist, bool reparam = true )` `[virtual]`

append a curve to this curve, with eventual reparametrization

#### Parameters

<i>cv</i>	the curve to append to 'this' curve.
<i>continuity</i>	the required continuity at the transition. Can be $G^{(-1)}$ and upwards.
<i>dist</i>	a measure of the local distortion around the transition in order to achieve the specified continuity.
<i>reparam</i>	specify whether or not there should be reparametrization

Implements [Go::ParamCurve](#).

29.217.3.3 virtual **BoundingBox** Go::Hyperbola::boundingBox ( ) const [virtual]

Return the object's bounding box.

Implements [Go::GeomObject](#).

29.217.3.4 static **ClassType** Go::Hyperbola::classType ( ) [static]

29.217.3.5 virtual **Hyperbola\*** Go::Hyperbola::clone ( ) const [virtual]

The clone-function is herited from [GeomObject](#), but overridden here to get a covariant return type (for those compilers that allow this).

Implements [Go::ElementaryCurve](#).

29.217.3.6 virtual void Go::Hyperbola::closestPoint ( const **Point** & *pt*, double *tmin*, double *tmax*, double & *clo\_t*, **Point** & *clo\_pt*, double & *clo\_dist*, double const \* *seed* = 0 ) const [virtual]

Compute the closest point from an interval of this curve to a specified point.

Parameters

<i>pt</i>	point we want to find the closest point to
<i>tmin</i>	start parameter of search interval
<i>tmax</i>	end parameter of search interval
<i>clo_t</i>	upon function return, 'clo_t' will contain the parameter value of the closest point found.
<i>clo_pt</i>	upon function return, 'clo_pt' will contain the position of the closest point found.
<i>clo_dist</i>	upon function return, 'clo_dist' will contain the distance between 'pt' and the closest point found.
<i>seed</i>	pointer to initial guess value, provided by the user (can be 0, for which the algorithm will determine a (hopefully) reasonable choice).

Implements [Go::ParamCurve](#).

29.217.3.7 virtual **SplineCurve\*** Go::Hyperbola::createSplineCurve ( ) const [virtual]

Fetch spline representation of curve.

Implements [Go::ElementaryCurve](#).

29.217.3.8 virtual int Go::Hyperbola::dimension ( ) const [virtual]

Return the dimension of the space in which the object lies (usually 2 or 3)

Implements [Go::GeomObject](#).

29.217.3.9 virtual **DirectionCone** Go::Hyperbola::directionCone ( ) const [virtual]

Creates a [DirectionCone](#) which covers all tangent directions of this curve.

#### Returns

the smallest [DirectionCone](#) containing all tangent directions of this curve.

Implements [Go::ParamCurve](#).

29.217.3.10 virtual **double** Go::Hyperbola::endparam ( ) const [virtual]

Query the end parameter of the curve

#### Returns

the curve's end parameter

Implements [Go::ParamCurve](#).

29.217.3.11 virtual **SplineCurve\*** Go::Hyperbola::geometryCurve ( ) [virtual]

If the definition of this [ParamCurve](#) contains a [SplineCurve](#) describing its spatial shape, then this function will return a pointer to this [SplineCurve](#). Otherwise it will return a null pointer. The returned curve is NEWed, so the user is responsible for deleting it. This function may have side-effects.

#### Returns

a pointer to a [SplineCurve](#) representation of the [ParamCurve](#), if it exists. Null pointer otherwise.

Implements [Go::ParamCurve](#).

29.217.3.12 virtual **ClassType** Go::Hyperbola::instanceType ( ) const [virtual]

Return the class type identifier of a given, derived instance of [GeomObject](#).

Implements [Go::GeomObject](#).

29.217.3.13 **bool** Go::Hyperbola::isBounded ( ) const

Query if parametrization is bounded. Both upper and lower parameter bounds must be finite for this to be true.

#### Returns

*true* if bounded, *false* otherwise

29.217.3.14 virtual **bool** Go::Hyperbola::isDegenerate ( **double** *degenerate\_epsilon* ) [virtual]

Query whether the curve is degenerate (collapsed into a single point).

## Parameters

<i>degenerate_epsilon</i>	the tolerance used in determine whether the curve is degenerate. A curve is considered degenerate if its total length is shorter than this value.
---------------------------	---------------------------------------------------------------------------------------------------------------------------------------------------

## Returns

`true` if the curve is degenerate, `false` otherwise.

Implements [Go::ParamCurve](#).

29.217.3.15 `virtual bool Go::Hyperbola::isInPlane ( const Point & norm, double eps, Point & pos ) const` [virtual]

Check if the hyperbola lies in a plane with a given normal.

Reimplemented from [Go::ParamCurve](#).

29.217.3.16 `virtual double Go::Hyperbola::length ( double tol )` [virtual]

Compute the total length of this curve up to some tolerance

## Parameters

<i>tol</i>	the relative tolerance when approximating the length, i.e. stop iteration when error becomes smaller than <code>tol/(curve length)</code>
------------	-------------------------------------------------------------------------------------------------------------------------------------------

## Returns

the length calculated

Implements [Go::ParamCurve](#).

29.217.3.17 `virtual void Go::Hyperbola::point ( Point & pt, double tpar ) const` [virtual]

Evaluate the curve's position at a given parameter

## Parameters

<i>pt</i>	the evaluated position will be written to this <a href="#">Point</a>
<i>tpar</i>	the parameter for which we wish to evaluate the curve

Implements [Go::ParamCurve](#).

29.217.3.18 `virtual void Go::Hyperbola::point ( std::vector< Point > & pts, double tpar, int derivs, bool from_right = true ) const [virtual]`

Evaluate the curve's position and a certain number of derivatives at a given parameter.

#### Parameters

<i>pts</i>	the evaluated position and derivatives (tangent, curvature vector, etc.) will be written to this vector. The first entry will be the position, the second entry will be the first derivative, etc. The size of this vector must be set to 'derivs'+ 1 prior to calling this function.
<i>tpar</i>	the parameter for which we want to evaluate the curve
<i>derivs</i>	the number of derivatives we want to have calculated
<i>from_right</i>	specify whether we should calculate derivatives 'from the right' or 'from the left' (default is from the right). This matters only when the curve presents discontinuities in its derivatives.

Implements [Go::ParamCurve](#).

29.217.3.19 `virtual void Go::Hyperbola::read ( std::istream & is ) [virtual]`

Read object from stream

#### Parameters

<i>is</i>	stream from which object is read
-----------	----------------------------------

Implements [Go::Streamable](#).

29.217.3.20 `virtual void Go::Hyperbola::setParamBounds ( double startpar, double endpar ) [virtual]`

Set bounds for the parametrization of the [Hyperbola](#).

#### Parameters

<i>startpar</i>	start parameter
<i>endpar</i>	end parameter

Implements [Go::ElementaryCurve](#).

29.217.3.21 `virtual void Go::Hyperbola::setParameterInterval ( double t1, double t2 ) [virtual]`

Limit the curve by limiting the parameter interval.

Implements [Go::ParamCurve](#).



29.217.3.22 void Go::Hyperbola::setSpanningVectors ( ) [protected]

29.217.3.23 virtual double Go::Hyperbola::startparam ( ) const [virtual]

Query the start parameter of the curve

#### Returns

the curve's start parameter

Implements [Go::ParamCurve](#).

29.217.3.24 virtual Hyperbola\* Go::Hyperbola::subCurve ( double *from\_par*, double *to\_par*, double *fuzzy* = DEFAULT\_PARAMETER\_EPSILON ) const [virtual]

Returns a curve which is a part of this curve. The result is NEWed, so the user is responsible for deleting it. NB: It is not guaranteed that the [ParamCurve](#) that is returned is of the same type as the curve itself. Thus, the returned curve might be a [SplineCurve](#).

#### Parameters

<i>from_par</i>	start value of parameter interval that will define the subcurve
<i>to_par</i>	end value of parameter interval that will define the subcurve
<i>fuzzy</i>	since subCurve works on those curves who are spline-based, this tolerance defines how close the start and end parameter must be to an existing knot in order to be considered <i>on</i> the knot.

#### Returns

a pointer to a new subcurve which represents the part of the curve between 'from\_par' and 'to\_par'. It will be spline-based and have a k-regular knotvector. The user is responsible for deleting this subcurve when it is no longer needed.

Implements [Go::ElementaryCurve](#).

29.217.3.25 virtual void Go::Hyperbola::swapParameters2D ( ) [virtual]

If the curve is 2 dimensional, x and y coordinates will be swapped. Used when curve is a parameter curve.

Implements [Go::ElementaryCurve](#).

29.217.3.26 virtual void Go::Hyperbola::translateCurve ( const Point & *dir* ) [virtual]

Implements [Go::ElementaryCurve](#).

29.217.3.27 virtual void Go::Hyperbola::write ( std::ostream & *os* ) const [virtual]

Write object to stream

## Parameters

<i>os</i>	stream to which object is written
-----------	-----------------------------------

Implements [Go::Streamable](#).

#### 29.217.4 Member Data Documentation

29.217.4.1 **double** `Go::Hyperbola::endparam_` [protected]

Definition at line 175 of file Hyperbola.h.

29.217.4.2 **Point** `Go::Hyperbola::location_` [protected]

Definition at line 166 of file Hyperbola.h.

29.217.4.3 **Point** `Go::Hyperbola::normal_` [protected]

Definition at line 169 of file Hyperbola.h.

29.217.4.4 **double** `Go::Hyperbola::r1_` [protected]

Definition at line 171 of file Hyperbola.h.

29.217.4.5 **double** `Go::Hyperbola::r2_` [protected]

Definition at line 172 of file Hyperbola.h.

29.217.4.6 **double** `Go::Hyperbola::startparam_` [protected]

Definition at line 174 of file Hyperbola.h.

29.217.4.7 **Point** `Go::Hyperbola::vec1_` [protected]

Definition at line 167 of file Hyperbola.h.

29.217.4.8 **Point** `Go::Hyperbola::vec2_` [protected]

Definition at line 168 of file Hyperbola.h.

The documentation for this class was generated from the following file:

- [gotools-core/include/GoTools/geometry/Hyperbola.h](#)

## 29.218 Go::Identity Class Reference

Check coincidence.

```
#include <Identity.h>
```

### Public Member Functions

- int `identicalSfs` (shared\_ptr< [ParamSurface](#) > sf1, shared\_ptr< [ParamSurface](#) > sf2, double tol)
- int `identicalCvs` (shared\_ptr< [ParamCurve](#) > cv1, double start1, double end1, shared\_ptr< [ParamCurve](#) > cv2, double start2, double end2, double tol)

### 29.218.1 Detailed Description

Check coincidence.

Definition at line 58 of file `Identity.h`.

### 29.218.2 Member Function Documentation

29.218.2.1 int `Go::Identity::identicalCvs` ( shared\_ptr< [ParamCurve](#) > cv1, double start1, double end1, shared\_ptr< [ParamCurve](#) > cv2, double start2, double end2, double tol )

Return value = 0 : Not coincident = 1 : Coincident curves = 2 : Curve one is embedded in curve two = 3 : Curve two is embedded in curve one

29.218.2.2 int `Go::Identity::identicalSfs` ( shared\_ptr< [ParamSurface](#) > sf1, shared\_ptr< [ParamSurface](#) > sf2, double tol )

Return value = 0 : Not coincident = 1 : Coincident surfaces = 2 : Surface one is embedded in surface two = 3 : Surface two is embedded in surface one

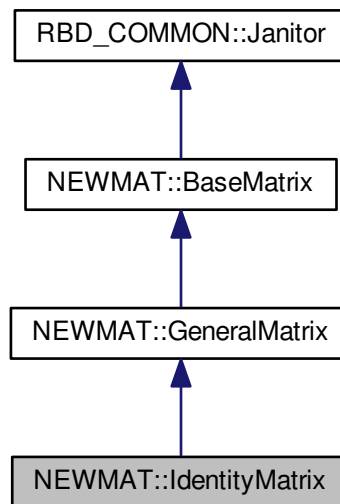
The documentation for this class was generated from the following file:

- [intersections/include/GoTools/intersections/Identity.h](#)

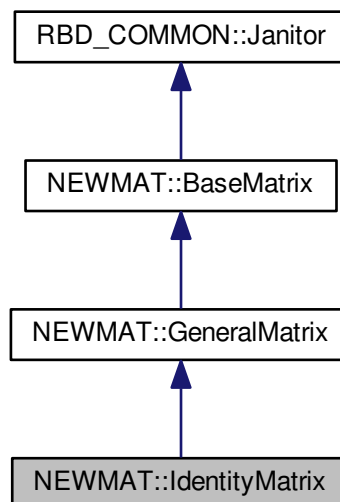
## 29.219 NEWMAT::IdentityMatrix Class Reference

```
#include <newmat.h>
```

Inheritance diagram for NEWMAT::IdentityMatrix:



Collaboration diagram for NEWMAT::IdentityMatrix:



## Public Member Functions

- [IdentityMatrix \(\)](#)
- [~IdentityMatrix \(\)](#)
- [IdentityMatrix \(ArrayLengthSpecifier n\)](#)
- [IdentityMatrix \(const IdentityMatrix &gm\)](#)
- [IdentityMatrix \(const BaseMatrix &\)](#)
- [void operator= \(const BaseMatrix &\)](#)
- [void operator= \(Real f\)](#)
- [MatrixType Type \(\) const](#)
- [LogAndSign LogDeterminant \(\) const](#)
- [Real Trace \(\) const](#)
- [Real SumSquare \(\) const](#)
- [Real SumAbsoluteValue \(\) const](#)
- [Real Sum \(\) const](#)
- [void GetRow \(MatrixRowCol &\)](#)
- [void GetCol \(MatrixRowCol &\)](#)
- [void GetCol \(MatrixColX &\)](#)
- [void NextRow \(MatrixRowCol &\)](#)
- [void NextCol \(MatrixRowCol &\)](#)
- [void NextCol \(MatrixColX &\)](#)
- [GeneralMatrix \\* MakeSolver \(\)](#)
- [void Solver \(MatrixColX &, const MatrixColX &\)](#)
- [GeneralMatrix \\* Transpose \(TransposedMatrix \\*, MatrixType\)](#)
- [void ReSize \(int n\)](#)
- [void ReSize \(const GeneralMatrix &A\)](#)
- [MatrixBandWidth BandWidth \(\) const](#)

## Additional Inherited Members

### 29.219.1 Detailed Description

Definition at line 1101 of file newmat.h.

### 29.219.2 Constructor & Destructor Documentation

**29.219.2.1** `NEWMAT::IdentityMatrix::IdentityMatrix ( )` [\[inline\]](#)

Definition at line 1105 of file newmat.h.

**29.219.2.2** `NEWMAT::IdentityMatrix::~~IdentityMatrix ( )` [\[inline\]](#)

Definition at line 1106 of file newmat.h.

**29.219.2.3** `NEWMAT::IdentityMatrix::IdentityMatrix ( ArrayLengthSpecifier n )` [\[inline\]](#)

Definition at line 1107 of file newmat.h.

29.219.2.4 `NEWMAT::IdentityMatrix::IdentityMatrix ( const IdentityMatrix & gm ) [inline]`

Definition at line 1109 of file newmat.h.

29.219.2.5 `NEWMAT::IdentityMatrix::IdentityMatrix ( const BaseMatrix & )`

### 29.219.3 Member Function Documentation

29.219.3.1 `MatrixBandWidth IdentityMatrix::BandWidth ( ) const [virtual]`

Reimplemented from [NEWMAT::BaseMatrix](#).

Definition at line 404 of file newmat4.cpp.

29.219.3.2 `void NEWMAT::IdentityMatrix::GetCol ( MatrixRowCol & ) [virtual]`

Implements [NEWMAT::GeneralMatrix](#).

29.219.3.3 `void NEWMAT::IdentityMatrix::GetCol ( MatrixCoIX & ) [virtual]`

Implements [NEWMAT::GeneralMatrix](#).

29.219.3.4 `void IdentityMatrix::GetRow ( MatrixRowCol & mrc ) [virtual]`

Implements [NEWMAT::GeneralMatrix](#).

Definition at line 784 of file newmat3.cpp.

29.219.3.5 `LogAndSign IdentityMatrix::LogDeterminant ( ) const [virtual]`

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 672 of file newmat8.cpp.

29.219.3.6 `GeneralMatrix* NEWMAT::IdentityMatrix::MakeSolver ( ) [inline],[virtual]`

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 1126 of file newmat.h.

29.219.3.7 `void NEWMAT::IdentityMatrix::NextCol ( MatrixRowCol & ) [virtual]`

Reimplemented from [NEWMAT::GeneralMatrix](#).

29.219.3.8 void NEWMAT::IdentityMatrix::NextCol ( MatrixColX & ) [virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

29.219.3.9 void IdentityMatrix::NextRow ( MatrixRowCol & mrc ) [virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 806 of file newmat3.cpp.

29.219.3.10 void NEWMAT::IdentityMatrix::operator= ( const BaseMatrix & )

29.219.3.11 void NEWMAT::IdentityMatrix::operator= ( Real f ) [inline]

Definition at line 1112 of file newmat.h.

29.219.3.12 void NEWMAT::IdentityMatrix::ReSize ( int n )

29.219.3.13 void IdentityMatrix::ReSize ( const GeneralMatrix & A ) [virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 327 of file newmat4.cpp.

29.219.3.14 void IdentityMatrix::Solver ( MatrixColX & mrc, const MatrixColX & mrc1 ) [virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 512 of file newmat2.cpp.

29.219.3.15 Real NEWMAT::IdentityMatrix::Sum ( ) const [inline],[virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 1119 of file newmat.h.

29.219.3.16 Real IdentityMatrix::SumAbsoluteValue ( ) const [virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 408 of file newmat8.cpp.

29.219.3.17 **Real IdentityMatrix::SumSquare ( ) const** [virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 424 of file newmat8.cpp.

29.219.3.18 **Real IdentityMatrix::Trace ( ) const** [virtual]

Reimplemented from [NEWMAT::BaseMatrix](#).

Definition at line 600 of file newmat8.cpp.

29.219.3.19 **GeneralMatrix \* IdentityMatrix::Transpose ( TransposedMatrix \*, MatrixType mt )** [virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 76 of file newmat5.cpp.

29.219.3.20 **MatrixType IdentityMatrix::Type ( ) const** [virtual]

Implements [NEWMAT::GeneralMatrix](#).

Definition at line 398 of file newmat4.cpp.

The documentation for this class was generated from the following files:

- [newmat/include/newmat.h](#)
- [newmat/src/newmat2.cpp](#)
- [newmat/src/newmat3.cpp](#)
- [newmat/src/newmat4.cpp](#)
- [newmat/src/newmat5.cpp](#)
- [newmat/src/newmat8.cpp](#)

## 29.220 Go::IGESconverter Class Reference

```
#include <IGESconverter.h>
```



## Public Member Functions

- [IGESconverter](#) ()  
*Default constructor.*
- [IGESconverter](#) (std::istream &is, [FileFormat](#) from\_type, std::ostream &os, [FileFormat](#) to\_type)  
*This constructor calls the appropriate read and write members.*
- [~IGESconverter](#) ()  
*Destructure.*
- void [readgo](#) (std::istream &is)  
*Read a g2-file.*
- void [readsislrfs](#) (std::istream &is)  
*Read a number of sisl surfaces.*
- void [readsislcrvs](#) (std::istream &is)  
*Read a number of sisl curves.*
- void [readdisp](#) (std::istream &is)
- void [readIGES](#) (std::istream &is)  
*Read an IGES file.*
- void [writego](#) (std::ostream &os)  
*Write the content of this converter to a g2-file.*
- void [writedisp](#) (std::ostream &os)
- void [writeIGES](#) (std::ostream &os)  
*Write the content of this converter to an IGES-file.*
- void [addGeom](#) (shared\_ptr< [Go::GeomObject](#) > sp)  
*Add one more geometry object to this converter.*
- const std::vector< shared\_ptr< [Go::GeomObject](#) > > & [getGoGeom](#) ()  
*Get all geometry objects stored in this converter.*
- const std::vector< double > & [getColour](#) (int i)  
*Get the colour of entity number i.*
- const std::vector< [ftGroupGeom](#) > & [getGroup](#) ()  
*Get group information given in the IGES file (specified by form number)*
- double [minResolution](#) () const  
*Fetch the geometry tolerance given in an IGES file.*
- int [num\\_geom](#) ()  
*Number of geometric entities.*
- int [num\\_group](#) ()  
*Number of groups of entities.*
- [IGESheader](#) & [header](#) ()  
*Gives dangerous but necessary direct access to header info.*

### 29.220.1 Detailed Description

The converter between the IGES fileformat, the file format used in [GoTools](#), a file format used for SISEL. The disp file format is outdated.

Definition at line 215 of file IGESconverter.h.

### 29.220.2 Constructor & Destructor Documentation

#### 29.220.2.1 Go::IGESconverter::IGESconverter ( )

Default constructor.

29.220.2.2 `Go::IGESconverter::IGESconverter ( std::istream & is, FileFormat from_type, std::ostream & os, FileFormat to_type )`

This constructor calls the appropriate read and write members.

29.220.2.3 `Go::IGESconverter::~~IGESconverter ( )`

Destructure.

### 29.220.3 Member Function Documentation

29.220.3.1 `void Go::IGESconverter::addGeom ( shared_ptr< Go::GeomObject > sp )`

Add one more geometry object to this converter.

29.220.3.2 `const std::vector<double>& Go::IGESconverter::getColour ( int i ) [inline]`

Get the colour of entity number i.

Definition at line 252 of file IGESconverter.h.

29.220.3.3 `const std::vector<shared_ptr<Go::GeomObject> >& Go::IGESconverter::getGoGeom ( ) [inline]`

Get all geometry objects stored in this converter.

Definition at line 248 of file IGESconverter.h.

29.220.3.4 `const std::vector<ftGroupGeom>& Go::IGESconverter::getGroup ( ) [inline]`

Get group information given in the IGES file (specified by form number)

Definition at line 256 of file IGESconverter.h.

29.220.3.5 `IGESheader& Go::IGESconverter::header ( ) [inline]`

Gives dangerous but necessary direct access to header info.

Definition at line 268 of file IGESconverter.h.

29.220.3.6 `double Go::IGESconverter::minResolution ( ) const [inline]`

Fetch the geometry tolerance given in an IGES file.

Definition at line 260 of file IGESconverter.h.

29.220.3.7 `int Go::IGESconverter::num_geom ( ) [inline]`

Number of geometric entities.

Definition at line 264 of file IGESconverter.h.

29.220.3.8 `int Go::IGESconverter::num_group ( ) [inline]`

Number of groups of entities.

Definition at line 266 of file IGESconverter.h.

29.220.3.9 `void Go::IGESconverter::readdisp ( std::istream & is )`

29.220.3.10 `void Go::IGESconverter::readgo ( std::istream & is )`

Read a g2-file.

29.220.3.11 `void Go::IGESconverter::readIGES ( std::istream & is )`

Read an IGES file.

29.220.3.12 `void Go::IGESconverter::readsislcrvs ( std::istream & is )`

Read a number of sisl curves.

29.220.3.13 `void Go::IGESconverter::readsislrfs ( std::istream & is )`

Read a number of sisl surfaces.

29.220.3.14 `void Go::IGESconverter::writedisp ( std::ostream & os )`

29.220.3.15 `void Go::IGESconverter::writego ( std::ostream & os )`

Write the content of this converter to a g2-file.

29.220.3.16 `void Go::IGESconverter::writeIGES ( std::ostream & os )`

Write the content of this converter to an IGES-file.

The documentation for this class was generated from the following file:

- [igeslib/include/GoTools/igeslib/IGESconverter.h](#)

## 29.221 Go::IGESdirentry Struct Reference

Storage of all data contained in an IGES directory entity.

```
#include <IGESconverter.h>
```

### Public Attributes

- int [entity\\_type\\_number](#)
- int [param\\_data\\_start](#)
- int [structure](#)
- int [line\\_font\\_pattern](#)
- int [level](#)
- int [view](#)
- int [trans\\_matrix](#)
- int [label\\_display](#)
- std::string [status](#)
- double [line\\_weight](#)
- int [color](#)
- int [line\\_count](#)
- int [form](#)
- std::string [entity\\_label](#)
- int [entity\\_number](#)

### 29.221.1 Detailed Description

Storage of all data contained in an IGES directory entity.

Definition at line 191 of file IGESconverter.h.

### 29.221.2 Member Data Documentation

#### 29.221.2.1 int Go::IGESdirentry::color

Definition at line 204 of file IGESconverter.h.

#### 29.221.2.2 std::string Go::IGESdirentry::entity\_label

Definition at line 207 of file IGESconverter.h.

#### 29.221.2.3 int Go::IGESdirentry::entity\_number

Definition at line 208 of file IGESconverter.h.

#### 29.221.2.4 int Go::IGESdirentry::entity\_type\_number

Definition at line 194 of file IGESconverter.h.

29.221.2.5 `int Go::IGESdirentry::form`

Definition at line 206 of file IGESconverter.h.

29.221.2.6 `int Go::IGESdirentry::label_display`

Definition at line 201 of file IGESconverter.h.

29.221.2.7 `int Go::IGESdirentry::level`

Definition at line 198 of file IGESconverter.h.

29.221.2.8 `int Go::IGESdirentry::line_count`

Definition at line 205 of file IGESconverter.h.

29.221.2.9 `int Go::IGESdirentry::line_font_pattern`

Definition at line 197 of file IGESconverter.h.

29.221.2.10 `double Go::IGESdirentry::line_weight`

Definition at line 203 of file IGESconverter.h.

29.221.2.11 `int Go::IGESdirentry::param_data_start`

Definition at line 195 of file IGESconverter.h.

29.221.2.12 `std::string Go::IGESdirentry::status`

Definition at line 202 of file IGESconverter.h.

29.221.2.13 `int Go::IGESdirentry::structure`

Definition at line 196 of file IGESconverter.h.

29.221.2.14 `int Go::IGESdirentry::trans_matrix`

Definition at line 200 of file IGESconverter.h.

29.221.2.15 int Go::IGESdirentry::view

Definition at line 199 of file IGESconverter.h.

The documentation for this struct was generated from the following file:

- [igeslib/include/GoTools/igeslib/IGESconverter.h](#)

## 29.222 Go::IGESheader Struct Reference

Storage of all data contained in the IGES header.

```
#include <IGESconverter.h>
```

### Public Member Functions

- [IGESheader \(\)](#)  
*Constructor. Needed to handle write to file.*

### Public Attributes

- char [pardel](#)
- char [recdel](#)
- std::string [sending\\_system\\_prodid](#)
- std::string [original\\_filename](#)
- std::string [creator\\_system\\_prodid](#)
- std::string [preprocessor\\_version\\_number](#)
- int [integer\\_bits](#)
- int [single\\_prec\\_magn](#)
- int [single\\_prec\\_sign](#)
- int [double\\_prec\\_magn](#)
- int [double\\_prec\\_sign](#)
- std::string [receiving\\_system\\_prodid](#)
- double [model\\_space\\_scale](#)
- int [unit\\_flag](#)
- std::string [unit\\_description](#)
- int [max\\_weight\\_grads](#)
- double [max\\_line\\_width](#)
- std::string [timestamp\\_file\\_changed](#)
- double [min\\_resolution](#)
- double [max\\_coordinate](#)
- std::string [author](#)
- std::string [organisation](#)
- int [IGES\\_version](#)
- int [drafting\\_standard](#)
- std::string [timestamp\\_model\\_changed](#)
- std::string [application\\_protocol\\_description](#)
- int [num\\_parameters](#)

### 29.222.1 Detailed Description

Storage of all data contained in the IGES header.

Definition at line 80 of file IGESconverter.h.

### 29.222.2 Constructor & Destructor Documentation

#### 29.222.2.1 Go::IGESheader::IGESheader ( )

Constructor. Needed to handle write to file.

### 29.222.3 Member Data Documentation

#### 29.222.3.1 std::string Go::IGESheader::application\_protocol\_description

Definition at line 111 of file IGESconverter.h.

#### 29.222.3.2 std::string Go::IGESheader::author

Definition at line 106 of file IGESconverter.h.

#### 29.222.3.3 std::string Go::IGESheader::creator\_system\_prodid

Definition at line 90 of file IGESconverter.h.

#### 29.222.3.4 int Go::IGESheader::double\_prec\_magn

Definition at line 95 of file IGESconverter.h.

#### 29.222.3.5 int Go::IGESheader::double\_prec\_sign

Definition at line 96 of file IGESconverter.h.

#### 29.222.3.6 int Go::IGESheader::drafting\_standard

Definition at line 109 of file IGESconverter.h.

#### 29.222.3.7 int Go::IGESheader::IGES\_version

Definition at line 108 of file IGESconverter.h.

**29.222.3.8** int Go::IGESheader::integer\_bits

Definition at line 92 of file IGESconverter.h.

**29.222.3.9** double Go::IGESheader::max\_coordinate

Definition at line 105 of file IGESconverter.h.

**29.222.3.10** double Go::IGESheader::max\_line\_width

Definition at line 102 of file IGESconverter.h.

**29.222.3.11** int Go::IGESheader::max\_weight\_grads

Definition at line 101 of file IGESconverter.h.

**29.222.3.12** double Go::IGESheader::min\_resolution

Definition at line 104 of file IGESconverter.h.

**29.222.3.13** double Go::IGESheader::model\_space\_scale

Definition at line 98 of file IGESconverter.h.

**29.222.3.14** int Go::IGESheader::num\_parameters

Definition at line 113 of file IGESconverter.h.

**29.222.3.15** std::string Go::IGESheader::organisation

Definition at line 107 of file IGESconverter.h.

**29.222.3.16** std::string Go::IGESheader::original\_filename

Definition at line 89 of file IGESconverter.h.

**29.222.3.17** char Go::IGESheader::pardel

Definition at line 86 of file IGESconverter.h.



29.222.3.18 `std::string Go::IGESheader::preprocessor_version_number`

Definition at line 91 of file IGESconverter.h.

29.222.3.19 `char Go::IGESheader::recdel`

Definition at line 87 of file IGESconverter.h.

29.222.3.20 `std::string Go::IGESheader::receiving_system_prodid`

Definition at line 97 of file IGESconverter.h.

29.222.3.21 `std::string Go::IGESheader::sending_system_prodid`

Definition at line 88 of file IGESconverter.h.

29.222.3.22 `int Go::IGESheader::single_prec_magn`

Definition at line 93 of file IGESconverter.h.

29.222.3.23 `int Go::IGESheader::single_prec_sign`

Definition at line 94 of file IGESconverter.h.

29.222.3.24 `std::string Go::IGESheader::timestamp_file_changed`

Definition at line 103 of file IGESconverter.h.

29.222.3.25 `std::string Go::IGESheader::timestamp_model_changed`

Definition at line 110 of file IGESconverter.h.

29.222.3.26 `std::string Go::IGESheader::unit_description`

Definition at line 100 of file IGESconverter.h.

29.222.3.27 `int Go::IGESheader::unit_flag`

Definition at line 99 of file IGESconverter.h.

The documentation for this struct was generated from the following file:

- [igeslib/include/GoTools/igeslib/IGESconverter.h](#)

## 29.223 Go::ImplicitizeCurveAlgo Class Reference

```
#include <ImplicitizeCurveAlgo.h>
```

### Public Member Functions

- [ImplicitizeCurveAlgo](#) ()  
*Default constructor.*
- [ImplicitizeCurveAlgo](#) (int deg)
- [ImplicitizeCurveAlgo](#) (const [SplineCurve](#) &curve, int deg)
- void [useSplineCurve](#) (const [SplineCurve](#) &curve)
- void [setDegree](#) (int deg)
- void [setTolerance](#) (double tol)
- void [perform](#) ()
- void [getResultData](#) ([BernsteinTriangularPoly](#) &implicit, [BaryCoordSystem2D](#) &bc, double &sigma\_min)

### 29.223.1 Detailed Description

Class that implements an implicitization algorithm for spline curves.

Input: A curve of type [SplineCurve](#), and the degree that is chosen for the implicit representation.

Output: A barycentric coordinate system, and the Bernstein polynomial that represents the implicit curve.

The algorithm first defines a barycentric coordinate system in terms of a triangle that is slightly larger than the bounding box of the spline curve. It then computes the Bernstein polynomial representing the implicit curve. If the chosen degree is too low to produce the exact result, an approximate implicit curve is produced. If the degree is too high, the resulting polynomial will be reducible and contain the exact implicitization as a factor. If the input curve consists of multiple segments, the output polynomial will be a product of the separate implicit curves, or an approximation of it.

Definition at line 73 of file [ImplicitizeCurveAlgo.h](#).

### 29.223.2 Constructor & Destructor Documentation

**29.223.2.1** [Go::ImplicitizeCurveAlgo::ImplicitizeCurveAlgo](#) ( ) `[inline]`

Default constructor.

Definition at line 76 of file [ImplicitizeCurveAlgo.h](#).

**29.223.2.2** [Go::ImplicitizeCurveAlgo::ImplicitizeCurveAlgo](#) ( int *deg* ) `[inline], [explicit]`

Constructor

Parameters

<i>deg</i>	degree of the implicit representation
------------	---------------------------------------

Definition at line 79 of file ImplicitizeCurveAlgo.h.

29.223.2.3 Go::ImplicitizeCurveAlgo::ImplicitizeCurveAlgo ( const SplineCurve & curve, int deg ) [inline]

Constructor

Parameters

<i>curve</i>	spline curve to be implicitized
<i>deg</i>	degree of the implicit representation

Definition at line 83 of file ImplicitizeCurveAlgo.h.

### 29.223.3 Member Function Documentation

29.223.3.1 void Go::ImplicitizeCurveAlgo::getResultData ( BernsteinTriangularPoly & implicit, BaryCoordSystem2D & bc, double & sigma\_min ) [inline]

Get the result of the implicitization

Return values

<i>implicit</i>	a <a href="#">BernsteinTriangularPoly</a> representing the implicit curve
<i>bc</i>	the barycentric coordinate system in which the <a href="#">BernsteinTriangularPoly</a> is defined

Definition at line 110 of file ImplicitizeCurveAlgo.h.

29.223.3.2 void Go::ImplicitizeCurveAlgo::perform ( )

Perform the implicitization. This function runs the implicitization algorithm.

29.223.3.3 void Go::ImplicitizeCurveAlgo::setDegree ( int deg ) [inline]

Choose the degree of the implicit representation

Parameters

<i>deg</i>	degree
------------	--------

Definition at line 93 of file ImplicitizeCurveAlgo.h.

29.223.3.4 void Go::ImplicitizeCurveAlgo::setTolerance ( double tol ) [inline]

Set the tolerance

## Parameters

<i>tol</i>	tolerance; default value is 3.0e-15
------------	-------------------------------------

Definition at line 98 of file ImplicitizeCurveAlgo.h.

29.223.3.5 void Go::ImplicitizeCurveAlgo::useSplineCurve ( const SplineCurve & curve ) [inline]

Load the spline curve to be implicitized

## Parameters

<i>curve</i>	curve on <a href="#">SplineCurve</a> form
--------------	-------------------------------------------

Definition at line 88 of file ImplicitizeCurveAlgo.h.

The documentation for this class was generated from the following file:

- [implicitization/include/GoTools/implicitization/ImplicitizeCurveAlgo.h](#)

## 29.224 Go::ImplicitizeCurveAndVectorAlgo Class Reference

```
#include <ImplicitizeCurveAndVectorAlgo.h>
```

### Public Member Functions

- [ImplicitizeCurveAndVectorAlgo](#) ()  
*Default constructor.*
- [ImplicitizeCurveAndVectorAlgo](#) (int deg)
- [ImplicitizeCurveAndVectorAlgo](#) (const [SplineCurve](#) &crv, const [Point](#) &pt, int deg)
- void [useSplineCurve](#) (const [SplineCurve](#) &crv)
- void [useVector](#) (const [Point](#) &pt)
- void [setDegree](#) (int deg)
- void [setTolerance](#) (double tol)
- int [perform](#) ()
- void [getResultData](#) ([BernsteinTetrahedralPoly](#) &implicit, [BaryCoordSystem3D](#) &bc, double &sigma\_min)

### 29.224.1 Detailed Description

Class that implements an algorithm for finding an implicit ruled surface defined by a curve and a vector.

Input: A curve of type [SplineCurve](#), a vector of type [Point](#), and the degree that is chosen for the implicit surface.

Output: A barycentric coordinate system, and the Bernstein polynomial that represents the implicit surface.

The algorithm first defines a barycentric coordinate system in terms of a tetrahedron that is slightly larger than the bounding box of the spline curve. It then computes the Bernstein polynomial representing the implicit ruled surface defined by the spline curve and vector. If the chosen degree is too low to produce the exact result, an approximate implicit surface is produced. If the degree is too high, the resulting polynomial will be reducible and contain the exact implicitization as a factor. If the input curve consists of multiple segments, the output polynomial will be a product of the separate implicit surfaces, or an approximation of it.

Definition at line 75 of file ImplicitizeCurveAndVectorAlgo.h.

## 29.224.2 Constructor & Destructor Documentation

### 29.224.2.1 Go::ImplicitizeCurveAndVectorAlgo::ImplicitizeCurveAndVectorAlgo ( ) [inline]

Default constructor.

Definition at line 78 of file ImplicitizeCurveAndVectorAlgo.h.

### 29.224.2.2 Go::ImplicitizeCurveAndVectorAlgo::ImplicitizeCurveAndVectorAlgo ( int *deg* ) [inline],[explicit]

Constructor

Parameters

<i>deg</i>	degree of the implicit representation
------------	---------------------------------------

Definition at line 81 of file ImplicitizeCurveAndVectorAlgo.h.

### 29.224.2.3 Go::ImplicitizeCurveAndVectorAlgo::ImplicitizeCurveAndVectorAlgo ( const SplineCurve & *crv*, const Point & *pt*, int *deg* ) [inline]

Constructor

Parameters

<i>crv</i>	3D spline curve defining the implicit surface
<i>pt</i>	3D vector defining the implicit surface
<i>deg</i>	degree of the implicit surface

Definition at line 87 of file ImplicitizeCurveAndVectorAlgo.h.

## 29.224.3 Member Function Documentation

### 29.224.3.1 void Go::ImplicitizeCurveAndVectorAlgo::getResultData ( BernsteinTetrahedralPoly & *implicit*, BaryCoordSystem3D & *bc*, double & *sigma\_min* ) [inline]

Get the result of the implicitization.

Return values

<i>implicit</i>	a <a href="#">BernsteinTetrahedralPoly</a> representing the implicit surface
<i>bc</i>	the barycentric coordinate system in which the <a href="#">BernsteinTetrahedralPoly</a> is defined

Definition at line 120 of file ImplicitizeCurveAndVectorAlgo.h.

29.224.3.2 `int Go::ImplicitizeCurveAndVectorAlgo::perform ( )`

Perform the implicitization. This function runs the implicitization algorithm.

29.224.3.3 `void Go::ImplicitizeCurveAndVectorAlgo::setDegree ( int deg )` `[inline]`

Choose the degree of the implicit representation.

#### Parameters

<i>deg</i>	degree
------------	--------

Definition at line 103 of file `ImplicitizeCurveAndVectorAlgo.h`.

29.224.3.4 `void Go::ImplicitizeCurveAndVectorAlgo::setTolerance ( double tol )` `[inline]`

Set the tolerance.

#### Parameters

<i>tol</i>	tolerance; default value is 3.0e-15
------------	-------------------------------------

Definition at line 108 of file `ImplicitizeCurveAndVectorAlgo.h`.

29.224.3.5 `void Go::ImplicitizeCurveAndVectorAlgo::useSplineCurve ( const SplineCurve & crv )` `[inline]`

Load the spline curve for the implicitization

#### Parameters

<i>crv</i>	curve on <a href="#">SplineCurve</a> form
------------	-------------------------------------------

Definition at line 93 of file `ImplicitizeCurveAndVectorAlgo.h`.

29.224.3.6 `void Go::ImplicitizeCurveAndVectorAlgo::useVector ( const Point & pt )` `[inline]`

Load the vector for the implicitization.

#### Parameters

<i>pt</i>	vector of type <a href="#">Point</a>
-----------	--------------------------------------

Definition at line 98 of file `ImplicitizeCurveAndVectorAlgo.h`.

The documentation for this class was generated from the following file:

- [implicitization/include/GoTools/implicitization/ImplicitizeCurveAndVectorAlgo.h](#)

## 29.225 Go::ImplicitizePointCloudAlgo Class Reference

```
#include <ImplicitizePointCloudAlgo.h>
```

### Public Member Functions

- [ImplicitizePointCloudAlgo](#) ()  
*Default constructor.*
- [ImplicitizePointCloudAlgo](#) (int deg)
- [ImplicitizePointCloudAlgo](#) (const [PointCloud3D](#) &cloud, int deg)
- void [usePointCloud](#) (const [PointCloud3D](#) &cloud)
- void [setDegree](#) (int deg)
- void [setTolerance](#) (double tol)
- void [perform](#) ()
- void [getResultData](#) ([BernsteinTetrahedralPoly](#) &implicit, [BaryCoordSystem3D](#) &bc, double &sigma\_min)

### 29.225.1 Detailed Description

Class that implements an algorithm for finding an implicit surface from a 3D point cloud.

Input: A point cloud of type [PointCloud3D](#), and the degree that is chosen for the implicit representation.

Output: A barycentric coordinate system, and the Bernstein polynomial that represents the implicit surface.

The algorithm first defines a barycentric coordinate system in terms of a tetrahedron that is slightly larger than the bounding box of the point cloud. It then computes the Bernstein polynomial representing the implicit surface. If the chosen degree is too low to produce the exact result, an approximate implicit surface is produced. If the degree is too high, the resulting polynomial will be reducible and contain the exact implicitization as a factor.

Definition at line 71 of file [ImplicitizePointCloudAlgo.h](#).

### 29.225.2 Constructor & Destructor Documentation

**29.225.2.1** [Go::ImplicitizePointCloudAlgo::ImplicitizePointCloudAlgo](#) ( ) [[inline](#)]

Default constructor.

Definition at line 74 of file [ImplicitizePointCloudAlgo.h](#).

**29.225.2.2** [Go::ImplicitizePointCloudAlgo::ImplicitizePointCloudAlgo](#) ( int *deg* ) [[inline](#)], [[explicit](#)]

Constructor.

## Parameters

<i>deg</i>	degree of the implicit representation
------------	---------------------------------------

Definition at line 77 of file ImplicitizePointCloudAlgo.h.

29.225.2.3 `Go::ImplicitizePointCloudAlgo::ImplicitizePointCloudAlgo ( const PointCloud3D & cloud, int deg )`  
`[inline]`

Constructor.

## Parameters

<i>cloud</i>	point cloud to be implicitized
<i>deg</i>	degree of the implicit representation

Definition at line 82 of file ImplicitizePointCloudAlgo.h.

## 29.225.3 Member Function Documentation

29.225.3.1 `void Go::ImplicitizePointCloudAlgo::getResultData ( BernsteinTetrahedralPoly & implicit, BaryCoordSystem3D & bc, double & sigma_min )` `[inline]`

Get the result of the implicitization.

## Return values

<i>implicit</i>	a <a href="#">BernsteinTetrahedralPoly</a> representing the implicit surface
<i>bc</i>	the barycentric coordinate system in which the <a href="#">BernsteinTetrahedralPoly</a> is defined

Definition at line 109 of file ImplicitizePointCloudAlgo.h.

29.225.3.2 `void Go::ImplicitizePointCloudAlgo::perform ( )`

Perform the implicitization. This function runs the implicitization algorithm.

29.225.3.3 `void Go::ImplicitizePointCloudAlgo::setDegree ( int deg )` `[inline]`

Choose the degree of the implicit representation.

## Parameters

<i>deg</i>	degree
------------	--------



Definition at line 92 of file ImplicitizePointCloudAlgo.h.

29.225.3.4 void Go::ImplicitizePointCloudAlgo::setTolerance ( double *tol* ) [inline]

Set the tolerance.

#### Parameters

<i>tol</i>	tolerance; default value is 3.0e-15
------------	-------------------------------------

Definition at line 97 of file ImplicitizePointCloudAlgo.h.

29.225.3.5 void Go::ImplicitizePointCloudAlgo::usePointCloud ( const PointCloud3D & *cloud* ) [inline]

Load the point cloud to be implicitized.

#### Parameters

<i>cloud</i>	point cloud on PointCloud3D form
--------------	----------------------------------

Definition at line 87 of file ImplicitizePointCloudAlgo.h.

The documentation for this class was generated from the following file:

- [implicitization/include/GoTools/implicitization/ImplicitizePointCloudAlgo.h](#)

## 29.226 Go::ImplicitizeSurfaceAlgo Class Reference

```
#include <ImplicitizeSurfaceAlgo.h>
```

### Public Member Functions

- [ImplicitizeSurfaceAlgo](#) ()  
*Default constructor.*
- [ImplicitizeSurfaceAlgo](#) (int deg)
- [ImplicitizeSurfaceAlgo](#) (const [SplineSurface](#) &surf, int deg)
- void [useSplineSurface](#) (const [SplineSurface](#) &surf)
- void [setDegree](#) (int deg)
- void [setTolerance](#) (double tol)
- void [perform](#) ()
- void [getResultData](#) ([BernsteinTetrahedralPoly](#) &implicit, [BaryCoordSystem3D](#) &bc, double &sigma\_min)

### 29.226.1 Detailed Description

Class that implements an implicitization algorithm for spline surfaces

Input: A surface of type [SplineSurface](#), and the degree that is chosen for the implicitization.

Output: A barycentric coordinate system, and the Bernstein polynomial that represents the implicit surface.

The algorithm first defines a barycentric coordinate system in terms of a tetrahedron that is slightly larger than the bounding box of the spline surface. It then computes the Bernstein polynomial representing the implicit surface. If the chosen degree is too low to produce the exact result, an approximate implicit surface is produced. If the degree is too high, the resulting polynomial will be reducible and contain the exact implicitization as a factor. If the input surface consists of multiple patches, the output polynomial will be a product of the separate implicit surfaces, or an approximation of it.

Definition at line 74 of file ImplicitizeSurfaceAlgo.h.

### 29.226.2 Constructor & Destructor Documentation

29.226.2.1 `Go::ImplicitizeSurfaceAlgo::ImplicitizeSurfaceAlgo ( )` `[inline]`

Default constructor.

Definition at line 77 of file ImplicitizeSurfaceAlgo.h.

29.226.2.2 `Go::ImplicitizeSurfaceAlgo::ImplicitizeSurfaceAlgo ( int deg )` `[inline],[explicit]`

Constructor.

Parameters

<i>deg</i>	degree of the implicit representation
------------	---------------------------------------

Definition at line 80 of file ImplicitizeSurfaceAlgo.h.

29.226.2.3 `Go::ImplicitizeSurfaceAlgo::ImplicitizeSurfaceAlgo ( const SplineSurface & surf, int deg )` `[inline]`

Constructor.

Parameters

<i>surf</i>	spline surface to be implicitized
<i>deg</i>	degree of the implicit representation

Definition at line 84 of file ImplicitizeSurfaceAlgo.h.

### 29.226.3 Member Function Documentation

29.226.3.1 void Go::ImplicitizeSurfaceAlgo::getResultData ( BernsteinTetrahedralPoly & *implicit*, BaryCoordSystem3D & *bc*, double & *sigma\_min* ) [inline]

Get the result of the implicitization.

Return values

<i>implicit</i>	a BernsteinTetrahedralPoly representing the implicit surface
<i>bc</i>	the barycentric coordinate system in which the BernsteinTetrahedralPoly is defined

Definition at line 111 of file ImplicitizeSurfaceAlgo.h.

29.226.3.2 void Go::ImplicitizeSurfaceAlgo::perform ( )

Perform the implicitization. This function runs the implicitization algorithm.

29.226.3.3 void Go::ImplicitizeSurfaceAlgo::setDegree ( int *deg* ) [inline]

Choose the degree of the implicit representation.

Parameters

<i>deg</i>	degree
------------	--------

Definition at line 94 of file ImplicitizeSurfaceAlgo.h.

29.226.3.4 void Go::ImplicitizeSurfaceAlgo::setTolerance ( double *tol* ) [inline]

Set the tolerance.

Parameters

<i>tol</i>	tolerance; default value is 3.0e-15
------------	-------------------------------------

Definition at line 99 of file ImplicitizeSurfaceAlgo.h.

29.226.3.5 void Go::ImplicitizeSurfaceAlgo::useSplineSurface ( const SplineSurface & *surf* ) [inline]

Load the spline surface to be implicitized.

Parameters

<i>surf</i>	surface on SplineSurface form
-------------	-------------------------------

Definition at line 89 of file ImplicitizeSurfaceAlgo.h.

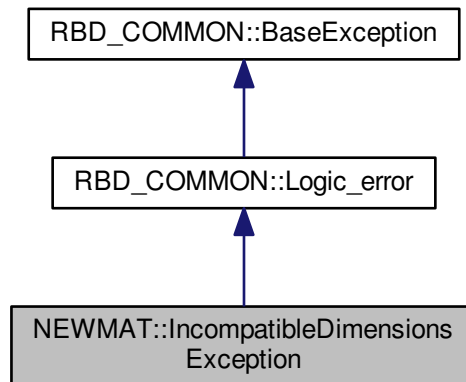
The documentation for this class was generated from the following file:

- [implicitization/include/GoTools/implicitization/ImplicitizeSurfaceAlgo.h](#)

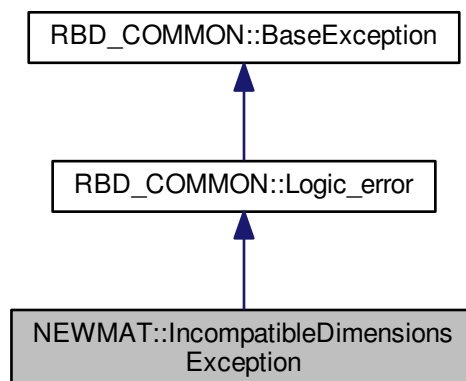
## 29.227 NEWMAT::IncompatibleDimensionsException Class Reference

```
#include <newmat.h>
```

Inheritance diagram for NEWMAT::IncompatibleDimensionsException:



Collaboration diagram for NEWMAT::IncompatibleDimensionsException:



## Public Member Functions

- [IncompatibleDimensionsException](#) ()
- [IncompatibleDimensionsException](#) (const [GeneralMatrix](#) &, const [GeneralMatrix](#) &)

## Static Public Attributes

- static unsigned long [Select](#)

## Additional Inherited Members

### 29.227.1 Detailed Description

Definition at line 1669 of file newmat.h.

### 29.227.2 Constructor & Destructor Documentation

#### 29.227.2.1 [IncompatibleDimensionsException::IncompatibleDimensionsException](#) ( )

Definition at line 160 of file newmatex.cpp.

#### 29.227.2.2 [NEWMAT::IncompatibleDimensionsException::IncompatibleDimensionsException](#) ( const [GeneralMatrix](#) & , const [GeneralMatrix](#) & )

### 29.227.3 Member Data Documentation

#### 29.227.3.1 unsigned long [IncompatibleDimensionsException::Select](#) [static]

Definition at line 1672 of file newmat.h.

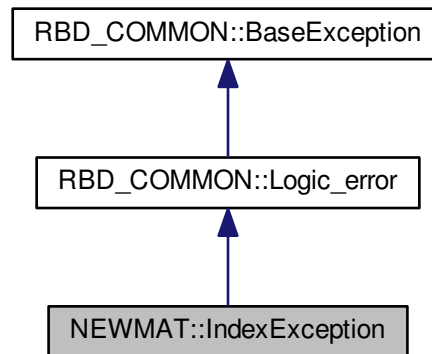
The documentation for this class was generated from the following files:

- newmat/include/[newmat.h](#)
- newmat/src/[newmatex.cpp](#)

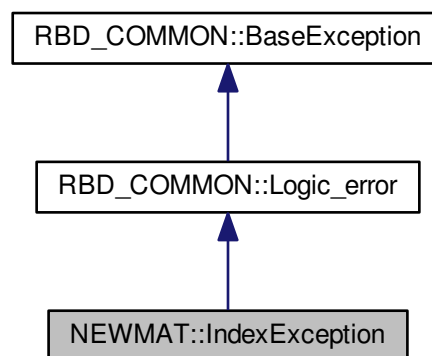
## 29.228 NEWMAT::IndexException Class Reference

```
#include <newmat.h>
```

Inheritance diagram for NEWMAT::IndexException:



Collaboration diagram for NEWMAT::IndexException:



### Public Member Functions

- [IndexException](#) (int i, const GeneralMatrix &A)
- [IndexException](#) (int i, int j, const GeneralMatrix &A)
- [IndexException](#) (int i, const GeneralMatrix &A, bool)
- [IndexException](#) (int i, int j, const GeneralMatrix &A, bool)

## Static Public Attributes

- static unsigned long [Select](#)

## Additional Inherited Members

### 29.228.1 Detailed Description

Definition at line 1636 of file newmat.h.

### 29.228.2 Constructor & Destructor Documentation

#### 29.228.2.1 IndexException::IndexException ( int *i*, const GeneralMatrix & A )

Definition at line 199 of file newmatex.cpp.

#### 29.228.2.2 IndexException::IndexException ( int *i*, int *j*, const GeneralMatrix & A )

Definition at line 209 of file newmatex.cpp.

#### 29.228.2.3 IndexException::IndexException ( int *i*, const GeneralMatrix & A, bool )

Definition at line 221 of file newmatex.cpp.

#### 29.228.2.4 IndexException::IndexException ( int *i*, int *j*, const GeneralMatrix & A, bool )

Definition at line 232 of file newmatex.cpp.

### 29.228.3 Member Data Documentation

#### 29.228.3.1 unsigned long IndexException::Select [static]

Definition at line 1639 of file newmat.h.

The documentation for this class was generated from the following files:

- newmat/include/[newmat.h](#)
- newmat/src/[newmatex.cpp](#)

## 29.229 Go::IndexMesh2DIterator Class Reference

```
#include <IndexMesh2DIterator.h>
```

## Public Member Functions

- [IndexMesh2DIterator](#) ([const Mesh2D](#) &m, int kind\_u=0, int kmul\_u=1, int kind\_v=0, int kmul\_v=1)
- [IndexMesh2DIterator](#) & operator++ ()
- [bool operator==](#) ([const IndexMesh2DIterator](#) &rhs) [const](#)
- [bool operator!=](#) ([const IndexMesh2DIterator](#) &rhs) [const](#)
- [const std::array< int, 8 > & operator\\*](#) () [const](#)

### 29.229.1 Detailed Description

Definition at line 50 of file [IndexMesh2DIterator.h](#).

### 29.229.2 Constructor & Destructor Documentation

29.229.2.1 [Go::IndexMesh2DIterator::IndexMesh2DIterator](#) ( [const Mesh2D](#) & m, int *kind\_u* = 0, int *kmul\_u* = 1, int *kind\_v* = 0, int *kmul\_v* = 1 )

### 29.229.3 Member Function Documentation

29.229.3.1 [bool Go::IndexMesh2DIterator::operator!=](#) ( [const IndexMesh2DIterator](#) & *rhs* ) [const](#) [\[inline\]](#)

Definition at line 77 of file [IndexMesh2DIterator.h](#).

29.229.3.2 [const std::array<int,8>& Go::IndexMesh2DIterator::operator\\*](#) ( ) [const](#) [\[inline\]](#)

Definition at line 98 of file [IndexMesh2DIterator.h](#).

29.229.3.3 [IndexMesh2DIterator& Go::IndexMesh2DIterator::operator++](#) ( )

29.229.3.4 [bool Go::IndexMesh2DIterator::operator==](#) ( [const IndexMesh2DIterator](#) & *rhs* ) [const](#) [\[inline\]](#)

Definition at line 71 of file [IndexMesh2DIterator.h](#).

The documentation for this class was generated from the following file:

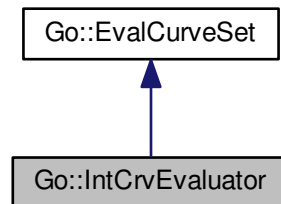
- [Irsplines2D/include/GoTools/Irsplines2D/IndexMesh2DIterator.h](#)



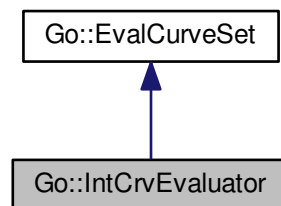
## 29.230 Go::IntCrvEvaluator Class Reference

```
#include <IntCrvEvaluator.h>
```

Inheritance diagram for Go::IntCrvEvaluator:



Collaboration diagram for Go::IntCrvEvaluator:



### Public Member Functions

- `IntCrvEvaluator` (`shared_ptr< CurveOnSurface > sfcv1`, `double start1`, `double end1`, `shared_ptr< CurveOnSurface > sfcv2`, `double start2`, `double end2`, `bool same_orientation`, `int keep_crv=0`)

*Constructor.*

- virtual `~IntCrvEvaluator` ()

*Destructor.*

- virtual `std::vector< Point > eval` (`double t`)
- virtual `void eval` (`double t`, `int n`, `std::vector< std::vector< Point > > &der`)
- virtual `double start` ()
- virtual `double end` ()
- virtual `int dim` ()
- virtual `bool approximationOK` (`double par`, `const std::vector< Point > &approxpos`, `double tol1`, `double tol2`)
- virtual `int nmbCvs` ()
- virtual `void resetErr` ()

*Reset intermediate error. New iterationa.*

- `double getMaxErr` ()

*Get maximum approximation error.*

### 29.230.1 Detailed Description

This class represents an intersection curve approximated by two curve on surface instances

Definition at line 51 of file IntCrvEvaluator.h.

### 29.230.2 Constructor & Destructor Documentation

29.230.2.1 `Go::IntCrvEvaluator::IntCrvEvaluator ( shared_ptr< CurveOnSurface > sfcv1, double start1, double end1, shared_ptr< CurveOnSurface > sfcv2, double start2, double end2, bool same_orientation, int keep_crv = 0 )`

Constructor.

29.230.2.2 `virtual Go::IntCrvEvaluator::~IntCrvEvaluator ( ) [virtual]`

Destructor.

### 29.230.3 Member Function Documentation

29.230.3.1 `virtual bool Go::IntCrvEvaluator::approximationOK ( double par, const std::vector< Point > & approxpos, double tol1, double tol2 ) [virtual]`

Whether the approximation is within tolerances in input parameter.

Parameters

<i>par</i>	parameter in which to evaluate.
<i>approxpos</i>	whether the input points are within tolerance from the evaluated points (as given by <a href="#">eval()</a> ).
<i>tol1</i>	tolerance used to decide approximation accuracy.
<i>tol2</i>	tolerance used to decide approximation accuracy.

Returns

whether the approximation is within tolerances in input parameter.

Implements [Go::EvalCurveSet](#).

29.230.3.2 `virtual int Go::IntCrvEvaluator::dim ( ) [virtual]`

The geometric dimension of the spline curves.

Returns

geometric dimension of the total space.

Implements [Go::EvalCurveSet](#).

29.230.3.3 `virtual double Go::IntCrvEvaluator::end ( ) [virtual]`

End parameter of domain.

Returns

end parameter of the spline space.

Implements [Go::EvalCurveSet](#).

29.230.3.4 `virtual std::vector<Point> Go::IntCrvEvaluator::eval ( double t ) [virtual]`

Evaluate the curves.

Parameters

<i>t</i>	parameter in which to evaluate.
----------	---------------------------------

Returns

the evaluated points for the curve set.

Implements [Go::EvalCurveSet](#).

29.230.3.5 `virtual void Go::IntCrvEvaluator::eval ( double t, int n, std::vector< std::vector< Point > > & der ) [virtual]`

Evaluate the curve derivatives.

Parameters

<i>t</i>	parameter in which to evaluate.
<i>n</i>	number of derivatives to compute.
<i>der</i>	the evaluated points up to the n'th derivative for the curve set.

Implements [Go::EvalCurveSet](#).

29.230.3.6 `double Go::IntCrvEvaluator::getMaxErr ( ) [inline]`

Get maximum approximation error.

Definition at line 109 of file IntCrvEvaluator.h.

29.230.3.7 `virtual int Go::IntCrvEvaluator::nmbCvs ( ) [virtual]`

The number of curves in the curve set.

**Returns**

the number of curves in the curve set.

Implements [Go::EvalCurveSet](#).

**29.230.3.8** `virtual void Go::IntCrvEvaluator::resetErr( ) [inline],[virtual]`

Reset intermediate error. New iterationa.

Reimplemented from [Go::EvalCurveSet](#).

Definition at line 103 of file IntCrvEvaluator.h.

**29.230.3.9** `virtual double Go::IntCrvEvaluator::start( ) [virtual]`

Start parameter of domain.

**Returns**

start parameter of the spline space.

Implements [Go::EvalCurveSet](#).

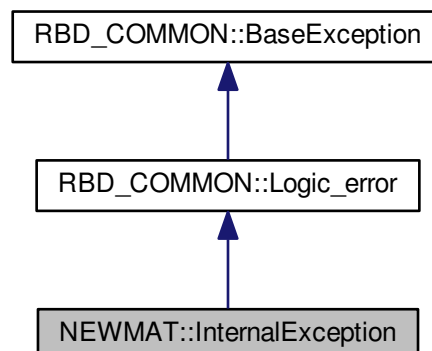
The documentation for this class was generated from the following file:

- `gotools-core/include/GoTools/creators/IntCrvEvaluator.h`

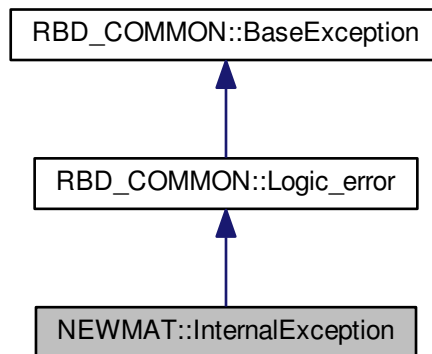
## 29.231 NEWMAT::InternalException Class Reference

```
#include <newmat.h>
```

Inheritance diagram for NEWMAT::InternalException:



Collaboration diagram for NEWMAT::InternalException:



### Public Member Functions

- [InternalException](#) (`const char *c`)

### Static Public Attributes

- static unsigned long [Select](#)

### Additional Inherited Members

#### 29.231.1 Detailed Description

Definition at line 1692 of file newmat.h.

#### 29.231.2 Constructor & Destructor Documentation

##### 29.231.2.1 `InternalException::InternalException ( const char * c )`

Definition at line 244 of file newmatex.cpp.

#### 29.231.3 Member Data Documentation

##### 29.231.3.1 `unsigned long InternalException::Select [static]`

Definition at line 1695 of file newmat.h.

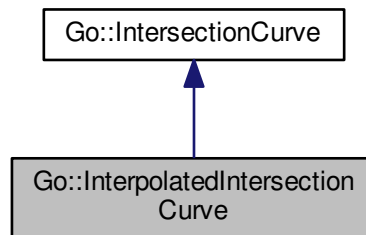
The documentation for this class was generated from the following files:

- newmat/include/newmat.h
- newmat/src/newmatex.cpp

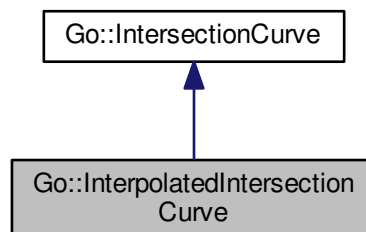
## 29.232 Go::InterpolatedIntersectionCurve Class Reference

```
#include <IntersectionCurve.h>
```

Inheritance diagram for Go::InterpolatedIntersectionCurve:



Collaboration diagram for Go::InterpolatedIntersectionCurve:



### Public Member Functions

- virtual [~InterpolatedIntersectionCurve](#) ()
- virtual void [refine](#) (const double &pos\_tol, const double &angle\_tol)
- virtual shared\_ptr< [ParamCurve](#) > [getCurve](#) () const
- virtual shared\_ptr< [ParamCurve](#) > [getParamCurve](#) (int obj\_nmb) const
- virtual [bool](#) [isIsocurve](#) () const
- virtual [bool](#) [isDegenerated](#) () const
- virtual void [getParamSpan](#) (double &start, double &end) const
- virtual void [evaluateAt](#) (double pval, [Point](#) &pos, [Point](#) &tan)
- virtual [bool](#) [getGuidePointTangent](#) (shared\_ptr< [IntersectionPoint](#) > pt, [Point](#) &tan, int type) const
- void [write](#) (std::ostream &os) const
- void [read](#) (std::istream &is) const

## Friends

- `template<class iterator >`  
`shared_ptr< IntersectionCurve > constructIntersectionCurve (const iterator begin, const iterator end)`

## Additional Inherited Members

### 29.232.1 Detailed Description

[IntersectionCurve](#) that is defined by Hermite interpolation of a number of guidepoints.

Definition at line 393 of file `IntersectionCurve.h`.

### 29.232.2 Constructor & Destructor Documentation

29.232.2.1 `virtual Go::InterpolatedIntersectionCurve::~InterpolatedIntersectionCurve ( ) [inline], [virtual]`

Definition at line 397 of file `IntersectionCurve.h`.

### 29.232.3 Member Function Documentation

29.232.3.1 `virtual void Go::InterpolatedIntersectionCurve::evaluateAt ( double pval, Point & pos, Point & tan ) [virtual]`

Evaluate the [IntersectionCurve](#) at parameter value *pval*. Its position at this parameter value will be returned in *pos* and its tangent direction will be returned in *tan*.

#### Parameters

<i>pval</i>	the parameter value for which to evaluate the <a href="#">IntersectionCurve</a> . It should be within the parametric span (which can be obtained from <code>GetParamSpan()</code> ).
-------------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

#### Return values

<i>pos</i>	the calculated position of the point on the curve at parameter <i>pval</i> .
------------	------------------------------------------------------------------------------

#### Parameters

<i>tan</i>	the calculated tangent of the curve at parameter <i>pval</i> .
------------	----------------------------------------------------------------

Implements [Go::IntersectionCurve](#).

29.232.3.2 `virtual shared_ptr<ParamCurve> Go::InterpolatedIntersectionCurve::getCurve ( ) const [virtual]`

Get out a (shared) pointer to a parametric curve that approximates this [IntersectionCurve](#).

Implements [Go::IntersectionCurve](#).

```
29.232.3.3 virtual bool Go::InterpolatedIntersectionCurve::getGuidePointTangent (shared_ptr< IntersectionPoint > pt,
Point & tan, int type) const [virtual]
```

Get the tangent of a guidepoint that we know lies on the curve (just using the `getTangent()` function of the guidepoint can be deceiving, as it is not necessarily unique).

#### Parameters

<i>pt</i>	shared pointer to the guidepoint. It is supposed to be part of the definition of the curve.
<i>type</i>	specifies what kind of tangent the user wants. '1' means the parameter plane tangent for the first object. '2' means the parameter plane tangent for the second object. Other values of 'type' means the tangent in 3D space.

#### Return values

<i>tan</i>	the requested tangent
------------	-----------------------

#### Returns

'true' if a tangent was found (i.e. *pt* was indeed a guidepoint of the [IntersectionCurve](#)). In the opposite case, 'false' is returned.

Reimplemented from [Go::IntersectionCurve](#).

```
29.232.3.4 virtual shared_ptr<ParamCurve> Go::InterpolatedIntersectionCurve::getParamCurve (int obj_nmb) const
[virtual]
```

Get out a (shared) pointer to the curve in the parametric plane of the first or second object. Should only be called when the concerned object is 2-parametric.

#### Parameters

<i>obj_nmb</i>	should be either 1 or 2, depending on whether you want to get the parametric curve in object 1 or object 2.
----------------	-------------------------------------------------------------------------------------------------------------

#### Returns

a shared pointer to a curve that approximates the intersection curve in the parametric domain of the specified object.

Implements [Go::IntersectionCurve](#).

```
29.232.3.5 virtual void Go::InterpolatedIntersectionCurve::getParamSpan (double & start, double & end) const
[inline],[virtual]
```

Get the start and end value for the parametric span of the [IntersectionCurve](#).



## Return values

<i>start</i>	upon function return this variable will contain the start value of the parametric span.
<i>end</i>	upon function return this variable will contain the end value of the parametric span.

Implements [Go::IntersectionCurve](#).

Definition at line 411 of file IntersectionCurve.h.

**29.232.3.6** `virtual bool Go::InterpolatedIntersectionCurve::isDegenerated ( ) const [inline],[virtual]`

Determine if the [IntersectionCurve](#) is in fact a degenerated curve (a single point in space).

## Returns

'true' if the [IntersectionCurve](#) is degenerated. 'false' otherwise.

Implements [Go::IntersectionCurve](#).

Definition at line 408 of file IntersectionCurve.h.

**29.232.3.7** `virtual bool Go::InterpolatedIntersectionCurve::isIsocurve ( ) const [inline],[virtual]`

Determine if the [IntersectionCurve](#) is in fact representing an isoparametric intersection.

## Returns

'true' if the [IntersectionCurve](#) is part of an isocurve for one of the objects, 'false' otherwise.

Implements [Go::IntersectionCurve](#).

Definition at line 405 of file IntersectionCurve.h.

**29.232.3.8** `void Go::InterpolatedIntersectionCurve::read ( std::istream & is ) const`

**29.232.3.9** `virtual void Go::InterpolatedIntersectionCurve::refine ( const double & pos_tol, const double & angle_tol ) [virtual]`

Refine the curve (which means to increase the number of [IntersectionPoint](#) s defining it), so that the curve matches a specific positional and angle tolerance.

## Parameters

<i>pos_tol</i>	the positional tolerance that the curve should match after refinement.
<i>angle_tol</i>	the angular tolerance that the tangent of the curve should match after refinement.

Implements [Go::IntersectionCurve](#).

29.232.3.10 `void Go::InterpolatedIntersectionCurve::write ( std::ostream & os ) const`

## 29.232.4 Friends And Related Function Documentation

29.232.4.1 `template<class iterator > shared_ptr<IntersectionCurve> constructIntersectionCurve ( const iterator begin, const iterator end ) [friend]`

Definition at line 601 of file IntersectionCurve.h.

The documentation for this class was generated from the following file:

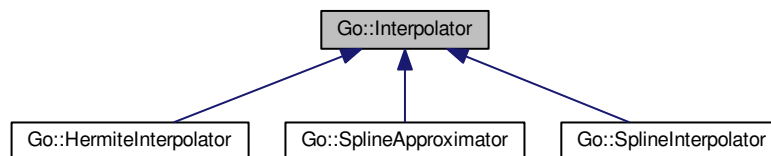
- [intersections/include/GoTools/intersections/IntersectionCurve.h](#)

## 29.233 Go::Interpolator Class Reference

Base class for spline interpolators or approximators.

```
#include <Interpolator.h>
```

Inheritance diagram for Go::Interpolator:



### Public Member Functions

- virtual `~Interpolator ()`  
*virtual desctructor assures safe inheritance*
- virtual void `interpolate (int num_points, int dimension, const double *param_start, const double *data_start, std::vector< double > &coefs)=0`
- virtual `const BsplineBasis & basis ()=0`

### 29.233.1 Detailed Description

Base class for spline interpolators or approximators.

Definition at line 52 of file Interpolator.h.

## 29.233.2 Constructor & Destructor Documentation

29.233.2.1 virtual `Go::Interpolator::~~Interpolator ( )` [virtual]

virtual desctructor assures safe inheritance

## 29.233.3 Member Function Documentation

29.233.3.1 virtual `const BsplineBasis& Go::Interpolator::basis ( )` [pure virtual]

after the function `interpolate()` has been successfully run, this function can be called to get the `BsplineBasis` of the generated curve.

### Returns

a constant reference to the `BsplineBasis` of the curve previously generated by `interpolate()`.

Implemented in `Go::SplineInterpolator`, `Go::SplineApproximator`, and `Go::HermitInterpolator`.

29.233.3.2 virtual `void Go::Interpolator::interpolate ( int num_points, int dimension, const double * param_start, const double * data_start, std::vector< double > & coefs )` [pure virtual]

Interpolate or approximate a set of point data by a spline

### Parameters

<i>num_points</i>	the number of data points to approximate
<i>dimension</i>	the dimension of each data point
<i>param_start</i>	pointer to the array where the points' parameters are consecutively stored
<i>data_start</i>	pointer to the array where the points are consecutively stored
<i>coefs</i>	upon function completion, this vector will contain the control points of the interpolating (or approximating spline). (In order to get access to the associated <code>BsplineBasis</code> , use the function <code>basis()</code> ).

Implemented in `Go::SplineInterpolator`, `Go::HermitInterpolator`, and `Go::SplineApproximator`.

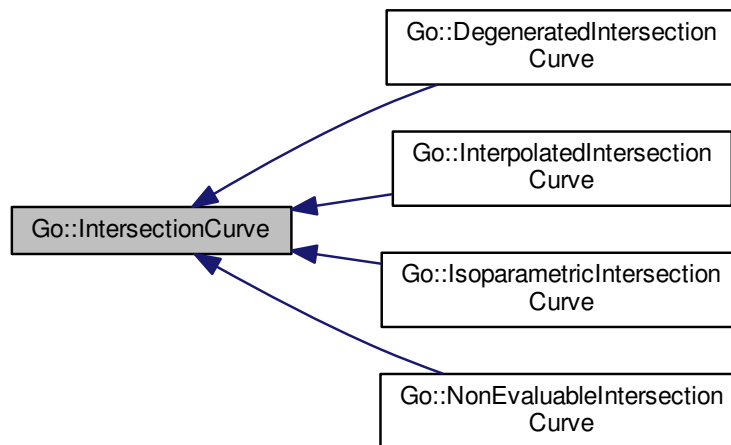
The documentation for this class was generated from the following file:

- `gtools-core/include/GoTools/geometry/Interpolator.h`

## 29.234 Go::IntersectionCurve Class Reference

```
#include <IntersectionCurve.h>
```

Inheritance diagram for Go::IntersectionCurve:



## Public Member Functions

- virtual [~IntersectionCurve](#) ()  
*Virtual destructor.*
- virtual [shared\\_ptr< ParamCurve > getCurve](#) () const =0
- virtual [shared\\_ptr< ParamCurve > getParamCurve](#) (int obj\_nmb) const =0
- virtual void [getParamSpan](#) (double &start, double &end) const =0
- virtual void [evaluateAt](#) (double pval, [Point](#) &pos, [Point](#) &tan)=0
- virtual void [refine](#) (const double &pos\_tol, const double &angle\_tol)=0
- virtual [bool isIsocurve](#) () const =0
- virtual [bool isDegenerated](#) () const =0
- [shared\\_ptr< IntersectionPoint > getGuidePoint](#) (int index) const
- virtual [bool getGuidePointTangent](#) (shared\_ptr< [IntersectionPoint](#) > pt, [Point](#) &tan, int type=0) const
- int [numGuidePoints](#) () const
- void [writeIPointsToStream](#) (std::ostream &os) const

## Protected Member Functions

- template<class iterator >  
[IntersectionCurve](#) (iterator begin, iterator end)

## Protected Attributes

- std::list< [shared\\_ptr< IntersectionPoint >](#) > [ipoints\\_](#)

## Friends

- template<class iterator >  
[shared\\_ptr< IntersectionCurve > constructIntersectionCurve](#) (const iterator begin, const iterator end)

### 29.234.1 Detailed Description

Object describing the curve that results from an intersection of two geometrical objects.

Definition at line 96 of file IntersectionCurve.h.

### 29.234.2 Constructor & Destructor Documentation

29.234.2.1 `virtual Go::IntersectionCurve::~IntersectionCurve ( ) [inline],[virtual]`

Virtual destructor.

Definition at line 100 of file IntersectionCurve.h.

29.234.2.2 `template<class iterator > Go::IntersectionCurve::IntersectionCurve ( iterator begin, iterator end ) [inline],[protected]`

Definition at line 200 of file IntersectionCurve.h.

### 29.234.3 Member Function Documentation

29.234.3.1 `virtual void Go::IntersectionCurve::evaluateAt ( double pval, Point & pos, Point & tan ) [pure virtual]`

Evaluate the [IntersectionCurve](#) at parameter value *pval*. Its position at this parameter value will be returned in *pos* and its tangent direction will be returned in *tan*.

#### Parameters

<i>pval</i>	the parameter value for which to evaluate the <a href="#">IntersectionCurve</a> . It should be within the parametric span (which can be obtained from <code>GetParamSpan()</code> ).
-------------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

#### Return values

<i>pos</i>	the calculated position of the point on the curve at parameter <i>pval</i> .
------------	------------------------------------------------------------------------------

#### Parameters

<i>tan</i>	the calculated tangent of the curve at parameter <i>pval</i> .
------------	----------------------------------------------------------------

Implemented in [Go::InterpolatedIntersectionCurve](#), [Go::IsoparametricIntersectionCurve](#), [Go::NonEvaluableIntersectionCurve](#), and [Go::DegeneratedIntersectionCurve](#).

29.234.3.2 `virtual shared_ptr<ParamCurve> Go::IntersectionCurve::getCurve ( ) const [pure virtual]`

Get out a (shared) pointer to a parametric curve that approximates this [IntersectionCurve](#).

Implemented in [Go::InterpolatedIntersectionCurve](#), [Go::IsoparametricIntersectionCurve](#), [Go::NonEvaluableIntersectionCurve](#), and [Go::DegeneratedIntersectionCurve](#).

29.234.3.3 `shared_ptr<IntersectionPoint> Go::IntersectionCurve::getGuidePoint ( int index ) const`

Get one of the specific guidepoints ([IntersectionPoint](#)) that define the curve.

#### Parameters

<i>index</i>	the index of the requested <a href="#">IntersectionPoint</a> (from 0 and up to the number of defining <a href="#">IntersectionPoint</a> s - 1). To find out how many <a href="#">IntersectionPoint</a> s participate in the definition of the curve, use the <a href="#">numGuidePoints()</a> function.
--------------	---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

#### Returns

a shared pointer to the requested [IntersectionPoint](#).

29.234.3.4 `virtual bool Go::IntersectionCurve::getGuidePointTangent ( shared_ptr< IntersectionPoint > pt, Point & tan, int type = 0 ) const [virtual]`

Get the tangent of a guidepoint that we know lies on the curve (just using the `getTangent()` function of the guidepoint can be deceiving, as it is not necessarily unique).

#### Parameters

<i>pt</i>	shared pointer to the guidepoint. It is supposed to be part of the definition of the curve.
<i>type</i>	specifies what kind of tangent the user wants. '1' means the parameter plane tangent for the first object. '2' means the parameter plane tangent for the second object. Other values of 'type' means the tangent in 3D space.

#### Return values

<i>tan</i>	the requested tangent
------------	-----------------------

#### Returns

'true' if a tangent was found (i.e. *pt* was indeed a guidepoint of the [IntersectionCurve](#)). In the opposite case, 'false' is returned.

Reimplemented in [Go::InterpolatedIntersectionCurve](#).

29.234.3.5 `virtual shared_ptr<ParamCurve> Go::IntersectionCurve::getParamCurve ( int obj_nmb ) const [pure virtual]`

Get out a (shared) pointer to the curve in the parametric plane of the first or second object. Should only be called when the concerned object is 2-parametric.

## Parameters

<i>obj_nmb</i>	should be either 1 or 2, depending on whether you want to get the parametric curve in object 1 or object 2.
----------------	-------------------------------------------------------------------------------------------------------------

## Returns

a shared pointer to a curve that approximates the intersection curve in the parametric domain of the specified object.

Implemented in [Go::InterpolatedIntersectionCurve](#), [Go::IsoparametricIntersectionCurve](#), [Go::NonEvaluableIntersectionCurve](#), and [Go::DegeneratedIntersectionCurve](#).

```
29.234.3.6 virtual void Go::IntersectionCurve::getParamSpan (double & start, double & end) const [pure virtual]
```

Get the start and end value for the parametric span of the [IntersectionCurve](#).

## Return values

<i>start</i>	upon function return this variable will contain the start value of the parametric span.
<i>end</i>	upon function return this variable will contain the end value of the parametric span.

Implemented in [Go::InterpolatedIntersectionCurve](#), [Go::IsoparametricIntersectionCurve](#), [Go::NonEvaluableIntersectionCurve](#), and [Go::DegeneratedIntersectionCurve](#).

```
29.234.3.7 virtual bool Go::IntersectionCurve::isDegenerated () const [pure virtual]
```

Determine if the [IntersectionCurve](#) is in fact a degenerated curve (a single point in space).

## Returns

'true' if the [IntersectionCurve](#) is degenerated. 'false' otherwise.

Implemented in [Go::InterpolatedIntersectionCurve](#), [Go::IsoparametricIntersectionCurve](#), [Go::NonEvaluableIntersectionCurve](#), and [Go::DegeneratedIntersectionCurve](#).

```
29.234.3.8 virtual bool Go::IntersectionCurve::isIsocurve () const [pure virtual]
```

Determine if the [IntersectionCurve](#) is in fact representing an isoparametric intersection.

## Returns

'true' if the [IntersectionCurve](#) is part of an isocurve for one of the objects, 'false' otherwise.

Implemented in [Go::InterpolatedIntersectionCurve](#), [Go::IsoparametricIntersectionCurve](#), [Go::NonEvaluableIntersectionCurve](#), and [Go::DegeneratedIntersectionCurve](#).

29.234.3.9 `int Go::IntersectionCurve::numGuidePoints ( ) const [inline]`

Get number of guide points defining the curve.

#### Returns

the number of guide points currently defining the curve (may increase if the [refine\(\)](#) function is run).

Definition at line 190 of file `IntersectionCurve.h`.

29.234.3.10 `virtual void Go::IntersectionCurve::refine ( const double & pos_tol, const double & angle_tol ) [pure virtual]`

Refine the curve (which means to increase the number of [IntersectionPoint](#) s defining it), so that the curve matches a specific positional and angle tolerance.

#### Parameters

<i>pos_tol</i>	the positional tolerance that the curve should match after refinement.
<i>angle_tol</i>	the angular tolerance that the tangent of the curve should match after refinement.

Implemented in [Go::InterpolatedIntersectionCurve](#), [Go::IsoparametricIntersectionCurve](#), [Go::NonEvaluableIntersectionCurve](#), and [Go::DegeneratedIntersectionCurve](#).

29.234.3.11 `void Go::IntersectionCurve::writePointsToStream ( std::ostream & os ) const`

Write the [IntersectionPoints](#) defining this [IntersectionCurve](#) to stream

#### Parameters

<i>os</i>	output stream
-----------	---------------

## 29.234.4 Friends And Related Function Documentation

29.234.4.1 `template<class iterator > shared_ptr<IntersectionCurve> constructIntersectionCurve ( const iterator begin, const iterator end ) [friend]`

Definition at line 601 of file `IntersectionCurve.h`.

## 29.234.5 Member Data Documentation

29.234.5.1 `std::list<shared_ptr<IntersectionPoint> > Go::IntersectionCurve::ipoints_ [protected]`

Definition at line 204 of file `IntersectionCurve.h`.

The documentation for this class was generated from the following file:

- [intersections/include/GoTools/intersections/IntersectionCurve.h](#)



## 29.235 Go::IntersectionLink Class Reference

```
#include <IntersectionLink.h>
```

### Public Member Functions

- [IntersectionLink](#) ([IntersectionPoint](#) \*p1, [IntersectionPoint](#) \*p2)  
*Constructor.*
- void [getIntersectionPoints](#) ([IntersectionPoint](#) \*&p1, [IntersectionPoint](#) \*&p2)
- void [getIntersectionPoints](#) ([const IntersectionPoint](#) \*&p1, [const IntersectionPoint](#) \*&p2) [const](#)
- [IntersectionPoint](#) \* [getOtherPoint](#) ([const IntersectionPoint](#) \*p1)
- bool [isAttachedTo](#) ([const IntersectionPoint](#) \*ip)  
*Returns true if the link is attached to the input point.*
- void [copyMetaInformation](#) ([const IntersectionLink](#) &rhs)
- bool [delimitsPAC](#) () [const](#)
- void [setPAC](#) (bool value)
- void [setIsoparametricIn](#) (int dir, bool iso)
- bool [isIsoparametricIn](#) (int dir) [const](#)
- bool [isIsoparametric](#) () [const](#)
- int [numParams](#) () [const](#)
- int [crosses](#) ([const shared\\_ptr](#)< [IntersectionLink](#) > &link) [const](#)  
*Test for crossing between 'this' and 'link'.*
- [const LinkType](#) & [linkType](#) () [const](#)
- [LinkType](#) & [linkType](#) ()
- void [writeInfo](#) () [const](#)  
*Writes info about the link to standard output.*
- void [dumpToStream](#) (std::ostream &os)

### 29.235.1 Detailed Description

This class represents a link between two intersection points. It is owned by both of the points in question, and can therefore refer to them only by means of ordinary pointers (no shared\_ptr stuff here...)

Definition at line 59 of file IntersectionLink.h.

### 29.235.2 Constructor & Destructor Documentation

**29.235.2.1** [Go::IntersectionLink::IntersectionLink](#) ([IntersectionPoint](#) \* p1, [IntersectionPoint](#) \* p2 ) [\[inline\]](#)

Constructor.

Definition at line 62 of file IntersectionLink.h.

### 29.235.3 Member Function Documentation

**29.235.3.1** void [Go::IntersectionLink::copyMetaInformation](#) ([const IntersectionLink](#) & rhs ) [\[inline\]](#)

Set all meta-information (all private info except IntersectionPoint-pointers) equal to the one in 'rhs'

Definition at line 104 of file IntersectionLink.h.

**29.235.3.2** `int Go::IntersectionLink::crosses ( const shared_ptr< IntersectionLink > & link ) const [inline]`

Test for crossing between 'this' and 'link'.

Definition at line 164 of file IntersectionLink.h.

**29.235.3.3** `bool Go::IntersectionLink::delimitsPAC ( ) const [inline]`

Find out whether this link is participating in defining a partial coincidence area

Definition at line 114 of file IntersectionLink.h.

**29.235.3.4** `void Go::IntersectionLink::dumpToStream ( std::ostream & os ) [inline]`

Definition at line 320 of file IntersectionLink.h.

**29.235.3.5** `void Go::IntersectionLink::getIntersectionPoints ( IntersectionPoint *& p1, IntersectionPoint *& p2 ) [inline]`

This function will set the 'p1' and 'p2' to point to the IntersectionPoints participating in this link.

Definition at line 73 of file IntersectionLink.h.

**29.235.3.6** `void Go::IntersectionLink::getIntersectionPoints ( const IntersectionPoint *& p1, const IntersectionPoint *& p2 ) const [inline]`

This function will set the 'p1' and 'p2' to point to the IntersectionPoints participating in this link.

Definition at line 82 of file IntersectionLink.h.

**29.235.3.7** `IntersectionPoint* Go::IntersectionLink::getOtherPoint ( const IntersectionPoint * p1 ) [inline]`

When one of the two points in the link is input, this function returns the other point.

Definition at line 91 of file IntersectionLink.h.

**29.235.3.8** `bool Go::IntersectionLink::isAttachedTo ( const IntersectionPoint * ip ) [inline]`

Returns true if the link is attached to the input point.

Definition at line 97 of file IntersectionLink.h.

**29.235.3.9** `bool Go::IntersectionLink::isIsoparametric ( ) const [inline]`

Definition at line 147 of file IntersectionLink.h.

**29.235.3.10** `bool Go::IntersectionLink::isIsoparametricIn ( int dir ) const [inline]`

Definition at line 141 of file IntersectionLink.h.

**29.235.3.11** `const LinkType& Go::IntersectionLink::linkType ( ) const [inline]`

Returns the link type of the link

## Return values

<i>a</i>	LinkType enum
----------	---------------

Definition at line 253 of file IntersectionLink.h.

**29.235.3.12 LinkType& Go::IntersectionLink::linkType ( )** [inline]

Returns the link type of the link

## Return values

<i>a</i>	LinkType enum
----------	---------------

Definition at line 258 of file IntersectionLink.h.

**29.235.3.13 int Go::IntersectionLink::numParams ( ) const** [inline]

Definition at line 158 of file IntersectionLink.h.

**29.235.3.14 void Go::IntersectionLink::setIsoparametricIn ( int *dir*, bool *iso* )** [inline]

Definition at line 124 of file IntersectionLink.h.

**29.235.3.15 void Go::IntersectionLink::setPAC ( bool *value* )** [inline]

Definition at line 119 of file IntersectionLink.h.

**29.235.3.16 void Go::IntersectionLink::writeInfo ( ) const** [inline]

Writes info about the link to standard output.

Definition at line 262 of file IntersectionLink.h.

The documentation for this class was generated from the following file:

- intersections/include/GoTools/intersections/[IntersectionLink.h](#)

## 29.236 Go::IntersectionPoint Class Reference

```
#include <IntersectionPoint.h>
```

## Public Member Functions

- [IntersectionPoint](#) ([const ParamObjectInt \\*obj1](#), [const ParamObjectInt \\*obj2](#), [const shared\\_ptr< GeoTol > epsge](#), [const double \\*obj1\\_params](#), [const double \\*obj2\\_params](#))
- [IntersectionPoint](#) ([const ParamObjectInt \\*obj1](#), [const ParamObjectInt \\*obj2](#), [const shared\\_ptr< IntersectionPoint > ip](#), [int missing\\_param](#))
- [~IntersectionPoint](#) ()
  - Destructor.*
- [IntersectionPoint](#) ()
  - Default constructor.*
- void [write](#) ([std::ostream &os](#)) [const](#)
- void [read](#) ([std::istream &is](#))
- void [writeParams](#) ([std::ostream &os](#)) [const](#)
- void [writeInfo](#) () [const](#)
- void [replaceParameter](#) ([double \\*param](#))
- [Point](#) [getPoint](#) () [const](#)
- [Point](#) [getPoint1](#) () [const](#)
- [Point](#) [getPoint2](#) () [const](#)
- void [projectToParamPlanes](#) ([const Point &dir](#), [Point &par\\_1\\_dir](#), [Point &par\\_2\\_dir](#)) [const](#)
- [bool](#) [hasUniqueTangentDirection](#) () [const](#)
- [bool](#) [tangentIsOriented](#) () [const](#)
- [double](#) [getDist](#) () [const](#)
- [const std::vector< double > &](#) [getPar](#) () [const](#)
- [double](#) [getPar](#) ([int idx](#)) [const](#)
- [const double \\*](#) [getPar1](#) () [const](#)
- [const double \\*](#) [getPar2](#) () [const](#)
- [Point](#) [getPar1Point](#) () [const](#)
- [Point](#) [getPar2Point](#) () [const](#)
- [int](#) [numParams1](#) () [const](#)
  - Get number of parameter values in object number 1.*
- [int](#) [numParams2](#) () [const](#)
  - Get number of parameter values in object number 2.*
- [Point](#) [getTangent](#) ([bool second\\_branch=false](#)) [const](#)
- [Point](#) [getPar1Dir](#) ([bool second\\_branch=false](#)) [const](#)
- [Point](#) [getPar2Dir](#) ([bool second\\_branch=false](#)) [const](#)
- [const ParamObjectInt \\*](#) [getObj1](#) () [const](#)
  - Get pointer to first object.*
- [const ParamObjectInt \\*](#) [getObj2](#) () [const](#)
  - Get pointer to second object.*
- [double](#) [startParam](#) ([int pdir](#))
- [double](#) [endParam](#) ([int pdir](#))
- [bool](#) [pointIsSingular](#) () [const](#)
- [bool](#) [isNearSingular](#) () [const](#)
- [double](#) [parameterTolerance](#) ([int pdir](#))
- void [fixTangentOrientation](#) ([bool flip](#))
- void [isBoundaryPoint](#) ([bool &first](#), [bool &second](#)) [const](#)
- void [tangentPointingInwards](#) ([bool &first](#), [bool &second](#), [bool &first\\_along\\_boundary](#), [bool &second\\_along\\_←\\_boundary](#)) [const](#)
- [bool](#) [isConnectedTo](#) ([shared\\_ptr< IntersectionPoint > p](#)) [const](#)
  - Check if this point is topologically connected with another [IntersectionPoint](#) (i.e. that there is an [IntersectionLink](#) between them).*
- [bool](#) [isConnectedTo](#) ([const IntersectionPoint \\*point](#)) [const](#)
  - Check if this point is topologically connected with another [IntersectionPoint](#) (i.e. that there is an [IntersectionLink](#) between them).*

- `shared_ptr< IntersectionLink > connectTo (IntersectionPoint *const point, LinkType type, shared_ptr< IntersectionLink > model_link=shared_ptr< IntersectionLink >(), int added_parameter_dir=-1)`
- `shared_ptr< IntersectionLink > connectTo (shared_ptr< IntersectionPoint > point, LinkType type, shared_ptr< IntersectionLink > model_link=shared_ptr< IntersectionLink >(), int added_parameter_dir=-1)`
- `void disconnectFrom (IntersectionPoint *point)`
- `int numBranches () const`
- `IntPtClassification getClassification (const ParamSurface *surf1, const ParamSurface *surf2, int branch_num=0) const`
- `IntPtClassification getClassification (const ParamSurface *par_func, double C, int branch_num=0) const`
- `SingularityType getSingularityType () const`  
*Get the SingularityType of this IntersectionPoint.*
- `shared_ptr< GeoTol > getTolerance ()`
- `shared_ptr< const GeoTol > getTolerance () const`
- `shared_ptr< IntersectionPoint > parentPoint ()`
- `void setParentPoint (shared_ptr< IntersectionPoint > p)`
- `bool hasParentPoint ()`
- `void flipObjects ()`
- `double getInfluenceArea (int dir, bool forward, bool first_outside, double aeps=0.0) const`
- `int inInfluenceArea (int paddir, double par, bool first_outside) const`
- `void shareInfluenceAreaWith (shared_ptr< IntersectionPoint > other_pt, int paddir)`
- `bool inInfluenceAreaBracket (int paddir, double parval)`
- `shared_ptr< IntersectionLink > getIntersectionLink (const IntersectionPoint *point)`
- `void getNeighbours (std::vector< IntersectionPoint * > &pnts) const`
- `void getNeighbourLinks (std::vector< shared_ptr< IntersectionLink > > &links) const`
- `int numNeighbours () const`
- `template<class bool_iterator > void setDifferentiateFromLeft (bool_iterator it) const`
- `bool differentiatesFromLeft (int paddir) const`
- `int hasIsoLinks (int *const iso_par_dirs) const`
- `int commonIsoLinks (int *const iso_par_dirs) const`
- `bool containsG2Discontinuity () const`
- `bool isDegenerate () const`
- `bool isSamePoint (const IntersectionPoint *point) const`
- `bool checkIntersectionPoint (IntPtInfo &int_pt_info) const`
- `bool isInDomain (double *frompar, double *topar) const`

## Static Public Attributes

- static `const double tangent_tol`

### 29.236.1 Detailed Description

Object describing a point located on the intersection of two geometrical objects.

Definition at line 65 of file IntersectionPoint.h.

### 29.236.2 Constructor & Destructor Documentation

- 29.236.2.1 `Go::IntersectionPoint::IntersectionPoint ( const ParamObjectInt * obj1, const ParamObjectInt * obj2, const shared_ptr< GeoTol > epsge, const double * obj1_params, const double * obj2_params )`

Constructor. This constructor specifies an [IntersectionPoint](#) from all its defining components.

## Parameters

<i>obj1</i>	the first of the two intersecting objects on which the <a href="#">IntersectionPoint</a> lies.
<i>obj2</i>	the second of the two intersecting objects on which the <a href="#">IntersectionPoint</a> lies.
<i>epsge</i>	an object defining various tolerances
<i>obj_1_params</i>	parameters of the intersection point for the first object
<i>obj_2_params</i>	parameters of the intersection point for the second object

29.236.2.2 `Go::IntersectionPoint::IntersectionPoint ( const ParamObjectInt * obj1, const ParamObjectInt * obj2, const shared_ptr< IntersectionPoint > ip, int missing_param )`

Constructor. This constructor is for creating an [IntersectionPoint](#) that is a replica of an existing one, but where one of the objects has one parameter less (i.e. is 'picked' from one of the objects of the existing [IntersectionPoint](#) by fixing one of its parameters).

## Parameters

<i>obj1</i>	the first of the two intersecting objects on which the <a href="#">IntersectionPoint</a> lies. It is either the same as the first object defining the point <i>ip</i> , or picked from it (one parameter less).
<i>obj2</i>	the second of the two intersecting objects on which the <a href="#">IntersectionPoint</a> lies. It is either the same as the second object defining the point <i>ip</i> , or picked from it (one parameter less).
<i>ip</i>	The already-existing <a href="#">IntersectionPoint</a> that 'this' <a href="#">IntersectionPoint</a> will represent.
<i>missing_param</i>	indicates which parameter of the point <i>ip</i> is 'missing' for 'this' <a href="#">IntersectionPoint</a> . (Parameters are numbered from 0 and up to the total number of parameters in 'obj1' and 'obj2').

29.236.2.3 `Go::IntersectionPoint::~~IntersectionPoint ( )`

Destructor.

29.236.2.4 `Go::IntersectionPoint::IntersectionPoint ( ) [inline]`

Default constructor.

Definition at line 112 of file `IntersectionPoint.h`.

## 29.236.3 Member Function Documentation

29.236.3.1 `bool Go::IntersectionPoint::checkIntersectionPoint ( IntPtInfo & int_pt_info ) const`

Check if the intersection point is OK, and get some info about the point.

## Parameters

<i>int_pt_info</i>	struct of type <a href="#">IntPtInfo</a> with info about the point. Indicates if the point is ok, the number of neighbours, the singularity type, and the location of the point is the parameter domains of the objects
--------------------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

**Returns**

'true' if the intersection points is OK, 'false' otherwise

**29.236.3.2** `int Go::IntersectionPoint::commonIsoLinks ( int *const iso_par_dirs ) const`

Returns the number of parameter directions for which all [IntersectionLink](#) s of this [IntersectionPoint](#) are isoparametric. The indices of these parameter directions are written to the array pointed to by 'iso\_par\_dirs' (allocated by the user).

**Parameters**

<code>iso_par_dirs</code>	pointer to the array where the indices will be written.
---------------------------	---------------------------------------------------------

**Returns**

the number of parameter directions for which all [IntersectionLink](#) s of this [IntersectionPoint](#) are isoparametric.

**29.236.3.3** `shared_ptr<IntersectionLink> Go::IntersectionPoint::connectTo ( IntersectionPoint *const point, LinkType type, shared_ptr< IntersectionLink > model_link = shared_ptr< IntersectionLink > (), int added_parameter_dir = -1 )`

This function topologically "connects" this point with another [IntersectionPoint](#). It generates a new [IntersectionLink](#) combining these points. This [IntersectionLink](#) can inherit the meta-information from `model_link`. This is typically useful when an [IntersectionPoint](#) is "inserted" between two already-connected points, and the two new links should contain the same information as the older, split-up link. Another example where information from `model_link` needs to be copied is when parents of two connected points needs to be connected. In that case, it is important that we know which parameter direction has been introduced when moving to the parent, so that we can "shift" the meta-information pertaining to individual parameters accordingly. If no new parameter information has been introduced, `added_parameter_dir` should be left at its default value of -1.

**Parameters**

<code>point</code>	the <a href="#">IntersectionPoint</a> we want to connect to <code>this</code> point
<code>type</code>	the type of the link in terms of a <code>LinkType</code>
<code>model_link</code>	if this parameter is specified, the information associated with <code>model_link</code> will be copied by the newly created <a href="#">IntersectionLink</a> .
<code>added_parameter_dir</code>	relevant parameter when <code>model_link</code> contains one parameter direction less than the new <a href="#">IntersectionLink</a> does. In this case, <code>added_parameter_dir</code> will specify the index of the parameter that is lacking.

**Returns**

shared pointer to the generated link

29.236.3.4 `shared_ptr<IntersectionLink> Go::IntersectionPoint::connectTo ( shared_ptr< IntersectionPoint > point, LinkType type, shared_ptr< IntersectionLink > model_link = shared_ptr<IntersectionLink> (), int added_parameter_dir = -1 ) [inline]`

This function topologically "connects" this point with another [IntersectionPoint](#). It generates a new [IntersectionLink](#) combining these points. This [IntersectionLink](#) can inherit the meta-information from *model\_link*. This is typically useful when an [IntersectionPoint](#) is "inserted" between two already-connected points, and the two new links should contain the same information as the older, split-up link. Another example where information from *model\_link* needs to be copied is when parents of two connected points needs to be connected. In that case, it is important that we know which parameter direction has been introduced when moving to the parent, so that we can "shift" the meta-information pertaining to individual parameters accordingly. If no new parameter information has been introduced, *added\_parameter\_dir* should be left at its default value of -1.

#### Parameters

<i>point</i>	the <a href="#">IntersectionPoint</a> we want to connect to <code>this</code> point
<i>type</i>	the type of the link in terms of a <code>LinkType</code>
<i>model_link</i>	if this parameter is specified, the information associated with <i>model_link</i> will be copied by the newly created <a href="#">IntersectionLink</a> .
<i>added_parameter_dir</i>	relevant parameter when <i>model_link</i> contains one parameter direction less than the new <a href="#">IntersectionLink</a> does. In this case, <i>added_parameter_dir</i> will specify the index of the parameter that is lacking.

#### Returns

shared pointer to the generated link

Definition at line 457 of file `IntersectionPoint.h`.

29.236.3.5 `bool Go::IntersectionPoint::containsG2Discontinuity ( ) const [inline]`

Check if there is a G2-discontinuity for one of the parameters at this [IntersectionPoint](#).

#### Returns

'true' if at least one G2-discontinuity was found, 'false' otherwise.

Definition at line 702 of file `IntersectionPoint.h`.

29.236.3.6 `bool Go::IntersectionPoint::differentiatesFromLeft ( int pdir ) const [inline]`

Returns whether the internal state for parameter 'pdir' is set so that it differentiates from left.

#### Parameters

<i>pdir</i>	the concerned parameter direction
-------------	-----------------------------------



**Returns**

'true' if differentiation is set to be carried out 'from the left' for this parameter, 'false' otherwise.

Definition at line 669 of file IntersectionPoint.h.

**29.236.3.7 void Go::IntersectionPoint::disconnectFrom ( IntersectionPoint \* *point* )**

Topologically disconnect 'this' [IntersectionPoint](#) from another [IntersectionPoint](#) (which means removing whatever [IntersectionLink](#) there might be between them).

**Parameters**

<i>point</i>	the <a href="#">IntersectionPoint</a> we want to disconnect from 'this' <a href="#">IntersectionPoint</a> .
--------------	-------------------------------------------------------------------------------------------------------------

**29.236.3.8 double Go::IntersectionPoint::endParam ( int *parDir* )**

Get the end value of the parameter interval for the specified parameter.

**Parameters**

<i>parDir</i>	the parameter for which we seek the end value.
---------------	------------------------------------------------

**Returns**

the end value of the parameter interval for the parameter 'parDir'.

**29.236.3.9 void Go::IntersectionPoint::fixTangentOrientation ( bool *flip* )**

If the tangent is not oriented (because of singular intersection), it can be set manually with this command. If the argument is "true", the tangent's orientation will be flipped. In the opposite case, it will be kept as it is. Subsequent calls to the [tangentIsOriented\(\)](#) function will now report that the tangent is oriented. Not to be called for an ORDINARY\_POINT (which should have been handled already). See [SingularityType](#) for a definition of what ORDINARY\_POINT means.

**Parameters**

<i>flip</i>	if 'true', then the tangent's orientation will be flipped, otherwise it will be kept as it is. In any case, the <a href="#">IntersectionPoint</a> will now consider the orientation of its tangent to be 'fixed'.
-------------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

**29.236.3.10 void Go::IntersectionPoint::flipObjects ( )**

Flip the two intersecting objects (useful sometimes for self-intersections, should be avoided otherwise).

**29.236.3.11** `IntPtClassification` `Go::IntersectionPoint::getClassification ( const ParamSurface * surf1, const ParamSurface * surf2, int branch_num = 0 ) const`

Classifies an [IntersectionPoint](#) according to the direction of the corresponding intersection curve with respect to the parameter domain of two surfaces. *branch\_num* indicates which tangential branch to use (usually there's just one (*branch\_num* should be 0), but for branchpoints there are two (*branch\_num* can be 0 or 1)).

#### Parameters

<i>surf1</i>	the first surface
<i>surf2</i>	the second surface
<i>branch_num</i>	indicates which tangential branch to use

#### Returns

the classification of this [IntersectionPoint](#) according to the domain of the given surfaces. See also [IntPtClassification](#).

**29.236.3.12** `IntPtClassification` `Go::IntersectionPoint::getClassification ( const ParamSurface * par_func, double C, int branch_num = 0 ) const`

Classifies an [IntersectionPoint](#) according to the direction of the corresponding intersection curve with respect to the parameter domain of a *function*. *branch\_num* indicates which tangential branch to use (usually there's just one (*branch\_num* should be 0), but for branchpoints there are two (*branch\_num* can be 0 or 1)).

#### Parameters

<i>par_func</i>	the function, expressed as a 1D <a href="#">ParamSurface</a>
<i>C</i>	the iso-value that defines the intersection on this function.
<i>branch_num</i>	indicates which tangential branch to use

#### Returns

the classification of this [IntersectionPoint](#) according to the domain of the given function. See also [IntPtClassification](#).

**29.236.3.13** `double` `Go::IntersectionPoint::getDist ( ) const` `[inline]`

Distance between the position of the [IntersectionPoint](#) as described in the first object and as described in the second object.

#### Returns

the distance between the two points describing the [IntersectionPoint](#) in each of the objects.

Definition at line 203 of file `IntersectionPoint.h`.

29.236.3.14 `double Go::IntersectionPoint::getInfluenceArea ( int dir, bool forward, bool first_outside, double aeps = 0.0 ) const`

Returns the parameter value corresponding to the influence area of the intersection point in a specified direction and orientation. Since the bounds of the influence area are bracketed, we do not have the exact value of its limits. If 'first\_outside' is 'true', we will return the first parameter value detected outside the influence area. Otherwise, we will return the last parameter detected inside the influence area.

#### Parameters

<i>dir</i>	the parameter direction for which we seek the influence area.
<i>forward</i>	'true' if we want to locate the upper end of the point's influence area, 'false' if we want to locate the lower end.
<i>first_outside</i>	'true' if we want the returned value to be the first parameter value <i>outside</i> the influence area that was found by the function. 'false' if we seek the last detected value <i>inside</i> the influence area.

#### Returns

a value that (according to the function arguments) either represents the last found parameter value *inside* or first found parameter value *outside* the influence area of this [IntersectionPoint](#) along the specified parameter direction.

29.236.3.15 `shared_ptr<IntersectionLink> Go::IntersectionPoint::getIntersectionLink ( const IntersectionPoint * point )`

Return the intersection link connecting `this` point with *point*. If no such link exists, a null pointer will be returned.

#### Parameters

<i>point</i>	the point for which we seek the <a href="#">IntersectionLink</a> to this point.
--------------	---------------------------------------------------------------------------------

#### Returns

the [IntersectionLink](#) in question.

29.236.3.16 `void Go::IntersectionPoint::getNeighbourLinks ( std::vector< shared_ptr< IntersectionLink > > & links ) const`

Get a vector containing all the [IntersectionLinks](#) that defines connections from `this` [IntersectionPoint](#).

#### Return values

<i>links</i>	upon function return, this will be the vector sought for.
--------------	-----------------------------------------------------------

29.236.3.17 `void Go::IntersectionPoint::getNeighbours ( std::vector< IntersectionPoint * > & pnts ) const`

Get a vector containing pointers of all [IntersectionPoints](#) topologically connected with this [IntersectionPoint](#). [IntersectionPoints](#) are topologically connected if they share an [IntersectionLink](#).

Return values

<i>pnts</i>	upon function return, this will be the vector sought for.
-------------	-----------------------------------------------------------

29.236.3.18 `const ParamObjectInt* Go::IntersectionPoint::getObj1 ( ) const` `[inline]`

Get pointer to first object.

Definition at line 287 of file [IntersectionPoint.h](#).

29.236.3.19 `const ParamObjectInt* Go::IntersectionPoint::getObj2 ( ) const` `[inline]`

Get pointer to second object.

Definition at line 291 of file [IntersectionPoint.h](#).

29.236.3.20 `const std::vector<double>& Go::IntersectionPoint::getPar ( ) const` `[inline]`

Get a vector containing the [IntersectionPoint](#)'s parameter values.

Definition at line 208 of file [IntersectionPoint.h](#).

29.236.3.21 `double Go::IntersectionPoint::getPar ( int idx ) const` `[inline]`

Get a specified parameter value of the [IntersectionPoint](#)

Parameters

<i>idx</i>	the number of their parameter for which we seek the value
------------	-----------------------------------------------------------

Returns

the specified parameter value

Definition at line 215 of file [IntersectionPoint.h](#).

29.236.3.22 `const double* Go::IntersectionPoint::getPar1 ( ) const`

Get a pointer to the parameter values in the first object

Returns

a pointer to the array where the parameter values for the first object are (consecutively) stored.

29.236.3.23 Point Go::IntersectionPoint::getPar1Dir ( bool *second\_branch* = false ) const

Get the parameter direction along intersection in first object at this [IntersectionPoint](#). If this point is a branch point (several intersection curves meet at this point), then the user has to specify the branch that she wants the tangent to.

## Parameters

<i>second_branch</i>	usually set to 'false' (default). If the <a href="#">IntersectionPoint</a> lies at a branchpoint, then the user can set this parameter to 'true' in order to get the parameter direction for the <i>second</i> of the two branches that supposedly meet in this point. However, setting this parameter to 'true' <i>will</i> cause an error if the <a href="#">IntersectionPoint</a> is <i>not</i> a branch point.
----------------------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

## Returns

the parameter direction along the intersection in the first object at this point.

## 29.236.3.24 Point Go::IntersectionPoint::getPar1Point ( ) const

Get a [Point](#) representing the [IntersectionPoint](#) in the parameter domain of the first object

## 29.236.3.25 const double\* Go::IntersectionPoint::getPar2 ( ) const

Get a pointer to the parameter values in the second object

## Returns

a pointer to the array where the parameter values for the second object are (consecutively) stored.

29.236.3.26 Point Go::IntersectionPoint::getPar2Dir ( bool *second\_branch* = false ) const

Get the parameter direction along intersection in second object at this [IntersectionPoint](#). If this point is a branch point (several intersection curves meet at this point), then the user has to specify the branch that she wants the tangent to.

## Parameters

<i>second_branch</i>	usually set to 'false' (default). If the <a href="#">IntersectionPoint</a> lies at a branchpoint, then the user can set this parameter to 'true' in order to get the parameter direction for the <i>second</i> of the two branches that supposedly meet in this point. However, setting this parameter to 'true' <i>will</i> cause an error if the <a href="#">IntersectionPoint</a> is <i>not</i> a branch point.
----------------------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

## Returns

the parameter direction along the intersection in the second object at this point.

**29.236.3.27 Point Go::IntersectionPoint::getPar2Point ( ) const**

Get a [Point](#) representing the [IntersectionPoint](#) in the parameter domain of the second object

**29.236.3.28 Point Go::IntersectionPoint::getPoint ( ) const**

Get average of the [IntersectionPoint](#)'s position in space as evaluated in the two underlying objects.

**Returns**

the position of the [IntersectionPoint](#) in space

**29.236.3.29 Point Go::IntersectionPoint::getPoint1 ( ) const** `[inline]`

Get the [IntersectionPoint](#)'s position in space as evaluated in the first underlying object.

**Returns**

the position of the [IntersectionPoint](#) in space

Definition at line 153 of file `IntersectionPoint.h`.

**29.236.3.30 Point Go::IntersectionPoint::getPoint2 ( ) const** `[inline]`

Get the [IntersectionPoint](#)'s position in space as evaluated in the second underlying object.

**Returns**

the position of the [IntersectionPoint](#) in space

Definition at line 159 of file `IntersectionPoint.h`.

**29.236.3.31 SingularityType Go::IntersectionPoint::getSingularityType ( ) const**

Get the `SingularityType` of this [IntersectionPoint](#).

**29.236.3.32 Point Go::IntersectionPoint::getTangent ( bool *second\_branch* = false ) const**

Get the tangent of the intersection at the point. If the point is a branch point (several intersection curves meet at this point), then the user has to specify the branch that she wants the tangent to.

## Parameters

<i>second_branch</i>	usually set to 'false' (default). If the <a href="#">IntersectionPoint</a> lies at a branchpoint, then the user can set this parameter to 'true' in order to get the tangent of the <i>second</i> of the two branches that supposedly meet in this point. However, setting this parameter to 'true' <i>will</i> cause an error if the <a href="#">IntersectionPoint</a> is <i>not</i> a branch point.
----------------------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

## Returns

the tangent of intersection curve at this point.

**29.236.3.33** `shared_ptr<GeoTol> Go::IntersectionPoint::getTolerance ( ) [inline]`

Get the object containing the *tolerances* used by this [IntersectionPoint](#)

Definition at line 516 of file IntersectionPoint.h.

**29.236.3.34** `shared_ptr<const GeoTol> Go::IntersectionPoint::getTolerance ( ) const [inline]`

Get the object containing the *tolerances* used by this [IntersectionPoint](#)

Definition at line 521 of file IntersectionPoint.h.

**29.236.3.35** `int Go::IntersectionPoint::hasIsoLinks ( int *const iso_par_dirs ) const`

Returns the number of parameter directions for which this point is connected with links that are isoparametric. The indices of the isoparametric parameter directions are written to an array pointed to by 'iso\_par\_dirs' (allocated by the user).

## Parameters

<i>iso_par_dirs</i>	pointer to the array where the indices will be written.
---------------------	---------------------------------------------------------

## Returns

the number of parameter directions for which this point is connected with links that are isoparametric.

**29.236.3.36** `bool Go::IntersectionPoint::hasParentPoint ( ) [inline]`

Definition at line 540 of file IntersectionPoint.h.

**29.236.3.37** `bool Go::IntersectionPoint::hasUniqueTangentDirection ( ) const`

Check if there is a unique tangent direction for the intersection at the position of this [IntersectionPoint](#). This is the case for ORDINARY\_POINTS and TANGENTIAL\_POINTS, and false in all other cases (see [SingularityType](#)). See also [tangentsOriented\(\)](#).

**Returns**

'true' if the [IntersectionPoint](#) has a unique tangent direction (not necessarily uniquely oriented). 'false' otherwise.

**29.236.3.38** `int Go::IntersectionPoint::inInfluenceArea ( int pardir, double par, bool first_outside ) const`

Returns 0 if the given parameter value *par* was not in the point's influence area for the point's parameter *pardir*. Returns 1 if it is inside the point's influence area, but not exactly on the point. Returns 2 if it is exactly on the point. Since the bounds of the influence area are bracketed, we do not have the exact value of its limits. If *first\_outside* is 'true', we will check against the first parameter value detected outside the influence area. Otherwise, we will check against the last parameter detected inside the influence area.

**Parameters**

<i>pardir</i>	the concerned parameter directon
<i>par</i>	the concerned parameter value
<i>first_outside</i>	specify whether we want to check against the first detected value <i>outside</i> the influence area ('true'), or the last detected value <i>inside</i> it ('false').

**Returns**

0, 1 or 2, according to the above description.

**29.236.3.39** `bool Go::IntersectionPoint::inInfluenceAreaBracket ( int pardir, double parval ) [inline]`

This function is not properly defined at the moment.

**Parameters**

<i>pardir</i>	the concerned parameter directon
<i>parval</i>	the concerned parameter value

Definition at line 609 of file `IntersectionPoint.h`.

**29.236.3.40** `void Go::IntersectionPoint::isBoundaryPoint ( bool & first, bool & second ) const`

The first and second argument are set to 'true' or 'false' according to whether the point is lying on the boundary of respectively the first and second object.

**Return values**

<i>first</i>	'true' if this <a href="#">IntersectionPoint</a> is located on the boundary of the first object, 'false' otherwise.
<i>second</i>	'true' if this <a href="#">IntersectionPoint</a> is located on the boundary of the second object, 'false' otherwise.



29.236.3.41 `bool Go::IntersectionPoint::isConnectedTo ( shared_ptr< IntersectionPoint > p ) const [inline]`

Check if this point is topologically connected with another [IntersectionPoint](#) (i.e. that there is an [IntersectionLink](#) between them).

#### Parameters

<i>p</i>	a shared pointer to the point we are checking against.
----------	--------------------------------------------------------

#### Returns

'true' if there is an [IntersectionLink](#) between 'this' point and *p*. 'false' otherwise.

Definition at line 385 of file IntersectionPoint.h.

29.236.3.42 `bool Go::IntersectionPoint::isConnectedTo ( const IntersectionPoint * point ) const`

Check if this point is topologically connected with another [IntersectionPoint](#) (i.e. that there is an [IntersectionLink](#) between them).

#### Parameters

<i>point</i>	a pointer to the point we are checking against.
--------------	-------------------------------------------------

#### Returns

'true' if there is an [IntersectionLink](#) between 'this' point and *point*. 'false' otherwise.

29.236.3.43 `bool Go::IntersectionPoint::isDegenerate ( ) const`

Check if the point lies on a degenerate edge (or other degenerate part) of the surface.

#### Returns

'true' if degenerate, 'false' otherwise.

29.236.3.44 `bool Go::IntersectionPoint::isInDomain ( double * frompar, double * topar ) const`

Check if the point is in the domain specified by a parameter box.

#### Parameters

<i>frompar</i>	the array of parameters defining the lower corner of the box in question.
<i>topar</i>	the array of parameters defining the upper corner of the box in question.

## Return values

--	--

29.236.3.45 **bool** Go::IntersectionPoint::isNearSingular ( ) const

Return 'true' if the point was found to be near-singular, that is, the tangent calculated in this point is very small.

29.236.3.46 **bool** Go::IntersectionPoint::isSamePoint ( const IntersectionPoint \* *point* ) const

Check if the given intersection point represents the same point as 'this' point within the tolerance.

## Parameters

<i>point</i>	pointer to the intersection point we wish to test on
--------------	------------------------------------------------------

## Return values

'true'	if the points are the same, 'false' otherwise
--------	-----------------------------------------------

29.236.3.47 **int** Go::IntersectionPoint::numBranches ( ) const

Returns how many 'tangential branches' emanates from this point (usually 1, but 2 for branch points, and 0 for isolated- or higher-order points). See [SingularityType](#) for a classification of [IntersectionPoint](#) s.

29.236.3.48 **int** Go::IntersectionPoint::numNeighbours ( ) const `[inline]`

Query the number of IntersectionPoints ('neighbours') that are connected to `this` [IntersectionPoint](#) via an [IntersectionLink](#).

Definition at line 638 of file IntersectionPoint.h.

29.236.3.49 **int** Go::IntersectionPoint::numParams1 ( ) const

Get number of parameter values in object number 1.

29.236.3.50 **int** Go::IntersectionPoint::numParams2 ( ) const

Get number of parameter values in object number 2.

29.236.3.51 **double** Go::IntersectionPoint::parameterTolerance ( int *pardir* )

Return the tolerance for the parameter in the direction 'pardir'. This is equal to `GeoTol::rel_par_res_` multiplied by the span of the parameter.

## Parameters

<i>parDir</i>	the parameter for which we want to find the parameter tolerance
---------------	-----------------------------------------------------------------

## Returns

the tolerance for the specified parameter.

29.236.3.52 `shared_ptr<IntersectionPoint> Go::IntersectionPoint::parentPoint ( ) [inline]`

Get a shared pointer to the parent point of this [IntersectionPoint](#). If it has no parent, the shared pointer will be a null-pointer.

Definition at line 527 of file `IntersectionPoint.h`.

29.236.3.53 `bool Go::IntersectionPoint::pointIsSingular ( ) const [inline]`

Return 'true' if the point was found to be singular, ie. not on a normal, transversal intersection (see [SingularityType](#)).

Definition at line 311 of file `IntersectionPoint.h`.

29.236.3.54 `void Go::IntersectionPoint::projectToParamPlanes ( const Point & dir, Point & par_1_dir, Point & par_2_dir ) const`

This function currently only works (and makes sense) for 3D geometrical objects (not functions). Given a vector 'dir', it projects this vector onto the two objects at the [IntersectionPoint](#), and calculates this direction in the two parameter spaces. These normalized directions are returned in 'par\_1\_dir' and 'par\_2\_dir'.

## Parameters

<i>dir</i>	the direction vector that the user wants to project on the objects
------------	--------------------------------------------------------------------

## Return values

<i>par_1_dir</i>	the direction of the projected vector in the parameter space of the first object.
<i>par_2_dir</i>	the direction of the projected vector in the parameter space of the second object.

29.236.3.55 `void Go::IntersectionPoint::read ( std::istream & is )`

Read [IntersectionPoint](#) from stream (NB: no topological or parent information)

## Parameters

<i>is</i>	input stream
-----------	--------------

29.236.3.56 `void Go::IntersectionPoint::replaceParameter ( double * param )`

Replace the parameters defining this [IntersectionPoint](#) and recalculate internal information (the underlying geometric objects are kept).

#### Parameters

<i>param</i>	pointer to an array of <code>double</code> s, defining the new values for the point's parameters. The array should of course have length equal to the total number of parameters defining this <a href="#">IntersectionPoint</a> .
--------------	------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

29.236.3.57 `template<class bool_iterator > void Go::IntersectionPoint::setDifferentiateFromLeft ( bool_iterator it ) const [inline]`

*it* should point to a p-array of `bool`, where *p* is the total number of parameters for this [IntersectionPoint](#). The value at *it*[*i*] determines whether second-order differentiation for the parameter *i* should be carried out from the left, rather than from the right (default). This sets an internal state variable that will not be changed until the next call of [setDifferentiateFromLeft\(\)](#). (The reason a template was used here is to be able to use iterators to vectors of `bool`, which are special in that they cannot be accessed in the usual pointer way).

#### Parameters

<i>it</i>	an iterator (pointer, etc.) to the aforementioned array of <code>bool</code> .
-----------	--------------------------------------------------------------------------------

Definition at line 653 of file `IntersectionPoint.h`.

29.236.3.58 `void Go::IntersectionPoint::setParentPoint ( shared_ptr< IntersectionPoint > p ) [inline]`

Set a given [IntersectionPoint](#) to be the parent point of this [IntersectionPoint](#).

#### Parameters

<i>p</i>	the point that shall be the parent point of this <a href="#">IntersectionPoint</a> .
----------	--------------------------------------------------------------------------------------

Definition at line 534 of file `IntersectionPoint.h`.

29.236.3.59 `void Go::IntersectionPoint::shareInfluenceAreaWith ( shared_ptr< IntersectionPoint > other_pt, int pdir )`

This function is mostly for optimizing reasons. This point is supposed to lie inside the influence area of *other\_pt*, and by this function, this influence interval is split up and shared between this point and the other point. By deducing this interval directly from *other\_pt*, we avoid having to march it out again. The concerned parameter direction is *pdir*.

#### Parameters

<i>other_pt</i>	refers to the <a href="#">IntersectionPoint</a> that we assume share influence area with <code>this</code> <a href="#">IntersectionPoint</a> .
<i>pdir</i>	the concerned parameter directon

29.236.3.60 `double Go::IntersectionPoint::startParam ( int pardir )`

Get the start value of the parameter interval for the specified parameter.

## Parameters

<i>pardir</i>	the parameter for which we seek the start value.
---------------	--------------------------------------------------

## Returns

the start value of the parameter interval for the parameter '*pardir*'.

29.236.3.61 `bool Go::IntersectionPoint::tangentsOriented ( ) const`

Returns true if this [IntersectionPoint](#) is sure about the orientation of its tangent. It is sure about the orientation if the intersection is transversal, or if the intersection is singular and the orientation has been set manually through the [fixTangentOrientation\(\)](#) command. See also [hasUniqueTangentDirection\(\)](#).

## Returns

'true' if the *tangent orientation* is clear (a prerequisite is of course that the *tangent direction* is clear). 'false' otherwise.

29.236.3.62 `void Go::IntersectionPoint::tangentPointingInwards ( bool & first, bool & second, bool & first_along_boundary, bool & second_along_boundary ) const`

The first and second argument are set to 'true' or 'false' according to whether the tangent direction in this point is clearly oriented toward the inside of the domain for respectively the first and second surface. It can only be pointing outwards if the point is lying on the boundary. If the point is on the boundary and its tangent is close to parallel with the boundary, '*first\_along\_boundary*' and/or '*second\_along\_boundary*' will be set to true, otherwise they are false. Of course, points whose tangents are directed along the boundary will return 'first' and/or 'second' false.

## Return values

<i>first</i>	'true' if the tangent direction of the intersection at this point is clearly oriented toward the inside of the domain of the first object. 'false' if not.
<i>second</i>	'true' if the tangent direction of the intersection at this point is clearly oriented toward the inside of the domain of the second object. 'false' if not.
<i>first_along_boundary</i>	'true' if the point is on the boundary of the first object, and with a tangent that is close to parallel with the boundary. 'false' otherwise.
<i>second_along_boundary</i>	'true' if the point is on the boundary of the second object, and with at tangent that is close to parallel with the boundary. 'false' otherwise.

29.236.3.63 `void Go::IntersectionPoint::write ( std::ostream & os ) const`

Write [IntersectionPoint](#) to stream (NB: topological and parent information will be lost)

## Parameters

<i>os</i>	output stream
-----------	---------------

29.236.3.64 void `Go::IntersectionPoint::writeInfo ( )` const

Write available info about an [IntersectionPoint](#) to standard output. Makes use of the struct [IntPtInfo](#).

29.236.3.65 void `Go::IntersectionPoint::writeParams ( std::ostream & os )` const

Write the parameters of the intersection point to stream

## Parameters

<i>os</i>	output stream
-----------	---------------

## 29.236.4 Member Data Documentation

29.236.4.1 const double `Go::IntersectionPoint::tangent_tol` [*static*]

Global tolerance for tangent length (considered singular if length is less than this value).

Definition at line 744 of file `IntersectionPoint.h`.

The documentation for this class was generated from the following file:

- [intersections/include/GoTools/intersections/IntersectionPoint.h](#)

## 29.237 Go::IntersectionPool Class Reference

```
#include <IntersectionPool.h>
```

### Public Member Functions

- [IntersectionPool](#) ([shared\\_ptr](#)< [ParamObjectInt](#) > obj1, [shared\\_ptr](#)< [ParamObjectInt](#) > obj2, [shared\\_ptr](#)< [IntersectionPool](#) > parent=[shared\\_ptr](#)< [IntersectionPool](#) >(), int missing\_dir=-1, double missing\_value=0)
- virtual [~IntersectionPool](#) ()
  - Destructor.*
- void [addParamPoints](#) (int nmb\_int\_pts, double \*pointpar1, double \*pointpar2, [shared\\_ptr](#)< [GeoTol](#) > epsge)
- [template](#)<class ip\_iterator > void [addCurves](#) (int nmb\_int\_cvs, ip\_iterator \*start\_iterators, ip\_iterator \*end\_iterators)
- [template](#)<class ip\_iterator > void [addCurve](#) (ip\_iterator start, ip\_iterator end)
- void [getBoundaryIntersections](#) ([std::vector](#)< [shared\\_ptr](#)< [IntersectionPoint](#) > > &bd\_ints)

- `bool checkIfBothPointsLieOnOneEndpointAndNotOfTheSameCurve ()`
- `void getSortedIntersections (std::vector< shared_ptr< IntersectionPoint > > &int_pts)`
- `std::vector< double > getSortedInnerInts (int paddir)`
- `void getSortedBdInts (const Point &vec, std::vector< shared_ptr< IntersectionPoint > > &result, int obj_↵  
nmb=-1)`
- `bool checkSortingDir (Point &vec, int sorting_obj)`
- `void getSortedInts (std::vector< shared_ptr< IntersectionPoint > > &result)`
- `void getAllIntersections (std::vector< shared_ptr< IntersectionPoint > > &result)`
- `void getBranchPoints (std::vector< shared_ptr< IntersectionPoint > > &pts)`
- `void getMidParameter (double *mid)`
- `shared_ptr< IntersectionPoint > addIntersectionPoint (shared_ptr< ParamObjectInt > obj_int1_, shared_↵  
ptr< ParamObjectInt > obj_int2_, shared_ptr< GeoTol > epsge, double *par1, double *par2)`
- `void includeReducedInts (shared_ptr< IntersectionPool > lower_order_pool)`
- `void selfIntersectParamReorganise (shared_ptr< IntersectionPool > sub_pool)`
- `void mirrorIntersectionPoints (size_t first)`
- `bool hasPointsInInner (int paddir)`
- `bool hasIntersectionPoints () const`
- `int numIntersectionPoints () const`
- `int numSingularIntersectionPoints ()`  
*Get the number of IntersectionPoints in this pool who are not classified as ORDINARY\_POINTS (see SingularityType for more info), i.e. points that lie on non-transversal intersections.*
- `bool isInDomain (IntersectionPoint *pnt) const`
- `bool closestInDomain (double param[], shared_ptr< IntersectionPoint > &pnt) const`
- `bool closestInDomain (const double param[], shared_ptr< IntersectionPoint > &pnt) const`
- `bool existIntersectionPoint (int dir, double par)`
- `int inInfluenceArea (int paddir, double par, bool first_outside=true)`
- `int inInfluenceArea (int paddir, double par, std::vector< shared_ptr< IntersectionPoint > > &int_pts, bool  
first_outside=true)`
- `void getIntersectionPoints (std::vector< shared_ptr< IntersectionPoint > > &int_pts) const`
- `void getIntersectionCurves (std::vector< shared_ptr< IntersectionCurve > > &int_curves) const`
- `std::vector< shared_ptr< IntersectionPoint > > & getIntersectionPoints ()`
- `const std::vector< shared_ptr< IntersectionPoint > > & getIntersectionPoints () const`
- `const std::vector< shared_ptr< IntersectionCurve > > & getIntersectionCurves () const`
- `void getIntersectionPoints (int dir, double par, std::vector< shared_ptr< IntersectionPoint > > &result) const`
- `bool hasIntersectionPoints (int dir, double par) const`
- `bool checkIntersectionPoints (std::vector< IntPtInfo > &int_pt_info) const`
- `bool checkIntersectionLinks () const`
- `void removeDoublePoints ()`
- `void removeLooseEnds ()`
- `void removeFalseCorners ()`
- `void removeDefectLinks ()`
- `void getAllLinks (std::set< shared_ptr< IntersectionLink > > &links)`
- `void getResult (std::vector< shared_ptr< IntersectionPoint > > &int_points, std::vector< shared_ptr<  
IntersectionCurve > > &int_curves)`
- `int lackingParameter () const`
- `double lackingParameterValue () const`
- `bool atSameBoundary (shared_ptr< IntersectionPoint > pt1, shared_ptr< IntersectionPoint > pt2)`
- `bool atDifferentBoundary (shared_ptr< IntersectionPoint > pt1, shared_ptr< IntersectionPoint > pt2)`
- `bool isBoundaryIntersection (shared_ptr< IntersectionPoint > pt1, shared_ptr< IntersectionPoint > pt2)`
- `bool isBoundaryPoint (shared_ptr< IntersectionPoint > pt1)`  
*Check if a point lies at a boundary in the current domain(s)*
- `bool isBoundaryPoint (IntersectionPoint *pt1)`
- `bool fetchLoops (std::vector< std::vector< shared_ptr< IntersectionPoint > > > &loop_ints)`
- `bool isPAC (std::vector< shared_ptr< IntersectionPoint > > &int_loop)`
- `void setCoincidence (std::vector< shared_ptr< IntersectionPoint > > &int_loop)`

- [bool isConnectedInside](#) (shared\_ptr< [IntersectionPoint](#) > pt1, shared\_ptr< [IntersectionPoint](#) > pt2)
- void [makeIntersectionCurves](#) ()
- void [getOrigPoints](#) (std::vector< shared\_ptr< [IntersectionPoint](#) > > &int\_pts) const
- void [writeDebug](#) (int singular=0)
- void [splitIntersectionLinks](#) (int fixed\_dir, double fixed\_val)
- int [numParams](#) () const
- double [startParam](#) (int dir) const
- double [endParam](#) (int dir) const
- void [includeCoveredNeighbourPoints](#) ()
- void [insertInInfluenceInterval](#) (shared\_ptr< [IntersectionPoint](#) > pt\_in\_pool, double \*parvals, int pardir)
- void [synchronizePool](#) ()
- void [cleanUpPool](#) (int first\_idx=0, double epsge=1.0e-15)
- void [removeIntPoint](#) (shared\_ptr< [IntersectionPoint](#) > int\_point)
- void [removeIntPoint2](#) (shared\_ptr< [IntersectionPoint](#) > int\_point)
- void [removeIntPoints](#) (double \*frompar, double \*topar, bool only\_inner=false)
- void [intersectAlongCommonBoundary](#) (double frac, std::vector< [BoundaryIntersectionData](#) > &isects)
 

*Check if two surfaces intersect along a common boundary (i.e. a linked intersection list exist along one boundary in both objects).*
- void [removeBoundaryIntersections](#) (bool single\_pt=true)
- bool [verifyIntersectionLink](#) (const shared\_ptr< [IntersectionLink](#) > &link, int recursion\_limit=10) const
- bool [checkIntersectionChain](#) ([IntersectionPoint](#) \*pnt, [IntersectionPoint](#) \*first, [IntersectionPoint](#) \*prev=0)
- bool [validate](#) () const
- void [weedOutClutterPoints](#) ()
- void [writeIntersectionPoints](#) () const
- void [writeIntersectionLinks](#) () const

## Friends

- class [SfSfIntersector](#)

### 29.237.1 Detailed Description

Class providing access to the intersection points and surfaces related to a particular subproblem.

Definition at line 64 of file [IntersectionPool.h](#).

### 29.237.2 Constructor & Destructor Documentation

**29.237.2.1** `Go::IntersectionPool::IntersectionPool ( shared_ptr< ParamObjectInt > obj1, shared_ptr< ParamObjectInt > obj2, shared_ptr< IntersectionPool > parent = shared_ptr< IntersectionPool > (), int missing_dir = -1, double missing_value = 0 )`

Constructor

Parameters

<i>obj1</i>	the first object of the (possible) intersection
<i>obj2</i>	the second object of the (possible) intersection
<i>parent</i>	pointer to parent pool (null pointer if no parent)
<i>missing_dir</i>	index of the missing parameter (negative if no missing parameter)
<i>missing_value</i>	value of the missing parameter



29.237.2.2 `virtual Go::IntersectionPool::~~IntersectionPool ( ) [inline],[virtual]`

Destructor.

Definition at line 83 of file IntersectionPool.h.

### 29.237.3 Member Function Documentation

29.237.3.1 `template<class ip_iterator > void Go::IntersectionPool::addCurve ( ip_iterator start, ip_iterator end ) [inline]`

Add one [IntersectionCurve](#) to the pool, by specifying a range of [IntersectionPoints](#) that make up the curve.

Parameters

<i>start</i>	iterator to the start of the range of <a href="#">IntersectionPoints</a> .
<i>end</i>	iterator to the end of the range of <a href="#">IntersectionPoints</a> .

Definition at line 940 of file IntersectionPool.h.

29.237.3.2 `template<class ip_iterator > void Go::IntersectionPool::addCurves ( int nmb_int_cvs, ip_iterator * start_iterators, ip_iterator * end_iterators ) [inline]`

Add a number of [IntersectionCurves](#) to the pool by specifying iterators to the [IntersectionPoint](#) making up the curves.

Parameters

<i>nmb_int_cvs</i>	number of curves to add
<i>start_iterators</i>	pointer to a range of iterators to <a href="#">IntersectionPoint</a> . The range should have length equal to the number of curves to add. Each iterator represents the start of a range of <a href="#">IntersectionPoints</a> that defines an <a href="#">IntersectionCurve</a> to add to the pool. (The corresponding end of the range is specified by the corresponding iterator in <i>end_iterators</i> ).
<i>end_iterators</i>	pointer to a range of iterators to <a href="#">IntersectionPoint</a> . The range should have length equal to the number of curves to add. Each iterator represents the one-past-end of a range of <a href="#">IntersectionPoint</a> s that defines an <a href="#">IntersectionCurve</a> to add to the pool. (The corresponding start of the range is specified by the corresponding iterator in <i>start_iterators</i> )

Definition at line 927 of file IntersectionPool.h.

29.237.3.3 `shared_ptr<IntersectionPoint> Go::IntersectionPool::addIntersectionPoint ( shared_ptr< ParamObjectInt > obj_int1, shared_ptr< ParamObjectInt > obj_int2, shared_ptr< GeoTol > epsge, double * par1, double * par2 )`

Construct a new [IntersectionPoint](#) and add it to this pool, as well as the pool's parents (and older ancestors). NB: The objects given as input should be equal to - or sub-objects of the objects that are already pointed to by this pool.

## Parameters

<i>obj_↔</i> <i>int1_</i>	The first of the intersecting objects of the new <a href="#">IntersectionPoint</a>
<i>obj_↔</i> <i>int2_</i>	The second of the intersecting objects of the new <a href="#">IntersectionPoint</a>
<i>epsge</i>	the tolerances used for the new IntesectionPoint
<i>par1</i>	pointer to the <a href="#">IntersectionPoint</a> 's parameters in the first object
<i>par2</i>	pointer to the <a href="#">IntersectionPoint</a> 's parameters in the second object

29.237.3.4 `void Go::IntersectionPool::addParamPoints ( int nmb_int_pts, double * pointpar1, double * pointpar2, shared_ptr< GeoTol > epsge )`

Add a number of [IntersectionPoint](#) s to the pool by specifying their parameter values.

## Parameters

<i>nmb_int_pts</i>	the number of <a href="#">IntersectionPoint</a> s that shall be added to the pool.
<i>pointpar1</i>	pointer to an array containing the consecutive parameter values in the first object for the <a href="#">IntersectionPoint</a> s to add.
<i>pointpar2</i>	pointer to an array containing the consecutive parameter values in the second object for the <a href="#">IntersectionPoint</a> s to add.
<i>epsge</i>	shared pointer to the object specifying the tolerances to use for the added <a href="#">IntersectionPoint</a> s.

29.237.3.5 `bool Go::IntersectionPool::atDifferentBoundary ( shared_ptr< IntersectionPoint > pt1, shared_ptr< IntersectionPoint > pt2 )`

Check if two [IntersectionPoints](#) can be found on two different boundaries of the parametric domain of this [IntersectionPools](#) objects (*obj1\_* and *obj2\_*)

## Parameters

<i>pt1</i>	the first <a href="#">IntersectionPoint</a>
<i>pt2</i>	the second <a href="#">IntersectionPoint</a>

## Returns

'true' if *pt1* and *pt2* could be found to lie on two distinct borders (i.e. *pt1* lay on one border and *pt2* lay on another).

29.237.3.6 `bool Go::IntersectionPool::atSameBoundary ( shared_ptr< IntersectionPoint > pt1, shared_ptr< IntersectionPoint > pt2 )`

Check if two [IntersectionPoints](#) lie on the same boundary of the parametric domain of this [IntersectionPools](#) objects (*obj1\_* and *obj2\_*).

## Parameters

<i>pt1</i>	the first <a href="#">IntersectionPoint</a>
<i>pt2</i>	the second <a href="#">IntersectionPoint</a>

## Returns

'true' if *pt1* and *pt2* both were found to lie on the same (parametric) boundary. 'false' otherwise.

**29.237.3.7** `bool Go::IntersectionPool::checkIfBothPointsLieOnOneEndpointAndNotOfTheSameCurve ( )`

**29.237.3.8** `bool Go::IntersectionPool::checkIntersectionChain ( IntersectionPoint * pt, IntersectionPoint * first, IntersectionPoint * prev = 0 )`

Check consistence of chain of intersection links with respect to number of neighbours

**29.237.3.9** `bool Go::IntersectionPool::checkIntersectionLinks ( ) const`

Check if the intersection links are OK. If they are not, give further diagnostics.

**29.237.3.10** `bool Go::IntersectionPool::checkIntersectionPoints ( std::vector< IntPtInfo > & int_pt_info ) const`

Check if the intersection points are OK. If they are not, give further diagnostics.

## Parameters

<i>int_pt_info</i>	vector with information about the intersection points. This information has the type of struct <a href="#">IntPtInfo</a> , which indicates if the point is ok, the number of neighbours, the singularity type, and the location of the point is the parameter domains of the objects
--------------------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

## Returns

'true' if all intersection points are OK, 'false' if at least one point is not OK

**29.237.3.11** `bool Go::IntersectionPool::checkSortingDir ( Point & vec, int sorting_obj )`

Check the consistency between the direction of the intersection curve indicated by *vec* and the direction computed in the existing intersection points. Turn the direction of *vec* if necessary.

## Parameters

<i>vec</i>	the direction of the intersection curve. This may be turned if necessary.
<i>sorting_obj</i>	the object - 0 or 1 - used for sorting

## Return values

'true'	if ???, 'false' if ???
--------	------------------------

29.237.3.12 `void Go::IntersectionPool::cleanUpPool ( int first_idx = 0, double epsge = 1.0e-15 )`

Remove redundant [IntersectionPoints](#) from the pool, starting from the point indexed *first\_idx* and upwards. An [IntersectionPoint](#) is considered redundant if:

- it has exactly two neighbours *AND*
- it lies on an isoparametric intersection curve *AND*
- both of its (two) neighbours are in the pool

## Parameters

<i>first_idx</i>	index of the first point
------------------	--------------------------

29.237.3.13 `bool Go::IntersectionPool::closestInDomain ( double param[], shared_ptr< IntersectionPoint > & pnt ) const`

Locate the [IntersectionPoint](#) in this [IntersectionPool](#) whose parameter values are closest (in the Euclidean norm on the parameter space) to a given set of parameters.

## Parameters

<i>param[]</i>	pointer to an array containing the parameter values that we want to find the closest <a href="#">IntersectionPoint</a> to.
----------------	----------------------------------------------------------------------------------------------------------------------------

## Return values

<i>pnt</i>	upon function return, if a closest <a href="#">IntersectionPoint</a> was found, <i>pnt</i> will be set to that <a href="#">IntersectionPoint</a> .
------------	----------------------------------------------------------------------------------------------------------------------------------------------------

## Returns

'true' if a closest [IntersectionPoint](#) was found, 'false' otherwise.

29.237.3.14 `bool Go::IntersectionPool::closestInDomain ( const double param[], shared_ptr< IntersectionPoint > & pnt ) const`

Locate the [IntersectionPoint](#) in this [IntersectionPool](#) whose parameter values are closest (in the Euclidean norm on the parameter space) to a given set of parameters.

## Parameters

<i>param[]</i>	pointer to an array containing the parameter values that we want to find the closest <a href="#">IntersectionPoint</a> to.
----------------	----------------------------------------------------------------------------------------------------------------------------

## Return values

<i>pnt</i>	upon function return, if a closest <a href="#">IntersectionPoint</a> was found, <i>pnt</i> will be set to that <a href="#">IntersectionPoint</a> .
------------	----------------------------------------------------------------------------------------------------------------------------------------------------

## Returns

'true' if a closest [IntersectionPoint](#) was found, 'false' otherwise.

29.237.3.15 `double Go::IntersectionPool::endParam ( int dir ) const`

Get the end of the parameter interval for the specified parameter direction

## Parameters

<i>dir</i>	the specified parameter direction
------------	-----------------------------------

## Returns

the end of the parameter interval for the specified parameter direction.

29.237.3.16 `bool Go::IntersectionPool::existIntersectionPoint ( int dir, double par )`

Checks the existence of any intersection points with the given parameter in the given direction.

## Parameters

<i>dir</i>	specifies the concerned parameter direction.
<i>par</i>	specifies the value of this parameter.

## Returns

'true' if the pool contains an [IntersectionPoint](#) whose parameter *dir* is approximately equal to *par*.

29.237.3.17 `bool Go::IntersectionPool::fetchLoops ( std::vector< std::vector< shared_ptr< IntersectionPoint > > > & loop_ints )`

Get all the intersection loops contained in this [IntersectionPool](#). This function assumes that the topologies of the loops in this pool are not too 'warped'. Notably, it might fail if two loops share some edges, since in that case there will be ambiguous choices for what constitutes the loops. (This is a fact from graph theory, and if we want to work around it, we might have to consider additional information of a non-topological nature, like looking at the geometric/parametric position of the points).

## Return values

<i>loop_ints</i>	upon function return, this vector will contain all the loops that were found in this <a href="#">IntersectionPool</a> (each loop is represented as a vector of <a href="#">IntersectionPoints</a> ).
------------------	------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

**Returns**

'true' if at least one loop was found. 'false' otherwise.

29.237.3.18 `void Go::IntersectionPool::getAllIntersections ( std::vector< shared_ptr< IntersectionPoint > > & result )`  
`[inline]`

Definition at line 194 of file IntersectionPool.h.

29.237.3.19 `void Go::IntersectionPool::getAllLinks ( std::set< shared_ptr< IntersectionLink > > & links )`

29.237.3.20 `void Go::IntersectionPool::getBoundaryIntersections ( std::vector< shared_ptr< IntersectionPoint > > & bd_ints )`

Get a vector containing shared pointers to all IntersectionPoints in the pool that is located on the boundary of one of the pool's objects.

**Return values**

<i>bd_ints</i>	vector of shared pointers to boundary intersection points
----------------	-----------------------------------------------------------

29.237.3.21 `void Go::IntersectionPool::getBranchPoints ( std::vector< shared_ptr< IntersectionPoint > > & pts )`

Get a vector containing all those IntersectionPoints in the pool that are branchpoints.

**Return values**

<i>pts</i>	vector of shared pointers to IntersectionPointss
------------	--------------------------------------------------

29.237.3.22 `void Go::IntersectionPool::getIntersectionCurves ( std::vector< shared_ptr< IntersectionCurve > > & int_curves ) const` `[inline]`

Fill the argument vector with shared pointers to all the IntersectionCurves in 'this' [IntersectionPool](#).

**Parameters**

<i>int_curves</i>	vector of shared pointers to all of the pool's IntersectionCurves.
-------------------	--------------------------------------------------------------------

Definition at line 891 of file IntersectionPool.h.

29.237.3.23 `const std::vector< shared_ptr< IntersectionCurve > > & Go::IntersectionPool::getIntersectionCurves ( )`  
`const` `[inline]`

Get a reference to the vector containing (shared pointers to) 'this' IntersectionPools IntersectionCurves.

## Return values

<i>reference</i>	to vector of shared pointers to IntersectionCurves
------------------	----------------------------------------------------

Definition at line 918 of file IntersectionPool.h.

```
29.237.3.24 void Go::IntersectionPool::getIntersectionPoints (std::vector< shared_ptr< IntersectionPoint > > & int_pts)
 const [inline]
```

Fill the vector given as argument with shared pointers to all the IntersectionPoints in 'this' [IntersectionPool](#).

## Return values

<i>int_pts</i>	vector of shared pointers to all of the pool's IntersectionPoints.
----------------	--------------------------------------------------------------------

Definition at line 881 of file IntersectionPool.h.

```
29.237.3.25 std::vector< shared_ptr< IntersectionPoint > > & Go::IntersectionPool::getIntersectionPoints ()
 [inline]
```

Get a reference to the vector containing (shared pointers to) 'this' [IntersectionPool](#)'s IntersectionPoints.

## Return values

<i>reference</i>	to vector of shared pointers to IntersectionPoints
------------------	----------------------------------------------------

Definition at line 900 of file IntersectionPool.h.

```
29.237.3.26 const std::vector< shared_ptr< IntersectionPoint > > & Go::IntersectionPool::getIntersectionPoints ()
 const [inline]
```

Get a reference to the vector containing (shared pointers to) 'this' IntersectionPools [IntersectionPoint](#) s.

## Return values

<i>reference</i>	to vector of shared pointers to IntersectionPoints
------------------	----------------------------------------------------

Definition at line 909 of file IntersectionPool.h.

```
29.237.3.27 void Go::IntersectionPool::getIntersectionPoints (int dir, double par, std::vector< shared_ptr<
 IntersectionPoint > > & result) const
```

Fetch the IntersectionPoints with a specified parameter value in a specified direction

## Parameters

<i>dir</i>	the specified parameter direction
<i>par</i>	the specified parameter value

## Return values

<i>result</i>	the vector containing those of the pool's <code>IntersectionPoints</code> that (approximately) had the specified parameter value in the specified parameter direction.
---------------	------------------------------------------------------------------------------------------------------------------------------------------------------------------------

29.237.3.28 `void Go::IntersectionPool::getMidParameter ( double * mid )`

Compute the middle parameter value of all intersection points of this intersection pool

## Return values

<i>mid</i>	points to an array of <code>doubles</code> where the result will be written. The number of elements in the array is equal to the total number of parameter directions for this pool.
------------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

29.237.3.29 `void Go::IntersectionPool::getOrigPoints ( std::vector< shared_ptr< IntersectionPoint > > & int_pts ) const`

Get the "original" intersection points of this intersection pool. If this pool has no parents with the same number of parameters as itself, then return the `IntersectionPoints` of this pool. Otherwise, return the `IntersectionPoints` of the parent pool.

## Return values

<i>int_pts</i>	the returned <code>IntersectionPoints</code> .
----------------	------------------------------------------------

29.237.3.30 `void Go::IntersectionPool::getResult ( std::vector< shared_ptr< IntersectionPoint > > & int_points, std::vector< shared_ptr< IntersectionCurve > > & int_curves )`

Get all isolated `IntersectionPoints` in the pool (those with no neighbours), as well as all `IntersectionCurves`.

## Return values

<i>int_points</i>	upon function return, this vector will contain shared pointers to all the <code>IntersectionPoints</code> in this pool that had no neighbours.
<i>int_curves</i>	upon function return, this vector will contain shared pointers to all the <code>IntersectionCurves</code> contained in this pool.



29.237.3.31 `void Go::IntersectionPool::getSortedBdInts ( const Point & vec, std::vector< shared_ptr< IntersectionPoint > > & result, int obj_nmb = -1 )`

Get a vector containing shared pointers to the pool's IntersectionPoints, sorted along a given spatial or parametrical direction. NOTE: The name of this function contains 'Bd' for 'boundary' for historical reasons. In fact, we get *all* intersections.

#### Parameters

<i>vec</i>	the direction along which to sort the <a href="#">IntersectionPoint</a> . If the number of vector elements is 3, it is interpreted as a 'spatial' vector; otherwise it is interpreted as a vector in the combined parametric domain of the two objects.
------------	---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

#### Return values

<i>result</i>	vector of shared pointers to IntersectionPoints
---------------	-------------------------------------------------

29.237.3.32 `std::vector<double> Go::IntersectionPool::getSortedInnerInts ( int padir )`

Get a vector containing the parametric values along a specific parameter direction for all the pool's IntersectionPoints that do not lie 'on' or 'close to' the boundary along this parameter. The returned values will be sorted according to increasing value.

#### Parameters

<i>padir</i>	the concerned parameter direction
--------------	-----------------------------------

#### Returns

the vector containing the requested parameter values.

29.237.3.33 `void Go::IntersectionPool::getSortedIntersections ( std::vector< shared_ptr< IntersectionPoint > > & int_pts )`

Get a vector containing shared pointers to all the IntersectionPoints in the pool. The entries in the vector will be sorted according to parameter values.

#### Return values

<i>int_pts</i>	vector containing the pool's IntersectionPoints in a sorted manner.
----------------	---------------------------------------------------------------------

29.237.3.34 `void Go::IntersectionPool::getSortedInts ( std::vector< shared_ptr< IntersectionPoint > > & result )`  
`[inline]`

Definition at line 189 of file IntersectionPool.h.

29.237.3.35 **bool** Go::IntersectionPool::hasIntersectionPoints ( ) const [inline]

Check if this [IntersectionPool](#) contains any [IntersectionPoints](#) at all.

#### Returns

'true' if 'this' pool contains at least one [IntersectionPoint](#).

Definition at line 290 of file IntersectionPool.h.

29.237.3.36 **bool** Go::IntersectionPool::hasIntersectionPoints ( int *dir*, double *par* ) const

Check if the [IntersectionPool](#) contains at least one [IntersectionPoint](#) with the specified parameter value in the specified direction.

#### Parameters

<i>dir</i>	the specified parameter direction
<i>par</i>	the specified parameter value

#### Returns

'true' if at least one [IntersectionPoint](#) with the specified parameter value in the specified parameter direction was found in the pool. 'false' otherwise.

29.237.3.37 **bool** Go::IntersectionPool::hasPointsInInner ( int *pardir* )

Check if this [IntersectionPool](#) contains [IntersectionPoints](#) that are not "boundary points" with respect to the parameter *pardir*.

#### Parameters

<i>pardir</i>	the parameter direction we want to check for
---------------	----------------------------------------------

#### Returns

'true' if there was found at least one [IntersectionPoint](#) whose *pardir* parameter did not lie on the extremal values of its interval. 'false' otherwise.

29.237.3.38 **void** Go::IntersectionPool::includeCoveredNeighbourPoints ( )

Check all the neighbours of the [IntersectionPoints](#) in this pool, and add those of them that are not already in the pool, but whose parameters are covered by the pool's parameter domain.

29.237.3.39 void Go::IntersectionPool::includeReducedInts ( shared\_ptr< IntersectionPool > lower\_order\_pool )

Include the IntersectionPoints in the lower\_order\_pool into 'this' pool, generating new representations of the points which includes the missing parameter. It is assumed that the IntersectionPoints in lower\_order\_pool have no [IntersectionLink](#) with points outside the pool.

## Parameters

<i>lower_order_pool</i>	a pointer to the lower-order pool from which we want to include its IntersectionPoints.
-------------------------	-----------------------------------------------------------------------------------------

29.237.3.40 `int Go::IntersectionPool::inInfluenceArea ( int pardir, double par, bool first_outside = true )`

Check if a given parameter value in a given parameter direction lies in the influence area of any IntersectionPoints in the pool.

## Parameters

<i>pardir</i>	the concerned parameter direction
<i>par</i>	the parameter value to check against
<i>first_outside</i>	if this is set to 'true' then the outer brackets of the influence intervals will be used (which in practice means that if the function returns 'false', then <i>par</i> is <i>guaranteed</i> to be outside the influence area of all points). If this argument is set to 'false', then the inner brackets of the influence interval will be used (which in practice means that if the function returns 'true', then <i>par</i> is <i>guaranteed</i> to be inside the influence area of at least one point).

## Returns

- 0 if no [IntersectionPoint](#) with an influence area covering the specified value was found.
- 1 if we found an [IntersectionPoint](#) whose influence area covered the specified value, but where the specified value did not hit *exactly* on the [IntersectionPoint](#)'s own parameter.
- 2 if we found an [IntersectionPoint](#) whose parameter value in the specified direction coincide with the specified value.

29.237.3.41 `int Go::IntersectionPool::inInfluenceArea ( int pardir, double par, std::vector< shared_ptr< IntersectionPoint > > & int_pts, bool first_outside = true )`

Check if a given parameter value in a given parameter direction lies in the influence area of any IntersectionPoints in the specified vector.

## Parameters

<i>pardir</i>	the concerned parameter direction
<i>par</i>	the parameter value to check against
<i>int_pts</i>	the vector of IntersectionPoints to check against.
<i>first_outside</i>	if this is set to 'true' then the outer brackets of the influence intervals will be used (which in practice means that if the function returns 'false', then <i>par</i> is <i>guaranteed</i> to be outside the influence area of all points). If this argument is set to 'false', then the inner brackets of the influence interval will be used (which in practice means that if the function returns 'true', then <i>par</i> is <i>guaranteed</i> to be inside the influence area of at least one point).

## Returns

- 0 if no [IntersectionPoint](#) with an influence area covering the specified value was found.

- 1 if we found an [IntersectionPoint](#) whose influence area covered the specified value, but where the specified value did not hit *exactly* on the [IntersectionPoint](#)'s own parameter.
- 2 if we found an [IntersectionPoint](#) whose parameter value in the specified direction coincide with the specified value.

29.237.3.42 `void Go::IntersectionPool::insertInInfluenceInterval ( shared_ptr< IntersectionPoint > pt_in_pool, double * parvals, int pardir )`

Create a new [IntersectionPoint](#) and add it to the the [IntersectionPool](#). The new [IntersectionPoint](#) (A) will lie inside the influence interval of another of the pool's points; the one (B) indicated by *pt\_in\_pool* (supposedly in the pool already). The parameter direction in which (A) lies inside the influence interval of (B) is indicated by *pardir*. The parameter values of the new point is pointed to by *parvals* (an array which also contains the parameter value in *pardir*. There will be inserted an [IntersectionLink](#) between (A) and (B), which will be isoparametric in the other parameter direction if *pardir* is on a two-parametric object (surface or function).

#### Parameters

<i>pt_in_pool</i>	the <a href="#">IntersectionPoint</a> (supposedly already in the pool) whose influence area the new <a href="#">IntersectionPoint</a> is going to lie.
<i>parvals</i>	pointer to an array specifying the parameter values for the new <a href="#">IntersectionPoint</a> to create.
<i>pardir</i>	the parameter direction that will be iso-parametric in the newly established <a href="#">IntersectionLink</a> (see above).

29.237.3.43 `void Go::IntersectionPool::intersectAlongCommonBoundary ( double frac, std::vector< BoundaryIntersectionData > & isects )`

Check if two surfaces intersect along a common boundary (i.e. a linked intersection list exist along one boundary in both objects).

Intersections whose parametric extent is smaller than a specified fraction (*frac*) of the surface's parameter intervals are ignored. The data on the found boundary intersections are returned as entries into a vector of [BoundaryIntersectionData](#).

#### Parameters

<i>frac</i>	the fraction of a parameter's total span that must be covered in order to consider this intersection "long enough" to be considered. (A value between 0 and 1, typically small).
-------------	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

#### Return values

<i>isects</i>	vector with objects containing information on the found boundary intersections.
---------------	---------------------------------------------------------------------------------

29.237.3.44 `bool Go::IntersectionPool::isBoundaryIntersection ( shared_ptr< IntersectionPoint > pt1, shared_ptr< IntersectionPoint > pt2 )`

Check if the two specified [IntersectionPoints](#) represent a boundary intersection in this pool. In order to be interpreted as such, they must both lie on the same (parametric) boundary, and be connected with an iso-parametric [IntersectionLink](#)

## Parameters

<i>pt1</i>	the first <a href="#">IntersectionPoint</a>
<i>pt2</i>	the second <a href="#">IntersectionPoint</a>

## Returns

'true' if *pt1* and *pt2* satisfy the criteria specified above, 'false' otherwise.

29.237.3.45 **bool** `Go::IntersectionPool::isBoundaryPoint ( shared_ptr< IntersectionPoint > pt1 )` `[inline]`

Check if a point lies at a boundary in the current domain(s)

Definition at line 546 of file `IntersectionPool.h`.

29.237.3.46 **bool** `Go::IntersectionPool::isBoundaryPoint ( IntersectionPoint * pt1 )`

29.237.3.47 **bool** `Go::IntersectionPool::isConnectedInside ( shared_ptr< IntersectionPoint > pt1, shared_ptr< IntersectionPoint > pt2 )`

Checks if there is a path of connected [IntersectionPoints](#) going between the two [IntersectionPoints](#) specified. The specified points are assumed to be in this [IntersectionPool](#), and all [IntersectionPoints](#) of the path must also be in the pool.

## Parameters

<i>pt1</i>	the first <a href="#">IntersectionPoint</a> . Must already be in the pool. Otherwise, an error will occur.
<i>pt2</i>	the second <a href="#">IntersectionPoint</a> . Must already be in the pool. Otherwise, an error will occur.

## Returns

'true' if a path was found connecting *pt1* with *pt2* using only [IntersectionPoints](#) in the pool. 'false' otherwise.

29.237.3.48 **bool** `Go::IntersectionPool::isInDomain ( IntersectionPoint * pnt ) const`

Check if a given [IntersectionPoint](#) is in this [IntersectionPool](#).

## Parameters

<i>pnt</i>	pointer to the <a href="#">IntersectionPoint</a> that we want to check.
------------	-------------------------------------------------------------------------

## Returns

'true' if this [IntersectionPoint](#) was found in this [IntersectionPool](#), 'false' otherwise.

**29.237.3.49** `bool Go::IntersectionPool::isPAC ( std::vector< shared_ptr< IntersectionPoint > > & int_loop )`

Check if a given loop delimits a partial coincidence area (PAC). To determine this the meta-information in the [IntersectionLink](#) s is examined.

#### Parameters

<code>int_loop</code>	the loop to check
-----------------------	-------------------

#### Returns

'true' if the loop delimits a PAC, 'false' otherwise.

**29.237.3.50** `int Go::IntersectionPool::lackingParameter ( ) const [inline]`

Get the index of the missing parameter (-1 if none missing)

#### Returns

the index of the missing parameter

Definition at line 949 of file IntersectionPool.h.

**29.237.3.51** `double Go::IntersectionPool::lackingParameterValue ( ) const [inline]`

Get value of missing parameter (if it exists)

#### Returns

value of the missing parameter

Definition at line 957 of file IntersectionPool.h.

**29.237.3.52** `void Go::IntersectionPool::makeIntersectionCurves ( )`

Prepare output by making IntersectionCurves. Points that have one or three or more IntersectionLinks are considered endpoints to curves, points with two IntersectionLinks lie in the inner of a curve.

**29.237.3.53** `void Go::IntersectionPool::mirrorIntersectionPoints ( size_t first )`

**29.237.3.54** `int Go::IntersectionPool::numIntersectionPoints ( ) const [inline]`

Query the number of IntersectionPoints contained in this [IntersectionPool](#).

#### Returns

the number of IntersectionPoints in this pool.

Definition at line 296 of file IntersectionPool.h.

29.237.3.55 `int Go::IntersectionPool::numParams ( ) const [inline]`

Get the combined number of parameter directions in the two objects.

#### Returns

the number of parameters in the first object plus the number of parameters in the second object.

Definition at line 633 of file IntersectionPool.h.

29.237.3.56 `int Go::IntersectionPool::numSingularIntersectionPoints ( )`

Get the number of IntersectionPoints in this pool who are not classified as ORDINARY\_POINTs (see [SingularityType](#) for more info), i.e. points that lie on non-transversal intersections.

#### Returns

number of singular IntersectionPoints.

29.237.3.57 `void Go::IntersectionPool::removeBoundaryIntersections ( bool single_pt = true )`

Removes IntersectionLinks positioned along the boundary of both objects, The concerned IntersectionPoints are also removed, except when they also links to points which are NOT on the boundary.

29.237.3.58 `void Go::IntersectionPool::removeDefectLinks ( )`

Remove defect intersection links. Whether or not a link is defect depends on the return value of [verifyIntersection↔Link\(\)](#).

29.237.3.59 `void Go::IntersectionPool::removeDoublePoints ( )`

Remove double points by merging into one single point. Double points are points that are the same within the tolerance. The new points that results from this will inherit all the links from the old points.

29.237.3.60 `void Go::IntersectionPool::removeFalseCorners ( )`

29.237.3.61 `void Go::IntersectionPool::removeIntPoint ( shared_ptr< IntersectionPoint > int_point )`

Remove an [IntersectionPoint](#) from the pool. Should only be called if the number of neighbours are less than or equal to two.

#### Parameters

<i>int_point</i>	the <a href="#">IntersectionPoint</a> to remove from the pool and its ancestors (as long as the ancestors have the same number of parameter directions). IntersectionLinks will be broken up and the removed <a href="#">IntersectionPoint</a> 's neighbours will be reconnected.
<i>int_point</i>	shared pointer to the <a href="#">IntersectionPoint</a> to remove.



29.237.3.62 void Go::IntersectionPool::removeIntPoint2 ( shared\_ptr< IntersectionPoint > int\_point )

Remove an [IntersectionPoint](#) from the pool.

#### Parameters

<i>int_point</i>	the <a href="#">IntersectionPoint</a> to remove from the pool and its ancestors (as long as the ancestors have the same number of parameter directions). IntersectionLinks will be broken up.
<i>int_point</i>	shared pointer to the <a href="#">IntersectionPoint</a> to remove.

29.237.3.63 void Go::IntersectionPool::removeIntPoints ( double \* frompar, double \* topar, bool only\_inner = false )

Remove all intersection points from the pool whose parameters lie in the box defined by *frompar* and *topar*. IntersectionLinks will be broken up.

#### Parameters

<i>frompar</i>	the array of parameters defining the lower corner of the box in question.
<i>topar</i>	the array of parameters defining the upper corner of the box in question.

29.237.3.64 void Go::IntersectionPool::removeLooseEnds ( )

29.237.3.65 void Go::IntersectionPool::selfIntersectParamReorganise ( shared\_ptr< IntersectionPool > sub\_pool )

Reorganize self-intersection parameters. A 'twin point' is a concept when working with self-intersection objects. It represent an [IntersectionPoint](#) that already exist in the parent pool, but whose ordering of parameters had to be switched in order to conform with its objects (which are really two parts of the same, global surface). If a pool contains twin points, we must assure that the curves where these points are included have a consistent ordering of the parameters. This function takes care of making this ordering consistent. Moreover, since the twin points are really just another representation of an already-existing point, they will go out of scope and disappear when the sub-pool in which they lie is destroyed. Therefore, we must make sure that points linked to by the twin points will now be linked to by the original points that the twin-points are representing.

**NB:** This is a very special function that is only used with the following preconditions:

- this pool should represent a self-intersection, ie. its first and second object should be the same.
- the 'sub\_pool' (given as argument) should be a child of 'this' pool.
- 'this' pool and the one pointed to by *sub\_pool* should have the same number of parameters.

#### Parameters

<i>sub_pool</i>	shared pointer to the sub-pool, as described above.
-----------------	-----------------------------------------------------

29.237.3.66 `void Go::IntersectionPool::setCoincidence ( std::vector< shared_ptr< IntersectionPoint > > & int_loop )`

Set the specified loop to be a partial coincidence area (PAC). This is written into the loop's `IntersectionLinks` as meta-information.

#### Parameters

<code>int_loop</code>	the loop in question
-----------------------	----------------------

29.237.3.67 `void Go::IntersectionPool::splitIntersectionLinks ( int fixed_dir, double fixed_val )`

Among all `IntersectionLinks` between `IntersectionPoints` in this pool, split those who cross a specified parameter direction at a specified parameter value. Split means to insert a new `IntersectionPoint` at this place.

#### Parameters

<code>fixed_dir</code>	the specified parameter direction
<code>fixed_val</code>	the specified parameter value

29.237.3.68 `double Go::IntersectionPool::startParam ( int dir ) const`

Get the start of the parameter interval for the specified parameter direction

#### Parameters

<code>dir</code>	the specified parameter direction
------------------	-----------------------------------

#### Returns

the start of the parameter interval for the `dir` parameter direction.

29.237.3.69 `void Go::IntersectionPool::synchronizePool ( )`

Synchronize pool with its parent. We check that the pool is "valid" by making sure that the intersection points on the current level is also present at previous levels. If there are intersection points that are redundant in this sense, we remove them. This function is useful when several sibling subintersector are around, and running compute on one of them has removed intersection points that are also present in some of the others.

29.237.3.70 `bool Go::IntersectionPool::validate ( ) const`

Checks if the pool is "valid", that is, if the intersection points on the present subdivision level exist also on previous levels.

#### Returns

`true` if the pool is valid, `false` otherwise

29.237.3.71 `bool Go::IntersectionPool::verifyIntersectionLink ( const shared_ptr< IntersectionLink > & link, int recursion_limit = 10 ) const`

Verify that the given intersection link is in the pool and represents a connected piece of the intersection curve.

#### Parameters

<i>link</i>	shared pointer to the intersection link we wish to verify
-------------	-----------------------------------------------------------

#### Return values

'true'	if the link represents a connected piece of the intersection curve, 'false' otherwise
--------	---------------------------------------------------------------------------------------

29.237.3.72 `void Go::IntersectionPool::weedOutClutterPoints ( )`

29.237.3.73 `void Go::IntersectionPool::writeDebug ( int singular = 0 )`

Write various debug information

#### Parameters

<i>singular</i>	use 1 for singular case, 0 is default
-----------------	---------------------------------------

29.237.3.74 `void Go::IntersectionPool::writeIntersectionLinks ( ) const`

Writes out a list of links between intersection points in the pool. Writes to cout.

29.237.3.75 `void Go::IntersectionPool::writeIntersectionPoints ( ) const`

Writes out a list of the current intersection points in the pool. Writes to cout.

## 29.237.4 Friends And Related Function Documentation

29.237.4.1 `friend class SfSfIntersector` [*friend*]

Definition at line 792 of file IntersectionPool.h.

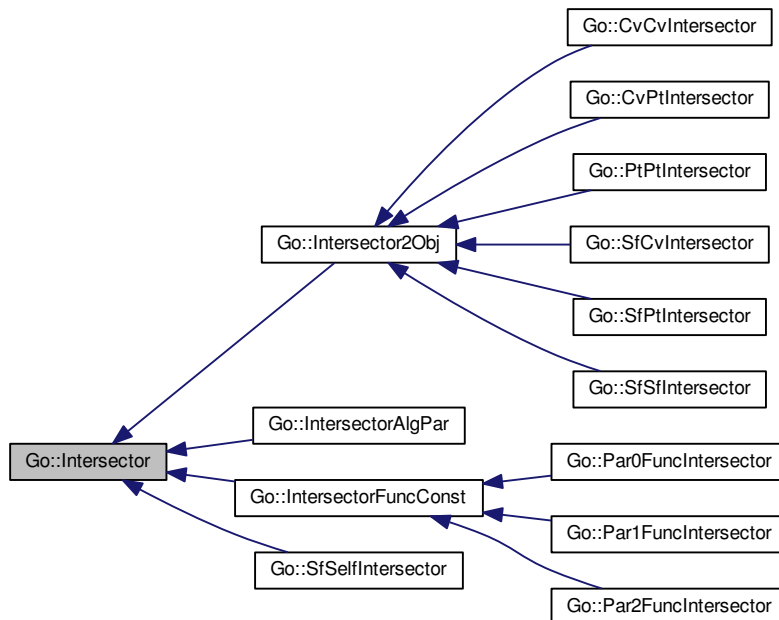
The documentation for this class was generated from the following file:

- intersections/include/GoTools/intersections/[IntersectionPool.h](#)

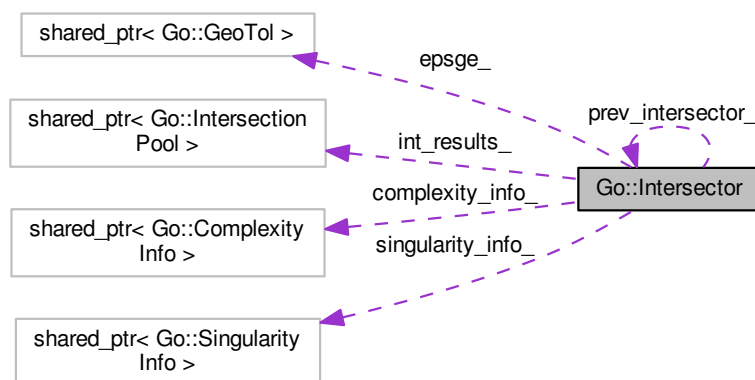
## 29.238 Go::Intersector Class Reference

```
#include <Intersector.h>
```

Inheritance diagram for Go::Intersector:



Collaboration diagram for Go::Intersector:



### Public Member Functions

- [Intersector](#) ()

*Default constructor.*

- [Intersector](#) (double epsge, [Intersector](#) \*prev=0)
- [Intersector](#) (shared\_ptr< [GeoTol](#) > epsge, [Intersector](#) \*prev=0)
- virtual ~[Intersector](#) ()

*Destructor.*

- virtual void [compute](#) (bool compute\_at\_boundary=true)
- virtual void [getResult](#) (std::vector< shared\_ptr< [IntersectionPoint](#) > > &int\_points, std::vector< shared\_ptr< [IntersectionCurve](#) > > &int\_curves)
- shared\_ptr< [IntersectionPool](#) > [getIntPool](#) ()
- bool [validateSiblingPools](#) ()
- shared\_ptr< [GeoTol](#) > [getTolerance](#) ()
- bool [hasSingularityInfo](#) ()
- shared\_ptr< [SingularityInfo](#) > [getSingularityInfo](#) ()
- void [setSingularityInfo](#) (shared\_ptr< [SingularityInfo](#) > previous, int missing\_dir)
- void [setHighPriSing](#) (double \*par)
- bool [hasComplexityInfo](#) ()
- shared\_ptr< [ComplexityInfo](#) > [getComplexityInfo](#) ()
- virtual int [numParams](#) () const =0
- virtual int [nmbBdObj](#) (int idx) const
- virtual [BoundaryGeomInt](#) \* [getBoundaryObject](#) (int idx, int bd\_idx) const
- int [nmbRecursions](#) ()
- virtual bool [isSelfIntersection](#) ()
- virtual int [isSelfintCase](#) ()
- virtual void [addComplexDomain](#) ([RectDomain](#) dom)
- void [writeIntersectionPoints](#) () const

*Write diagnostic information about the intersection points.*

## Protected Member Functions

- virtual void [print\\_objs](#) ()=0
- virtual int [getBoundaryIntersections](#) ()=0
- virtual int [performInterception](#) ()=0
- virtual int [simpleCase](#) ()=0
- virtual bool [isLinear](#) ()=0
- virtual bool [degTriangleSimple](#) ()
- virtual bool [complexityReduced](#) ()=0
- virtual void [handleComplexity](#) ()=0
- virtual int [checkCoincidence](#) ()=0
- virtual void [microCase](#) ()=0
- virtual int [updateIntersections](#) ()=0
- virtual int [repairIntersections](#) ()=0
- virtual int [linearCase](#) ()=0
- virtual int [doSubdivide](#) ()=0
- virtual int [complexIntercept](#) ()
- virtual int [complexSimpleCase](#) ()
- virtual void [doPostIterate](#) ()
- virtual void [printDebugInfo](#) ()=0

## Protected Attributes

- shared\_ptr< [IntersectionPool](#) > [int\\_results\\_](#)
- std::vector< shared\_ptr< [Intersector](#) > > [sub\\_intersectors\\_](#)
- [Intersector](#) \* [prev\\_intersector\\_](#)
- shared\_ptr< [GeoTol](#) > [epsge\\_](#)
- shared\_ptr< [SingularityInfo](#) > [singularity\\_info\\_](#)
- shared\_ptr< [ComplexityInfo](#) > [complexity\\_info\\_](#)

## Friends

- class [SfSfIntersector](#)
- class [IntersectionPool](#)

### 29.238.1 Detailed Description

This class is an abstract class providing an interface to the intersection functionality in [GoTools](#).

Definition at line 67 of file `Intersector.h`.

### 29.238.2 Constructor & Destructor Documentation

#### 29.238.2.1 `Go::Intersector::Intersector ( )` `[inline]`

Default constructor.

Definition at line 71 of file `Intersector.h`.

#### 29.238.2.2 `Go::Intersector::Intersector ( double epsg, Intersector * prev = 0 )`

Constructor.

##### Parameters

<i>epsg</i>	the geometric tolerance for the intersector.
<i>prev</i>	the previous intersector.

#### 29.238.2.3 `Go::Intersector::Intersector ( shared_ptr< GeoTol > epsg, Intersector * prev = 0 )`

Constructor.

##### Parameters

<i>epsg</i>	the geometric tolerance for the intersector.
<i>prev</i>	the previous intersector.

#### 29.238.2.4 `virtual Go::Intersector::~~Intersector ( )` `[inline]`, `[virtual]`

Destructor.

Definition at line 84 of file `Intersector.h`.

### 29.238.3 Member Function Documentation

29.238.3.1 `virtual void Go::Intersector::addComplexDomain ( RectDomain dom ) [inline],[virtual]`

Reimplemented in [Go::SfSelfIntersector](#).

Definition at line 202 of file `Intersector.h`.

29.238.3.2 `virtual int Go::Intersector::checkCoincidence ( ) [protected],[pure virtual]`

Implemented in [Go::Intersector2Obj](#), [Go::SfSelfIntersector](#), [Go::IntersectorAlgPar](#), [Go::SfCvIntersector](#), [Go::SfSfIntersector](#), [Go::IntersectorFuncConst](#), [Go::CvPtIntersector](#), [Go::CvCvIntersector](#), [Go::PtPtIntersector](#), [Go::Par1FuncIntersector](#), [Go::Par2FuncIntersector](#), [Go::Par0FuncIntersector](#), and [Go::SfPtIntersector](#).

29.238.3.3 `virtual int Go::Intersector::complexIntercept ( ) [inline],[protected],[virtual]`

Reimplemented in [Go::Intersector2Obj](#).

Definition at line 263 of file `Intersector.h`.

29.238.3.4 `virtual bool Go::Intersector::complexityReduced ( ) [protected],[pure virtual]`

Implemented in [Go::Intersector2Obj](#), [Go::SfSelfIntersector](#), [Go::IntersectorAlgPar](#), [Go::SfSfIntersector](#), and [Go::IntersectorFuncConst](#).

29.238.3.5 `virtual int Go::Intersector::complexSimpleCase ( ) [inline],[protected],[virtual]`

Reimplemented in [Go::Intersector2Obj](#).

Definition at line 268 of file `Intersector.h`.

29.238.3.6 `virtual void Go::Intersector::compute ( bool compute_at_boundary = true ) [virtual]`

Compute the current intersections (topology).

Parameters

<code><i>compute_at_boundary</i></code>	if true we will include computation of boundary intersections.
-----------------------------------------	----------------------------------------------------------------

Reimplemented in [Go::SfSelfIntersector](#), and [Go::IntersectorAlgPar](#).

29.238.3.7 `virtual bool Go::Intersector::degTriangleSimple ( ) [inline],[protected],[virtual]`

Reimplemented in [Go::SfSfIntersector](#).

Definition at line 241 of file `Intersector.h`.

29.238.3.8 `virtual void Go::Intersector::doPostIterate ( ) [inline],[protected],[virtual]`

Reimplemented in [Go::Intersector2Obj](#).

Definition at line 273 of file Intersector.h.

29.238.3.9 `virtual int Go::Intersector::doSubdivide ( ) [protected],[pure virtual]`

Implemented in [Go::Intersector2Obj](#), [Go::SfSelfIntersector](#), [Go::SfSfIntersector](#), [Go::IntersectorAlgPar](#), [Go::SfCvIntersector](#), [Go::IntersectorFuncConst](#), [Go::CvPtIntersector](#), [Go::CvCvIntersector](#), [Go::Par2FuncIntersector](#), [Go::PtPtIntersector](#), [Go::Par1FuncIntersector](#), [Go::Par0FuncIntersector](#), and [Go::SfPtIntersector](#).

29.238.3.10 `virtual int Go::Intersector::getBoundaryIntersections ( ) [protected],[pure virtual]`

Implemented in [Go::Intersector2Obj](#), [Go::SfSelfIntersector](#), [Go::IntersectorAlgPar](#), and [Go::IntersectorFuncConst](#).

29.238.3.11 `virtual BoundaryGeomInt* Go::Intersector::getBoundaryObject ( int idx, int bd_idx ) const [inline],[virtual]`

Get the specified boundary object belonging to the specified [ParamGeomInt](#).

#### Parameters

<i>idx</i>	refers to obj1 or obj2. Indexing starts at 0.
<i>bd_idx</i>	index of the boundary object in the <a href="#">ParamGeomInt</a> .

#### Returns

The boundary object.

Reimplemented in [Go::Intersector2Obj](#).

Definition at line 179 of file Intersector.h.

29.238.3.12 `shared_ptr<ComplexityInfo> Go::Intersector::getComplexityInfo ( ) [inline]`

Get info on the complexity of the problem.

#### Returns

Info on the complexity of the problem.

Definition at line 158 of file Intersector.h.



29.238.3.13 `shared_ptr<IntersectionPool> Go::Intersector::getIntPool ( ) [inline]`

Get the [IntersectionPool](#) for the intersector.

#### Returns

The [IntersectionPool](#).

Definition at line 104 of file `Intersector.h`.

29.238.3.14 `virtual void Go::Intersector::getResult ( std::vector< shared_ptr< IntersectionPoint > > &int_points, std::vector< shared_ptr< IntersectionCurve > > &int_curves ) [virtual]`

Get intersection points and curves. Sends request to [IntersectionPool](#). The intersection points are isolated.

#### Parameters

<i>int_points</i>	vector of intersection points
<i>int_curves</i>	vector of intersection curves

Reimplemented in [Go::IntersectorAlgPar](#).

29.238.3.15 `shared_ptr<SingularityInfo> Go::Intersector::getSingularityInfo ( ) [inline]`

Get info regarding singularities.

#### Returns

The singularify info for the intersector.

Definition at line 125 of file `Intersector.h`.

29.238.3.16 `shared_ptr<GeoTol> Go::Intersector::getTolerance ( ) [inline]`

Get the tolerance object used by this [Intersector](#)

#### Returns

The tolerance object for the intersector.

Definition at line 115 of file `Intersector.h`.

29.238.3.17 `virtual void Go::Intersector::handleComplexity ( ) [protected],[pure virtual]`

Implemented in [Go::Intersector2Obj](#), [Go::SfSelfIntersector](#), [Go::IntersectorAlgPar](#), [Go::SfSfIntersector](#), and [Go::↔IntersectorFuncConst](#).

**29.238.3.18** `bool Go::Intersector::hasComplexityInfo ( ) [inline]`

Verify whether there has been created info regarding the complexity of the problem.

#### Returns

True if complexity info has been created.

Definition at line 153 of file Intersector.h.

**29.238.3.19** `bool Go::Intersector::hasSingularityInfo ( ) [inline]`

Verify whether singularities has been set.

#### Returns

True if info on singularities has been set.

Definition at line 120 of file Intersector.h.

**29.238.3.20** `virtual bool Go::Intersector::isLinear ( ) [protected],[pure virtual]`

Implemented in [Go::Intersector2Obj](#), [Go::SfSelfIntersector](#), [Go::IntersectorAlgPar](#), and [Go::IntersectorFuncConst](#).

**29.238.3.21** `virtual int Go::Intersector::isSelfintCase ( ) [inline],[virtual]`

Check if this intersection algorithm is performed in a self-intersection context.

Reimplemented in [Go::Intersector2Obj](#).

Definition at line 199 of file Intersector.h.

**29.238.3.22** `virtual bool Go::Intersector::isSelfIntersection ( ) [inline],[virtual]`

Verify whether the surface is self-intersecting.

#### Returns

True if the surface is self-intersecting.

Reimplemented in [Go::SfSelfIntersector](#).

Definition at line 194 of file Intersector.h.

**29.238.3.23** `virtual int Go::Intersector::linearCase ( ) [protected],[pure virtual]`

Implemented in [Go::Intersector2Obj](#), [Go::SfSelfIntersector](#), [Go::SfSfIntersector](#), [Go::IntersectorAlgPar](#), [Go::SfCv↔Intersector](#), [Go::IntersectorFuncConst](#), [Go::CvPtIntersector](#), [Go::CvCvIntersector](#), [Go::PtPtIntersector](#), and [Go::↔SfPtIntersector](#).

**29.238.3.24** `virtual void Go::Intersector::microCase ( ) [protected],[pure virtual]`

Implemented in [Go::Intersector2Obj](#), [Go::SfSelfIntersector](#), [Go::IntersectorAlgPar](#), [Go::SfCvIntersector](#), [Go::SfSf↔Intersector](#), [Go::IntersectorFuncConst](#), [Go::CvPtIntersector](#), [Go::CvCvIntersector](#), [Go::PtPtIntersector](#), [Go::Par1↔FuncIntersector](#), [Go::Par2FuncIntersector](#), [Go::Par0FuncIntersector](#), and [Go::SfPtIntersector](#).

**29.238.3.25** `virtual int Go::Intersector::nmbBdObj ( int idx ) const [inline],[virtual]`

Count the number of boundary objects belonging to the specified [ParamGeomInt](#).

## Parameters

<i>idx</i>	refers to obj1 or obj2. Indexing starts at 0.
------------	-----------------------------------------------

## Returns

The number of boundary objects.

Reimplemented in [Go::Intersector2Obj](#).

Definition at line 170 of file Intersector.h.

**29.238.3.26** `int Go::Intersector::nmbRecurions ( ) [inline]`

The current recursion level.

## Returns

The current recursion level.

Definition at line 184 of file Intersector.h.

**29.238.3.27** `virtual int Go::Intersector::numParams ( ) const [pure virtual]`

Get the number of parameter directions for the intersection.

## Returns

The number of parameter directions for the intersection.

Implemented in [Go::SfSelfIntersector](#), [Go::CvPtIntersector](#), [Go::SfCvIntersector](#), [Go::IntersectorAlgPar](#), [Go::Cv↔CvIntersector](#), [Go::PtPtIntersector](#), [Go::SfSfIntersector](#), [Go::Par1FuncIntersector](#), [Go::Par2FuncIntersector](#), [Go::↔Par0FuncIntersector](#), and [Go::SfPtIntersector](#).

**29.238.3.28** `virtual int Go::Intersector::performInterception ( ) [protected],[pure virtual]`

Implemented in [Go::SfSelfIntersector](#), [Go::Intersector2Obj](#), [Go::IntersectorAlgPar](#), and [Go::IntersectorFuncConst](#).

**29.238.3.29** `virtual void Go::Intersector::print_objs ( ) [protected],[pure virtual]`

Implemented in [Go::SfSelfIntersector](#), [Go::Intersector2Obj](#), [Go::IntersectorAlgPar](#), and [Go::IntersectorFuncConst](#).

**29.238.3.30** `virtual void Go::Intersector::printDebugInfo ( ) [protected],[pure virtual]`

Implemented in [Go::Intersector2Obj](#), [Go::SfSelfIntersector](#), [Go::IntersectorAlgPar](#), and [Go::IntersectorFuncConst](#).

**29.238.3.31** `virtual int Go::Intersector::repairIntersections ( ) [protected],[pure virtual]`

Implemented in [Go::SfSelfIntersector](#), [Go::IntersectorAlgPar](#), [Go::SfCvIntersector](#), [Go::SfSfIntersector](#), [Go::CvPt↔Intersector](#), [Go::CvCvIntersector](#), [Go::PtPtIntersector](#), [Go::Par1FuncIntersector](#), [Go::Par2FuncIntersector](#), [Go::↔Par0FuncIntersector](#), and [Go::SfPtIntersector](#).

**29.238.3.32** `void Go::Intersector::setHighPriSing ( double * par )`

Instruct the intersector about known singular points.

## Parameters

<i>par</i>	the parameter value of the singularity. Size of array should be <a href="#">numParams()</a> .
------------	-----------------------------------------------------------------------------------------------

29.238.3.33 `void Go::Intersector::setSingularityInfo ( shared_ptr< SingularityInfo > previous, int missing_dir )`  
`[inline]`

Set info regarding singularities.

## Parameters

<i>previous</i>	the singularity info.
<i>missing_dir</i>	if the dimension of the intersection problem has been reduced from previous, the index tells us which parameter direction that was removed. Indexing starts at 0.

Definition at line 134 of file Intersector.h.

29.238.3.34 `virtual int Go::Intersector::simpleCase ( )` `[protected],[pure virtual]`

Implemented in [Go::Intersector2Obj](#), [Go::SfSelfIntersector](#), [Go::SfCvIntersector](#), [Go::IntersectorAlgPar](#), and [Go::IntersectorFuncConst](#).

29.238.3.35 `virtual int Go::Intersector::updateIntersections ( )` `[protected],[pure virtual]`

Implemented in [Go::Intersector2Obj](#), [Go::SfSelfIntersector](#), [Go::IntersectorAlgPar](#), [Go::SfCvIntersector](#), [Go::SfSfIntersector](#), [Go::IntersectorFuncConst](#), [Go::CvPtIntersector](#), [Go::CvCvIntersector](#), [Go::PtPtIntersector](#), [Go::Par1FuncIntersector](#), [Go::Par2FuncIntersector](#), [Go::Par0FuncIntersector](#), and [Go::SfPtIntersector](#).

29.238.3.36 `bool Go::Intersector::validateSiblingPools ( )`

Validate this pool and its siblings.

## Returns

*true* if all the sibling pools are valid, *false* otherwise

29.238.3.37 `void Go::Intersector::writeIntersectionPoints ( ) const`

Write diagnostic information about the intersection points.

## 29.238.4 Friends And Related Function Documentation

29.238.4.1 `friend class IntersectionPool` `[friend]`

Definition at line 209 of file Intersector.h.

29.238.4.2 friend class SfSfIntersector [friend]

Definition at line 208 of file Intersector.h.

## 29.238.5 Member Data Documentation

29.238.5.1 shared\_ptr<ComplexityInfo> Go::Intersector::complexity\_info\_ [protected]

Definition at line 222 of file Intersector.h.

29.238.5.2 shared\_ptr<GeoTol> Go::Intersector::epsge\_ [protected]

Definition at line 220 of file Intersector.h.

29.238.5.3 shared\_ptr<IntersectionPool> Go::Intersector::int\_results\_ [protected]

Definition at line 217 of file Intersector.h.

29.238.5.4 Intersector\* Go::Intersector::prev\_intersector\_ [protected]

Definition at line 219 of file Intersector.h.

29.238.5.5 shared\_ptr<SingularityInfo> Go::Intersector::singularity\_info\_ [protected]

Definition at line 221 of file Intersector.h.

29.238.5.6 std::vector<shared\_ptr<Intersector> > Go::Intersector::sub\_intersectors\_ [protected]

Definition at line 218 of file Intersector.h.

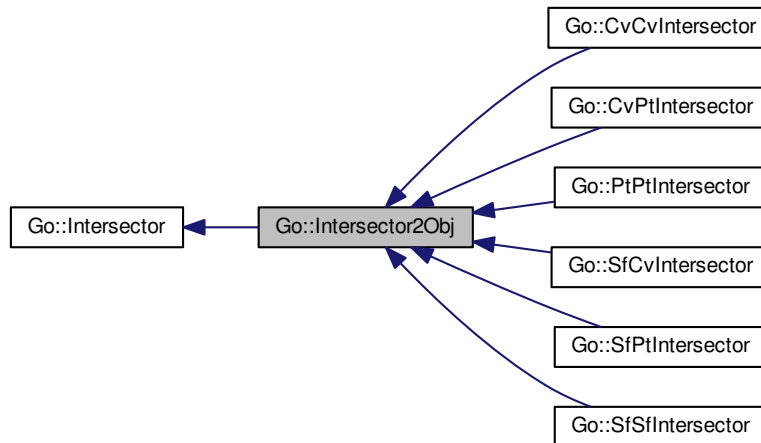
The documentation for this class was generated from the following file:

- intersections/include/GoTools/intersections/[Intersector.h](#)

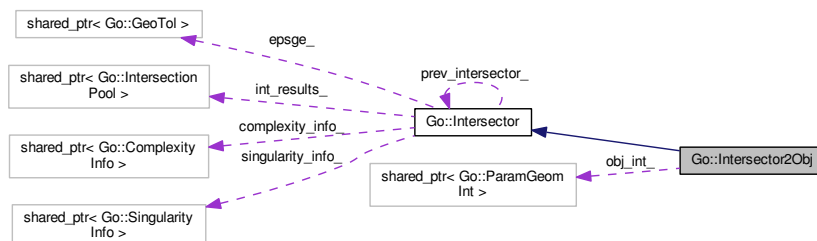
## 29.239 Go::Intersector2Obj Class Reference

```
#include <Intersector2Obj.h>
```

Inheritance diagram for Go::Intersector2Obj:



Collaboration diagram for Go::Intersector2Obj:



### Public Member Functions

- [Intersector2Obj](#) ()  
*Default constructor.*
- [Intersector2Obj](#) (shared\_ptr< [ParamGeomInt](#) > obj1, shared\_ptr< [ParamGeomInt](#) > obj2, shared\_ptr< [GeoTol](#) > epsge, [Intersector](#) \*prev=0, int eliminated\_parameter=-1, double eliminated\_value=0)
- [Intersector2Obj](#) (shared\_ptr< [ParamGeomInt](#) > obj1, shared\_ptr< [ParamGeomInt](#) > obj2, double epsge, [Intersector](#) \*prev=0, int eliminated\_parameter=-1, double eliminated\_value=0)
- virtual [~Intersector2Obj](#) ()  
*Destructor.*
- virtual int [nmbBdObj](#) (int idx) const
- virtual [BoundaryGeomInt](#) \* [getBoundaryObject](#) (int idx, int bd\_idx) const
- void [setSelfintCase](#) (int type)
- virtual int [isSelfintCase](#) ()

## Protected Member Functions

- virtual void `print_objs` ()
- virtual `shared_ptr< Intersector > lowerOrderIntersector` (`shared_ptr< ParamGeomInt > obj1`, `shared_ptr< ParamGeomInt > obj2`, `Intersector *prev=0`, `int eliminated_parameter=-1`, `double eliminated_value=0`)=0
- virtual `int getBoundaryIntersections` ()
- virtual `int performInterception` ()
- virtual `bool foundIntersectionNearBoundary` ()
- virtual `int performRotatedBoxTest` (`double eps1`, `double eps2`)
- virtual `int performInterceptionByImplicitization` ()
- virtual `int interceptionBySeparationSurface` ()
- virtual `int simpleCase` ()
- virtual `int simpleCase2` (`Point &axis1`, `Point &axis2`)
- virtual `int simpleCaseByImplicitization` ()
- virtual `bool isLinear` ()
- virtual `bool complexityReduced` ()
- virtual void `handleComplexity` ()
- virtual `int checkCoincidence` ()=0
- virtual void `microCase` ()=0
- virtual `int updateIntersections` ()=0
- virtual `int linearCase` ()=0
- virtual `int doSubdivide` ()=0
- virtual `int complexIntercept` ()
- virtual `int complexSimpleCase` ()
- virtual void `postIterate` (`int nmb_orig`, `int dir=-1`, `bool keep_endpt=true`)
- virtual void `doPostIterate` ()
- virtual void `removeDegenerateConnections` ()
- void `getSeedIteration` (`double seed[]`)
- `bool atSameBoundary` (`const double *par1`, `const double *par2`)
- `bool atBoundary` (`const double *par1`, `std::vector< bool > &boundaries`)
- virtual void `printDebugInfo` ()
- virtual void `writeOut` ()

## Protected Attributes

- `shared_ptr< ParamGeomInt > obj_int_ [2]`
- `int selfint_case_`

### 29.239.1 Detailed Description

This class is an abstract class providing an interface to the intersection functionality in [GoTools](#).

Definition at line 54 of file `Intersector2Obj.h`.

### 29.239.2 Constructor & Destructor Documentation

#### 29.239.2.1 Go::Intersector2Obj::Intersector2Obj( ) [inline]

Default constructor.

Definition at line 58 of file `Intersector2Obj.h`.

#### 29.239.2.2 Go::Intersector2Obj::Intersector2Obj( `shared_ptr< ParamGeomInt > obj1`, `shared_ptr< ParamGeomInt > obj2`, `shared_ptr< GeoTol > epsge`, `Intersector * prev = 0`, `int eliminated_parameter = -1`, `double eliminated_value = 0` )

Constructor. The last two variables are relevant only if the parent has one more parameter than the [Intersector](#) to be constructed.

## Parameters

<i>obj1</i>	object of type <a href="#">ParamSurfaceInt</a> , <a href="#">ParamCurveInt</a> or <a href="#">ParamPointInt</a> .
<i>obj2</i>	object of type <a href="#">ParamSurfaceInt</a> , <a href="#">ParamCurveInt</a> or <a href="#">ParamPointInt</a> .
<i>epsge</i>	the associated tolerance.
<i>prev</i>	the "parent" <a href="#">Intersector</a> (0 if there is no parent).
<i>eliminated_parameter</i>	the index of the parameter that was removed from the parent <i>prev</i> .
<i>eliminated_value</i>	the value of the parameter that was removed from the parent <i>prev</i> .

29.239.2.3 `Go::Intersector2Obj::Intersector2Obj ( shared_ptr< ParamGeomInt > obj1, shared_ptr< ParamGeomInt > obj2, double epsge, Intersector * prev = 0, int eliminated_parameter = -1, double eliminated_value = 0 )`

Constructor. The last two variables are relevant only if the parent has one more parameter than the [Intersector](#) to be constructed.

## Parameters

<i>obj1</i>	object of type <a href="#">ParamSurfaceInt</a> , <a href="#">ParamCurveInt</a> or <a href="#">ParamPointInt</a> .
<i>obj2</i>	object of type <a href="#">ParamSurfaceInt</a> , <a href="#">ParamCurveInt</a> or <a href="#">ParamPointInt</a> .
<i>epsge</i>	the associated tolerance.
<i>prev</i>	the "parent" <a href="#">Intersector</a> (0 if there is no parent).
<i>eliminated_parameter</i>	the index of the parameter that was removed from the parent <i>prev</i> .
<i>eliminated_value</i>	the value of the parameter that was removed from the parent <i>prev</i> .

29.239.2.4 `virtual Go::Intersector2Obj::~~Intersector2Obj ( ) [virtual]`

Destructor.

## 29.239.3 Member Function Documentation

29.239.3.1 `bool Go::Intersector2Obj::atBoundary ( const double * par1, std::vector< bool > & boundaries )`  
[protected]

29.239.3.2 `bool Go::Intersector2Obj::atSameBoundary ( const double * par1, const double * par2 )` [protected]

29.239.3.3 `virtual int Go::Intersector2Obj::checkCoincidence ( )` [protected],[pure virtual]

Implements [Go::Intersector](#).

Implemented in [Go::SfCvIntersector](#), [Go::SfSfIntersector](#), [Go::CvPtIntersector](#), [Go::CvCvIntersector](#), [Go::PtPtIntersector](#), and [Go::SfPtIntersector](#).

29.239.3.4 `virtual int Go::Intersector2Obj::complexIntercept ( )` [protected],[virtual]

Reimplemented from [Go::Intersector](#).



29.239.3.5 `virtual bool Go::Intersector2Obj::complexityReduced ( ) [inline],[protected],[virtual]`

Implements [Go::Intersector](#).

Reimplemented in [Go::SfSfIntersector](#).

Definition at line 188 of file `Intersector2Obj.h`.

29.239.3.6 `virtual int Go::Intersector2Obj::complexSimpleCase ( ) [protected],[virtual]`

Reimplemented from [Go::Intersector](#).

29.239.3.7 `virtual void Go::Intersector2Obj::doPostIterate ( ) [inline],[protected],[virtual]`

Reimplemented from [Go::Intersector](#).

Definition at line 216 of file `Intersector2Obj.h`.

29.239.3.8 `virtual int Go::Intersector2Obj::doSubdivide ( ) [protected],[pure virtual]`

Implements [Go::Intersector](#).

Implemented in [Go::SfSfIntersector](#), [Go::SfCvIntersector](#), [Go::CvPtIntersector](#), [Go::CvCvIntersector](#), [Go::PtPtIntersector](#), and [Go::SfPtIntersector](#).

29.239.3.9 `virtual bool Go::Intersector2Obj::foundIntersectionNearBoundary ( ) [inline],[protected],[virtual]`

Reimplemented in [Go::SfCvIntersector](#), [Go::CvCvIntersector](#), and [Go::SfSfIntersector](#).

Definition at line 159 of file `Intersector2Obj.h`.

29.239.3.10 `virtual int Go::Intersector2Obj::getBoundaryIntersections ( ) [protected],[virtual]`

Implements [Go::Intersector](#).

29.239.3.11 `virtual BoundaryGeomInt* Go::Intersector2Obj::getBoundaryObject ( int idx, int bd_idx ) const [inline],[virtual]`

Get the specified boundary object belonging to the specified [ParamGeomInt](#).

#### Parameters

<i>idx</i>	refers to <i>obj1</i> or <i>obj2</i> (i.e. 0 or 1).
<i>bd_idx</i>	index of the boundary object in the <a href="#">ParamGeomInt</a> .

**Returns**

The boundary object.

Reimplemented from [Go::Intersector](#).

Definition at line 121 of file Intersector2Obj.h.

29.239.3.12 `void Go::Intersector2Obj::getSeedIteration ( double seed[] )` [protected]

29.239.3.13 `virtual void Go::Intersector2Obj::handleComplexity ( )` [inline],[protected],[virtual]

Implements [Go::Intersector](#).

Reimplemented in [Go::SfSfIntersector](#).

Definition at line 194 of file Intersector2Obj.h.

29.239.3.14 `virtual int Go::Intersector2Obj::interceptionBySeparationSurface ( )` [inline],[protected],[virtual]

Reimplemented in [Go::SfSfIntersector](#).

Definition at line 174 of file Intersector2Obj.h.

29.239.3.15 `virtual bool Go::Intersector2Obj::isLinear ( )` [protected],[virtual]

Implements [Go::Intersector](#).

29.239.3.16 `virtual int Go::Intersector2Obj::isSelfintCase ( )` [inline],[virtual]

Check if this intersection is performed in self-intersection context.

**Returns**

`true` if this is a self-intersection, `false` otherwise

Reimplemented from [Go::Intersector](#).

Definition at line 134 of file Intersector2Obj.h.

29.239.3.17 `virtual int Go::Intersector2Obj::linearCase ( )` [protected],[pure virtual]

Implements [Go::Intersector](#).

Implemented in [Go::SfSfIntersector](#), [Go::SfCvIntersector](#), [Go::CvPtIntersector](#), [Go::CvCvIntersector](#), [Go::PtPtIntersector](#), and [Go::SfPtIntersector](#).

29.239.3.18 `virtual shared_ptr<Intersector> Go::Intersector2Obj::lowerOrderIntersector ( shared_ptr< ParamGeomInt > obj1, shared_ptr< ParamGeomInt > obj2, Intersector * prev = 0, int eliminated_parameter = -1, double eliminated_value = 0 )` [protected],[pure virtual]

Implemented in [Go::SfCvIntersector](#), [Go::CvPtIntersector](#), [Go::CvCvIntersector](#), [Go::PtPtIntersector](#), [Go::SfSfIntersector](#), and [Go::SfPtIntersector](#).

29.239.3.19 `virtual void Go::Intersector2Obj::microCase ( )` [protected],[pure virtual]

Implements [Go::Intersector](#).

Implemented in [Go::SfCvIntersector](#), [Go::SfSfIntersector](#), [Go::CvPtIntersector](#), [Go::CvCvIntersector](#), [Go::PtPtIntersector](#), and [Go::SfPtIntersector](#).

29.239.3.20 `virtual int Go::Intersector2Obj::nmbBdObj ( int idx ) const` [inline],[virtual]

Count the number of boundary objects belonging to the specified [ParamGeomInt](#).

#### Parameters

<i>idx</i>	refers to <i>obj1</i> or <i>obj2</i> (i.e. 0 or 1).
------------	-----------------------------------------------------

#### Returns

The number of boundary objects.

Reimplemented from [Go::Intersector](#).

Definition at line 112 of file [Intersector2Obj.h](#).

29.239.3.21 `virtual int Go::Intersector2Obj::performInterception ( )` [protected],[virtual]

Implements [Go::Intersector](#).

29.239.3.22 `virtual int Go::Intersector2Obj::performInterceptionByImplicitization ( )` [inline],[protected],[virtual]

Reimplemented in [Go::SfSfIntersector](#).

Definition at line 167 of file [Intersector2Obj.h](#).

29.239.3.23 `virtual int Go::Intersector2Obj::performRotatedBoxTest ( double eps1, double eps2 )` [protected],[virtual]

Reimplemented in [Go::SfCvIntersector](#), [Go::CvCvIntersector](#), [Go::SfSfIntersector](#), and [Go::SfPtIntersector](#).

29.239.3.24 `virtual void Go::Intersector2Obj::postIterate ( int nmb_orig, int dir = -1, bool keep_endpt = true )`  
[inline], [protected], [virtual]

Reimplemented in [Go::SfCvIntersector](#).

Definition at line 214 of file `Intersector2Obj.h`.

29.239.3.25 `virtual void Go::Intersector2Obj::print_objs ( )` [protected], [virtual]

Implements [Go::Intersector](#).

29.239.3.26 `virtual void Go::Intersector2Obj::printDebugInfo ( )` [protected], [virtual]

Implements [Go::Intersector](#).

29.239.3.27 `virtual void Go::Intersector2Obj::removeDegenerateConnections ( )` [inline], [protected],  
[virtual]

Definition at line 221 of file `Intersector2Obj.h`.

29.239.3.28 `void Go::Intersector2Obj::setSelfintCase ( int type )` [inline]

Mark that this intersection is performed in self-intersection context.

Definition at line 127 of file `Intersector2Obj.h`.

29.239.3.29 `virtual int Go::Intersector2Obj::simpleCase ( )` [protected], [virtual]

Implements [Go::Intersector](#).

Reimplemented in [Go::SfCvIntersector](#).

29.239.3.30 `virtual int Go::Intersector2Obj::simpleCase2 ( Point & axis1, Point & axis2 )` [protected], [virtual]

Reimplemented in [Go::SfCvIntersector](#), [Go::SfSfIntersector](#), and [Go::CvCvIntersector](#).

29.239.3.31 `virtual int Go::Intersector2Obj::simpleCaseByImplicitization ( )` [inline], [protected],  
[virtual]

Reimplemented in [Go::SfSfIntersector](#).

Definition at line 183 of file `Intersector2Obj.h`.

29.239.3.32 `virtual int Go::Intersector2Obj::updateIntersections ( )` [protected],[pure virtual]

Implements [Go::Intersector](#).

Implemented in [Go::SfCvIntersector](#), [Go::SfSfIntersector](#), [Go::CvPtIntersector](#), [Go::CvCvIntersector](#), [Go::PtPtIntersector](#), and [Go::SfPtIntersector](#).

29.239.3.33 `virtual void Go::Intersector2Obj::writeOut ( )` [inline],[protected],[virtual]

Reimplemented in [Go::CvCvIntersector](#).

Definition at line 231 of file `Intersector2Obj.h`.

#### 29.239.4 Member Data Documentation

29.239.4.1 `shared_ptr<ParamGeomInt> Go::Intersector2Obj::obj_int_2` [protected]

Definition at line 139 of file `Intersector2Obj.h`.

29.239.4.2 `int Go::Intersector2Obj::selfint_case_` [protected]

Definition at line 140 of file `Intersector2Obj.h`.

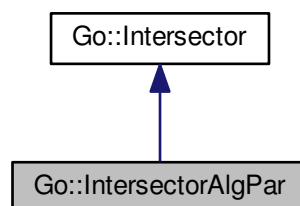
The documentation for this class was generated from the following file:

- `intersections/include/GoTools/intersections/Intersector2Obj.h`

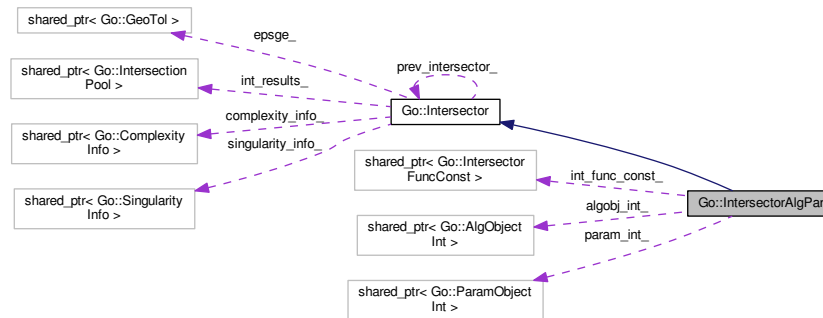
## 29.240 Go::IntersectorAlgPar Class Reference

```
#include <IntersectorAlgPar.h>
```

Inheritance diagram for `Go::IntersectorAlgPar`:



Collaboration diagram for Go::IntersectorAlgPar:



## Public Member Functions

- [IntersectorAlgPar](#) (shared\_ptr< [AlgObjectInt](#) > alg\_obj, shared\_ptr< [ParamObjectInt](#) > param\_obj, shared\_ptr< [GeoTol](#) > epsge, [Intersector](#) \*prev=0, int eliminated\_parameter=-1, double eliminated\_↔ value=0)
- virtual [~IntersectorAlgPar](#) ()  
*Destructor.*
- virtual void [compute](#) (bool compute\_at\_boundary=true)
- virtual void [getResult](#) (std::vector< shared\_ptr< [IntersectionPoint](#) > > &int\_points, std::vector< shared\_ptr< [IntersectionCurve](#) > > &int\_curves)
- virtual int [numParams](#) () const

## Protected Member Functions

- virtual void [print\\_objs](#) ()
- virtual int [getBoundaryIntersections](#) ()
- virtual int [performInterception](#) ()
- virtual int [simpleCase](#) ()
- virtual bool [isLinear](#) ()
- virtual bool [complexityReduced](#) ()
- virtual void [handleComplexity](#) ()
- virtual int [checkCoincidence](#) ()
- virtual void [microCase](#) ()
- virtual int [updateIntersections](#) ()
- virtual int [repairIntersections](#) ()
- virtual int [linearCase](#) ()
- virtual int [doSubdivide](#) ()
- virtual void [printDebugInfo](#) ()

## Protected Attributes

- shared\_ptr< [ParamObjectInt](#) > [param\\_int\\_](#)
- shared\_ptr< [AlgObjectInt](#) > [alobj\\_int\\_](#)
- shared\_ptr< [IntersectorFuncConst](#) > [int\\_func\\_const\\_](#)

### 29.240.1 Detailed Description

This class is an interface class used to compute the intersection between an algebraic object and a parametric object. The algebraic object is "plugged" into the parametric object, resulting in a scalar function. We then use the [IntersectorFuncConst](#) class to find the intersection.

Definition at line 65 of file `IntersectorAlgPar.h`.

### 29.240.2 Constructor & Destructor Documentation

**29.240.2.1** `Go::IntersectorAlgPar::IntersectorAlgPar ( shared_ptr< AlgObjectInt > alg_obj, shared_ptr< ParamObjectInt > param_obj, shared_ptr< GeoTol > epsge, Intersector * prev = 0, int eliminated_parameter = -1, double eliminated_value = 0 )`

Constructor. The last two variables are relevant only if the parent has one more parameter than the [Intersector](#) to be constructed.

#### Parameters

<i>alg_obj</i>	the algebraic object.
<i>param_obj</i>	the parametric object.
<i>epsge</i>	the associated tolerance.
<i>prev</i>	the "parent" <a href="#">Intersector</a> (0 if there is no parent).
<i>eliminated_parameter</i>	the index of the parameter that was removed from the parent <i>prev</i> .
<i>eliminated_value</i>	the value of the parameter that was removed from the parent <i>prev</i> .

**29.240.2.2** `virtual Go::IntersectorAlgPar::~~IntersectorAlgPar ( ) [inline],[virtual]`

Destructor.

Definition at line 88 of file `IntersectorAlgPar.h`.

### 29.240.3 Member Function Documentation

**29.240.3.1** `virtual int Go::IntersectorAlgPar::checkCoincidence ( ) [protected],[virtual]`

Implements [Go::Intersector](#).

**29.240.3.2** `virtual bool Go::IntersectorAlgPar::complexityReduced ( ) [protected],[virtual]`

Implements [Go::Intersector](#).

**29.240.3.3** `virtual void Go::IntersectorAlgPar::compute ( bool compute_at_boundary = true ) [virtual]`

Compute the current intersections (topology)

## Parameters

<i>compute_at_boundary</i>	flag to indicate that we compute at the boundary
----------------------------	--------------------------------------------------

Reimplemented from [Go::Intersector](#).

29.240.3.4 `virtual int Go::IntersectorAlgPar::doSubdivide ( ) [protected],[virtual]`

Implements [Go::Intersector](#).

29.240.3.5 `virtual int Go::IntersectorAlgPar::getBoundaryIntersections ( ) [protected],[virtual]`

Implements [Go::Intersector](#).

29.240.3.6 `virtual void Go::IntersectorAlgPar::getResult ( std::vector< shared_ptr< IntersectionPoint > > & int_points, std::vector< shared_ptr< IntersectionCurve > > & int_curves ) [virtual]`

Return intersection points and curves.

## Parameters

<i>int_points</i>	vector of shared pointers to IntersectionPoints
<i>int_curves</i>	vector of shared pointers to IntersectionCurves

Reimplemented from [Go::Intersector](#).

29.240.3.7 `virtual void Go::IntersectorAlgPar::handleComplexity ( ) [protected],[virtual]`

Implements [Go::Intersector](#).

29.240.3.8 `virtual bool Go::IntersectorAlgPar::isLinear ( ) [protected],[virtual]`

Implements [Go::Intersector](#).

29.240.3.9 `virtual int Go::IntersectorAlgPar::linearCase ( ) [protected],[virtual]`

Implements [Go::Intersector](#).

29.240.3.10 `virtual void Go::IntersectorAlgPar::microCase ( ) [protected],[virtual]`

Implements [Go::Intersector](#).



29.240.3.11 `virtual int Go::IntersectorAlgPar::numParams ( ) const [virtual]`

Return the number of parameter directions for the object.

Returns

thenumber of parameter directions

Implements [Go::Intersector](#).

29.240.3.12 `virtual int Go::IntersectorAlgPar::performInterception ( ) [protected],[virtual]`

Implements [Go::Intersector](#).

29.240.3.13 `virtual void Go::IntersectorAlgPar::print_objs ( ) [protected],[virtual]`

Implements [Go::Intersector](#).

29.240.3.14 `virtual void Go::IntersectorAlgPar::printDebugInfo ( ) [protected],[virtual]`

Implements [Go::Intersector](#).

29.240.3.15 `virtual int Go::IntersectorAlgPar::repairIntersections ( ) [inline],[protected],[virtual]`

Implements [Go::Intersector](#).

Definition at line 141 of file `IntersectorAlgPar.h`.

29.240.3.16 `virtual int Go::IntersectorAlgPar::simpleCase ( ) [protected],[virtual]`

Implements [Go::Intersector](#).

29.240.3.17 `virtual int Go::IntersectorAlgPar::updateIntersections ( ) [protected],[virtual]`

Implements [Go::Intersector](#).

## 29.240.4 Member Data Documentation

29.240.4.1 `shared_ptr<AlgObjectInt> Go::IntersectorAlgPar::algobj_int_ [protected]`

Definition at line 113 of file `IntersectorAlgPar.h`.

29.240.4.2 `shared_ptr<IntersectorFuncConst> Go::IntersectorAlgPar::int_func_const_` [protected]

Definition at line 119 of file IntersectorAlgPar.h.

29.240.4.3 `shared_ptr<ParamObjectInt> Go::IntersectorAlgPar::param_int_` [protected]

Definition at line 111 of file IntersectorAlgPar.h.

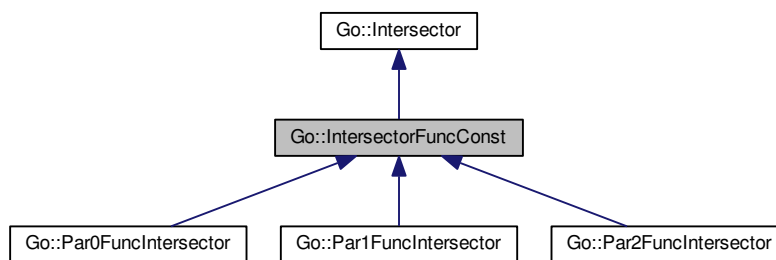
The documentation for this class was generated from the following file:

- [intersections/include/GoTools/intersections/IntersectorAlgPar.h](#)

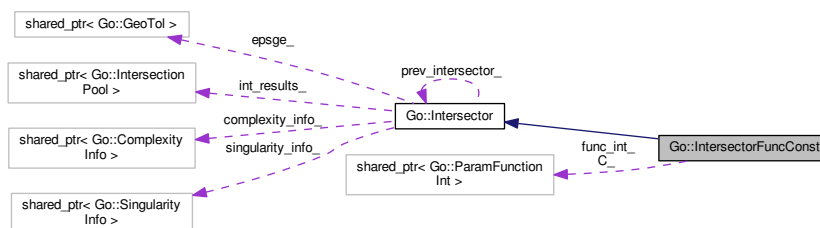
## 29.241 Go::IntersectorFuncConst Class Reference

```
#include <IntersectorFuncConst.h>
```

Inheritance diagram for Go::IntersectorFuncConst:



Collaboration diagram for Go::IntersectorFuncConst:



### Public Member Functions

- `IntersectorFuncConst` (`shared_ptr< ParamFunctionInt > func`, `shared_ptr< ParamFunctionInt > C`, `shared_ptr< GeoTol > epsge`, `Intersector *prev=0`, `int eliminated_parameter=-1`, `double eliminated_value=0`)
- `virtual ~IntersectorFuncConst ()`

*Destructor.*

## Protected Member Functions

- virtual void [print\\_objs](#) ()
- virtual shared\_ptr< [Intersector](#) > [lowerOrderIntersector](#) (shared\_ptr< [ParamFunctionInt](#) > obj1, shared\_ptr< [ParamFunctionInt](#) > obj2, [Intersector](#) \*prev=0, int eliminated\_parameter=-1, double eliminated\_value=0)=0
- virtual int [getBoundaryIntersections](#) ()
- virtual int [performInterception](#) ()
- virtual int [simpleCase](#) ()
- virtual bool [isLinear](#) ()
- virtual bool [complexityReduced](#) ()
- virtual void [handleComplexity](#) ()
- virtual int [checkCoincidence](#) ()=0
- virtual void [microCase](#) ()=0
- virtual int [updateIntersections](#) ()=0
- virtual int [linearCase](#) ()
- virtual int [doSubdivide](#) ()=0
- virtual void [printDebugInfo](#) ()

## Protected Attributes

- shared\_ptr< [ParamFunctionInt](#) > [func\\_int\\_](#)
- shared\_ptr< [ParamFunctionInt](#) > [C\\_](#)

## Friends

- class [IntersectorAlgPar](#)

### 29.241.1 Detailed Description

This class is an interface class for finding the intersection between a parametric function (with range R) and a constant.

Definition at line 56 of file [IntersectorFuncConst.h](#).

### 29.241.2 Constructor & Destructor Documentation

**29.241.2.1** [Go::IntersectorFuncConst::IntersectorFuncConst](#) ( shared\_ptr< [ParamFunctionInt](#) > *func*, shared\_ptr< [ParamFunctionInt](#) > *C*, shared\_ptr< [GeoTol](#) > *epsge*, [Intersector](#) \* *prev* = 0, int *eliminated\_parameter* = -1, double *eliminated\_value* = 0 )

Constructor. The last two variables are relevant only if the parent has one more parameter than the [Intersector](#) to be constructed.

#### Parameters

<i>func</i>	of type <a href="#">ParamFunctionInt</a> .
<i>C</i>	of type <a href="#">Param0FunctionInt</a> .
<i>epsge</i>	the associated tolerance.
<i>prev</i>	the "parent" <a href="#">Intersector</a> (0 if there is no parent).
<i>eliminated_parameter</i>	the index of the parameter that was removed from the parent <i>prev</i> .
<i>eliminated_value</i>	the value of the parameter that was removed from the parent <i>prev</i> .

29.241.2.2 virtual `Go::IntersectorFuncConst::~~IntersectorFuncConst ( )` [virtual]

Destructor.

### 29.241.3 Member Function Documentation

29.241.3.1 virtual `int Go::IntersectorFuncConst::checkCoincidence ( )` [protected],[pure virtual]

Implements [Go::Intersector](#).

Implemented in [Go::Par1FuncIntersector](#), [Go::Par2FuncIntersector](#), and [Go::Par0FuncIntersector](#).

29.241.3.2 virtual `bool Go::IntersectorFuncConst::complexityReduced ( )` [inline],[protected],[virtual]

Implements [Go::Intersector](#).

Definition at line 116 of file `IntersectorFuncConst.h`.

29.241.3.3 virtual `int Go::IntersectorFuncConst::doSubdivide ( )` [protected],[pure virtual]

Implements [Go::Intersector](#).

Implemented in [Go::Par2FuncIntersector](#), [Go::Par1FuncIntersector](#), and [Go::Par0FuncIntersector](#).

29.241.3.4 virtual `int Go::IntersectorFuncConst::getBoundaryIntersections ( )` [protected],[virtual]

Implements [Go::Intersector](#).

29.241.3.5 virtual `void Go::IntersectorFuncConst::handleComplexity ( )` [inline],[protected],[virtual]

Implements [Go::Intersector](#).

Definition at line 122 of file `IntersectorFuncConst.h`.

29.241.3.6 virtual `bool Go::IntersectorFuncConst::isLinear ( )` [protected],[virtual]

Implements [Go::Intersector](#).

29.241.3.7 virtual `int Go::IntersectorFuncConst::linearCase ( )` [protected],[virtual]

Implements [Go::Intersector](#).

29.241.3.8 `virtual shared_ptr<Intersector> Go::IntersectorFuncConst::lowerOrderIntersector ( shared_ptr< ParamFunctionInt > obj1, shared_ptr< ParamFunctionInt > obj2, Intersector * prev = 0, int eliminated_parameter = -1, double eliminated_value = 0 )` [protected],[pure virtual]

Implemented in [Go::Par1FuncIntersector](#), [Go::Par2FuncIntersector](#), and [Go::Par0FuncIntersector](#).

29.241.3.9 `virtual void Go::IntersectorFuncConst::microCase ( )` [protected],[pure virtual]

Implements [Go::Intersector](#).

Implemented in [Go::Par1FuncIntersector](#), [Go::Par2FuncIntersector](#), and [Go::Par0FuncIntersector](#).

29.241.3.10 `virtual int Go::IntersectorFuncConst::performInterception ( )` [protected],[virtual]

Implements [Go::Intersector](#).

29.241.3.11 `virtual void Go::IntersectorFuncConst::print_objs ( )` [protected],[virtual]

Implements [Go::Intersector](#).

29.241.3.12 `virtual void Go::IntersectorFuncConst::printDebugInfo ( )` [protected],[virtual]

Implements [Go::Intersector](#).

29.241.3.13 `virtual int Go::IntersectorFuncConst::simpleCase ( )` [protected],[virtual]

Implements [Go::Intersector](#).

29.241.3.14 `virtual int Go::IntersectorFuncConst::updateIntersections ( )` [protected],[pure virtual]

Implements [Go::Intersector](#).

Implemented in [Go::Par1FuncIntersector](#), [Go::Par2FuncIntersector](#), and [Go::Par0FuncIntersector](#).

## 29.241.4 Friends And Related Function Documentation

29.241.4.1 `friend class IntersectorAlgPar` [friend]

Definition at line 84 of file [IntersectorFuncConst.h](#).

### 29.241.5 Member Data Documentation

29.241.5.1 `shared_ptr<ParamFunctionInt> Go::IntersectorFuncConst::C_` [protected]

Definition at line 89 of file `IntersectorFuncConst.h`.

29.241.5.2 `shared_ptr<ParamFunctionInt> Go::IntersectorFuncConst::func_int_` [protected]

Definition at line 88 of file `IntersectorFuncConst.h`.

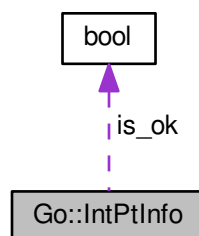
The documentation for this class was generated from the following file:

- [intersections/include/GoTools/intersections/IntersectorFuncConst.h](#)

### 29.242 Go::IntPtInfo Struct Reference

```
#include <IntersectionPointUtils.h>
```

Collaboration diagram for `Go::IntPtInfo`:



#### Public Member Functions

- [IntPtInfo \(\)](#)

#### Public Attributes

- `int` [neighbours](#)
- `SingularityType` [singularity\\_type](#)
- `IntPtClassification` [direction](#)
- `IntPtLocation` [location](#)
- `bool` [is\\_ok](#)

### 29.242.1 Detailed Description

Struct that holds diagnostic information about an intersection point. Used by [IntersectionPool::checkIntersectionPoints\(\)](#).

Definition at line 108 of file IntersectionPointUtils.h.

### 29.242.2 Constructor & Destructor Documentation

#### 29.242.2.1 Go::IntPtInfo::IntPtInfo ( ) `[inline]`

Definition at line 110 of file IntersectionPointUtils.h.

### 29.242.3 Member Data Documentation

#### 29.242.3.1 IntPtClassification Go::IntPtInfo::direction

Definition at line 114 of file IntersectionPointUtils.h.

#### 29.242.3.2 bool Go::IntPtInfo::is\_ok

Definition at line 116 of file IntersectionPointUtils.h.

#### 29.242.3.3 IntPtLocation Go::IntPtInfo::location

Definition at line 115 of file IntersectionPointUtils.h.

#### 29.242.3.4 int Go::IntPtInfo::nneighbours

Definition at line 112 of file IntersectionPointUtils.h.

#### 29.242.3.5 SingularityType Go::IntPtInfo::singularity\_type

Definition at line 113 of file IntersectionPointUtils.h.

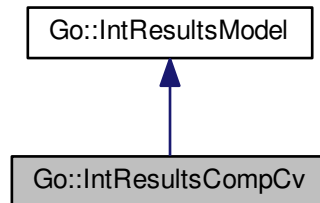
The documentation for this struct was generated from the following file:

- intersections/include/GoTools/intersections/[IntersectionPointUtils.h](#)

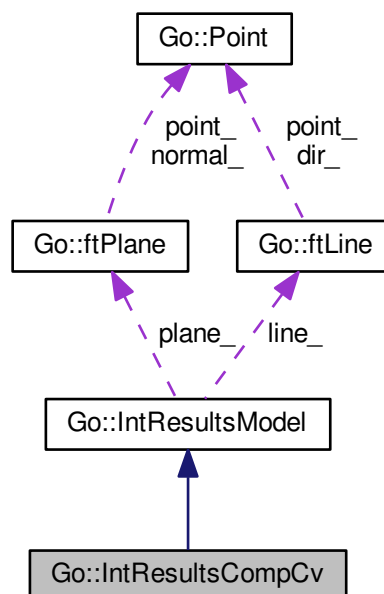
## 29.243 Go::IntResultsCompCv Class Reference

```
#include <IntResultsCompCv.h>
```

Inheritance diagram for Go::IntResultsCompCv:



Collaboration diagram for Go::IntResultsCompCv:



### Public Member Functions

- [IntResultsCompCv](#) ([CompositeCurve](#) \*compcv, [const ftLine](#) &line)  
*Constructor.*
- [IntResultsCompCv](#) ([CompositeCurve](#) \*compcv, [const ftPlane](#) &plane)



*Constructor.*

- `~IntResultsCompCv ()`

*Destructor.*

- void `addIntPt` (shared\_ptr< [ParamCurve](#) > cv, double \*parval)

*Add an intersection point.*

- void `addIntCv` (shared\_ptr< [ParamCurve](#) > cv, double \*startpar, double \*endpar)

*Add an intersection curve.*

- virtual `bool hasIntCurves () const`

*Check if any intersection curves are found.*

- virtual `int nmbCurveSegments () const`

*Return the number of intersection curve segments.*

- virtual `bool hasIntPoints () const`

*Check if any intersection point are found.*

- virtual `int nmbIntPoints () const`

*Return the number of intersection points.*

- void `getIntersectionPoints` (std::vector< [PointOnCurve](#) > &int\_points) `const`

*Fetch all intersection points.*

- void `getIntersectionCurves` (std::vector< std::pair< [PointOnCurve](#), [PointOnCurve](#) > > &int\_crvs) `const`

*Fetch all intersection curves.*

- virtual `bool hasIntersections () const`

*Check if any intersections are found.*

- virtual void `tessellate` (std::vector< shared\_ptr< [LineStrip](#) > > &meshes, [PointCloud3D](#) &points) `const`

- virtual void `tessellate` (int resolution, std::vector< shared\_ptr< [LineStrip](#) > > &meshes, [PointCloud3D](#) &points) `const`

- virtual void `tessellate` (double density, std::vector< shared\_ptr< [LineStrip](#) > > &meshes, [PointCloud3D](#) &points) `const`

*Tessellate with respect to a given density.*

## Additional Inherited Members

### 29.243.1 Detailed Description

Storage of results from intersection operations related to a [CompositeCurve](#)

Definition at line 58 of file `IntResultsCompCv.h`.

### 29.243.2 Constructor & Destructor Documentation

#### 29.243.2.1 Go::IntResultsCompCv::IntResultsCompCv ( [CompositeCurve](#) \* *compcv*, const [ftLine](#) & *line* )

Constructor.

#### 29.243.2.2 Go::IntResultsCompCv::IntResultsCompCv ( [CompositeCurve](#) \* *compcv*, const [ftPlane](#) & *plane* )

Constructor.

29.243.2.3 `Go::IntResultsCompCv::~~IntResultsCompCv ( )`

Destructor.

### 29.243.3 Member Function Documentation

29.243.3.1 `void Go::IntResultsCompCv::addIntCv ( shared_ptr< ParamCurve > cv, double * startpar, double * endpar )`

Add an intersection curve.

29.243.3.2 `void Go::IntResultsCompCv::addIntPt ( shared_ptr< ParamCurve > cv, double * parval )`

Add an intersection point.

29.243.3.3 `void Go::IntResultsCompCv::getIntersectionCurves ( std::vector< std::pair< PointOnCurve, PointOnCurve > & int_crvs ) const`

Fetch all intersection curves.

29.243.3.4 `void Go::IntResultsCompCv::getIntersectionPoints ( std::vector< PointOnCurve > & int_points ) const`

Fetch all intersection points.

29.243.3.5 `virtual bool Go::IntResultsCompCv::hasIntCurves ( ) const [inline],[virtual]`

Check if any intersection curves are found.

Implements [Go::IntResultsModel](#).

Definition at line 80 of file `IntResultsCompCv.h`.

29.243.3.6 `virtual bool Go::IntResultsCompCv::hasIntersections ( ) const [inline],[virtual]`

Check if any intersections are found.

Implements [Go::IntResultsModel](#).

Definition at line 111 of file `IntResultsCompCv.h`.

29.243.3.7 `virtual bool Go::IntResultsCompCv::hasIntPoints ( ) const [inline],[virtual]`

Check if any intersection point are found.

Implements [Go::IntResultsModel](#).

Definition at line 92 of file `IntResultsCompCv.h`.

29.243.3.8 `virtual int Go::IntResultsCompCv::nmbCurveSegments ( ) const [inline],[virtual]`

Return the number of intersection curve segments.

Implements [Go::IntResultsModel](#).

Definition at line 86 of file `IntResultsCompCv.h`.

29.243.3.9 `virtual int Go::IntResultsCompCv::nmbIntPoints ( ) const [inline],[virtual]`

Return the number of intersection points.

Implements [Go::IntResultsModel](#).

Definition at line 98 of file `IntResultsCompCv.h`.

29.243.3.10 `virtual void Go::IntResultsCompCv::tessellate ( std::vector< shared_ptr< LineStrip > > & meshes, PointCloud3D & points ) const [virtual]`

Tessellation Tessellate with respect to a default value

Implements [Go::IntResultsModel](#).

29.243.3.11 `virtual void Go::IntResultsCompCv::tessellate ( int resolution, std::vector< shared_ptr< LineStrip > > & meshes, PointCloud3D & points ) const [virtual]`

Tessellate with respect to a given resolution, the same for each intersection curve

Implements [Go::IntResultsModel](#).

29.243.3.12 `virtual void Go::IntResultsCompCv::tessellate ( double density, std::vector< shared_ptr< LineStrip > > & meshes, PointCloud3D & points ) const [virtual]`

Tessellate with respect to a given density.

Implements [Go::IntResultsModel](#).

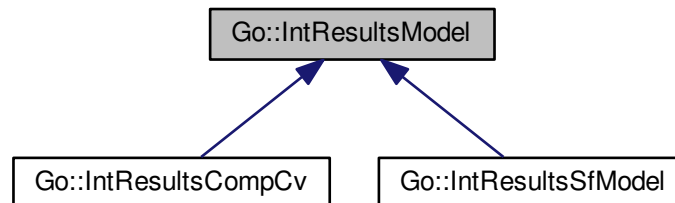
The documentation for this class was generated from the following file:

- `compositemodel/include/GoTools/compositemodel/IntResultsCompCv.h`

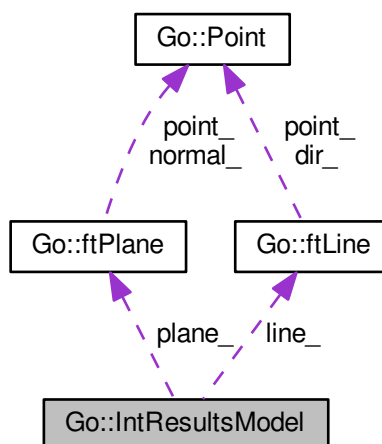
## 29.244 Go::IntResultsModel Class Reference

```
#include <IntResultsModel.h>
```

Inheritance diagram for Go::IntResultsModel:



Collaboration diagram for Go::IntResultsModel:



### Public Member Functions

- [IntResultsModel](#) ([IntersectionType](#) type)  
*Constructor.*
- [~IntResultsModel](#) ()  
*Destructor.*
- void [addPlaneInfo](#) ([const ftPlane](#) &plane)  
*Plane involved in the intersection.*
- void [addLineInfo](#) ([const ftLine](#) &line)

- virtual [bool](#) `hasIntCurves () const =0`  
*Check if any intersection curves are found.*
- virtual [int](#) `nmbCurveSegments () const =0`  
*Return the number of intersection curve segments.*
- virtual [bool](#) `hasIntPoints () const =0`  
*Check if any intersection point are found.*
- virtual [int](#) `nmbIntPoints () const =0`  
*Return the number of intersection points.*
- virtual [bool](#) `hasIntersections () const =0`  
*Check if any intersections are found.*
- virtual [void](#) `tessellate (std::vector< shared_ptr< LineStrip > > &meshes, PointCloud3D &points) const =0`
- virtual [void](#) `tessellate (int resolution, std::vector< shared_ptr< LineStrip > > &meshes, PointCloud3D &points) const =0`
- virtual [void](#) `tessellate (double density, std::vector< shared_ptr< LineStrip > > &meshes, PointCloud3D &points) const =0`  
*Tessellate with respect to a given density.*

## Protected Attributes

- [IntersectionType](#) `type_`
- [ftLine](#) `line_`
- [ftPlane](#) `plane_`
- [int](#) `numpar_`

### 29.244.1 Detailed Description

Storage of results from intersection operations related to a [CompositeModel](#)

Definition at line 72 of file `IntResultsModel.h`.

### 29.244.2 Constructor & Destructor Documentation

#### 29.244.2.1 `Go::IntResultsModel::IntResultsModel ( IntersectionType type )`

Constructor.

#### 29.244.2.2 `Go::IntResultsModel::~~IntResultsModel ( ) [inline]`

Destructor.

Definition at line 80 of file `IntResultsModel.h`.

### 29.244.3 Member Function Documentation

#### 29.244.3.1 `void Go::IntResultsModel::addLineInfo ( const ftLine & line ) [inline]`

[Line](#) involved in the intersection.

Definition at line 90 of file `IntResultsModel.h`.

29.244.3.2 void Go::IntResultsModel::addPlaneInfo ( const *ftPlane* & *plane* ) [inline]

[Plane](#) involved in the intersection.

Definition at line 84 of file IntResultsModel.h.

29.244.3.3 virtual bool Go::IntResultsModel::hasIntCurves ( ) const [pure virtual]

Check if any intersection curves are found.

Implemented in [Go::IntResultsSfModel](#), and [Go::IntResultsCompCv](#).

29.244.3.4 virtual bool Go::IntResultsModel::hasIntersections ( ) const [pure virtual]

Check if any intersections are found.

Implemented in [Go::IntResultsSfModel](#), and [Go::IntResultsCompCv](#).

29.244.3.5 virtual bool Go::IntResultsModel::hasIntPoints ( ) const [pure virtual]

Check if any intersection point are found.

Implemented in [Go::IntResultsSfModel](#), and [Go::IntResultsCompCv](#).

29.244.3.6 virtual int Go::IntResultsModel::nmbCurveSegments ( ) const [pure virtual]

Return the number of intersection curve segments.

Implemented in [Go::IntResultsSfModel](#), and [Go::IntResultsCompCv](#).

29.244.3.7 virtual int Go::IntResultsModel::nmbIntPoints ( ) const [pure virtual]

Return the number of intersection points.

Implemented in [Go::IntResultsSfModel](#), and [Go::IntResultsCompCv](#).

29.244.3.8 virtual void Go::IntResultsModel::tessellate ( std::vector< shared\_ptr< **LineStrip** > > & *meshes*,  
PointCloud3D & *points* ) const [pure virtual]

Tesselation Tessellate with respect to a default value

Implemented in [Go::IntResultsSfModel](#), and [Go::IntResultsCompCv](#).

29.244.3.9 `virtual void Go::IntResultsModel::tessellate ( int resolution, std::vector< shared_ptr< LineStrip > > & meshes, PointCloud3D & points ) const` [pure virtual]

Tessellate with respect to a given resolution, the same for each intersection curve

Implemented in [Go::IntResultsSfModel](#), and [Go::IntResultsCompCv](#).

29.244.3.10 `virtual void Go::IntResultsModel::tessellate ( double density, std::vector< shared_ptr< LineStrip > > & meshes, PointCloud3D & points ) const` [pure virtual]

Tessellate with respect to a given density.

Implemented in [Go::IntResultsSfModel](#), and [Go::IntResultsCompCv](#).

#### 29.244.4 Member Data Documentation

29.244.4.1 `ftLine Go::IntResultsModel::line_` [protected]

Definition at line 128 of file `IntResultsModel.h`.

29.244.4.2 `int Go::IntResultsModel::numpar_` [protected]

Definition at line 130 of file `IntResultsModel.h`.

29.244.4.3 `ftPlane Go::IntResultsModel::plane_` [protected]

Definition at line 129 of file `IntResultsModel.h`.

29.244.4.4 `IntersectionType Go::IntResultsModel::type_` [protected]

Definition at line 127 of file `IntResultsModel.h`.

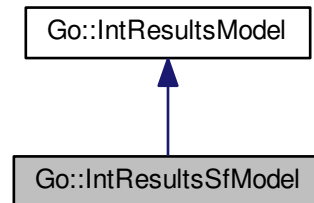
The documentation for this class was generated from the following file:

- `compositemodel/include/GoTools/compositemodel/IntResultsModel.h`

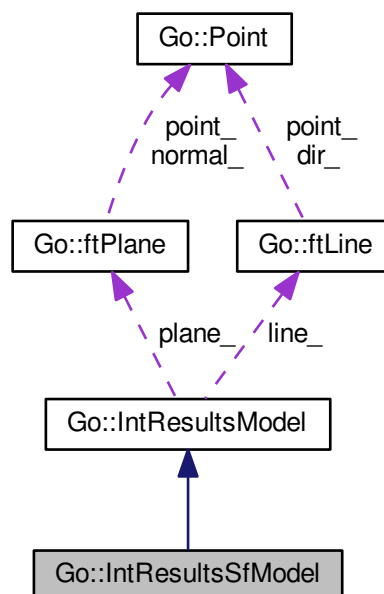
## 29.245 Go::IntResultsSfModel Class Reference

```
#include <IntResultsSfModel.h>
```

Inheritance diagram for Go::IntResultsSfModel:



Collaboration diagram for Go::IntResultsSfModel:



### Public Member Functions

- [IntResultsSfModel](#) ([SurfaceModel](#) \*sfmodel, [const ftLine](#) &line)  
*Constructor.*
- [IntResultsSfModel](#) ([SurfaceModel](#) \*sfmodel, [const ftPlane](#) &plane)



*Constructor.*

- `~IntResultsSfModel ()`

*Destructor.*

- void `addIntPts (std::vector< ftPoint > &intpts)`

*Add an intersection point.*

- void `addIntCvs (ftCurve &cvs)`

*Add an intersection curve.*

- virtual `bool hasIntCurves () const`

*Check if any intersection curves are found.*

- virtual `int nmbCurveSegments () const`

*Return the number of intersection curve segments.*

- `const ftCurve & getIntersectionCurves ()`

*Fetch all intersection curve segments joined in one curve.*

- virtual `bool hasIntPoints () const`

*Check if any intersection points are found.*

- `int nmbIntPoints () const`

*Return the number of intersection points.*

- `std::vector< ftPoint > & getIntersectionPoints ()`

*Fetch all intersection points.*

- virtual `bool hasIntersections () const`

*Check if any intersections are found.*

- virtual void `tessellate (std::vector< shared_ptr< LineStrip > > &meshes, PointCloud3D &points) const`

- virtual void `tessellate (int resolution, std::vector< shared_ptr< LineStrip > > &meshes, PointCloud3D &points) const`

- virtual void `tessellate (double density, std::vector< shared_ptr< LineStrip > > &meshes, PointCloud3D &points) const`

*Tessellate with respect to a given density.*

## Additional Inherited Members

### 29.245.1 Detailed Description

Storage of results from intersection operations related to a [SurfaceModel](#)

Definition at line 59 of file `IntResultsSfModel.h`.

### 29.245.2 Constructor & Destructor Documentation

#### 29.245.2.1 Go::IntResultsSfModel::IntResultsSfModel ( SurfaceModel \* sfmodel, const ftLine & line )

Constructor.

#### 29.245.2.2 Go::IntResultsSfModel::IntResultsSfModel ( SurfaceModel \* sfmodel, const ftPlane & plane )

Constructor.

29.245.2.3 `Go::IntResultsSfModel::~~IntResultsSfModel ( )`

Destructor.

### 29.245.3 Member Function Documentation

29.245.3.1 `void Go::IntResultsSfModel::addIntCvs ( ftCurve & cvs )`

Add an intersection curve.

29.245.3.2 `void Go::IntResultsSfModel::addIntPts ( std::vector< ftPoint > & intpts )`

Add an intersection point.

29.245.3.3 `const ftCurve& Go::IntResultsSfModel::getIntersectionCurves ( ) [inline]`

Fetch all intersection curve segments joined in one curve.

Definition at line 93 of file `IntResultsSfModel.h`.

29.245.3.4 `std::vector<ftPoint>& Go::IntResultsSfModel::getIntersectionPoints ( ) [inline]`

Fetch all intersection points.

Definition at line 111 of file `IntResultsSfModel.h`.

29.245.3.5 `virtual bool Go::IntResultsSfModel::hasIntCurves ( ) const [inline],[virtual]`

Check if any intersection curves are found.

Implements [Go::IntResultsModel](#).

Definition at line 81 of file `IntResultsSfModel.h`.

29.245.3.6 `virtual bool Go::IntResultsSfModel::hasIntersections ( ) const [inline],[virtual]`

Check if any intersections are found.

Implements [Go::IntResultsModel](#).

Definition at line 117 of file `IntResultsSfModel.h`.

29.245.3.7 `virtual bool Go::IntResultsSfModel::hasIntPoints ( ) const [inline],[virtual]`

Check if any intersection points are found.

Implements [Go::IntResultsModel](#).

Definition at line 99 of file `IntResultsSfModel.h`.

29.245.3.8 `virtual int Go::IntResultsSfModel::nmbCurveSegments ( ) const [inline],[virtual]`

Return the number of intersection curve segments.

Implements [Go::IntResultsModel](#).

Definition at line 87 of file `IntResultsSfModel.h`.

29.245.3.9 `int Go::IntResultsSfModel::nmbIntPoints ( ) const [inline],[virtual]`

Return the number of intersection points.

Implements [Go::IntResultsModel](#).

Definition at line 105 of file `IntResultsSfModel.h`.

29.245.3.10 `virtual void Go::IntResultsSfModel::tessellate ( std::vector< shared_ptr< LineStrip > > & meshes, PointCloud3D & points ) const [virtual]`

Tessellation Tessellate with respect to a default value

Implements [Go::IntResultsModel](#).

29.245.3.11 `virtual void Go::IntResultsSfModel::tessellate ( int resolution, std::vector< shared_ptr< LineStrip > > & meshes, PointCloud3D & points ) const [virtual]`

Tessellate with respect to a given resolution, the same for each intersection curve

Implements [Go::IntResultsModel](#).

29.245.3.12 `virtual void Go::IntResultsSfModel::tessellate ( double density, std::vector< shared_ptr< LineStrip > > & meshes, PointCloud3D & points ) const [virtual]`

Tessellate with respect to a given density.

Implements [Go::IntResultsModel](#).

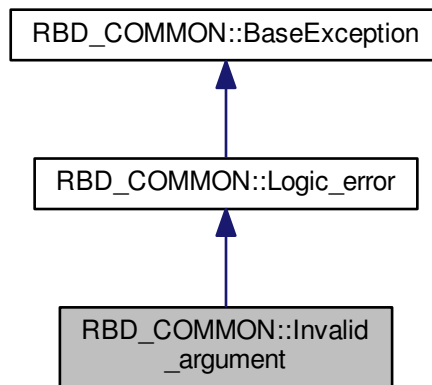
The documentation for this class was generated from the following file:

- `compositemodel/include/GoTools/compositemodel/IntResultsSfModel.h`

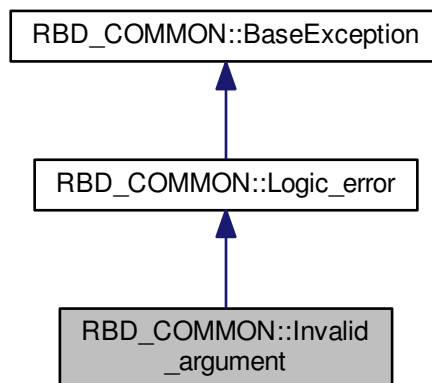
## 29.246 RBD\_COMMON::Invalid\_argument Class Reference

```
#include <myexcept.h>
```

Inheritance diagram for RBD\_COMMON::Invalid\_argument:



Collaboration diagram for RBD\_COMMON::Invalid\_argument:



- static unsigned long [Select](#)
- [Invalid\\_argument](#) (const char \*a\_what=0)

### Additional Inherited Members

#### 29.246.1 Detailed Description

Definition at line 378 of file `myexcept.h`.

## 29.246.2 Constructor & Destructor Documentation

### 29.246.2.1 Invalid\_argument::Invalid\_argument ( const char \* a\_what = 0 )

Definition at line 419 of file myexcept.cpp.

## 29.246.3 Member Data Documentation

### 29.246.3.1 unsigned long Invalid\_argument::Select [static]

Definition at line 381 of file myexcept.h.

The documentation for this class was generated from the following files:

- [newmat/include/myexcept.h](#)
- [newmat/src/myexcept.cpp](#)

## 29.247 Go::InverseFactorial< T, N > Struct Template Reference

```
#include <Factorial.h>
```

### Static Public Member Functions

- [static T val \(\)](#)

### 29.247.1 Detailed Description

```
template<typename T, int N>
struct Go::InverseFactorial< T, N >
```

Compute the inverse of the factorial of a given integer N, where T is typically 'float' or 'double'.

Definition at line 67 of file Factorial.h.

### 29.247.2 Member Function Documentation

#### 29.247.2.1 template<typename T, int N> static T Go::InverseFactorial< T, N >::val ( ) [inline], [static]

Definition at line 68 of file Factorial.h.

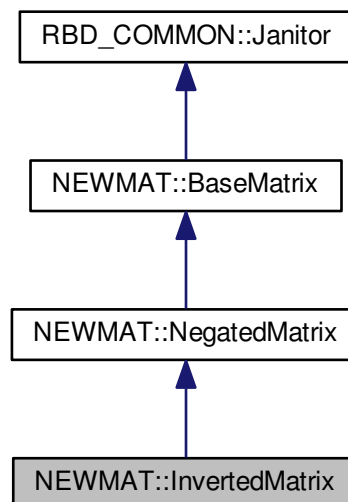
The documentation for this struct was generated from the following file:

- [gotools-core/include/GoTools/utis/Factorial.h](#)

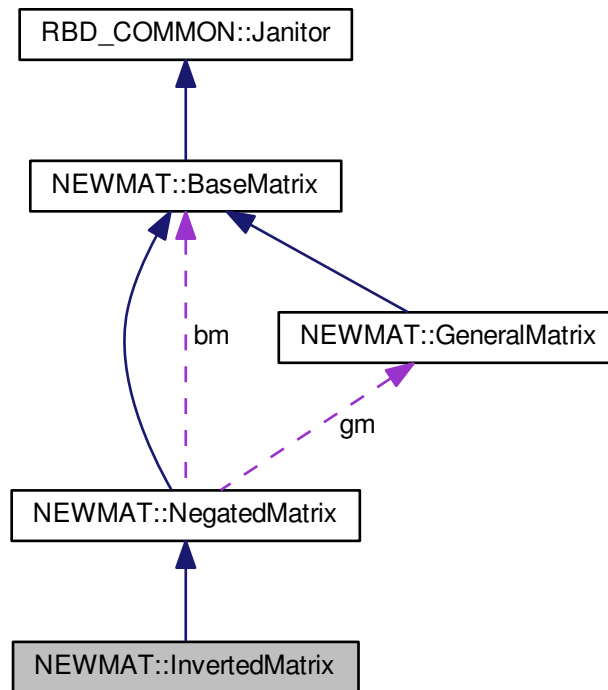
## 29.248 NEWMAT::InvertedMatrix Class Reference

```
#include <newmat.h>
```

Inheritance diagram for NEWMAT::InvertedMatrix:



Collaboration diagram for NEWMAT::InvertedMatrix:



## Public Member Functions

- [~InvertedMatrix \(\)](#)
- [SolvedMatrix operator\\* \(const BaseMatrix &\) const](#)
- [ScaledMatrix operator\\* \(Real t\) const](#)
- [GeneralMatrix \\* Evaluate \(MatrixType mt=MatrixTypeUnSp\)](#)
- [MatrixBandWidth BandWidth \(\) const](#)

## Friends

- class [BaseMatrix](#)

## Additional Inherited Members

### 29.248.1 Detailed Description

Definition at line 1402 of file newmat.h.

## 29.248.2 Constructor & Destructor Documentation

### 29.248.2.1 `NEWMAT::InvertedMatrix::~~InvertedMatrix ( )` [inline]

Definition at line 1406 of file `newmat.h`.

## 29.248.3 Member Function Documentation

### 29.248.3.1 `MatrixBandWidth InvertedMatrix::BandWidth ( )` const [virtual]

Reimplemented from [NEWMAT::NegatedMatrix](#).

Definition at line 474 of file `newmat4.cpp`.

### 29.248.3.2 `GeneralMatrix * InvertedMatrix::Evaluate ( MatrixType mt = MatrixTypeUnSp )` [virtual]

Reimplemented from [NEWMAT::NegatedMatrix](#).

Definition at line 491 of file `newmat7.cpp`.

### 29.248.3.3 `SolvedMatrix NEWMAT::InvertedMatrix::operator* ( const BaseMatrix & )` const

### 29.248.3.4 `ScaledMatrix NEWMAT::InvertedMatrix::operator* ( Real t )` const [inline]

Definition at line 1409 of file `newmat.h`.

## 29.248.4 Friends And Related Function Documentation

### 29.248.4.1 `friend class BaseMatrix` [friend]

Definition at line 1414 of file `newmat.h`.

The documentation for this class was generated from the following files:

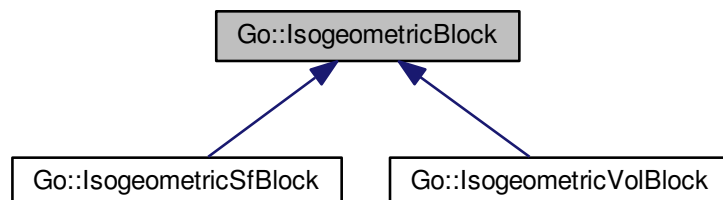
- [newmat/include/newmat.h](#)
- [newmat/src/newmat4.cpp](#)
- [newmat/src/newmat7.cpp](#)



## 29.249 Go::IsogeometricBlock Class Reference

```
#include <IsogeometricBlock.h>
```

Inheritance diagram for Go::IsogeometricBlock:



### Public Member Functions

- [IsogeometricBlock](#) ([IsogeometricModel](#) \**model*)
- virtual [~IsogeometricBlock](#) ()
- virtual [IsogeometricModel](#) \* *model* ()
- virtual [IsogeometricSfBlock](#) \* *asIsogeometricSfBlock* ()
- virtual [IsogeometricVolBlock](#) \* *asIsogeometricVolBlock* ()
- virtual int *nmbOfNeighbours* () *const* =0
- virtual [IsogeometricBlock](#) \* *getNeighbour* (int *bd\_nmb*) *const* =0
- virtual [bool](#) *isNeighbour* ([IsogeometricBlock](#) \**other*) *const* =0
- virtual int *nmbCoefs* () *const* =0
- virtual int *nmbCoefs* (int *pardir*) *const*
- virtual int *degree* (int *pardir*) *const*
- virtual [std::vector](#)< [double](#) > *knots* (int *pardir*) *const*
- virtual [std::vector](#)< [double](#) > *distinctKnots* (int *pardir*) *const*
- virtual [BsplineBasis](#) *basis* (int *pardir*) *const* =0
- virtual void *refineGeometry* ([std::vector](#)< [double](#) > *newknots*, int *pardir*)=0
- virtual void *refineGeometry* ([const](#) [BsplineBasis](#) &*other\_basis*, int *pardir*)=0
- virtual void *increaseGeometryDegree* (int *new\_degree*, int *pardir*)=0
- virtual void *erasePreEvaluatedBasisFunctions* ()=0
- virtual int *getNmbOfBoundaryConditions* () *const* =0
- virtual int *getNmbOfPointBdConditions* () *const* =0
- virtual [tpTolerances](#) *getTolerances* () *const*

### 29.249.1 Detailed Description

Definition at line 58 of file [IsogeometricBlock.h](#).

## 29.249.2 Constructor & Destructor Documentation

29.249.2.1 `Go::IsogeometricBlock::IsogeometricBlock ( IsogeometricModel * model ) [inline]`

Definition at line 62 of file `IsogeometricBlock.h`.

29.249.2.2 `virtual Go::IsogeometricBlock::~~IsogeometricBlock ( ) [virtual]`

## 29.249.3 Member Function Documentation

29.249.3.1 `virtual IsogeometricSfBlock* Go::IsogeometricBlock::asIsogeometricSfBlock ( ) [virtual]`

Reimplemented in [Go::IsogeometricSfBlock](#).

29.249.3.2 `virtual IsogeometricVolBlock* Go::IsogeometricBlock::asIsogeometricVolBlock ( ) [virtual]`

Reimplemented in [Go::IsogeometricVolBlock](#).

29.249.3.3 `virtual BsplineBasis Go::IsogeometricBlock::basis ( int pdir ) const [pure virtual]`

Implemented in [Go::IsogeometricSfBlock](#), and [Go::IsogeometricVolBlock](#).

29.249.3.4 `virtual int Go::IsogeometricBlock::degree ( int pdir ) const [virtual]`

29.249.3.5 `virtual std::vector<double> Go::IsogeometricBlock::distinctKnots ( int pdir ) const [virtual]`

29.249.3.6 `virtual void Go::IsogeometricBlock::erasePreEvaluatedBasisFunctions ( ) [pure virtual]`

Implemented in [Go::IsogeometricVolBlock](#), and [Go::IsogeometricSfBlock](#).

29.249.3.7 `virtual IsogeometricBlock* Go::IsogeometricBlock::getNeighbour ( int bd_nmb ) const [pure virtual]`

Implemented in [Go::IsogeometricSfBlock](#), and [Go::IsogeometricVolBlock](#).

29.249.3.8 `virtual int Go::IsogeometricBlock::getNmbOfBoundaryConditions ( ) const [pure virtual]`

Implemented in [Go::IsogeometricVolBlock](#), and [Go::IsogeometricSfBlock](#).

29.249.3.9 `virtual int Go::IsogeometricBlock::getNmbOfPointBdConditions ( ) const [pure virtual]`

Implemented in [Go::IsogeometricVolBlock](#), and [Go::IsogeometricSfBlock](#).

29.249.3.10 virtual `tpTolerances` Go::IsogeometricBlock::getTolerances ( ) const [virtual]

29.249.3.11 virtual void Go::IsogeometricBlock::increaseGeometryDegree ( int *new\_degree*, int *pardir* ) [pure virtual]

Implemented in [Go::IsogeometricVolBlock](#), and [Go::IsogeometricSfBlock](#).

29.249.3.12 virtual `bool` Go::IsogeometricBlock::isNeighbour ( `IsogeometricBlock * other` ) const [pure virtual]

Implemented in [Go::IsogeometricSfBlock](#), and [Go::IsogeometricVolBlock](#).

29.249.3.13 virtual `std::vector<double>` Go::IsogeometricBlock::knots ( int *pardir* ) const [virtual]

29.249.3.14 virtual `IsogeometricModel*` Go::IsogeometricBlock::model ( ) [inline],[virtual]

Definition at line 71 of file `IsogeometricBlock.h`.

29.249.3.15 virtual int Go::IsogeometricBlock::nmbCoefs ( ) const [pure virtual]

Implemented in [Go::IsogeometricSfBlock](#), and [Go::IsogeometricVolBlock](#).

29.249.3.16 virtual int Go::IsogeometricBlock::nmbCoefs ( int *pardir* ) const [virtual]

29.249.3.17 virtual int Go::IsogeometricBlock::nmbOfNeighbours ( ) const [pure virtual]

Implemented in [Go::IsogeometricSfBlock](#), and [Go::IsogeometricVolBlock](#).

29.249.3.18 virtual void Go::IsogeometricBlock::refineGeometry ( `std::vector< double > newknots`, int *pardir* ) [pure virtual]

Implemented in [Go::IsogeometricVolBlock](#), and [Go::IsogeometricSfBlock](#).

29.249.3.19 virtual void Go::IsogeometricBlock::refineGeometry ( `const BsplineBasis & other_basis`, int *pardir* ) [pure virtual]

Implemented in [Go::IsogeometricVolBlock](#), and [Go::IsogeometricSfBlock](#).

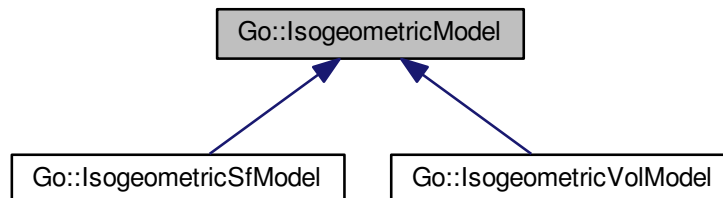
The documentation for this class was generated from the following file:

- [isogeometric\\_model/include/GoTools/isogeometric\\_model/IsogeometricBlock.h](#)

## 29.250 Go::IsogeometricModel Class Reference

```
#include <IsogeometricModel.h>
```

Inheritance diagram for Go::IsogeometricModel:



### Public Member Functions

- [IsogeometricModel](#) ([tpTolerances](#) toptol)
- virtual [~IsogeometricModel](#) ()
- virtual int [getNmbOfBoundaries](#) () const =0
- virtual void [setMinimumDegree](#) (int [degree](#), int solutionspace\_idx)=0
- virtual void [updateSolutionSplineSpace](#) ()=0
- virtual [tpTolerances](#) [getTolerances](#) () const
- virtual void [setTolerances](#) ([tpTolerances](#) t)

### 29.250.1 Detailed Description

Definition at line 51 of file IsogeometricModel.h.

### 29.250.2 Constructor & Destructor Documentation

29.250.2.1 [Go::IsogeometricModel::IsogeometricModel](#) ( [tpTolerances](#) *toptol* ) `[inline]`

Definition at line 55 of file IsogeometricModel.h.

29.250.2.2 virtual [Go::IsogeometricModel::~~IsogeometricModel](#) ( ) `[virtual]`

### 29.250.3 Member Function Documentation

29.250.3.1 virtual int [Go::IsogeometricModel::getNmbOfBoundaries](#) ( ) const `[pure virtual]`

Implemented in [Go::IsogeometricSfModel](#), and [Go::IsogeometricVolModel](#).

29.250.3.2 `virtual tpTolerances Go::IsogeometricModel::getTolerances ( ) const [inline],[virtual]`

Definition at line 73 of file IsogeometricModel.h.

29.250.3.3 `virtual void Go::IsogeometricModel::setMinimumDegree ( int degree, int solutionspace_idx ) [pure virtual]`

Implemented in [Go::IsogeometricSfModel](#), and [Go::IsogeometricVolModel](#).

29.250.3.4 `virtual void Go::IsogeometricModel::setTolerances ( tpTolerances t ) [inline],[virtual]`

Definition at line 77 of file IsogeometricModel.h.

29.250.3.5 `virtual void Go::IsogeometricModel::updateSolutionSplineSpace ( ) [pure virtual]`

Implemented in [Go::IsogeometricSfModel](#), and [Go::IsogeometricVolModel](#).

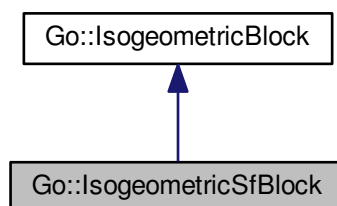
The documentation for this class was generated from the following file:

- [isogeometric\\_model/include/GoTools/isogeometric\\_model/IsogeometricModel.h](#)

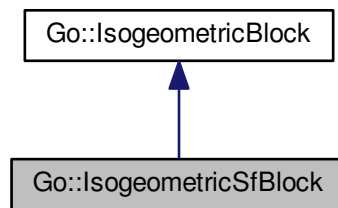
## 29.251 Go::IsogeometricSfBlock Class Reference

```
#include <IsogeometricSfBlock.h>
```

Inheritance diagram for Go::IsogeometricSfBlock:



Collaboration diagram for Go::IsogeometricSfBlock:



## Public Member Functions

- [IsogeometricSfBlock](#) ([IsogeometricModel](#) \*model, [shared\\_ptr](#)< [SplineSurface](#) > geom\_sf, [std::vector](#)< int > solution\_space\_dimension, int index)
- virtual [~IsogeometricSfBlock](#) ()
- virtual [IsogeometricSfBlock](#) \* [asIsogeometricSfBlock](#) ()
- void [addNeighbour](#) ([shared\\_ptr](#)< [IsogeometricSfBlock](#) > neighbour, int edge\_nmb\_this, int edge\_nmb\_other, [bool](#) equ\_orient)
- virtual int [nmbOfNeighbours](#) () const
- virtual [IsogeometricSfBlock](#) \* [getNeighbour](#) (int edge\_nmb) const
- virtual [bool](#) [isNeighbour](#) ([IsogeometricBlock](#) \*other) const
- virtual int [nmbCoefs](#) () const
- virtual [BsplineBasis](#) [basis](#) (int paddir) const
- [shared\\_ptr](#)< [SplineCurve](#) > [getGeomBoundaryCurve](#) (int edge\_number) const
- [double](#) [getParamOnBdCurve](#) (int edge\_number, [const Point](#) &position) const
- [bool](#) [geomIsDegenerate](#) ([std::vector](#)< int > &degen\_bd, [double](#) epsge)
- void [getDegenEnumeration](#) ([std::vector](#)< int > &degen\_bd, [std::vector](#)< [std::vector](#)< int > > &enumeration, [double](#) epsge)
- [bool](#) [geomIsPeriodic](#) (int per[], [double](#) epsge)
- [bool](#) [getPeriodicEnumeration](#) (int paddir, [std::vector](#)< [std::pair](#)< int, int > > &enumeration)
- virtual void [refineGeometry](#) ([std::vector](#)< [double](#) > newknots, int paddir)
- virtual void [refineGeometry](#) ([const BsplineBasis](#) &other\_basis, int paddir)
- virtual void [increaseGeometryDegree](#) (int new\_degree, int paddir)
- void [updateGeometry](#) ([shared\\_ptr](#)< [SplineCurve](#) > new\_boundary, int edge\_number)
- virtual void [erasePreEvaluatedBasisFunctions](#) ()
- virtual int [getNmbOfBoundaryConditions](#) () const
- void [getEdgeBoundaryConditions](#) (int edge\_number, [std::vector](#)< [shared\\_ptr](#)< [SfBoundaryCondition](#) > > &bd\_cond) const
- virtual int [getNmbOfPointBdConditions](#) () const
- void [getEdgePointBdConditions](#) (int edge\_number, [std::vector](#)< [shared\\_ptr](#)< [SfPointBdCond](#) > > &bd\_cond) const
- [shared\\_ptr](#)< [SfSolution](#) > [getSolutionSpace](#) (int solution\_index)
- [shared\\_ptr](#)< [SplineSurface](#) > [surface](#) () const
- virtual void [setMinimumDegree](#) (int degree, int solutionspace\_idx)
- virtual [bool](#) [updateSolutionSplineSpace](#) (int solutionspace\_idx)
- int [nmbSolutionSpaces](#) () const
- int [getEdgeOrientation](#) ([shared\\_ptr](#)< [ParamCurve](#) > crv, [double](#) tol)
- void [getNeighbourInfo](#) ([IsogeometricSfBlock](#) \*other, [std::vector](#)< int > &edges, [std::vector](#)< int > &edges\_↔\_other, [std::vector](#)< [bool](#) > &equal\_oriented)

### 29.251.1 Detailed Description

Definition at line 65 of file IsogeometricSfBlock.h.

### 29.251.2 Constructor & Destructor Documentation

29.251.2.1 **Go::IsogeometricSfBlock::IsogeometricSfBlock ( IsogeometricModel \* *model*, shared\_ptr< SplineSurface > *geom\_sf*, std::vector< int > *solution\_space\_dimension*, int *index* )**

29.251.2.2 **virtual Go::IsogeometricSfBlock::~IsogeometricSfBlock ( )** [virtual]

### 29.251.3 Member Function Documentation

29.251.3.1 **void Go::IsogeometricSfBlock::addNeighbour ( shared\_ptr< IsogeometricSfBlock > *neighbour*, int *edge\_nmb\_this*, int *edge\_nmb\_other*, bool *equ\_orient* )**

29.251.3.2 **virtual IsogeometricSfBlock\* Go::IsogeometricSfBlock::asIsogeometricSfBlock ( )** [virtual]

Reimplemented from [Go::IsogeometricBlock](#).

29.251.3.3 **virtual BsplineBasis Go::IsogeometricSfBlock::basis ( int *pardir* ) const** [virtual]

Implements [Go::IsogeometricBlock](#).

29.251.3.4 **virtual void Go::IsogeometricSfBlock::erasePreEvaluatedBasisFunctions ( )** [virtual]

Implements [Go::IsogeometricBlock](#).

29.251.3.5 **bool Go::IsogeometricSfBlock::geomsDegenerate ( std::vector< int > & *degen\_bd*, double *epsge* )**

29.251.3.6 **bool Go::IsogeometricSfBlock::geomsPeriodic ( int *per*[], double *epsge* )**

29.251.3.7 **void Go::IsogeometricSfBlock::getDegenEnumeration ( std::vector< int > & *degen\_bd*, std::vector< std::vector< int > > & *enumeration*, double *epsge* )**

29.251.3.8 **void Go::IsogeometricSfBlock::getEdgeBoundaryConditions ( int *edge\_number*, std::vector< shared\_ptr< SfBoundaryCondition > > & *bd\_cond* ) const**

29.251.3.9 **int Go::IsogeometricSfBlock::getEdgeOrientation ( shared\_ptr< ParamCurve > *crv*, double *tol* )**

29.251.3.10 **void Go::IsogeometricSfBlock::getEdgePointBdConditions ( int *edge\_number*, std::vector< shared\_ptr< SfPointBdCond > > & *bd\_cond* ) const**

29.251.3.11 **shared\_ptr< SplineCurve > Go::IsogeometricSfBlock::getGeomBoundaryCurve ( int *edge\_number* ) const**

29.251.3.12 **virtual IsogeometricSfBlock\* Go::IsogeometricSfBlock::getNeighbour ( int *edge\_nmb* ) const** [virtual]

Implements [Go::IsogeometricBlock](#).

29.251.3.13 void Go::IsogeometricSfBlock::getNeighbourInfo ( IsogeometricSfBlock \* other, std::vector< int > & edges, std::vector< int > & edges\_other, std::vector< bool > & equal\_oriented )

29.251.3.14 virtual int Go::IsogeometricSfBlock::getNmbOfBoundaryConditions ( ) const [virtual]

Implements [Go::IsogeometricBlock](#).

29.251.3.15 virtual int Go::IsogeometricSfBlock::getNmbOfPointBdConditions ( ) const [virtual]

Implements [Go::IsogeometricBlock](#).

29.251.3.16 double Go::IsogeometricSfBlock::getParamOnBdCurve ( int edge\_number, const Point & position ) const

29.251.3.17 bool Go::IsogeometricSfBlock::getPeriodicEnumeration ( int pardir, std::vector< std::pair< int, int > > & enumeration )

29.251.3.18 shared\_ptr<SfSolution> Go::IsogeometricSfBlock::getSolutionSpace ( int solution\_index )

29.251.3.19 virtual void Go::IsogeometricSfBlock::increaseGeometryDegree ( int new\_degree, int pardir ) [virtual]

Implements [Go::IsogeometricBlock](#).

29.251.3.20 virtual bool Go::IsogeometricSfBlock::isNeighbour ( IsogeometricBlock \* other ) const [virtual]

Implements [Go::IsogeometricBlock](#).

29.251.3.21 virtual int Go::IsogeometricSfBlock::nmbCoefs ( ) const [virtual]

Implements [Go::IsogeometricBlock](#).

29.251.3.22 virtual int Go::IsogeometricSfBlock::nmbOfNeighbours ( ) const [virtual]

Implements [Go::IsogeometricBlock](#).

29.251.3.23 int Go::IsogeometricSfBlock::nmbSolutionSpaces ( ) const

29.251.3.24 virtual void Go::IsogeometricSfBlock::refineGeometry ( std::vector< double > newknots, int pardir ) [virtual]

Implements [Go::IsogeometricBlock](#).



29.251.3.25 `virtual void Go::IsogeometricSfBlock::refineGeometry ( const BsplineBasis & other_basis, int pardir )`  
`[virtual]`

Implements [Go::IsogeometricBlock](#).

29.251.3.26 `virtual void Go::IsogeometricSfBlock::setMinimumDegree ( int degree, int solutionspace_idx )` `[virtual]`

29.251.3.27 `shared_ptr<SplineSurface> Go::IsogeometricSfBlock::surface ( ) const`

29.251.3.28 `void Go::IsogeometricSfBlock::updateGeometry ( shared_ptr< SplineCurve > new_boundary, int edge_number )`

29.251.3.29 `virtual bool Go::IsogeometricSfBlock::updateSolutionSplineSpace ( int solutionspace_idx )` `[virtual]`

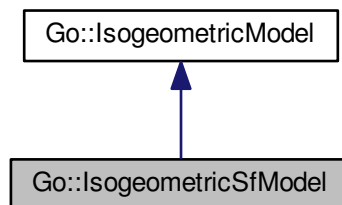
The documentation for this class was generated from the following file:

- [isogeometric\\_model/include/GoTools/isogeometric\\_model/IsogeometricSfBlock.h](#)

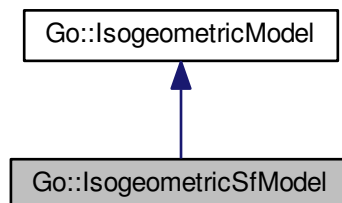
## 29.252 Go::IsogeometricSfModel Class Reference

```
#include <IsogeometricSfModel.h>
```

Inheritance diagram for Go::IsogeometricSfModel:



Collaboration diagram for Go::IsogeometricSfModel:



## Public Member Functions

- [IsogeometricSfModel](#) (shared\_ptr< [SurfaceModel](#) > sfmodel, std::vector< int > solution\_space\_dimension)
- virtual [~IsogeometricSfModel](#) ()
- [bool addBoundaryCond](#) (int boundary, std::vector< [Point](#) > pos, [BdConditionType](#) type, [BdCondFuncor](#) \*fbd, int solutionspace\_idx, [double](#) \*constant\_value=0)
- void [addDirichletPointBdCond](#) (int boundary, [Point](#) &pos, [Point](#) &condition\_value, int solutionspace\_idx)
- virtual int [getNmbOfBoundaries](#) () const
- [CurveLoop](#) [getOuterBoundary](#) () const
- [CurveLoop](#) [getBoundary](#) (int idx) const
- virtual void [setMinimumDegree](#) (int degree, int solutionspace\_idx)
- virtual void [updateSolutionSplineSpace](#) ()
- virtual void [updateSolutionSplineSpace](#) (int solutionspace\_idx)
- void [getIsogeometricBlocks](#) (std::vector< shared\_ptr< [IsogeometricSfBlock](#) > > &sfblock)

### 29.252.1 Detailed Description

Definition at line 63 of file [IsogeometricSfModel.h](#).

### 29.252.2 Constructor & Destructor Documentation

29.252.2.1 [Go::IsogeometricSfModel::IsogeometricSfModel](#) ( shared\_ptr< [SurfaceModel](#) > sfmodel, std::vector< int > solution\_space\_dimension )

29.252.2.2 virtual [Go::IsogeometricSfModel::~~IsogeometricSfModel](#) ( ) [virtual]

### 29.252.3 Member Function Documentation

29.252.3.1 [bool Go::IsogeometricSfModel::addBoundaryCond](#) ( int boundary, std::vector< [Point](#) > pos, [BdConditionType](#) type, [BdCondFuncor](#) \* fbd, int solutionspace\_idx, [double](#) \* constant\_value = 0 )

29.252.3.2 void [Go::IsogeometricSfModel::addDirichletPointBdCond](#) ( int boundary, [Point](#) & pos, [Point](#) & condition\_value, int solutionspace\_idx )

29.252.3.3 [CurveLoop](#) [Go::IsogeometricSfModel::getBoundary](#) ( int idx ) const

29.252.3.4 void [Go::IsogeometricSfModel::getIsogeometricBlocks](#) ( std::vector< shared\_ptr< [IsogeometricSfBlock](#) > > & sfblock )

29.252.3.5 virtual int [Go::IsogeometricSfModel::getNmbOfBoundaries](#) ( ) const [virtual]

Implements [Go::IsogeometricModel](#).

29.252.3.6 [CurveLoop](#) [Go::IsogeometricSfModel::getOuterBoundary](#) ( ) const

29.252.3.7 virtual void [Go::IsogeometricSfModel::setMinimumDegree](#) ( int degree, int solutionspace\_idx ) [virtual]

Implements [Go::IsogeometricModel](#).

29.252.3.8 virtual void Go::IsogeometricSfModel::updateSolutionSplineSpace ( ) [virtual]

Implements [Go::IsogeometricModel](#).

29.252.3.9 virtual void Go::IsogeometricSfModel::updateSolutionSplineSpace ( int *solutionspace\_idx* ) [virtual]

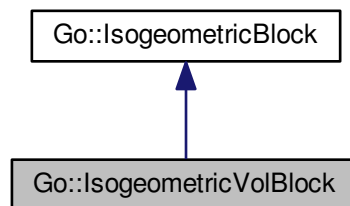
The documentation for this class was generated from the following file:

- [isogeometric\\_model/include/GoTools/isogeometric\\_model/IsogeometricSfModel.h](#)

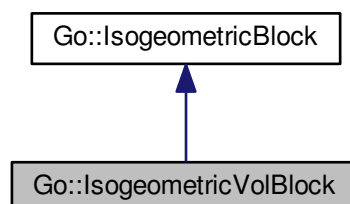
## 29.253 Go::IsogeometricVolBlock Class Reference

```
#include <IsogeometricVolBlock.h>
```

Inheritance diagram for Go::IsogeometricVolBlock:



Collaboration diagram for Go::IsogeometricVolBlock:



## Public Member Functions

- [IsogeometricVolBlock](#) ([IsogeometricModel](#) \*model, [shared\\_ptr](#)< [SplineVolume](#) > geom\_vol, [std::vector](#)< int > solution\_space\_dimension, int index)
- virtual [~IsogeometricVolBlock](#) ()
- virtual [IsogeometricVolBlock](#) \* [asIsogeometricVolBlock](#) ()
- void [addNeighbour](#) ([shared\\_ptr](#)< [IsogeometricVolBlock](#) > neighbour, int face\_nmb\_this, int face\_nmb\_other, int orientation, [bool](#) same\_dir\_order)
- virtual int [nmbOfNeighbours](#) () [const](#)
- [IsogeometricVolBlock](#) \* [getNeighbour](#) (int bd\_nmb) [const](#)
- virtual [bool](#) [isNeighbour](#) ([IsogeometricBlock](#) \*other) [const](#)
- virtual int [nmbCoefs](#) () [const](#)
- virtual [BsplineBasis](#) [basis](#) (int paddir) [const](#)
- [shared\\_ptr](#)< [SplineSurface](#) > [getGeomBoundarySurface](#) (int face\_number) [const](#)
- [std::vector](#)< [double](#) > [getParamOnBdSurf](#) (int face\_number, [const Point](#) &position) [const](#)
- [bool](#) [geomIsDegenerate](#) ([std::vector](#)< [std::pair](#)< int, int > > &degen\_bd, [double](#) epsge)
- void [getDegenEnumeration](#) ([std::vector](#)< [std::pair](#)< int, int > > &degen\_bd, [std::vector](#)< [std::vector](#)< int > > &enumeration, [double](#) epsge)
- [bool](#) [geomIsPeriodic](#) (int per[], [double](#) epsge)
- [bool](#) [getPeriodicEnumeration](#) (int paddir, [std::vector](#)< [std::pair](#)< int, int > > &enumeration)
- virtual void [refineGeometry](#) ([std::vector](#)< [double](#) > newknots, int paddir)
- virtual void [refineGeometry](#) ([const BsplineBasis](#) &other\_basis, int paddir)
- virtual void [increaseGeometryDegree](#) (int new\_degree, int paddir)
- void [updateGeometry](#) ([shared\\_ptr](#)< [SplineSurface](#) > new\_boundary, int face\_number)
- virtual void [erasePreEvaluatedBasisFunctions](#) ()
- virtual int [getNmbOfBoundaryConditions](#) () [const](#)
- [shared\\_ptr](#)< [VolBoundaryCondition](#) > [getBoundaryCondition](#) (int index) [const](#)
- void [getFaceBoundaryConditions](#) (int face\_number, [std::vector](#)< [shared\\_ptr](#)< [VolBoundaryCondition](#) > > &bd\_cond) [const](#)
- virtual int [getNmbOfPointBdConditions](#) () [const](#)
- [shared\\_ptr](#)< [VolPointBdCond](#) > [getPointBdCondition](#) (int index) [const](#)
- void [getFacePointBdConditions](#) (int face\_number, [std::vector](#)< [shared\\_ptr](#)< [VolPointBdCond](#) > > &bd\_cond) [const](#)
- [shared\\_ptr](#)< [VolSolution](#) > [getSolutionSpace](#) (int solution\_index)
- [shared\\_ptr](#)< [SplineVolume](#) > [volume](#) () [const](#)
- virtual void [setMinimumDegree](#) (int degree, int solutionspace\_idx)
- virtual [bool](#) [updateSolutionSplineSpace](#) (int solutionspace\_idx)
- int [nmbSolutionSpaces](#) () [const](#)
- int [getFaceOrientation](#) ([shared\\_ptr](#)< [ParamSurface](#) > srf, [double](#) tol)
- void [getNeighbourInfo](#) ([IsogeometricVolBlock](#) \*other, [std::vector](#)< int > &faces, [std::vector](#)< int > &faces←\_other, [std::vector](#)< int > &orientation, [std::vector](#)< [bool](#) > &same\_dir\_order)
- [bool](#) [sameDirOrder](#) (int face\_nmb) [const](#)

### 29.253.1 Detailed Description

Definition at line 61 of file [IsogeometricVolBlock.h](#).

## 29.253.2 Constructor & Destructor Documentation

29.253.2.1 `Go::IsogeometricVolBlock::IsogeometricVolBlock ( IsogeometricModel * model, shared_ptr< SplineVolume > geom_vol, std::vector< int > solution_space_dimension, int index )`

29.253.2.2 `virtual Go::IsogeometricVolBlock::~~IsogeometricVolBlock ( )` [virtual]

## 29.253.3 Member Function Documentation

29.253.3.1 `void Go::IsogeometricVolBlock::addNeighbour ( shared_ptr< IsogeometricVolBlock > neighbour, int face_nmb_this, int face_nmb_other, int orientation, bool same_dir_order )`

29.253.3.2 `virtual IsogeometricVolBlock* Go::IsogeometricVolBlock::asIsogeometricVolBlock ( )` [virtual]

Reimplemented from [Go::IsogeometricBlock](#).

29.253.3.3 `virtual BsplineBasis Go::IsogeometricVolBlock::basis ( int pardir ) const` [virtual]

Implements [Go::IsogeometricBlock](#).

29.253.3.4 `virtual void Go::IsogeometricVolBlock::erasePreEvaluatedBasisFunctions ( )` [virtual]

Implements [Go::IsogeometricBlock](#).

29.253.3.5 `bool Go::IsogeometricVolBlock::geomIsDegenerate ( std::vector< std::pair< int, int > > & degen_bd, double epsge )`

29.253.3.6 `bool Go::IsogeometricVolBlock::geomIsPeriodic ( int per[], double epsge )`

29.253.3.7 `shared_ptr< VolBoundaryCondition > Go::IsogeometricVolBlock::getBoundaryCondition ( int index ) const`

29.253.3.8 `void Go::IsogeometricVolBlock::getDegenEnumeration ( std::vector< std::pair< int, int > > & degen_bd, std::vector< std::vector< int > > & enumeration, double epsge )`

29.253.3.9 `void Go::IsogeometricVolBlock::getFaceBoundaryConditions ( int face_number, std::vector< shared_ptr< VolBoundaryCondition > > & bd_cond ) const`

29.253.3.10 `int Go::IsogeometricVolBlock::getFaceOrientation ( shared_ptr< ParamSurface > srf, double tol )`

29.253.3.11 `void Go::IsogeometricVolBlock::getFacePointBdConditions ( int face_number, std::vector< shared_ptr< VolPointBdCond > > & bd_cond ) const`

29.253.3.12 `shared_ptr< SplineSurface > Go::IsogeometricVolBlock::getGeomBoundarySurface ( int face_number ) const`

29.253.3.13 `IsogeometricVolBlock* Go::IsogeometricVolBlock::getNeighbour ( int bd_nmb ) const` [virtual]

Implements [Go::IsogeometricBlock](#).

29.253.3.14 void Go::IsogeometricVolBlock::getNeighbourInfo ( IsogeometricVolBlock \* other, std::vector< int > & faces, std::vector< int > & faces\_other, std::vector< int > & orientation, std::vector< bool > & same\_dir\_order )

29.253.3.15 virtual int Go::IsogeometricVolBlock::getNmbOfBoundaryConditions ( ) const [virtual]

Implements [Go::IsogeometricBlock](#).

29.253.3.16 virtual int Go::IsogeometricVolBlock::getNmbOfPointBdConditions ( ) const [virtual]

Implements [Go::IsogeometricBlock](#).

29.253.3.17 std::vector< double > Go::IsogeometricVolBlock::getParamOnBdSurf ( int face\_number, const Point & position ) const

29.253.3.18 bool Go::IsogeometricVolBlock::getPeriodicEnumeration ( int pardir, std::vector< std::pair< int, int > > & enumeration )

29.253.3.19 shared\_ptr< VolPointBdCond > Go::IsogeometricVolBlock::getPointBdCondition ( int index ) const

29.253.3.20 shared\_ptr< VolSolution > Go::IsogeometricVolBlock::getSolutionSpace ( int solution\_index )

29.253.3.21 virtual void Go::IsogeometricVolBlock::increaseGeometryDegree ( int new\_degree, int pardir ) [virtual]

Implements [Go::IsogeometricBlock](#).

29.253.3.22 virtual bool Go::IsogeometricVolBlock::isNeighbour ( IsogeometricBlock \* other ) const [virtual]

Implements [Go::IsogeometricBlock](#).

29.253.3.23 virtual int Go::IsogeometricVolBlock::nmbCoefs ( ) const [virtual]

Implements [Go::IsogeometricBlock](#).

29.253.3.24 virtual int Go::IsogeometricVolBlock::nmbOfNeighbours ( ) const [virtual]

Implements [Go::IsogeometricBlock](#).

29.253.3.25 int Go::IsogeometricVolBlock::nmbSolutionSpaces ( ) const

29.253.3.26 virtual void Go::IsogeometricVolBlock::refineGeometry ( std::vector< double > newknots, int pardir ) [virtual]

Implements [Go::IsogeometricBlock](#).

29.253.3.27 `virtual void Go::IsogeometricVolBlock::refineGeometry ( const BsplineBasis & other_basis, int pardir )`  
`[virtual]`

Implements [Go::IsogeometricBlock](#).

29.253.3.28 `bool Go::IsogeometricVolBlock::sameDirOrder ( int face_nmb ) const`

29.253.3.29 `virtual void Go::IsogeometricVolBlock::setMinimumDegree ( int degree, int solutionspace_idx )` `[virtual]`

29.253.3.30 `void Go::IsogeometricVolBlock::updateGeometry ( shared_ptr< SplineSurface > new_boundary, int face_number )`

29.253.3.31 `virtual bool Go::IsogeometricVolBlock::updateSolutionSplineSpace ( int solutionspace_idx )` `[virtual]`

29.253.3.32 `shared_ptr<SplineVolume> Go::IsogeometricVolBlock::volume ( ) const`

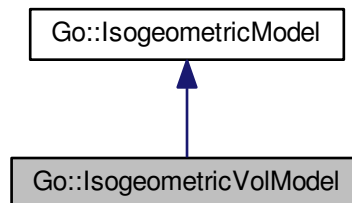
The documentation for this class was generated from the following file:

- [isogeometric\\_model/include/GoTools/isogeometric\\_model/IsogeometricVolBlock.h](#)

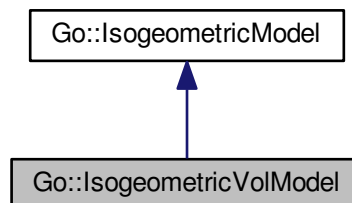
## 29.254 Go::IsogeometricVolModel Class Reference

```
#include <IsogeometricVolModel.h>
```

Inheritance diagram for Go::IsogeometricVolModel:



Collaboration diagram for Go::IsogeometricVolModel:



## Public Member Functions

- [IsogeometricVolModel](#) (shared\_ptr< [VolumeModel](#) > volmodel, std::vector< int > solution\_space\_↔ dimension)
- virtual [~IsogeometricVolModel](#) ()
- [bool addBoundaryCond](#) (std::vector< std::pair< [ParamSurface](#) \*, [Point](#) > > polygon, [BdConditionType](#) type, [BdCondFunc](#) \*fbd, int solutionspace\_idx, [double](#) \*constant\_value=0)
- [void addDirichletPointBdCond](#) ([ParamSurface](#) \*bd\_surf, [Point](#) &pos, [Point](#) &condition\_value, int solutionspace\_idx)
- virtual int [getNmbOfBoundaries](#) () const
- std::vector< shared\_ptr< [ParamSurface](#) > > [getOuterBoundary](#) () const
- std::vector< shared\_ptr< [ParamSurface](#) > > [getBoundary](#) (int idx) const
- virtual void [setMinimumDegree](#) (int degree, int solutionspace\_idx)
- virtual void [updateSolutionSplineSpace](#) ()
- virtual void [updateSolutionSplineSpace](#) (int solutionspace\_idx)
- [void getIsogeometricBlocks](#) (std::vector< shared\_ptr< [IsogeometricVolBlock](#) > > &volblock)

### 29.254.1 Detailed Description

Definition at line 63 of file [IsogeometricVolModel.h](#).

### 29.254.2 Constructor & Destructor Documentation

29.254.2.1 [Go::IsogeometricVolModel::IsogeometricVolModel](#) ( shared\_ptr< [VolumeModel](#) > volmodel, std::vector< int > solution\_space\_dimension )

29.254.2.2 virtual [Go::IsogeometricVolModel::~~IsogeometricVolModel](#) ( ) [virtual]

### 29.254.3 Member Function Documentation

29.254.3.1 [bool Go::IsogeometricVolModel::addBoundaryCond](#) ( std::vector< std::pair< [ParamSurface](#) \*, [Point](#) > > polygon, [BdConditionType](#) type, [BdCondFunc](#) \* fbd, int solutionspace\_idx, [double](#) \* constant\_value = 0 )

29.254.3.2 [void Go::IsogeometricVolModel::addDirichletPointBdCond](#) ( [ParamSurface](#) \* bd\_surf, [Point](#) & pos, [Point](#) & condition\_value, int solutionspace\_idx )

29.254.3.3 [std::vector<shared\\_ptr<ParamSurface>>](#) [Go::IsogeometricVolModel::getBoundary](#) ( int idx ) const

29.254.3.4 [void Go::IsogeometricVolModel::getIsogeometricBlocks](#) ( std::vector< shared\_ptr< [IsogeometricVolBlock](#) > > & volblock )

29.254.3.5 virtual int [Go::IsogeometricVolModel::getNmbOfBoundaries](#) ( ) const [virtual]

Implements [Go::IsogeometricModel](#).



29.254.3.6 `std::vector<shared_ptr<ParamSurface>>` Go::IsogeometricVolModel::getOuterBoundary ( ) const

29.254.3.7 `virtual void` Go::IsogeometricVolModel::setMinimumDegree ( *int degree*, *int solutionspace\_idx* ) [virtual]

Implements [Go::IsogeometricModel](#).

29.254.3.8 `virtual void` Go::IsogeometricVolModel::updateSolutionSplineSpace ( ) [virtual]

Implements [Go::IsogeometricModel](#).

29.254.3.9 `virtual void` Go::IsogeometricVolModel::updateSolutionSplineSpace ( *int solutionspace\_idx* ) [virtual]

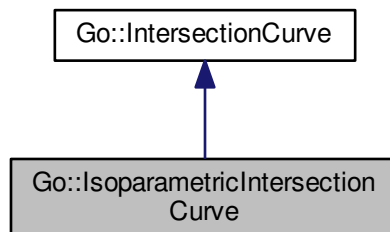
The documentation for this class was generated from the following file:

- [isogeometric\\_model/include/GoTools/isogeometric\\_model/IsogeometricVolModel.h](#)

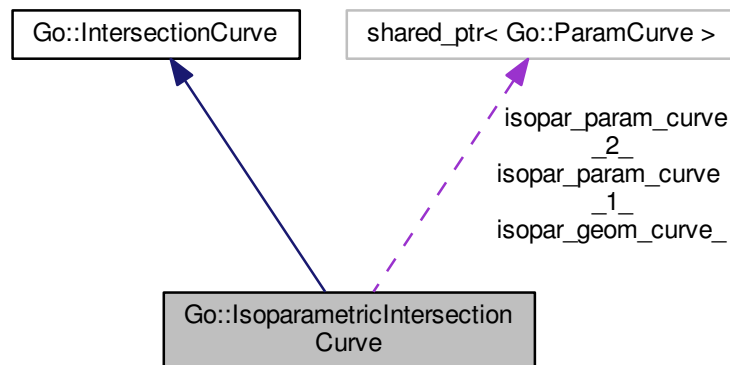
## 29.255 Go::IsoparametricIntersectionCurve Class Reference

```
#include <IntersectionCurve.h>
```

Inheritance diagram for Go::IsoparametricIntersectionCurve:



Collaboration diagram for Go::IsoparametricIntersectionCurve:



## Public Member Functions

- virtual `~IsoparametricIntersectionCurve ()`
- virtual `shared_ptr< ParamCurve > getCurve () const`
- virtual `shared_ptr< ParamCurve > getParamCurve (int obj_nmb) const`
- virtual `bool isIsocurve () const`
- virtual `bool isDegenerated () const`
- virtual `void refine (const double &pos_tol, const double &angle_tol)`
- virtual `void getParamSpan (double &start, double &end) const`
- virtual `void evaluateAt (double pval, Point &pos, Point &tan)`

## Protected Member Functions

- `template<class iterator >`  
`IsoparametricIntersectionCurve (const iterator begin, const iterator end, int isopar)`
- `void precalculate_iso_curves (int isopar)`

## Static Protected Member Functions

- `template<class iterator >`  
`static std::vector< int > resolve_isoparametric_directions (const iterator begin, const iterator end)`

## Protected Attributes

- `shared_ptr< ParamCurve > isopar_geom_curve_`
- `shared_ptr< ParamCurve > isopar_param_curve_1_`
- `shared_ptr< ParamCurve > isopar_param_curve_2_`
- `std::vector< Go::Point > temp_`

## Friends

- `template<class iterator >`  
`shared_ptr< IntersectionCurve > constructIntersectionCurve (const iterator begin, const iterator end)`

### 29.255.1 Detailed Description

[IntersectionCurve](#) defined by an isoparametric curve that can be picked from the underlying object

Definition at line 326 of file `IntersectionCurve.h`.

### 29.255.2 Constructor & Destructor Documentation

29.255.2.1 `virtual Go::IsoparametricIntersectionCurve::~IsoparametricIntersectionCurve ( )` `[inline]`, `[virtual]`

Definition at line 330 of file `IntersectionCurve.h`.

29.255.2.2 `template<class iterator > Go::IsoparametricIntersectionCurve::IsoparametricIntersectionCurve ( const iterator begin, const iterator end, int isopar )` `[inline]`, `[protected]`

Definition at line 368 of file `IntersectionCurve.h`.

### 29.255.3 Member Function Documentation

29.255.3.1 `virtual void Go::IsoparametricIntersectionCurve::evaluateAt ( double pval, Point & pos, Point & tan )`  
`[inline]`, `[virtual]`

Evaluate the [IntersectionCurve](#) at parameter value *pval*. Its position at this parameter value will be returned in *pos* and its tangent direction will be returned in *tan*.

#### Parameters

<i>pval</i>	the parameter value for which to evaluate the <a href="#">IntersectionCurve</a> . It should be within the parametric span (which can be obtained from <code>GetParamSpan()</code> ).
-------------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

#### Return values

<i>pos</i>	the calculated position of the point on the curve at parameter <i>pval</i> .
------------	------------------------------------------------------------------------------

#### Parameters

<i>tan</i>	the calculated tangent of the curve at parameter <i>pval</i> .
------------	----------------------------------------------------------------

Implements [Go::IntersectionCurve](#).

Definition at line 353 of file `IntersectionCurve.h`.

29.255.3.2 `virtual shared_ptr<ParamCurve> Go::IsoparametricIntersectionCurve::getCurve ( ) const [virtual]`

Get out a (shared) pointer to a parametric curve that approximates this [IntersectionCurve](#).

Implements [Go::IntersectionCurve](#).

29.255.3.3 `virtual shared_ptr<ParamCurve> Go::IsoparametricIntersectionCurve::getParamCurve ( int obj_nmb ) const [virtual]`

Get out a (shared) pointer to the curve in the parametric plane of the first or second object. Should only be called when the concerned object is 2-parametric.

#### Parameters

<i>obj_nmb</i>	should be either 1 or 2, depending on whether you want to get the parametric curve in object 1 or object 2.
----------------	-------------------------------------------------------------------------------------------------------------

#### Returns

a shared pointer to a curve that approximates the intersection curve in the parametric domain of the specified object.

Implements [Go::IntersectionCurve](#).

29.255.3.4 `virtual void Go::IsoparametricIntersectionCurve::getParamSpan ( double & start, double & end ) const [inline],[virtual]`

Get the start and end value for the parametric span of the [IntersectionCurve](#).

#### Return values

<i>start</i>	upon function return this variable will contain the start value of the parametric span.
<i>end</i>	upon function return this variable will contain the end value of the parametric span.

Implements [Go::IntersectionCurve](#).

Definition at line 347 of file `IntersectionCurve.h`.

29.255.3.5 `virtual bool Go::IsoparametricIntersectionCurve::isDegenerated ( ) const [inline],[virtual]`

Determine if the [IntersectionCurve](#) is in fact a degenerated curve (a single point in space).

#### Returns

'true' if the [IntersectionCurve](#) is degenerated. 'false' otherwise.

Implements [Go::IntersectionCurve](#).

Definition at line 339 of file `IntersectionCurve.h`.

29.255.3.6 `virtual bool Go::IsoparametricIntersectionCurve::isIsocurve ( ) const [inline], [virtual]`

Determine if the [IntersectionCurve](#) is in fact representing an isoparametric intersection.

#### Returns

'true' if the [IntersectionCurve](#) is part of an isocurve for one of the objects, 'false' otherwise.

Implements [Go::IntersectionCurve](#).

Definition at line 336 of file `IntersectionCurve.h`.

29.255.3.7 `void Go::IsoparametricIntersectionCurve::precalculate_iso_curves ( int isopar ) [protected]`

29.255.3.8 `virtual void Go::IsoparametricIntersectionCurve::refine ( const double & pos_tol, const double & angle_tol ) [inline], [virtual]`

Refine the curve (which means to increase the number of [IntersectionPoint](#) s defining it), so that the curve matches a specific positional and angle tolerance.

#### Parameters

<i>pos_tol</i>	the positional tolerance that the curve should match after refinement.
<i>angle_tol</i>	the angular tolerance that the tangent of the curve should match after refinement.

Implements [Go::IntersectionCurve](#).

Definition at line 342 of file `IntersectionCurve.h`.

29.255.3.9 `template<class iterator > std::vector< int > Go::IsoparametricIntersectionCurve::resolve_isoparametric_directions ( const iterator begin, const iterator end ) [static], [protected]`

Definition at line 658 of file `IntersectionCurve.h`.

## 29.255.4 Friends And Related Function Documentation

29.255.4.1 `template<class iterator > shared_ptr<IntersectionCurve> constructIntersectionCurve ( const iterator begin, const iterator end ) [friend]`

Definition at line 601 of file `IntersectionCurve.h`.

## 29.255.5 Member Data Documentation

29.255.5.1 `shared_ptr<ParamCurve> Go::IsoparametricIntersectionCurve::isopar_geom_curve_ [protected]`

Definition at line 362 of file `IntersectionCurve.h`.

29.255.5.2 `shared_ptr<ParamCurve> Go::IsoparametricIntersectionCurve::isopar_param_curve_1_` [protected]

Definition at line 363 of file IntersectionCurve.h.

29.255.5.3 `shared_ptr<ParamCurve> Go::IsoparametricIntersectionCurve::isopar_param_curve_2_` [protected]

Definition at line 364 of file IntersectionCurve.h.

29.255.5.4 `std::vector<Go::Point> Go::IsoparametricIntersectionCurve::temp_` [mutable],[protected]

Definition at line 365 of file IntersectionCurve.h.

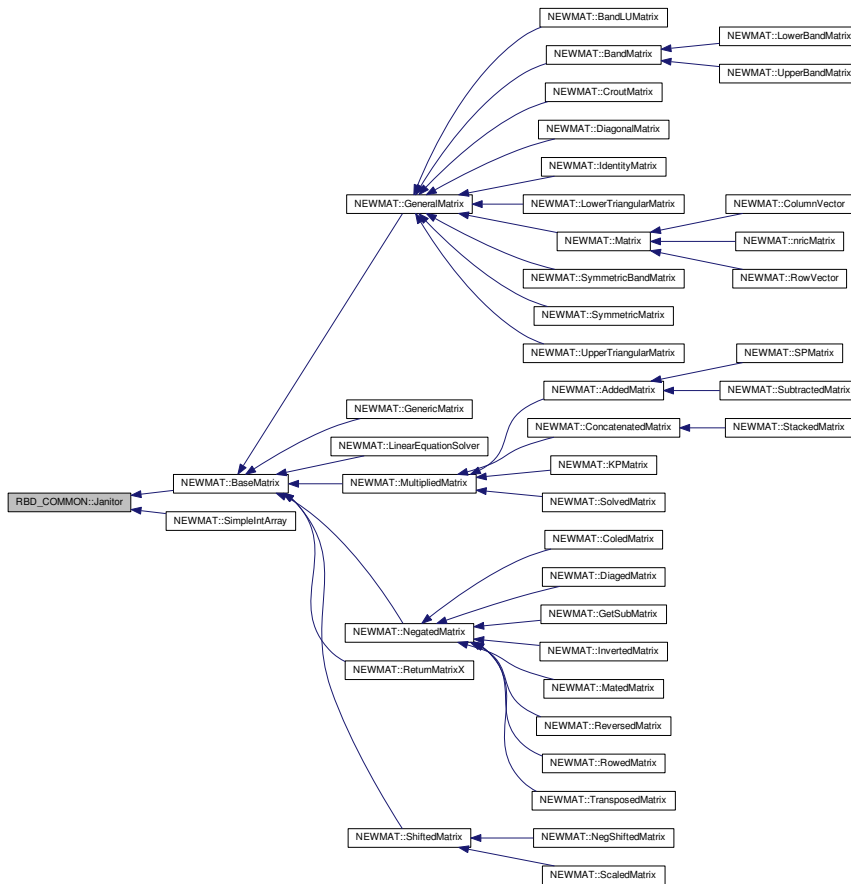
The documentation for this class was generated from the following file:

- intersections/include/GoTools/intersections/[IntersectionCurve.h](#)

## 29.256 RBD\_COMMON::Janitor Class Reference

```
#include <myexcept.h>
```

Inheritance diagram for RBD\_COMMON::Janitor:



## Public Member Functions

- virtual void [CleanUp](#) ()
- [Janitor](#) ()
- virtual [~Janitor](#) ()

### 29.256.1 Detailed Description

Definition at line 216 of file myexcept.h.

### 29.256.2 Constructor & Destructor Documentation

29.256.2.1 `RBD_COMMON::Janitor ( )` [[inline](#)]

Definition at line 220 of file myexcept.h.

29.256.2.2 `virtual RBD_COMMON::Janitor::~~Janitor ( )` [[inline](#)], [[virtual](#)]

Definition at line 221 of file myexcept.h.

### 29.256.3 Member Function Documentation

29.256.3.1 `virtual void RBD_COMMON::Janitor::CleanUp ( )` [[inline](#)], [[virtual](#)]

Reimplemented in [NEWMAT::SimpleIntArray](#), [NEWMAT::LinearEquationSolver](#), [NEWMAT::GenericMatrix](#), [NEWMAT::BandLUMatrix](#), [NEWMAT::CroutMatrix](#), [NEWMAT::ColumnVector](#), [NEWMAT::RowVector](#), [NEWMAT::nrncMatrix](#), [NEWMAT::GeneralMatrix](#), and [NEWMAT::BaseMatrix](#).

Definition at line 219 of file myexcept.h.

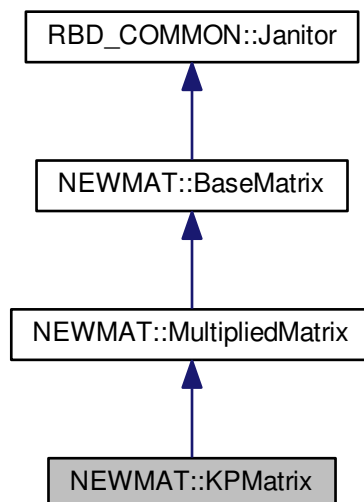
The documentation for this class was generated from the following file:

- [newmat/include/myexcept.h](#)

## 29.257 NEWMAT::KPMatrix Class Reference

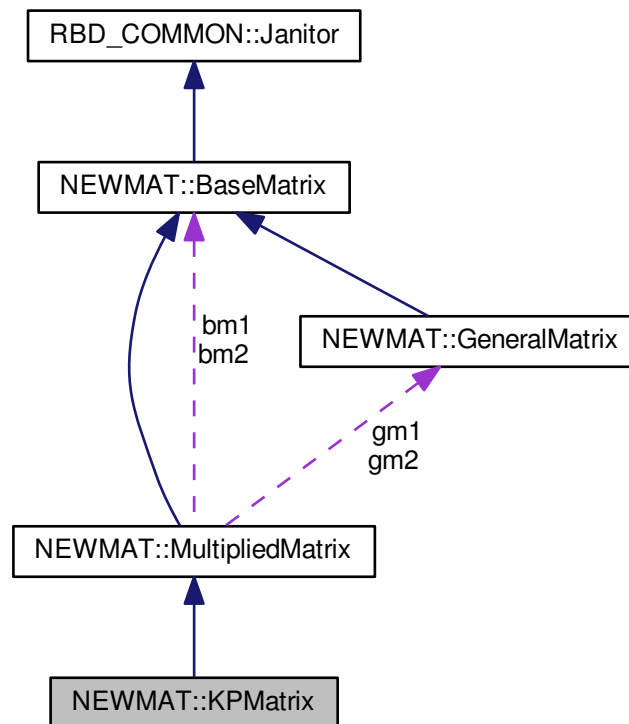
```
#include <newmat.h>
```

Inheritance diagram for NEWMAT::KPMatrix:





Collaboration diagram for NEWMAT::KPMatrix:



## Public Member Functions

- [~KPMatrix \(\)](#)
- [MatrixBandWidth BandWidth \(\) const](#)
- [GeneralMatrix \\* Evaluate \(MatrixType mt=MatrixTypeUnSp\)](#)

## Protected Member Functions

- [KPMatrix \(const BaseMatrix \\*bm1x, const BaseMatrix \\*bm2x\)](#)

## Friends

- class [BaseMatrix](#)
- class [GeneralMatrix](#)
- class [GenericMatrix](#)
- [KPMatrix KP \(const BaseMatrix &, const BaseMatrix &\)](#)

## Additional Inherited Members

### 29.257.1 Detailed Description

Definition at line 1233 of file newmat.h.

### 29.257.2 Constructor & Destructor Documentation

29.257.2.1 **NEWMAT::KPMatrix::KPMatrix** ( **const BaseMatrix** \* *bm1x*, **const BaseMatrix** \* *bm2x* ) [*inline*],  
[*protected*]

Definition at line 1236 of file newmat.h.

29.257.2.2 **NEWMAT::KPMatrix::~~KPMatrix** ( ) [*inline*]

Definition at line 1243 of file newmat.h.

### 29.257.3 Member Function Documentation

29.257.3.1 **MatrixBandWidth** **KPMatrix::BandWidth** ( ) **const** [*virtual*]

Reimplemented from [NEWMAT::MultipliedMatrix](#).

Definition at line 424 of file newmat4.cpp.

29.257.3.2 **GeneralMatrix** \* **KPMatrix::Evaluate** ( **MatrixType** *mt* = **MatrixTypeUnSp** ) [*virtual*]

Reimplemented from [NEWMAT::MultipliedMatrix](#).

Definition at line 130 of file newmat7.cpp.

### 29.257.4 Friends And Related Function Documentation

29.257.4.1 **friend class BaseMatrix** [*friend*]

Definition at line 1239 of file newmat.h.

29.257.4.2 **friend class GeneralMatrix** [*friend*]

Definition at line 1240 of file newmat.h.

29.257.4.3 friend class `GenericMatrix` [`friend`]

Definition at line 1241 of file `newmat.h`.

29.257.4.4 `KPMatrix KP ( const BaseMatrix &, const BaseMatrix & )` [`friend`]

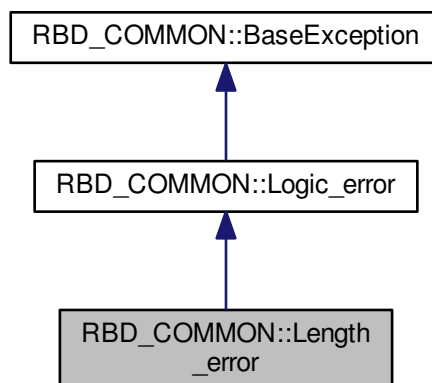
The documentation for this class was generated from the following files:

- [newmat/include/newmat.h](#)
- [newmat/src/newmat4.cpp](#)
- [newmat/src/newmat7.cpp](#)

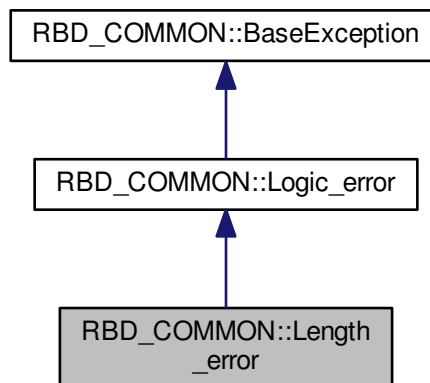
## 29.258 RBD\_COMMON::Length\_error Class Reference

```
#include <myexcept.h>
```

Inheritance diagram for `RBD_COMMON::Length_error`:



Collaboration diagram for RBD\_COMMON::Length\_error:



- static unsigned long [Select](#)
- [Length\\_error](#) (const char \*a\_what=0)

### Additional Inherited Members

#### 29.258.1 Detailed Description

Definition at line 385 of file myexcept.h.

#### 29.258.2 Constructor & Destructor Documentation

##### 29.258.2.1 Length\_error::Length\_error ( const char \* a\_what = 0 )

Definition at line 426 of file myexcept.cpp.

#### 29.258.3 Member Data Documentation

##### 29.258.3.1 unsigned long Length\_error::Select [static]

Definition at line 388 of file myexcept.h.

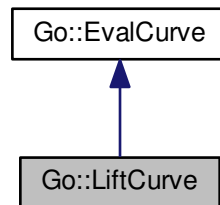
The documentation for this class was generated from the following files:

- [newmat/include/myexcept.h](#)
- [newmat/src/myexcept.cpp](#)

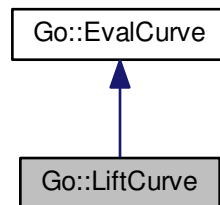
## 29.259 Go::LiftCurve Class Reference

```
#include <LiftCurve.h>
```

Inheritance diagram for Go::LiftCurve:



Collaboration diagram for Go::LiftCurve:



### Public Member Functions

- `LiftCurve` (`shared_ptr< Go::ParamCurve > &parameter_crv, shared_ptr< Go::ParamSurface > &surf, double epsgeo`)
- `virtual ~LiftCurve ()`  
*virtual destructor enables safe inheritance*
- `virtual Point eval (double t) const`
- `virtual void eval (double t, int n, Point der[]) const`
- `virtual double start () const`
- `virtual double end () const`
- `virtual int dim () const`  
*Dimension of the lifted curve (i.e. 3).*
- `virtual bool approximationOK (double par, Point approxpos, double tol1, double tol2) const`

### 29.259.1 Detailed Description

This class represents a "lift curve", generated from taking a surface and a 2D curve and then evaluate the surface at the parameter values obtained by evaluating the 2D curve.

Definition at line 59 of file LiftCurve.h.

### 29.259.2 Constructor & Destructor Documentation

29.259.2.1 `Go::LiftCurve::LiftCurve ( shared_ptr< Go::ParamCurve > & parameter_crv, shared_ptr< Go::ParamSurface > & surf, double epsgeo )`

Constructor, taking a 2D parameter curve and a surface.

#### Parameters

<i>parameter_crv</i>	the 2D parameter curve that will be 'lifted'
<i>surf</i>	the surface on which the resulting, 'lifted' curve will lie
<i>epsgeo</i>	geometrical tolerance used when running the 'approximationOK' function.

29.259.2.2 `virtual Go::LiftCurve::~~LiftCurve ( ) [virtual]`

virtual destructor enables safe inheritance

### 29.259.3 Member Function Documentation

29.259.3.1 `virtual bool Go::LiftCurve::approximationOK ( double par, Point approxpos, double tol1, double tol2 ) const [virtual]`

Inherited from [EvalCurve::approximationOK\(\)](#).

#### Parameters

<i>par</i>	the parameter at which to check the curve
<i>approxpos</i>	the position we want to check whether or not the curve approximates for parameter 'par'.
<i>tol1</i>	unused
<i>tol2</i>	unused

#### Returns

'true' if the curve approximates the point at the parameter (within the tolerance given in the constructor, 'epsgeo'). 'false' otherwise.

Implements [Go::EvalCurve](#).

29.259.3.2 virtual int Go::LiftCurve::dim ( ) const [virtual]

Dimension of the lifted curve (i.e. 3).

Implements [Go::EvalCurve](#).

29.259.3.3 virtual double Go::LiftCurve::end ( ) const [virtual]

Get the end parameter of the curve.

Returns

the end parameter of the curve.

Implements [Go::EvalCurve](#).

29.259.3.4 virtual Point Go::LiftCurve::eval ( double t ) const [virtual]

Evaluate a point on the curve for a given parameter

Parameters

<i>t</i>	the parameter for which to evaluate the curve.
----------	------------------------------------------------

Returns

the evaluated point

Implements [Go::EvalCurve](#).

29.259.3.5 virtual void Go::LiftCurve::eval ( double t, int n, Point der[] ) const [virtual]

Evaluate a point and a certain number of derivatives on the curve for a given parameter.

Parameters

<i>t</i>	the parameter for which to evaluate the curve.
<i>n</i>	the number of derivatives (0 or more)

Return values

<i>der</i>	pointer to an array of Points where the result will be written. The position will be stored first, then the first derivative (tangent), then the second, etc.. <b>NB:</b> For most (all) derived classes of ' <a href="#">EvalCurve</a> ', the implementation actually only supports the computation of one derivative, i.e. if $n > 1$ , only one derivative will be computed anyway.
------------	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Implements [Go::EvalCurve](#).

29.259.3.6 `virtual double Go::LiftCurve::start ( ) const [virtual]`

Get the start parameter of the curve.

#### Returns

the start parameter of the curve.

Implements [Go::EvalCurve](#).

The documentation for this class was generated from the following file:

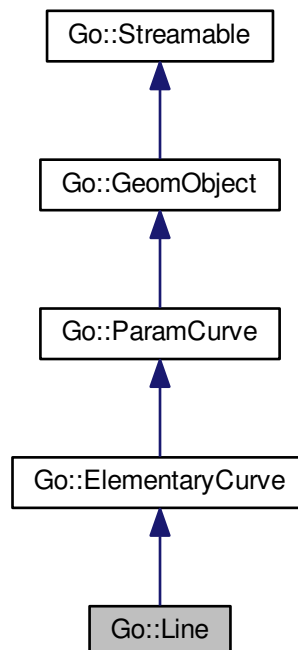
- `gotools-core/include/GoTools/creators/LiftCurve.h`

## 29.260 Go::Line Class Reference

Class that represents a line. It is a subclass of [ElementaryCurve](#) and thus has a parametrization.

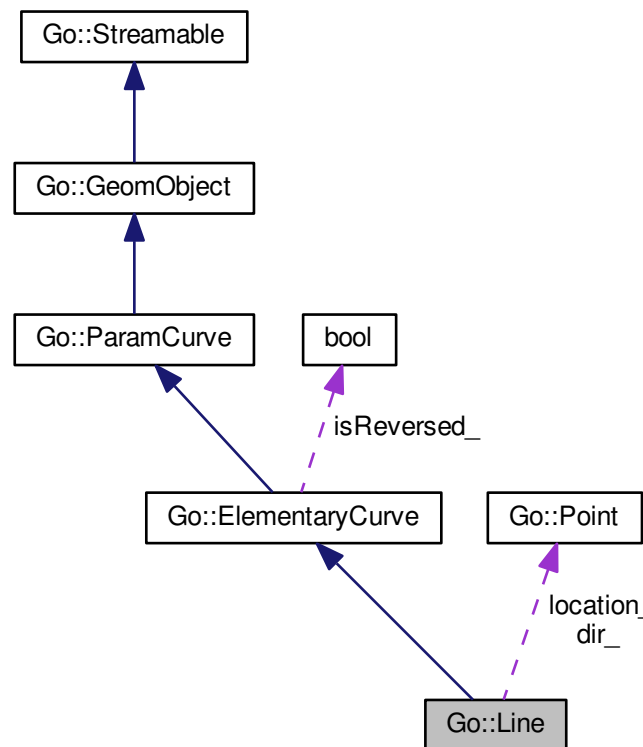
```
#include <Line.h>
```

Inheritance diagram for Go::Line:





Collaboration diagram for Go::Line:



## Public Member Functions

- [Line](#) ()
- [Line](#) ([Point](#) point, [Point](#) direction, [bool](#) isReversed=false)
- [Line](#) ([Point](#) point, [Point](#) direction, [double](#) length, [bool](#) isReversed=false)
  - Constructor. Bounded line.*
- [Line](#) ([Point](#) point1, [Point](#) point2, [double](#) par1, [double](#) par2, [bool](#) isReversed=false)
  - Constructor. Bounded line with parameterization.*
- virtual [~Line](#) ()
  - virtual destructor - ensures safe inheritance*
- virtual void [read](#) (std::istream &is)
- virtual void [write](#) (std::ostream &os) const
- virtual [BoundingBox](#) boundingBox () const
  - Return the object's bounding box.*
- virtual int [dimension](#) () const
  - Return the dimension of the space in which the object lies (usually 2 or 3)*
- virtual [ClassType](#) instanceType () const
  - Return the class type identifier of a given, derived instance of [GeomObject](#).*
- virtual [Line](#) \* [clone](#) () const
- virtual void [point](#) ([Point](#) &pt, [double](#) tpar) const
- virtual void [point](#) (std::vector< [Point](#) > &pts, [double](#) tpar, int derivs, [bool](#) from\_right=true) const

- virtual [double startparam](#) () const
- virtual [double endparam](#) () const
- virtual void [setParameterInterval](#) (double t1, double t2)
  - Limit the curve by limiting the parameter interval.*
- virtual [SplineCurve](#) \* [geometryCurve](#) ()
- virtual [SplineCurve](#) \* [createSplineCurve](#) () const
  - Fetch spline representation of curve.*
- virtual [bool isDegenerate](#) (double degenerate\_epsilon)
- virtual [Line](#) \* [subCurve](#) (double from\_par, double to\_par, double fuzzy=DEFAULT\_PARAMETER\_EPSILON) const
- virtual [DirectionCone](#) [directionCone](#) () const
- virtual void [appendCurve](#) ([ParamCurve](#) \*cv, bool reparam=true)
- virtual void [appendCurve](#) ([ParamCurve](#) \*cv, int continuity, double &dist, bool reparam=true)
- virtual void [closestPoint](#) (const [Point](#) &pt, double tmin, double tmax, double &clo\_t, [Point](#) &clo\_pt, double &clo\_dist, double const \*seed=0) const
- virtual [double length](#) (double tol)
- [Point](#) [getPoint](#) ()
  - Point on line.*
- [Point](#) [getDirection](#) ()
  - Direction vector.*
- virtual void [setParamBounds](#) (double startpar, double endpar)
- virtual void [translateCurve](#) (const [Point](#) &dir)
- [bool isBounded](#) () const
- virtual [bool isLinear](#) ([Point](#) &dir, double tol)
  - Confirm that the curve is linear.*
- virtual void [swapParameters2D](#) ()
- virtual [bool isInPlane](#) (const [Point](#) &loc, const [Point](#) &axis, double eps, [Point](#) &normal) const
  - Check if the line lies in a plane passing through a given axis.*
- virtual [bool isInPlane](#) (const [Point](#) &norm, double eps, [Point](#) &pos) const
  - Check if the line lies in a plane with a given normal.*

### Static Public Member Functions

- static [ClassType](#) [classType](#) ()

### Protected Attributes

- [Point](#) [location\\_](#)
- [Point](#) [dir\\_](#)
- [double](#) [startparam\\_](#)
- [double](#) [endparam\\_](#)

### Additional Inherited Members

#### 29.260.1 Detailed Description

Class that represents a line. It is a subclass of [ElementaryCurve](#) and thus has a parametrization.

A [Line](#) has a natural parametrization in terms of its location  $\mathbf{C}$  and direction vector  $\mathbf{V}$ :  $p(t) = \mathbf{C} + t\mathbf{V}$ .

Definition at line 56 of file [Line.h](#).

## 29.260.2 Constructor & Destructor Documentation

### 29.260.2.1 Go::Line::Line ( ) [inline]

Default constructor. Constructs an uninitialized [Line](#) which can only be assigned to or read into.

Definition at line 61 of file Line.h.

### 29.260.2.2 Go::Line::Line ( Point point, Point direction, bool isReversed = false )

Constructor. Input is point and direction. The default [Line](#) is unbounded in its parametrization. To bound it, use [setParamBounds\(\)](#).

### 29.260.2.3 Go::Line::Line ( Point point, Point direction, double length, bool isReversed = false )

Constructor. Bounded line.

### 29.260.2.4 Go::Line::Line ( Point point1, Point point2, double par1, double par2, bool isReversed = false )

Constructor. Bounded line with parameterization.

### 29.260.2.5 virtual Go::Line::~~Line ( ) [virtual]

virtual destructor - ensures safe inheritance

## 29.260.3 Member Function Documentation

### 29.260.3.1 virtual void Go::Line::appendCurve ( ParamCurve \* cv, bool reparam = true ) [virtual]

append a curve to this curve, with eventual reparametrization NB: This virtual member function currently only works for [SplineCurves](#) and [CurveOnSurfaces](#). Moreover, 'this' curve and the 'cv' curve must be of the same type.

#### Parameters

<i>cv</i>	the curve to append to 'this' curve.
<i>reparam</i>	specify whether or not there should be reparametrization

Implements [Go::ParamCurve](#).

### 29.260.3.2 virtual void Go::Line::appendCurve ( ParamCurve \* cv, int continuity, double & dist, bool reparam = true ) [virtual]

append a curve to this curve, with eventual reparametrization

## Parameters

<i>cv</i>	the curve to append to 'this' curve.
<i>continuity</i>	the required continuity at the transition. Can be $G^{-1}$ and upwards.
<i>dist</i>	a measure of the local distortion around the transition in order to achieve the specified continuity.
<i>reparam</i>	specify whether or not there should be reparametrization

Implements [Go::ParamCurve](#).

29.260.3.3 virtual **BoundingBox** Go::Line::boundingBox ( ) const [virtual]

Return the object's bounding box.

Implements [Go::GeomObject](#).

29.260.3.4 static **ClassType** Go::Line::classType ( ) [static]

29.260.3.5 virtual **Line\*** Go::Line::clone ( ) const [virtual]

The clone-function is inherited from [GeomObject](#), but overridden here to get a covariant return type (for those compilers that allow this).

Implements [Go::ElementaryCurve](#).

29.260.3.6 virtual void Go::Line::closestPoint ( const Point & *pt*, double *tmin*, double *tmax*, double & *clo\_t*, Point & *clo\_pt*, double & *clo\_dist*, double const \* *seed* = 0 ) const [virtual]

Compute the closest point from an interval of this curve to a specified point.

## Parameters

<i>pt</i>	point we want to find the closest point to
<i>tmin</i>	start parameter of search interval
<i>tmax</i>	end parameter of search interval
<i>clo_t</i>	upon function return, 'clo_t' will contain the parameter value of the closest point found.
<i>clo_pt</i>	upon function return, 'clo_pt' will contain the position of the closest point found.
<i>clo_dist</i>	upon function return, 'clo_dist' will contain the distance between 'pt' and the closest point found.
<i>seed</i>	pointer to initial guess value, provided by the user (can be 0, for which the algorithm will determine a (hopefully) reasonable choice).

Implements [Go::ParamCurve](#).

29.260.3.7 virtual **SplineCurve\*** Go::Line::createSplineCurve ( ) const [virtual]

Fetch spline representation of curve.

Implements [Go::ElementaryCurve](#).

29.260.3.8 `virtual int Go::Line::dimension ( ) const [virtual]`

Return the dimension of the space in which the object lies (usually 2 or 3)

Implements [Go::GeomObject](#).

29.260.3.9 `virtual DirectionCone Go::Line::directionCone ( ) const [virtual]`

Creates a [DirectionCone](#) which covers all tangent directions of this curve.

Returns

the smallest [DirectionCone](#) containing all tangent directions of this curve.

Implements [Go::ParamCurve](#).

29.260.3.10 `virtual double Go::Line::endparam ( ) const [virtual]`

Query the end parameter of the curve

Returns

the curve's end parameter

Implements [Go::ParamCurve](#).

29.260.3.11 `virtual SplineCurve* Go::Line::geometryCurve ( ) [virtual]`

If the definition of this [ParamCurve](#) contains a [SplineCurve](#) describing its spatial shape, then this function will return a pointer to this [SplineCurve](#). Otherwise it will return a null pointer. The returned curve is NEWed, so the user is responsible for deleting it. This function may have side-effects.

Returns

a pointer to a [SplineCurve](#) representation of the [ParamCurve](#), if it exists. Null pointer otherwise.

Implements [Go::ParamCurve](#).

29.260.3.12 `Point Go::Line::getDirection ( ) [inline]`

Direction vector.

Definition at line 151 of file Line.h.

29.260.3.13 **Point** `Go::Line::getPoint ( )` `[inline]`

[Point](#) on line.

Definition at line 147 of file Line.h.

29.260.3.14 **virtual ClassType** `Go::Line::instanceType ( ) const` `[virtual]`

Return the class type identifier of a given, derived instance of [GeomObject](#).

Implements [Go::GeomObject](#).

29.260.3.15 **bool** `Go::Line::isBounded ( ) const`

Query if parametrization is bounded. Both upper and lower parameter bounds must be finite for this to be true.

Returns

*true* if bounded, *false* otherwise

29.260.3.16 **virtual bool** `Go::Line::isDegenerate ( double degenerate_epsilon )` `[virtual]`

Query whether the curve is degenerate (collapsed into a single point).

Parameters

<i>degenerate_epsilon</i>	the tolerance used in determine whether the curve is degenerate. A curve is considered degenerate if its total length is shorter than this value.
---------------------------	---------------------------------------------------------------------------------------------------------------------------------------------------

Returns

*true* if the curve is degenerate, *false* otherwise.

Implements [Go::ParamCurve](#).

29.260.3.17 **virtual bool** `Go::Line::isInPlane ( const Point & loc, const Point & axis, double eps, Point & normal ) const` `[virtual]`

Check if the line lies in a plane passing through a given axis.

Reimplemented from [Go::ParamCurve](#).

29.260.3.18 **virtual bool** `Go::Line::isInPlane ( const Point & norm, double eps, Point & pos ) const` `[virtual]`

Check if the line lies in a plane with a given normal.

Reimplemented from [Go::ParamCurve](#).

29.260.3.19 `virtual bool Go::Line::isLinear ( Point & dir, double tol ) [virtual]`

Confirm that the curve is linear.

Reimplemented from [Go::ParamCurve](#).

29.260.3.20 `virtual double Go::Line::length ( double tol ) [virtual]`

Compute the total length of this curve up to some tolerance

#### Parameters

<i>tol</i>	the relative tolerance when approximating the length, i.e. stop iteration when error becomes smaller than $tol/(curve\ length)$
------------	---------------------------------------------------------------------------------------------------------------------------------

#### Returns

the length calculated

Implements [Go::ParamCurve](#).

29.260.3.21 `virtual void Go::Line::point ( Point & pt, double tpar ) const [virtual]`

Evaluate the curve's position at a given parameter

#### Parameters

<i>pt</i>	the evaluated position will be written to this <a href="#">Point</a>
<i>tpar</i>	the parameter for which we wish to evaluate the curve

Implements [Go::ParamCurve](#).

29.260.3.22 `virtual void Go::Line::point ( std::vector< Point > & pts, double tpar, int derivs, bool from_right = true ) const [virtual]`

Evaluate the curve's position and a certain number of derivatives at a given parameter.

#### Parameters

<i>pts</i>	the evaluated position and derivatives (tangent, curvature vector, etc.) will be written to this vector. The first entry will be the position, the second entry will be the first derivative, etc. The size of this vector must be set to 'derivs'+ 1 prior to calling this function.
<i>tpar</i>	the parameter for which we want to evaluate the curve
<i>derivs</i>	the number of derivatives we want to have calculated
<i>from_right</i>	specify whether we should calculate derivatives 'from the right' or 'from the left' (default is from the right). This matters only when the curve presents discontinuities in its derivatives.

Implements [Go::ParamCurve](#).

29.260.3.23 `virtual void Go::Line::read ( std::istream & is ) [virtual]`

Read object from stream

#### Parameters

<i>is</i>	stream from which object is read
-----------	----------------------------------

Implements [Go::Streamable](#).

29.260.3.24 `virtual void Go::Line::setParamBounds ( double startpar, double endpar ) [virtual]`

Set bounds for the parametrization of the [Line](#).

#### Parameters

<i>startpar</i>	start parameter
<i>endpar</i>	end parameter

Implements [Go::ElementaryCurve](#).

29.260.3.25 `virtual void Go::Line::setParameterInterval ( double t1, double t2 ) [virtual]`

Limit the curve by limiting the parameter interval.

Implements [Go::ParamCurve](#).

29.260.3.26 `virtual double Go::Line::startparam ( ) const [virtual]`

Query the start parameter of the curve

#### Returns

the curve's start parameter

Implements [Go::ParamCurve](#).

29.260.3.27 `virtual Line* Go::Line::subCurve ( double from_par, double to_par, double fuzzy = DEFAULT_PARAMETER_EPSILON ) const [virtual]`

Returns a curve which is a part of this curve. The result is NEWed, so the user is responsible for deleting it. NB: It is not guaranteed that the [ParamCurve](#) that is returned is of the same type as the curve itself. Thus, the returned curve might be a [SplineCurve](#).



## Parameters

<i>from_par</i>	start value of parameter interval that will define the subcurve
<i>to_par</i>	end value of parameter interval that will define the subcurve
<i>fuzzy</i>	since subCurve works on those curves who are spline-based, this tolerance defines how close the start and end parameter must be to an existing knot in order to be considered <i>on</i> the knot.

## Returns

a pointer to a new subcurve which represents the part of the curve between 'from\_par' and 'to\_par'. It will be spline-based and have a k-regular knotvector. The user is responsible for deleting this subcurve when it is no longer needed.

Implements [Go::ElementaryCurve](#).

**29.260.3.28** virtual void Go::Line::swapParameters2D ( ) [virtual]

If the curve is 2 dimensional, x and y coordinates will be swapped. Used when curve is a parameter curve.

Implements [Go::ElementaryCurve](#).

**29.260.3.29** virtual void Go::Line::translateCurve ( const Point & dir ) [virtual]

Implements [Go::ElementaryCurve](#).

**29.260.3.30** virtual void Go::Line::write ( std::ostream & os ) const [virtual]

Write object to stream

## Parameters

<i>os</i>	stream to which object is written
-----------	-----------------------------------

Implements [Go::Streamable](#).

**29.260.4 Member Data Documentation**

**29.260.4.1** Point Go::Line::dir\_ [protected]

Definition at line 185 of file Line.h.

**29.260.4.2** double Go::Line::endparam\_ [protected]

Definition at line 188 of file Line.h.

29.260.4.3 **Point** `Go::Line::location_` [protected]

Definition at line 184 of file Line.h.

29.260.4.4 **double** `Go::Line::startparam_` [protected]

Definition at line 187 of file Line.h.

The documentation for this class was generated from the following file:

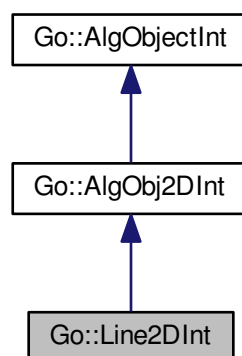
- [gtools-core/include/GoTools/geometry/Line.h](#)

## 29.261 `Go::Line2DInt` Class Reference

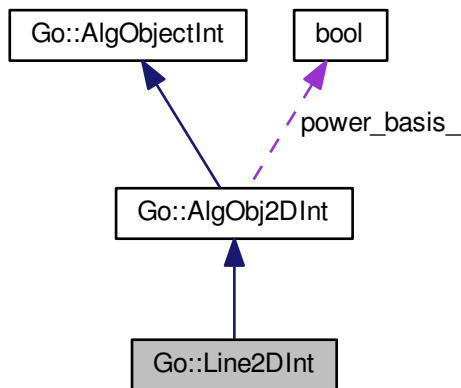
Class representing an algebraic line in 2-dimensional space.

```
#include <Line2DInt.h>
```

Inheritance diagram for `Go::Line2DInt`:



Collaboration diagram for Go::Line2DInt:



## Public Member Functions

- [Line2DInt](#) ([Point](#) point, [Point](#) dir)
- [Line2DInt](#) ([double](#) a, [double](#) b, [double](#) c)
- virtual [~Line2DInt](#) ()

*Destructor:*

- [double](#) a ()
- [double](#) b ()
- [double](#) c ()

## Additional Inherited Members

### 29.261.1 Detailed Description

Class representing an algebraic line in 2-dimensional space.

Definition at line 53 of file Line2DInt.h.

### 29.261.2 Constructor & Destructor Documentation

#### 29.261.2.1 Go::Line2DInt::Line2DInt ( [Point](#) point, [Point](#) dir )

Constructor.

Parameters

<i>point</i>	reference point which lies on the line.
<i>dir</i>	direction of the line.

### 29.261.2.2 `Go::Line2DInt::Line2DInt ( double a, double b, double c )`

Constructor. The line is given by the expression  $ax + by + c = 0$ .

#### Parameters

<i>a</i>	the x multiplier.
<i>b</i>	the y multiplier.
<i>c</i>	the constant

### 29.261.2.3 `virtual Go::Line2DInt::~~Line2DInt ( ) [inline],[virtual]`

Destructor.

Definition at line 68 of file Line2DInt.h.

## 29.261.3 Member Function Documentation

### 29.261.3.1 `double Go::Line2DInt::a ( )`

Get the x multiplier.

#### Returns

The x multiplier.

### 29.261.3.2 `double Go::Line2DInt::b ( )`

Get the y multiplier.

#### Returns

The y multiplier.

### 29.261.3.3 `double Go::Line2DInt::c ( )`

Get the constant.

#### Returns

The constant.

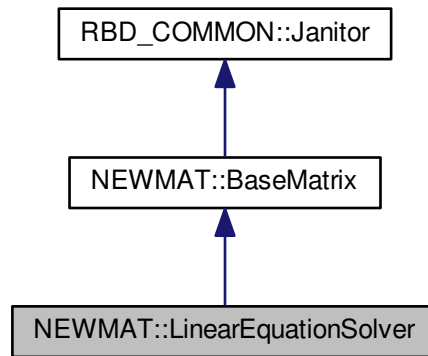
The documentation for this class was generated from the following file:

- intersections/include/GoTools/intersections/[Line2DInt.h](#)

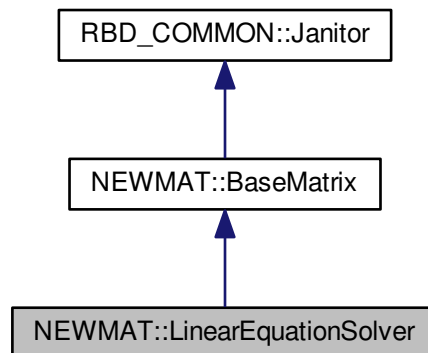
## 29.262 NEWMAT::LinearEquationSolver Class Reference

```
#include <newmat.h>
```

Inheritance diagram for NEWMAT::LinearEquationSolver:



Collaboration diagram for NEWMAT::LinearEquationSolver:



### Public Member Functions

- [LinearEquationSolver](#) (`const BaseMatrix &bm`)
- [~LinearEquationSolver](#) ()
- `void CleanUp` ()
- [GeneralMatrix \\* Evaluate](#) (`MatrixType`)

## Friends

- class [BaseMatrix](#)

## Additional Inherited Members

### 29.262.1 Detailed Description

Definition at line 1528 of file newmat.h.

### 29.262.2 Constructor & Destructor Documentation

#### 29.262.2.1 `LinearEquationSolver::LinearEquationSolver ( const BaseMatrix & bm )`

Definition at line 722 of file newmat8.cpp.

#### 29.262.2.2 `NEWMAT::LinearEquationSolver::~~LinearEquationSolver ( ) [inline]`

Definition at line 1535 of file newmat.h.

### 29.262.3 Member Function Documentation

#### 29.262.3.1 `void NEWMAT::LinearEquationSolver::Cleanup ( ) [inline],[virtual]`

Reimplemented from [NEWMAT::BaseMatrix](#).

Definition at line 1536 of file newmat.h.

#### 29.262.3.2 `GeneralMatrix* NEWMAT::LinearEquationSolver::Evaluate ( MatrixType ) [inline],[virtual]`

Implements [NEWMAT::BaseMatrix](#).

Definition at line 1537 of file newmat.h.

### 29.262.4 Friends And Related Function Documentation

#### 29.262.4.1 `friend class BaseMatrix [friend]`

Definition at line 1532 of file newmat.h.

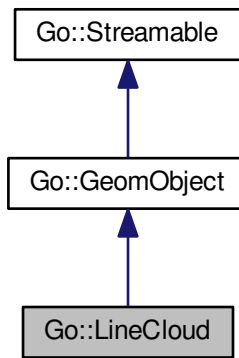
The documentation for this class was generated from the following files:

- newmat/include/newmat.h
- newmat/src/newmat8.cpp

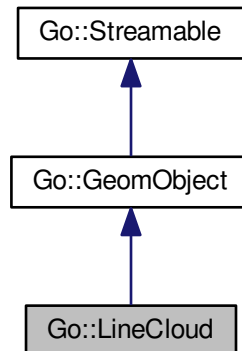
## 29.263 Go::LineCloud Class Reference

```
#include <LineCloud.h>
```

Inheritance diagram for Go::LineCloud:



Collaboration diagram for Go::LineCloud:



### Public Member Functions

- [LineCloud](#) ()  
*Makes an uninitialized [LineCloud](#) that can later be assigned or [read\(\)](#) into.*
- `template<typename ForwardIterator >`  
[LineCloud](#) (ForwardIterator start, int numlines)
- virtual `~LineCloud` ()

*Virtual destructor, enables safe inheritance.*

- virtual void [read](#) (std::istream &is)
- virtual void [write](#) (std::ostream &os) [const](#)
- virtual [BoundingBox](#) [boundingBox](#) () [const](#)
- virtual int [dimension](#) () [const](#)
- virtual [ClassType](#) [instanceType](#) () [const](#)
- virtual [LineCloud](#) \* [clone](#) () [const](#)
- void [setCloud](#) (const double \*points, int numlines)
- int [numLines](#) () [const](#)
- [Vector3D](#) & [point](#) (int i)
- const [Vector3D](#) & [point](#) (int i) [const](#)
- double \* [rawData](#) ()

## Static Public Member Functions

- static [ClassType](#) [classType](#) ()

### 29.263.1 Detailed Description

[GeomObject](#) representing a collection of line segments in space.

Definition at line 57 of file [LineCloud.h](#).

### 29.263.2 Constructor & Destructor Documentation

#### 29.263.2.1 `Go::LineCloud::LineCloud ( )` [\[inline\]](#)

Makes an uninitialized [LineCloud](#) that can later be assigned or [read\(\)](#) into.

Definition at line 62 of file [LineCloud.h](#).

#### 29.263.2.2 `template<typename ForwardIterator > Go::LineCloud::LineCloud ( ForwardIterator start, int numlines )` [\[inline\]](#)

`start` is supposed to point to the start of the array to be copied, its valuetype should be convertible to double. Generate a [LineCloud](#) based on values stored in memory. The lines should be stored as a sequence of point pairs indicating the start and end position of each line in the line cloud. Each point is stored as (x\_coord, y\_coord, z\_coord). The coordinates should be convertible to 'double'. Ex: p0\_start\_x, p0\_start\_y, p0\_start\_z, p0\_end\_x, p0\_end\_y, p0\_end\_z, p1\_start\_x, p1\_start\_y, p1\_start\_z,....

#### Parameters

<code>start</code>	pointer to the start of the memory area from which point information is to be copied
<code>numlines</code>	number of lines in the <a href="#">LineCloud</a>

Definition at line 77 of file [LineCloud.h](#).



29.263.2.3 virtual `Go::LineCloud::~~LineCloud ( )` [virtual]

Virtual destructor, enables safe inheritance.

### 29.263.3 Member Function Documentation

29.263.3.1 virtual `BoundingBox Go::LineCloud::boundingBox ( ) const` [virtual]

Get the [BoundingBox](#) of the [LineCloud](#)

#### Returns

the [BoundingBox](#) enclosing the [LineCloud](#)

Implements [Go::GeomObject](#).

29.263.3.2 static `ClassType Go::LineCloud::classType ( )` [inline],[static]

Get the `ClassType` identifier of [LineCloud](#).

#### Returns

the `ClassType` identifier of [LineCloud](#)

Definition at line 112 of file `LineCloud.h`.

29.263.3.3 virtual `LineCloud* Go::LineCloud::clone ( ) const` [inline],[virtual]

Clone this object

#### Returns

a pointer to a cloned [LineCloud](#).

Implements [Go::GeomObject](#).

Definition at line 126 of file `LineCloud.h`.

29.263.3.4 virtual `int Go::LineCloud::dimension ( ) const` [inline],[virtual]

Query the dimension of the space in which the [LineCloud](#) is embedded (currently, only dimension=3 is allowed...)

#### Returns

the dimension of the space (always 3 for now)

Implements [Go::GeomObject](#).

Definition at line 101 of file `LineCloud.h`.

```
29.263.3.5 virtual ClassType Go::LineCloud::instanceType () const [inline],[virtual]
```

Get the ClassType of this [GeomObject](#) (which is of course the ClassType identified [LineCloud](#)).

#### Returns

the ClassType identifier of [LineCloud](#)

Implements [Go::GeomObject](#).

Definition at line 107 of file LineCloud.h.

```
29.263.3.6 int Go::LineCloud::numLines () const [inline]
```

Query the number of lines in the [LineCloud](#)

#### Returns

the [LineCloud](#)'s number of lines.

Definition at line 140 of file LineCloud.h.

```
29.263.3.7 Vector3D& Go::LineCloud::point (int i) [inline]
```

Get a start or end point from a line in the [LineCloud](#) (non-const version)

#### Parameters

<i>i</i>	the index of the start/end point. If 'i' is pair, then the returned point is a start point, else it is an end point.
----------	----------------------------------------------------------------------------------------------------------------------

#### Returns

a reference to the requested start/end point

Definition at line 146 of file LineCloud.h.

```
29.263.3.8 const Vector3D& Go::LineCloud::point (int i) const [inline]
```

Get a start or end point from a line in the [LineCloud](#) (const version)

#### Parameters

<i>i</i>	the index of the start/end point. If 'i' is even, then the returned point is a start point, else it is an end point.
----------	----------------------------------------------------------------------------------------------------------------------

**Returns**

a const-reference to the requested start/end point

Definition at line 152 of file LineCloud.h.

**29.263.3.9 double\* Go::LineCloud::rawData ( ) [inline]**

Get a pointer to the start of the internal memory area where line information is stored.

**Returns**

a pointer to the beginning of the array where line information is stored.

Definition at line 157 of file LineCloud.h.

**29.263.3.10 virtual void Go::LineCloud::read ( std::istream & is ) [virtual]**

Read a [LineCloud](#) from an input stream

**Parameters**

<i>is</i>	the stream from which we will read the <a href="#">LineCloud</a>
-----------	------------------------------------------------------------------

Implements [Go::Streamable](#).

**29.263.3.11 void Go::LineCloud::setCloud ( const double \* points, int numlines )**

Fill a line cloud with information read from memory. The layout of the read information should be as for the [LineCloud](#) constructor: [LineCloud](#)(ForwardIterator start, int numlines)

**Parameters**

<i>points</i>	pointer to the memory area where the coordinates of the elements in the <a href="#">LineCloud</a> can be found. (This information will be copied).
<i>numlines</i>	number of lines in the <a href="#">LineCloud</a> .

**29.263.3.12 virtual void Go::LineCloud::write ( std::ostream & os ) const [virtual]**

Write the [LineCloud](#) to stream

**Parameters**

<i>os</i>	the stream that we will write the <a href="#">LineCloud</a> to
-----------	----------------------------------------------------------------

Implements [Go::Streamable](#).

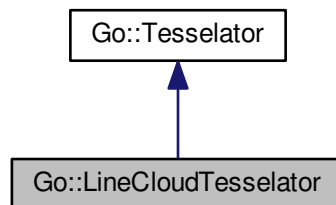
The documentation for this class was generated from the following file:

- [gotools-core/include/GoTools/geometry/LineCloud.h](#)

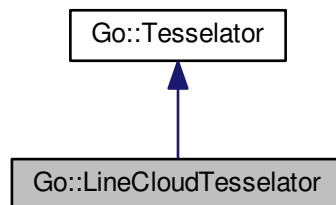
## 29.264 Go::LineCloudTesselator Class Reference

```
#include <LineCloudTesselator.h>
```

Inheritance diagram for Go::LineCloudTesselator:



Collaboration diagram for Go::LineCloudTesselator:



### Public Member Functions

- [LineCloudTesselator](#) (`const Go::LineCloud &lc`)  
*Constructor.*
- virtual [~LineCloudTesselator](#) ()  
*Destructor.*
- virtual void [tesselate](#) ()  
*Perform tessellation.*
- `const Go::LineCloud &` [getRenderCloud](#) ()  
*Fetch result.*
- void [setScale](#) (`double scale`)
- `double` [getScale](#) ()  
*Fetch scaling factor.*

### 29.264.1 Detailed Description

Tesselate line cloud. Preparation for visualization

Definition at line 52 of file LineCloudTesselator.h.

### 29.264.2 Constructor & Destructor Documentation

29.264.2.1 `Go::LineCloudTesselator::LineCloudTesselator ( const Go::LineCloud & lc ) [inline]`

Constructor.

Definition at line 56 of file LineCloudTesselator.h.

29.264.2.2 `virtual Go::LineCloudTesselator::~~LineCloudTesselator ( ) [virtual]`

Destructor.

### 29.264.3 Member Function Documentation

29.264.3.1 `const Go::LineCloud& Go::LineCloudTesselator::getRenderCloud ( ) [inline]`

Fetch result.

Definition at line 66 of file LineCloudTesselator.h.

29.264.3.2 `double Go::LineCloudTesselator::getScale ( ) [inline]`

Fetch scaling factor.

Definition at line 81 of file LineCloudTesselator.h.

29.264.3.3 `void Go::LineCloudTesselator::setScale ( double scale ) [inline]`

The difference between the given line cloud and the tessellated version is the scale. Set scaling factor.

Definition at line 73 of file LineCloudTesselator.h.

29.264.3.4 `virtual void Go::LineCloudTesselator::tesselate ( ) [virtual]`

Perform tessellation.

Implements [Go::Tesselator](#).

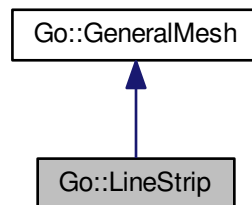
The documentation for this class was generated from the following file:

- [gotools-core/include/GoTools/tesselator/LineCloudTesselator.h](#)

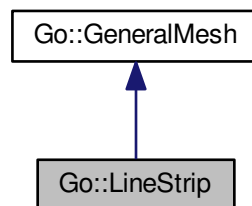
## 29.265 Go::LineStrip Class Reference

```
#include <LineStrip.h>
```

Inheritance diagram for Go::LineStrip:



Collaboration diagram for Go::LineStrip:



### Public Member Functions

- `LineStrip` (int `n`=200)  
*Constructor given size of mesh.*
- virtual `~LineStrip` ()  
*Destructor.*
- void `resize` (int `n`)  
*Change mesh size.*
- virtual int `numVertices` ()  
*Number of nodes.*
- virtual `double * vertexArray` ()  
*Fetch all nodes.*
- virtual `double * paramArray` ()  
*Fetch parameter values corresponding to the nodes.*
- virtual int `atBoundary` (int `idx`)

- Check if a given node lies at the boundary.*
  - unsigned int \* [stripArray](#) ()  
*Indices for each line strip.*
  - virtual unsigned int \* [triangleIndexArray](#) ()  
*Indices for triangles. Not used.*
  - virtual [LineStrip](#) \* [asLineStrip](#) ()  
*Casting. Return object as line strip.*

## Additional Inherited Members

### 29.265.1 Detailed Description

[LineStrip](#): Structure for storing values for a line strip, i.e. result of curve tessellation

Definition at line 54 of file [LineStrip.h](#).

### 29.265.2 Constructor & Destructor Documentation

#### 29.265.2.1 `Go::LineStrip::LineStrip ( int n = 200 )`

Constructor given size of mesh.

#### 29.265.2.2 `virtual Go::LineStrip::~~LineStrip ( )` [virtual]

Destructor.

### 29.265.3 Member Function Documentation

#### 29.265.3.1 `virtual LineStrip* Go::LineStrip::asLineStrip ( )` [virtual]

Casting. Return object as line strip.

Reimplemented from [Go::GeneralMesh](#).

#### 29.265.3.2 `virtual int Go::LineStrip::atBoundary ( int idx )` [virtual]

Check if a given node lies at the boundary.

Implements [Go::GeneralMesh](#).

#### 29.265.3.3 `virtual int Go::LineStrip::numVertices ( )` [inline],[virtual]

Number of nodes.

Implements [Go::GeneralMesh](#).

Definition at line 66 of file [LineStrip.h](#).

**29.265.3.4** `virtual double* Go::LineStrip::paramArray ( ) [inline],[virtual]`

Fetch parameter values corresponding to the nodes.

Implements [Go::GeneralMesh](#).

Definition at line 70 of file LineStrip.h.

**29.265.3.5** `void Go::LineStrip::resize ( int n )`

Change mesh size.

**29.265.3.6** `unsigned int* Go::LineStrip::stripArray ( ) [inline]`

Indices for each line strip.

Definition at line 73 of file LineStrip.h.

**29.265.3.7** `virtual unsigned int* Go::LineStrip::triangleIndexArray ( ) [inline],[virtual]`

Indices for triangles. Not used.

Implements [Go::GeneralMesh](#).

Definition at line 75 of file LineStrip.h.

**29.265.3.8** `virtual double* Go::LineStrip::vertexArray ( ) [inline],[virtual]`

Fetch all nodes.

Implements [Go::GeneralMesh](#).

Definition at line 69 of file LineStrip.h.

The documentation for this class was generated from the following file:

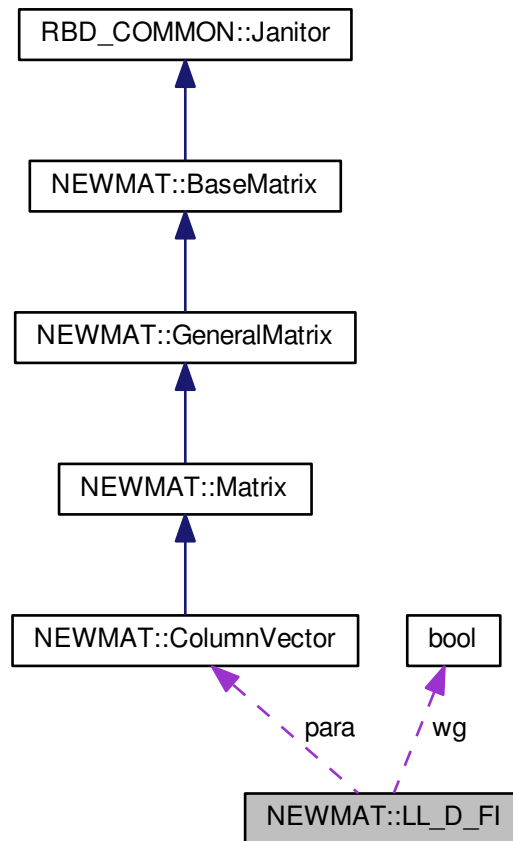
- [gotools-core/include/GoTools/tesselator/LineStrip.h](#)



## 29.266 NEWMAT::LL\_D\_FI Class Reference

```
#include <newmatnl.h>
```

Collaboration diagram for NEWMAT::LL\_D\_FI:



### Public Member Functions

- virtual void [Set](#) (const [ColumnVector](#) &X)
- virtual void [WG](#) (bool wgx)
- virtual bool [IsValid](#) ()
- bool [IsValid](#) (const [ColumnVector](#) &X, bool wgx=true)
- virtual [Real](#) [LogLikelihood](#) ()=0
- [Real](#) [LogLikelihood](#) (const [ColumnVector](#) &X, bool wgx=true)
- virtual [ReturnMatrix](#) [Derivatives](#) ()=0
- virtual [ReturnMatrix](#) [FI](#) ()=0
- virtual [~LL\\_D\\_FI](#) ()

### Protected Attributes

- [ColumnVector](#) para
- bool wg

### 29.266.1 Detailed Description

Definition at line 253 of file newmatnl.h.

### 29.266.2 Constructor & Destructor Documentation

29.266.2.1 virtual `NEWMAT::LL_D_FI::~~LL_D_FI( )` [`inline`],[`virtual`]

Definition at line 277 of file newmatnl.h.

### 29.266.3 Member Function Documentation

29.266.3.1 virtual `ReturnMatrix` `NEWMAT::LL_D_FI::Derivatives( )` [`pure virtual`]

29.266.3.2 virtual `ReturnMatrix` `NEWMAT::LL_D_FI::FI( )` [`pure virtual`]

29.266.3.3 virtual `bool` `NEWMAT::LL_D_FI::IsValid( )` [`inline`],[`virtual`]

Definition at line 265 of file newmatnl.h.

29.266.3.4 `bool` `NEWMAT::LL_D_FI::IsValid( const ColumnVector & X, bool wgx = true )` [`inline`]

Definition at line 267 of file newmatnl.h.

29.266.3.5 virtual `Real` `NEWMAT::LL_D_FI::LogLikelihood( )` [`pure virtual`]

29.266.3.6 `Real` `NEWMAT::LL_D_FI::LogLikelihood( const ColumnVector & X, bool wgx = true )` [`inline`]

Definition at line 271 of file newmatnl.h.

29.266.3.7 virtual `void` `NEWMAT::LL_D_FI::Set( const ColumnVector & X )` [`inline`],[`virtual`]

Definition at line 260 of file newmatnl.h.

29.266.3.8 virtual `void` `NEWMAT::LL_D_FI::WG( bool wgx )` [`inline`],[`virtual`]

Definition at line 262 of file newmatnl.h.

### 29.266.4 Member Data Documentation

29.266.4.1 `ColumnVector` `NEWMAT::LL_D_FI::para` [`protected`]

Definition at line 256 of file newmatnl.h.

29.266.4.2 `bool NEWMAT::LL_D_Fl::wg` [protected]

Definition at line 257 of file `newmatnl.h`.

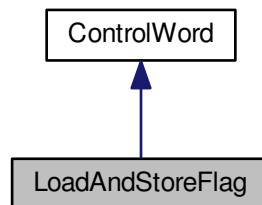
The documentation for this class was generated from the following file:

- `newmat/include/newmatnl.h`

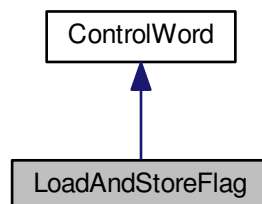
## 29.267 LoadAndStoreFlag Class Reference

```
#include <newmatrc.h>
```

Inheritance diagram for LoadAndStoreFlag:



Collaboration diagram for LoadAndStoreFlag:



### Public Member Functions

- `LoadAndStoreFlag` ()
- `LoadAndStoreFlag` (int i)
- `LoadAndStoreFlag` (LSF lsf)
- `LoadAndStoreFlag` (const `ControlWord` &cwx)

## Additional Inherited Members

### 29.267.1 Detailed Description

Definition at line 24 of file newmatrc.h.

### 29.267.2 Constructor & Destructor Documentation

#### 29.267.2.1 LoadAndStoreFlag::LoadAndStoreFlag ( ) [inline]

Definition at line 27 of file newmatrc.h.

#### 29.267.2.2 LoadAndStoreFlag::LoadAndStoreFlag ( int *i* ) [inline]

Definition at line 28 of file newmatrc.h.

#### 29.267.2.3 LoadAndStoreFlag::LoadAndStoreFlag ( LSF *lsf* ) [inline]

Definition at line 29 of file newmatrc.h.

#### 29.267.2.4 LoadAndStoreFlag::LoadAndStoreFlag ( const ControlWord & *cwx* ) [inline]

Definition at line 30 of file newmatrc.h.

The documentation for this class was generated from the following file:

- [newmat/include/newmatrc.h](#)

## 29.268 Go::LockedDirDistFunc Class Reference

```
#include <IntersectionPoolUtils.h>
```

### Public Member Functions

- [LockedDirDistFunc](#) (const ParamObjectInt \*const o1, const ParamObjectInt \*const o2, int fixed\_dir, double fixed\_value)
- [double operator\(\)](#) (const double \*arg) const
- [void grad](#) (const double \*arg, double \*grad) const
- [double minPar](#) (int n) const
- [double maxPar](#) (int n) const

### 29.268.1 Detailed Description

Definition at line 356 of file IntersectionPoolUtils.h.

### 29.268.2 Constructor & Destructor Documentation

29.268.2.1 `Go::LockedDirDistFunc::LockedDirDistFunc ( const ParamObjectInt *const o1, const ParamObjectInt *const o2, int fixed_dir, double fixed_value ) [inline]`

Definition at line 360 of file IntersectionPoolUtils.h.

### 29.268.3 Member Function Documentation

29.268.3.1 `void Go::LockedDirDistFunc::grad ( const double * arg, double * grad ) const`

Definition at line 592 of file IntersectionPoolUtils.h.

29.268.3.2 `double Go::LockedDirDistFunc::maxPar ( int n ) const [inline]`

Definition at line 387 of file IntersectionPoolUtils.h.

29.268.3.3 `double Go::LockedDirDistFunc::minPar ( int n ) const [inline]`

Definition at line 386 of file IntersectionPoolUtils.h.

29.268.3.4 `double Go::LockedDirDistFunc::operator() ( const double * arg ) const`

Definition at line 581 of file IntersectionPoolUtils.h.

The documentation for this class was generated from the following file:

- [intersections/include/GoTools/intersections/IntersectionPoolUtils.h](#)

## 29.269 NEWMAT::LogAndSign Class Reference

```
#include <newmat.h>
```

### Public Member Functions

- [LogAndSign \(\)](#)
- [LogAndSign \(Real\)](#)
- `void operator*= (Real)`
- `void PowEq (int k)`
- `void ChangeSign ()`
- `Real LogValue () const`
- `int Sign () const`
- `Real Value () const`

### 29.269.1 Detailed Description

Definition at line 46 of file newmat.h.

### 29.269.2 Constructor & Destructor Documentation

#### 29.269.2.1 NEWMAT::LogAndSign::LogAndSign ( ) [inline]

Definition at line 53 of file newmat.h.

#### 29.269.2.2 NEWMAT::LogAndSign::LogAndSign ( Real )

### 29.269.3 Member Function Documentation

#### 29.269.3.1 void NEWMAT::LogAndSign::ChangeSign ( ) [inline]

Definition at line 57 of file newmat.h.

#### 29.269.3.2 Real NEWMAT::LogAndSign::LogValue ( ) const [inline]

Definition at line 58 of file newmat.h.

#### 29.269.3.3 void LogAndSign::operator\*=( Real x )

Definition at line 615 of file newmat8.cpp.

#### 29.269.3.4 void LogAndSign::PowEq ( int k )

Definition at line 622 of file newmat8.cpp.

#### 29.269.3.5 int NEWMAT::LogAndSign::Sign ( ) const [inline]

Definition at line 59 of file newmat.h.

#### 29.269.3.6 Real LogAndSign::Value ( ) const

Definition at line 631 of file newmat8.cpp.

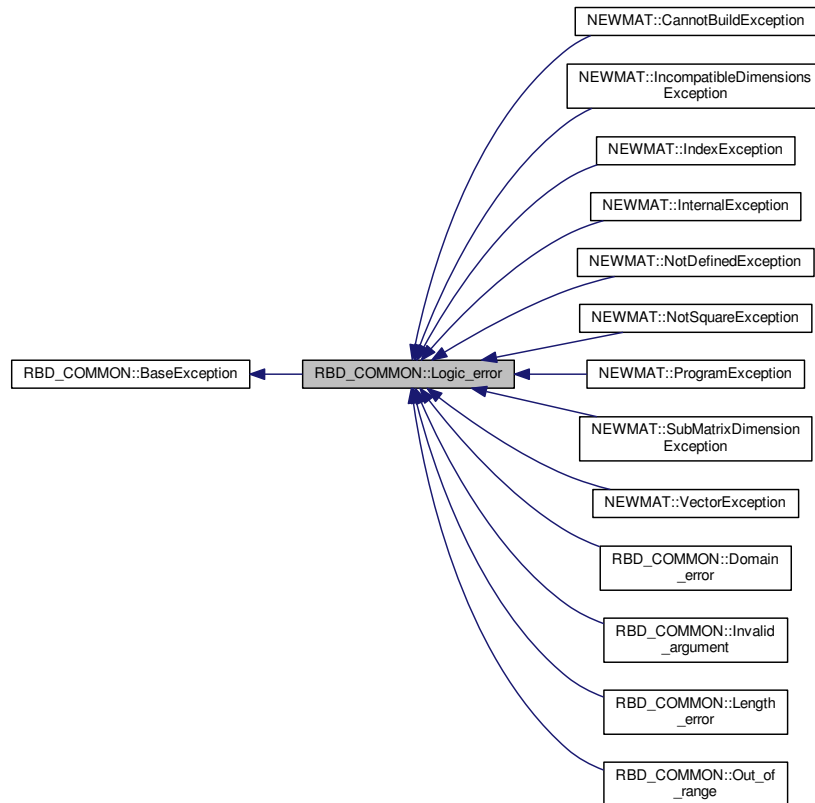
The documentation for this class was generated from the following files:

- [newmat/include/newmat.h](#)
- [newmat/src/newmat8.cpp](#)

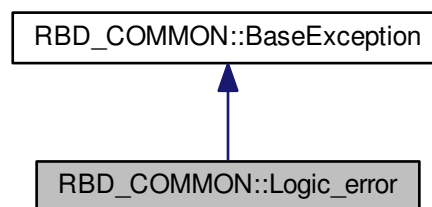
## 29.270 RBD\_COMMON::Logic\_error Class Reference

```
#include <myexcept.h>
```

Inheritance diagram for RBD\_COMMON::Logic\_error:



Collaboration diagram for RBD\_COMMON::Logic\_error:



- static unsigned long [Select](#)
- [Logic\\_error](#) (const char \*a\_what=0)

## Additional Inherited Members

### 29.270.1 Detailed Description

Definition at line 357 of file myexcept.h.

### 29.270.2 Constructor & Destructor Documentation

#### 29.270.2.1 Logic\_error::Logic\_error ( const char \* a\_what = 0 )

Definition at line 397 of file myexcept.cpp.

### 29.270.3 Member Data Documentation

#### 29.270.3.1 unsigned long Logic\_error::Select [static]

Definition at line 360 of file myexcept.h.

The documentation for this class was generated from the following files:

- [newmat/include/myexcept.h](#)
- [newmat/src/myexcept.cpp](#)

## 29.271 Go::Loop Class Reference

Primarily a loop connected to a face in a boundary represented solid or face set. May also be used to represent a general closed sequence of edges.

```
#include <Loop.h>
```

### Public Member Functions

- [Loop](#) ([ftFaceBase](#) \*face, [CurveLoop](#) &curve\_loop, [double](#) kink, [bool](#) split\_in\_kinks=true, [bool](#) no\_split=false)  
*Constructor.*
- [Loop](#) ([ftFaceBase](#) \*face, [std::vector](#)< [shared\\_ptr](#)< [ftEdgeBase](#) > > &edges, [double](#) space\_epsilon)
- [Loop](#) ([std::vector](#)< [shared\\_ptr](#)< [ftEdgeBase](#) > > &edges, [double](#) space\_epsilon)
- [~Loop](#) ()  
*Destructor.*
- [size\\_t](#) size ()  
*Number of edges in loop.*
- [std::vector](#)< [shared\\_ptr](#)< [ftEdgeBase](#) > > & [getEdges](#) ()  
*Get all edges oriented head to tail.*
- [shared\\_ptr](#)< [ftEdgeBase](#) > [getEdge](#) ([size\\_t](#) idx)
- [std::vector](#)< [shared\\_ptr](#)< [Vertex](#) > > [getVertices](#) () [const](#)  
*Fetch all vertices between edges in the loop.*
- [std::vector](#)< [shared\\_ptr](#)< [Vertex](#) > > [getSeqVertices](#) () [const](#)



- Fetch all vertices between edges in the loop in sequence.*

  - `double getTol ()`  

*Get tolerance.*
  - `bool isFaceConsistent ()`  

*Check consistency with regard to face.*
  - `void setFace (ftFaceBase *face)`  

*Set face pointer.*
  - `const ftFaceBase * getFace ()`  

*The face associated to this loop.*
  - `void updateLoop (shared_ptr< ftEdgeBase > new_edge)`  

*Make sure that the edges belonging to this loop is complete.*
  - `void split (int ind, double par)`  

*Split loop in a given parameter of an edge given by index.*
  - `bool isClose (ftEdge *edge, RectDomain *domain, double tol) const`  

*Test if an edge is close to the surface within some tolerance.*
  - `void getBadDistance (std::vector< std::pair< ftSurface *, ftEdge * > > &badPairs, RectDomain *domain, double tol) const`  

*Collect all pairs of surface and edges where some part of the edge has a distance to the surface greater than a tolerance.*
  - `void getBadDistance (std::vector< std::pair< ftEdge *, shared_ptr< Vertex > > > &badPairs, double tol) const`
  - `void getPosTangentSurfaceDiscont (std::vector< ftEdge * > &badPos, std::vector< ftEdge * > &badTangent, double tol, double kink, double bend, int leastSurfIndex, shared_ptr< SurfaceModel > sm) const`  

*Fetch information on continuity.*
  - `bool checkConsistency () const`
  - `void getAcuteEdges (std::vector< std::pair< ftEdge *, ftEdge * > > &acute_edges, double angtol) const`  

*Check for acute edges in boundary loop.*
  - `void getLoopIntersections (shared_ptr< Loop > loop2, double tol, std::vector< std::pair< shared_ptr< PointOnEdge > >, shared_ptr< PointOnEdge > > > &int_pt) const`  

*Compute intersections between boundary loops.*
  - `void getLoopSelfIntersections (double tol, std::vector< std::pair< shared_ptr< PointOnEdge > >, shared_ptr< PointOnEdge > > > &int_pt) const`  

*Compute self intersections of boundary loop.*
  - `bool correspondingEdges (shared_ptr< Loop > other, double tol, ftEdgeBase *&first1, ftEdgeBase *&first2, bool &same_dir, bool no_snap=true)`
  - `bool hasRadialEdges () const`
  - `bool allRadialEdges () const`  

*All edges is connected a radial edge.*
  - `void closestPoint (const Point &pt, int &clo_ind, double &clo_par, Point &clo_pt, double &clo_dist) const`
  - `bool isInLoop (ftEdgeBase *edge)`  

*Check if a given edge is in this loop.*
  - `void groupSmoothEdges (double tol, double angtol, std::vector< std::vector< shared_ptr< ftEdgeBase > > > &edge_groups)`

### 29.271.1 Detailed Description

Primarily a loop connected to a face in a boundary represented solid or face set. May also be used to represent a general closed sequence of edges.

Definition at line 70 of file Loop.h.

## 29.271.2 Constructor & Destructor Documentation

29.271.2.1 `Go::Loop::Loop ( ftFaceBase * face, CurveLoop & curve_loop, double kink, bool split_in_kinks = true, bool no_split = false )`

Constructor.

29.271.2.2 `Go::Loop::Loop ( ftFaceBase * face, std::vector< shared_ptr< ftEdgeBase > > & edges, double space_epsilon )`

This constructor takes an ordered sequence of edges as input Note that the function may throw

29.271.2.3 `Go::Loop::Loop ( std::vector< shared_ptr< ftEdgeBase > > & edges, double space_epsilon )`

Constructor that takes an ordered sequence of edges as input. There is no reference to a face object.

29.271.2.4 `Go::Loop::~~Loop ( )`

Destructor.

## 29.271.3 Member Function Documentation

29.271.3.1 `bool Go::Loop::allRadialEdges ( ) const`

All edges is connected a radial edge.

29.271.3.2 `bool Go::Loop::checkConsistency ( ) const`

Check if the edges of the loop are consistent with the corresponding curves with regard to orientation

29.271.3.3 `void Go::Loop::closestPoint ( const Point & pt, int & clo_ind, double & clo_par, Point & clo_pt, double & clo_dist ) const`

Find the closest point on the curve loop to a point specified by the user.

### Parameters

<code>pt</code>	The point given by the user. We want to determine the closest point to this on the <a href="#">CurveLoop</a> .
<code>clo_ind</code>	Upon return: the index of the curve segment on which the closest point was found.
<code>clo_par</code>	Upon return: the parameter of the detected closest point on the curve containing it.
<code>clo_pt</code>	Upon return: the geometric position of the detected closest point
<code>clo_dist</code>	Upon return: the distance to the detected closest point.

29.271.3.4 **bool** Go::Loop::correspondingEdges ( *shared\_ptr*< Loop > *other*, *double tol*, *ftEdgeBase* \* & *first1*, *ftEdgeBase* \* & *first2*, *bool* & *same\_dir*, *bool no\_snap = true* )

Non-manifold functionality Check for loop correspondance, split edges if required and return the start edges for corresponding loops

29.271.3.5 **void** Go::Loop::getAcuteEdges ( *std::vector*< *std::pair*< *ftEdge* \*, *ftEdge* \* > > & *acute\_edges*, *double angtol* ) *const*

Check for acute edges in boundary loop.

29.271.3.6 **void** Go::Loop::getBadDistance ( *std::vector*< *std::pair*< *ftSurface* \*, *ftEdge* \* > > & *badPairs*, *RectDomain* \* *domain*, *double tol* ) *const*

Collect all pairs of surface and edges where some part of the edge has a distance to the surface greater than a tolerance.

29.271.3.7 **void** Go::Loop::getBadDistance ( *std::vector*< *std::pair*< *ftEdge* \*, *shared\_ptr*< Vertex > > > & *badPairs*, *double tol* ) *const*

29.271.3.8 *shared\_ptr*<*ftEdgeBase*> Go::Loop::getEdge ( *size\_t idx* ) [*inline*]

Fetch an edge from the loop specified by the index in the sequence of edges

Definition at line 107 of file Loop.h.

29.271.3.9 *std::vector*<*shared\_ptr*<*ftEdgeBase*> > & Go::Loop::getEdges ( ) [*inline*]

Get all edges oriented head to tail.

Definition at line 100 of file Loop.h.

29.271.3.10 **const** *ftFaceBase*\* Go::Loop::getFace ( ) [*inline*]

The face associated to this loop.

Definition at line 136 of file Loop.h.

29.271.3.11 **void** Go::Loop::getLoopIntersections ( *shared\_ptr*< Loop > *loop2*, *double tol*, *std::vector*< *std::pair*< *shared\_ptr*< PointOnEdge >, *shared\_ptr*< PointOnEdge > > > & *int\_pt* ) *const*

Compute intersections between boundary loops.

```
29.271.3.12 void Go::Loop::getLoopSelfIntersections (double tol, std::vector< std::pair< shared_ptr< PointOnEdge >,
shared_ptr< PointOnEdge > > > & int_pt) const
```

Compute self intersections of boundary loop.

```
29.271.3.13 void Go::Loop::getPosTangentSurfaceDiscont (std::vector< ftEdge * > & badPos, std::vector< ftEdge * > &
badTangent, double tol, double kink, double bend, int leastSurfIndex, shared_ptr< SurfaceModel > sm)
const
```

Fetch information on continuity.

```
29.271.3.14 std::vector<shared_ptr<Vertex> > Go::Loop::getSeqVertices () const
```

Fetch all vertices between edges in the loop in sequence.

```
29.271.3.15 double Go::Loop::getTol () [inline]
```

Get tolerance.

Definition at line 123 of file Loop.h.

```
29.271.3.16 std::vector<shared_ptr<Vertex> > Go::Loop::getVertices () const
```

Fetch all vertices between edges in the loop.

```
29.271.3.17 void Go::Loop::groupSmoothEdges (double tol, double angtol, std::vector< std::vector< shared_ptr<
ftEdgeBase > > > & edge_groups)
```

Group edges that are smoothly joined together. The sequence of edges corresponds to the sequence in the [Loop](#), i.e. head to tail connected

```
29.271.3.18 bool Go::Loop::hasRadialEdges () const
```

Check for radial edges Existence

```
29.271.3.19 bool Go::Loop::isClose (ftEdge * edge, RectDomain * domain, double tol) const
```

Test if an edge is close to the surface within some tolerance.

```
29.271.3.20 bool Go::Loop::isFaceConsistent ()
```

Check consistency with regard to face.

29.271.3.21 `bool Go::Loop::isInLoop ( ftEdgeBase * edge ) [inline]`

Check if a given edge is in this loop.

Definition at line 212 of file Loop.h.

29.271.3.22 `void Go::Loop::setFace ( ftFaceBase * face )`

Set face pointer.

29.271.3.23 `size_t Go::Loop::size ( ) [inline]`

Number of edges in loop.

Definition at line 94 of file Loop.h.

29.271.3.24 `void Go::Loop::split ( int ind, double par )`

Split loop in a given parameter of an edge given by index.

29.271.3.25 `void Go::Loop::updateLoop ( shared_ptr< ftEdgeBase > new_edge )`

Make sure that the edges belonging to this loop is complete.

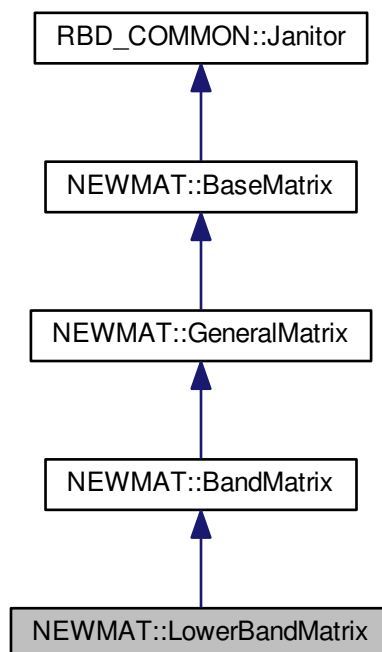
The documentation for this class was generated from the following file:

- `compositemodel/include/GoTools/compositemodel/Loop.h`

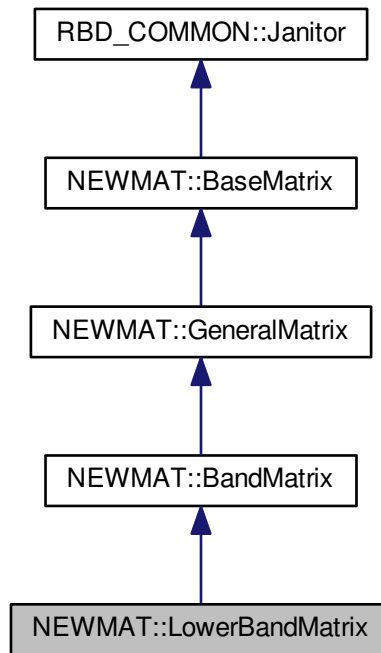
## 29.272 NEWMAT::LowerBandMatrix Class Reference

```
#include <newmat.h>
```

Inheritance diagram for NEWMAT::LowerBandMatrix:



Collaboration diagram for NEWMAT::LowerBandMatrix:



## Public Member Functions

- [LowerBandMatrix](#) ()
- [~LowerBandMatrix](#) ()
- [LowerBandMatrix](#) (int n, int lbw)
- [LowerBandMatrix](#) (const [BaseMatrix](#) &)
- void [operator=](#) (const [BaseMatrix](#) &)
- void [operator=](#) (Real f)
- void [operator=](#) (const [LowerBandMatrix](#) &m)
- [MatrixType](#) [Type](#) () const
- [LowerBandMatrix](#) (const [LowerBandMatrix](#) &gm)
- [GeneralMatrix](#) \* [MakeSolver](#) ()
- void [Solver](#) ([MatrixColX](#) &, const [MatrixColX](#) &)
- [LogAndSign](#) [LogDeterminant](#) () const
- void [ReSize](#) (int, int, int)
- void [ReSize](#) (int n, int lbw)
- void [ReSize](#) (const [GeneralMatrix](#) &A)
- [Real](#) & [operator\(\)](#) (int, int)
- [Real](#) [operator\(\)](#) (int, int) const
- [Real](#) & [element](#) (int, int)
- [Real](#) [element](#) (int, int) const

## Additional Inherited Members

### 29.272.1 Detailed Description

Definition at line 984 of file newmat.h.

### 29.272.2 Constructor & Destructor Documentation

29.272.2.1 `NEWMAT::LowerBandMatrix::LowerBandMatrix ( )` `[inline]`

Definition at line 988 of file newmat.h.

29.272.2.2 `NEWMAT::LowerBandMatrix::~~LowerBandMatrix ( )` `[inline]`

Definition at line 989 of file newmat.h.

29.272.2.3 `NEWMAT::LowerBandMatrix::LowerBandMatrix ( int n, int lbw )` `[inline]`

Definition at line 990 of file newmat.h.

29.272.2.4 `NEWMAT::LowerBandMatrix::LowerBandMatrix ( const BaseMatrix & )`

29.272.2.5 `NEWMAT::LowerBandMatrix::LowerBandMatrix ( const LowerBandMatrix & gm )` `[inline]`

Definition at line 998 of file newmat.h.

### 29.272.3 Member Function Documentation

29.272.3.1 `Real& NEWMAT::LowerBandMatrix::element ( int , int )`

29.272.3.2 `Real NEWMAT::LowerBandMatrix::element ( int , int ) const`

29.272.3.3 `LogAndSign LowerBandMatrix::LogDeterminant ( ) const` `[virtual]`

Reimplemented from [NEWMAT::BandMatrix](#).

Definition at line 403 of file bandmat.cpp.

29.272.3.4 `GeneralMatrix* NEWMAT::LowerBandMatrix::MakeSolver ( )` `[inline]`, `[virtual]`

Reimplemented from [NEWMAT::BandMatrix](#).

Definition at line 999 of file newmat.h.



29.272.3.5 `Real& NEWMAT::LowerBandMatrix::operator()( int , int )`

29.272.3.6 `Real NEWMAT::LowerBandMatrix::operator()( int , int ) const`

29.272.3.7 `void NEWMAT::LowerBandMatrix::operator= ( const BaseMatrix & )`

29.272.3.8 `void NEWMAT::LowerBandMatrix::operator= ( Real f ) [inline]`

Definition at line 994 of file newmat.h.

29.272.3.9 `void NEWMAT::LowerBandMatrix::operator= ( const LowerBandMatrix & m ) [inline]`

Definition at line 995 of file newmat.h.

29.272.3.10 `void NEWMAT::LowerBandMatrix::ReSize ( int , int , int ) [virtual]`

Reimplemented from [NEWMAT::BandMatrix](#).

29.272.3.11 `void NEWMAT::LowerBandMatrix::ReSize ( int n, int bw ) [inline]`

Definition at line 1003 of file newmat.h.

29.272.3.12 `void NEWMAT::LowerBandMatrix::ReSize ( const GeneralMatrix & A ) [inline],[virtual]`

Reimplemented from [NEWMAT::BandMatrix](#).

Definition at line 1005 of file newmat.h.

29.272.3.13 `void LowerBandMatrix::Solver ( MatrixColX & mcout, const MatrixColX & mcin ) [virtual]`

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 375 of file bandmat.cpp.

29.272.3.14 `MatrixType LowerBandMatrix::Type ( ) const [virtual]`

Reimplemented from [NEWMAT::BandMatrix](#).

Definition at line 395 of file newmat4.cpp.

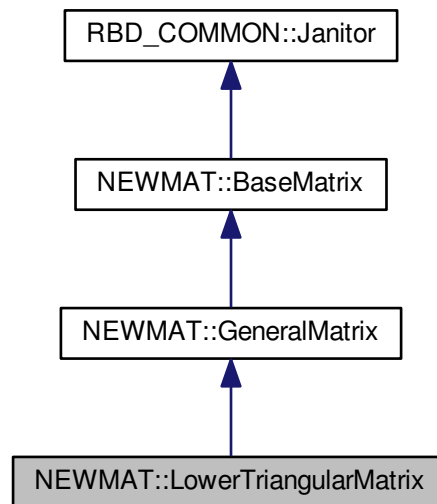
The documentation for this class was generated from the following files:

- [newmat/include/newmat.h](#)
- [newmat/src/bandmat.cpp](#)
- [newmat/src/newmat4.cpp](#)

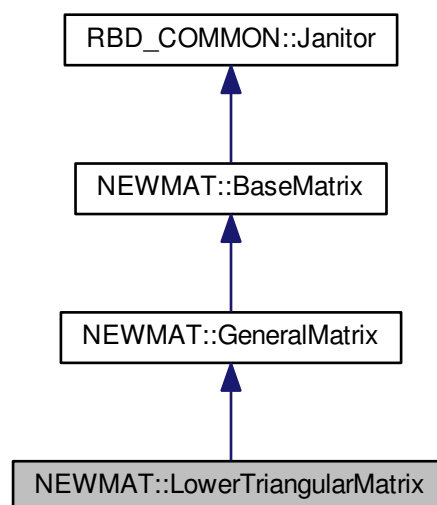
## 29.273 NEWMAT::LowerTriangularMatrix Class Reference

```
#include <newmat.h>
```

Inheritance diagram for NEWMAT::LowerTriangularMatrix:



Collaboration diagram for NEWMAT::LowerTriangularMatrix:



## Public Member Functions

- [LowerTriangularMatrix \(\)](#)
- [~LowerTriangularMatrix \(\)](#)
- [LowerTriangularMatrix \(ArrayLengthSpecifier\)](#)
- [LowerTriangularMatrix \(const LowerTriangularMatrix &gm\)](#)
- [LowerTriangularMatrix \(const BaseMatrix &M\)](#)
- [void operator= \(const BaseMatrix &\)](#)
- [void operator= \(Real f\)](#)
- [void operator= \(const LowerTriangularMatrix &m\)](#)
- [Real & operator\(\) \(int, int\)](#)
- [Real & element \(int, int\)](#)
- [Real operator\(\) \(int, int\) const](#)
- [Real element \(int, int\) const](#)
- [MatrixType Type \(\) const](#)
- [GeneralMatrix \\* MakeSolver \(\)](#)
- [void Solver \(MatrixColX &, const MatrixColX &\)](#)
- [LogAndSign LogDeterminant \(\) const](#)
- [Real Trace \(\) const](#)
- [void GetRow \(MatrixRowCol &\)](#)
- [void GetCol \(MatrixRowCol &\)](#)
- [void GetCol \(MatrixColX &\)](#)
- [void RestoreCol \(MatrixRowCol &\)](#)
- [void RestoreCol \(MatrixColX &c\)](#)
- [void NextRow \(MatrixRowCol &\)](#)
- [void ReSize \(int\)](#)
- [void ReSize \(const GeneralMatrix &A\)](#)
- [MatrixBandWidth BandWidth \(\) const](#)

## Additional Inherited Members

### 29.273.1 Detailed Description

Definition at line 708 of file newmat.h.

### 29.273.2 Constructor & Destructor Documentation

**29.273.2.1** `NEWMAT::LowerTriangularMatrix::LowerTriangularMatrix ( )` `[inline]`

Definition at line 712 of file newmat.h.

**29.273.2.2** `NEWMAT::LowerTriangularMatrix::~~LowerTriangularMatrix ( )` `[inline]`

Definition at line 713 of file newmat.h.

29.273.2.3 **NEWMAT::LowerTriangularMatrix::LowerTriangularMatrix** ( **ArrayLengthSpecifier** )

29.273.2.4 **NEWMAT::LowerTriangularMatrix::LowerTriangularMatrix** ( **const LowerTriangularMatrix & gm** )  
[inline]

Definition at line 715 of file newmat.h.

29.273.2.5 **LowerTriangularMatrix::LowerTriangularMatrix** ( **const BaseMatrix & M** )

Definition at line 108 of file newmat4.cpp.

### 29.273.3 Member Function Documentation

29.273.3.1 **MatrixBandWidth** **LowerTriangularMatrix::BandWidth** ( ) **const** [virtual]

Reimplemented from [NEWMAT::BaseMatrix](#).

Definition at line 409 of file newmat4.cpp.

29.273.3.2 **Real&** **NEWMAT::LowerTriangularMatrix::element** ( **int**, **int** )

29.273.3.3 **Real** **NEWMAT::LowerTriangularMatrix::element** ( **int**, **int** ) **const**

29.273.3.4 **void** **NEWMAT::LowerTriangularMatrix::GetCol** ( **MatrixRowCol &** ) [virtual]

Implements [NEWMAT::GeneralMatrix](#).

29.273.3.5 **void** **NEWMAT::LowerTriangularMatrix::GetCol** ( **MatrixCoIX &** ) [virtual]

Implements [NEWMAT::GeneralMatrix](#).

29.273.3.6 **void** **LowerTriangularMatrix::GetRow** ( **MatrixRowCol & mrc** ) [virtual]

Implements [NEWMAT::GeneralMatrix](#).

Definition at line 334 of file newmat3.cpp.

29.273.3.7 **LogAndSign** **LowerTriangularMatrix::LogDeterminant** ( ) **const** [virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 655 of file newmat8.cpp.

29.273.3.8 **GeneralMatrix\*** NEWMAT::LowerTriangularMatrix::MakeSolver ( ) [inline],[virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 730 of file newmat.h.

29.273.3.9 void LowerTriangularMatrix::NextRow ( **MatrixRowCol** & *mrc* ) [virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 392 of file newmat3.cpp.

29.273.3.10 **Real&** NEWMAT::LowerTriangularMatrix::operator() ( *int* , *int* )

29.273.3.11 **Real** NEWMAT::LowerTriangularMatrix::operator() ( *int* , *int* ) const

29.273.3.12 void NEWMAT::LowerTriangularMatrix::operator= ( **const BaseMatrix** & )

29.273.3.13 void NEWMAT::LowerTriangularMatrix::operator= ( **Real** *f* ) [inline]

Definition at line 718 of file newmat.h.

29.273.3.14 void NEWMAT::LowerTriangularMatrix::operator= ( **const LowerTriangularMatrix** & *m* ) [inline]

Definition at line 719 of file newmat.h.

29.273.3.15 void NEWMAT::LowerTriangularMatrix::ReSize ( *int* )

29.273.3.16 void LowerTriangularMatrix::ReSize ( **const GeneralMatrix** & *A* ) [virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 315 of file newmat4.cpp.

29.273.3.17 void NEWMAT::LowerTriangularMatrix::RestoreCol ( **MatrixRowCol** & ) [virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

29.273.3.18 void NEWMAT::LowerTriangularMatrix::RestoreCol ( **MatrixColX** & *c* ) [inline],[virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 738 of file newmat.h.

29.273.3.19 `void LowerTriangularMatrix::Solver ( MatrixCoIX & mcout, const MatrixCoIX & mcin ) [virtual]`

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 70 of file newmat7.cpp.

29.273.3.20 `Real LowerTriangularMatrix::Trace ( ) const [virtual]`

Reimplemented from [NEWMAT::BaseMatrix](#).

Definition at line 563 of file newmat8.cpp.

29.273.3.21 `MatrixType LowerTriangularMatrix::Type ( ) const [virtual]`

Implements [NEWMAT::GeneralMatrix](#).

Definition at line 388 of file newmat4.cpp.

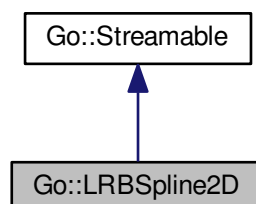
The documentation for this class was generated from the following files:

- [newmat/include/newmat.h](#)
- [newmat/src/newmat3.cpp](#)
- [newmat/src/newmat4.cpp](#)
- [newmat/src/newmat7.cpp](#)
- [newmat/src/newmat8.cpp](#)

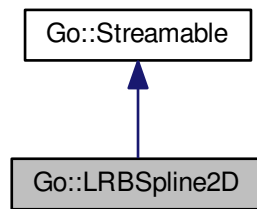
## 29.274 Go::LRBSpline2D Class Reference

```
#include <LRBSpline2D.h>
```

Inheritance diagram for Go::LRBSpline2D:



Collaboration diagram for Go::LRBSpline2D:



## Public Member Functions

- [LRBSpline2D](#) ()  
*Constructor to create an empty (invalid) LRBSpline2D.*
- `template<typename Iterator >`  
[LRBSpline2D](#) (`const Point &c_g`, `double weight`, `int deg_u`, `int deg_v`, `Iterator kvec_u_start`, `Iterator kvec_v_start`, `double gamma`, `const Mesh2D *mesh`, `bool rational=false`)
- [LRBSpline2D](#) (`const LRBSpline2D &rhs`)  
*Copy constructor.*
- `void swap` (`LRBSpline2D &rhs`)  
*Swap the contents of two LRBSpline2Ds.*
- [~LRBSpline2D](#) ()
- `virtual void write` (`std::ostream &os`) `const`  
*Write the LRBSpline2D to a stream.*
- `virtual void read` (`std::istream &is`)  
*Read the LRBSpline2D from a stream.*
- `double evalBasisFunc` (`double u`, `double v`) `const`
- `double evalBasisFunction` (`double u`, `double v`, `int u_deriv=0`, `int v_deriv=0`, `bool u_at_end=false`, `bool v_at_end=false`) `const`
- `Point eval` (`double u`, `double v`, `int u_deriv=0`, `int v_deriv=0`, `bool u_at_end=false`, `bool v_at_end=false`) `const`
- `void evalpos` (`double u`, `double v`, `double pos[]`)
- `void evalBasisGridDer` (`int nmb_der`, `const std::vector< double > &par1`, `const std::vector< double > &par2`, `std::vector< double > &derivs`) `const`
- `void evalBasisLineDer` (`int nmb_der`, `Direction2D d`, `const std::vector< double > &parval`, `std::vector< double > &derivs`) `const`
- `Point & coefTimesGamma` ()
- `const Point & coefTimesGamma` () `const`
- `Point Coef` ()
- `const Point Coef` () `const`
- `double & gamma` ()
- `const double & gamma` () `const`
- `double & weight` ()
- `const double & weight` () `const`
- `const bool rational` () `const`
- `const int dimension` () `const`
- `const std::vector< int > & kvec` (`Direction2D d`) `const`
- `std::vector< int > & kvec` (`Direction2D d`)

- `const int degree (Direction2D d) const`
- `const int suppMin (Direction2D d) const`  
*Get the index to the knot that defines the start (end) of the LRBSpline2D's support.*
- `const int suppMax (Direction2D d) const`
- `double umin () const`  
*Information about the domain covered by this B-spline.*
- `double umax () const`
- `double vmin () const`
- `double vmax () const`
- `int coefFixed () const`
- `void setFixCoef (int coef_fixed)`
- `int endmult_u (bool atstart) const`
- `int endmult_v (bool atstart) const`
- `bool coversCorner (int u_ix, int v_ix) const`
- `Point getGrevilleParameter () const`
- `double getGrevilleParameter (Direction2D d) const`
- `bool overlaps (Element2D *el) const`
- `bool overlaps (double domain[]) const`
- `bool addSupport (Element2D *el)`
- `void removeSupport (Element2D *el)`
- `std::vector< Element2D * >::iterator supportedElementBegin ()`
- `std::vector< Element2D * >::iterator supportedElementEnd ()`
- `std::vector< Element2D * > getExtendedSupport ()`
- `std::vector< Element2D * > getMinimalExtendedSupport ()`
- `const std::vector< Element2D * > & supportedElements ()`
- `void setSupport (std::vector< Element2D * > elements)`
- `bool hasSupportedElement (Element2D *el)`
- `int nmbSupportedElements ()`
- `void setMesh (const Mesh2D *mesh)`
- `const Mesh2D * getMesh ()`
- `void subtractKnotIdx (int u_del, int v_del)`
- `bool operator< (const LRBSpline2D &rhs) const`
- `bool operator== (const LRBSpline2D &rhs) const`
- `void setCoefAndGamma (Point &coef, double gamma)`
- `void reverseParameterDirection (bool dir_is_u)`
- `void swapParameterDirection ()`
- `std::vector< double > unitSquareBernsteinBasis (double start_u, double stop_u, double start_v, double stop_v) const`

### 29.274.1 Detailed Description

Definition at line 66 of file LRBSpline2D.h.

### 29.274.2 Constructor & Destructor Documentation

#### 29.274.2.1 Go::LRBSpline2D::LRBSpline2D ( ) [inline]

Constructor to create an empty (invalid) LRBSpline2D.

Definition at line 76 of file LRBSpline2D.h.



29.274.2.2 `template<typename Iterator > Go::LRBSpline2D::LRBSpline2D ( const Point & c_g, double weight, int deg_u, int deg_v, Iterator kvec_u_start, Iterator kvec_v_start, double gamma, const Mesh2D * mesh, bool rational = false ) [inline]`

Definition at line 80 of file LRBSpline2D.h.

29.274.2.3 `Go::LRBSpline2D::LRBSpline2D ( const LRBSpline2D & rhs )`

Copy constructor.

29.274.2.4 `Go::LRBSpline2D::~~LRBSpline2D ( ) [inline]`

Definition at line 110 of file LRBSpline2D.h.

### 29.274.3 Member Function Documentation

29.274.3.1 `bool Go::LRBSpline2D::addSupport ( Element2D * el )`

29.274.3.2 `Point Go::LRBSpline2D::Coef ( ) [inline]`

Definition at line 203 of file LRBSpline2D.h.

29.274.3.3 `const Point Go::LRBSpline2D::Coef ( ) const [inline]`

Definition at line 204 of file LRBSpline2D.h.

29.274.3.4 `int Go::LRBSpline2D::coefFixed ( ) const [inline]`

Definition at line 255 of file LRBSpline2D.h.

29.274.3.5 `Point& Go::LRBSpline2D::coefTimesGamma ( ) [inline]`

Definition at line 200 of file LRBSpline2D.h.

29.274.3.6 `const Point& Go::LRBSpline2D::coefTimesGamma ( ) const [inline]`

Definition at line 201 of file LRBSpline2D.h.

29.274.3.7 `bool Go::LRBSpline2D::coversCorner ( int u_ix, int v_ix ) const [inline]`

Definition at line 273 of file LRBSpline2D.h.

29.274.3.8 `const int Go::LRBSpline2D::degree ( Direction2D d ) const [inline]`

Definition at line 229 of file LRBSpline2D.h.

29.274.3.9 `const int Go::LRBSpline2D::dimension ( ) const [inline]`

Definition at line 219 of file LRBSpline2D.h.

29.274.3.10 `int Go::LRBSpline2D::endmult_u ( bool atstart ) const`

29.274.3.11 `int Go::LRBSpline2D::endmult_v ( bool atstart ) const`

29.274.3.12 `Point Go::LRBSpline2D::eval ( double u, double v, int u_deriv = 0, int v_deriv = 0, bool u_at_end = false, bool v_at_end = false ) const [inline]`

Definition at line 151 of file LRBSpline2D.h.

29.274.3.13 `double Go::LRBSpline2D::evalBasisFunc ( double u, double v ) const`

29.274.3.14 `double Go::LRBSpline2D::evalBasisFunction ( double u, double v, int u_deriv = 0, int v_deriv = 0, bool u_at_end = false, bool v_at_end = false ) const`

Similar to 'eval' below, but returns the value of the [LRBSpline2D](#)'s underlying *basis* function, rather than the function value itself. (In other words, the basis function's value is not multiplied by the spline coefficient, nor gamma, before it is returned.)

29.274.3.15 `void Go::LRBSpline2D::evalBasisGridDer ( int nmb_der, const std::vector< double > & par1, const std::vector< double > & par2, std::vector< double > & derivs ) const`

Compute a number of derivatives of the LRBSpline in a grid of parameter values specified in the two parameter directions. The sequence of the output is: du for all points, then dv, duu, duv, dvv, ... The position of the basis function is NOT stored. Rationals are NOT handled

29.274.3.16 `void Go::LRBSpline2D::evalBasisLineDer ( int nmb_der, Direction2D d, const std::vector< double > & parval, std::vector< double > & derivs ) const`

29.274.3.17 `void Go::LRBSpline2D::evalpos ( double u, double v, double pos[] ) [inline]`

Evaluate position only, array of correct size is expected as input. No size checking

Definition at line 173 of file LRBSpline2D.h.

29.274.3.18 `double& Go::LRBSpline2D::gamma ( ) [inline]`

Definition at line 208 of file LRBSpline2D.h.

29.274.3.19 `const double& Go::LRBSpline2D::gamma ( ) const [inline]`

Definition at line 209 of file LRBSpline2D.h.

29.274.3.20 `std::vector<Element2D*> Go::LRBSpline2D::getExtendedSupport ( )`

29.274.3.21 `Point Go::LRBSpline2D::getGrevilleParameter ( ) const`

29.274.3.22 `double Go::LRBSpline2D::getGrevilleParameter ( Direction2D d ) const`

29.274.3.23 `const Mesh2D* Go::LRBSpline2D::getMesh ( ) [inline]`

Definition at line 308 of file LRBSpline2D.h.

29.274.3.24 `std::vector<Element2D*> Go::LRBSpline2D::getMinimalExtendedSupport ( )`

29.274.3.25 `bool Go::LRBSpline2D::hasSupportedElement ( Element2D * el )`

29.274.3.26 `const std::vector<int>& Go::LRBSpline2D::kvec ( Direction2D d ) const [inline]`

Definition at line 225 of file LRBSpline2D.h.

29.274.3.27 `std::vector<int>& Go::LRBSpline2D::kvec ( Direction2D d ) [inline]`

Definition at line 226 of file LRBSpline2D.h.

29.274.3.28 `int Go::LRBSpline2D::nmbSupportedElements ( ) [inline]`

Definition at line 301 of file LRBSpline2D.h.

29.274.3.29 `bool Go::LRBSpline2D::operator< ( const LRBSpline2D & rhs ) const`

29.274.3.30 `bool Go::LRBSpline2D::operator==( const LRBSpline2D & rhs ) const`

29.274.3.31 `bool Go::LRBSpline2D::overlaps ( Element2D * el ) const`

29.274.3.32 `bool Go::LRBSpline2D::overlaps ( double domain[ ] ) const`

29.274.3.33 `const bool Go::LRBSpline2D::rational ( ) const [inline]`

Definition at line 215 of file LRBSpline2D.h.

29.274.3.34 `virtual void Go::LRBSpline2D::read ( std::istream & is )` [virtual]

Read the [LRBSpline2D](#) from a stream.

Implements [Go::Streamable](#).

29.274.3.35 `void Go::LRBSpline2D::removeSupport ( Element2D * el )`

29.274.3.36 `void Go::LRBSpline2D::reverseParameterDirection ( bool dir_is_u )`

29.274.3.37 `void Go::LRBSpline2D::setCoefAndGamma ( Point & coef, double gamma )` [inline]

Definition at line 325 of file LRBSpline2D.h.

29.274.3.38 `void Go::LRBSpline2D::setFixCoef ( int coef_fixed )` [inline]

Definition at line 260 of file LRBSpline2D.h.

29.274.3.39 `void Go::LRBSpline2D::setMesh ( const Mesh2D * mesh )` [inline]

Definition at line 303 of file LRBSpline2D.h.

29.274.3.40 `void Go::LRBSpline2D::setSupport ( std::vector< Element2D * > elements )` [inline]

Definition at line 295 of file LRBSpline2D.h.

29.274.3.41 `void Go::LRBSpline2D::subtractKnotIdx ( int u_del, int v_del )`

29.274.3.42 `const int Go::LRBSpline2D::suppMax ( Direction2D d ) const` [inline]

Definition at line 235 of file LRBSpline2D.h.

29.274.3.43 `const int Go::LRBSpline2D::suppMin ( Direction2D d ) const` [inline]

Get the index to the knot that defines the start (end) of the [LRBSpline2D](#)'s support.

Definition at line 234 of file LRBSpline2D.h.

29.274.3.44 `std::vector<Element2D*>::iterator Go::LRBSpline2D::supportedElementBegin ( )`

29.274.3.45 `std::vector<Element2D*>::iterator Go::LRBSpline2D::supportedElementEnd ( )`

29.274.3.46 `const std::vector<Element2D*>& Go::LRBSpline2D::supportedElements ( )` [inline]

Definition at line 291 of file LRBSpline2D.h.

29.274.3.47 `void Go::LRBSpline2D::swap ( LRBSpline2D & rhs ) [inline]`

Swap the contents of two LRBSpline2Ds.

Definition at line 98 of file LRBSpline2D.h.

29.274.3.48 `void Go::LRBSpline2D::swapParameterDirection ( )`

29.274.3.49 `double Go::LRBSpline2D::umax ( ) const [inline]`

Definition at line 242 of file LRBSpline2D.h.

29.274.3.50 `double Go::LRBSpline2D::umin ( ) const [inline]`

Information about the domain covered by this B-spline.

Definition at line 238 of file LRBSpline2D.h.

29.274.3.51 `std::vector<double> Go::LRBSpline2D::unitSquareBernsteinBasis ( double start_u, double stop_u, double start_v, double stop_v ) const`

For a given rectangle inside one of the elements cut out by the knot lines, this B-spline is a bi-degree polynomial. After transforming the rectangle to the unit square, this polynomial times the coefficient times gamma can be expressed by the Bernstein basis. This function returns the coefficients for this expression.

#### Parameters

<code>start_u</code>	the minimum u-value of the given rectangle
<code>stop_u</code>	the maximum u-value of the given rectangle
<code>start_v</code>	the minimum v-value of the given rectangle
<code>stop_v</code>	the maximum v-value of the given rectangle

#### Returns

a vector of the coefficients of the control points  $p_{ij}$  in order  $p_{00}[0]$ ,  $p_{00}[1]$ , ...,  $p_{10}[0]$ , ...,  $p_{01}[0]$ , ...

29.274.3.52 `double Go::LRBSpline2D::vmax ( ) const [inline]`

Definition at line 250 of file LRBSpline2D.h.

29.274.3.53 `double Go::LRBSpline2D::vmin ( ) const [inline]`

Definition at line 246 of file LRBSpline2D.h.

29.274.3.54 `double& Go::LRBSpline2D::weight ( ) [inline]`

Definition at line 212 of file LRBSpline2D.h.

29.274.3.55 `const double& Go::LRBSpline2D::weight ( ) const [inline]`

Definition at line 213 of file LRBSpline2D.h.

29.274.3.56 `virtual void Go::LRBSpline2D::write ( std::ostream & os ) const [virtual]`

Write the [LRBSpline2D](#) to a stream.

Implements [Go::Streamable](#).

The documentation for this class was generated from the following file:

- [Irsplines2D/include/GoTools/Irsplines2D/LRBSpline2D.h](#)

## 29.275 Go::LRSplineEvalGrid Class Reference

```
#include <LRSplineEvalGrid.h>
```

### Public Types

- typedef [double param\\_float\\_type](#)

### Public Member Functions

- [LRSplineEvalGrid \(\)](#)
- [LRSplineEvalGrid \(LRSplineSurface &lr\\_spline\)](#)
- [std::vector< Element2D >::iterator elements\\_begin \(\)](#)
- [std::vector< Element2D >::iterator elements\\_end \(\)](#)
- [void evaluate \(Element2D &elem, double u, double v, double \\*res\) const](#)
- [int numElements \(\) const](#)
- [int dim \(\) const](#)
- [int orderU \(\) const](#)
- [int orderV \(\) const](#)
- [void low \(const Element2D &e, double &u, double &v\)](#)
- [void high \(const Element2D &e, double &u, double &v\)](#)
- [void testCoefComputation \(\)](#)
- [void computeBezCoefs \(int dim, const double \\*points, int orderU, int orderV, double \\*coefs\)](#)
- [template<class V > void evaluateGrid \(V &element, double \\*points\)](#)
- [RectDomain origDom \(\)](#)

### 29.275.1 Detailed Description

Definition at line 70 of file LRSplineEvalGrid.h.

### 29.275.2 Member Typedef Documentation

#### 29.275.2.1 typedef double Go::LRSplineEvalGrid::param\_float\_type

Definition at line 77 of file LRSplineEvalGrid.h.

### 29.275.3 Constructor & Destructor Documentation

#### 29.275.3.1 Go::LRSplineEvalGrid::LRSplineEvalGrid ( )

#### 29.275.3.2 Go::LRSplineEvalGrid::LRSplineEvalGrid ( LRSplineSurface & lr\_spline )

### 29.275.4 Member Function Documentation

#### 29.275.4.1 void Go::LRSplineEvalGrid::computeBezCoefs ( int dim, const double \* points, int orderU, int orderV, double \* coefs ) [inline]

Definition at line 186 of file LRSplineEvalGrid.h.

#### 29.275.4.2 int Go::LRSplineEvalGrid::dim ( ) const [inline]

Definition at line 148 of file LRSplineEvalGrid.h.

#### 29.275.4.3 std::vector<Element2D>::iterator Go::LRSplineEvalGrid::elements\_begin ( ) [inline]

Definition at line 83 of file LRSplineEvalGrid.h.

#### 29.275.4.4 std::vector<Element2D>::iterator Go::LRSplineEvalGrid::elements\_end ( ) [inline]

Definition at line 88 of file LRSplineEvalGrid.h.

#### 29.275.4.5 void Go::LRSplineEvalGrid::evaluate ( Element2D & elem, double u, double v, double \* res ) const [inline]

Definition at line 93 of file LRSplineEvalGrid.h.

#### 29.275.4.6 template<class V > void Go::LRSplineEvalGrid::evaluateGrid ( V & element, double \* points ) [inline]

Definition at line 245 of file LRSplineEvalGrid.h.

29.275.4.7 void Go::LRSplineEvalGrid::high ( const Element2D & e, double & u, double & v ) [inline]

Definition at line 173 of file LRSplineEvalGrid.h.

29.275.4.8 void Go::LRSplineEvalGrid::low ( const Element2D & e, double & u, double & v ) [inline]

Definition at line 163 of file LRSplineEvalGrid.h.

29.275.4.9 int Go::LRSplineEvalGrid::numElements ( ) const [inline]

Definition at line 143 of file LRSplineEvalGrid.h.

29.275.4.10 int Go::LRSplineEvalGrid::orderU ( ) const [inline]

Definition at line 153 of file LRSplineEvalGrid.h.

29.275.4.11 int Go::LRSplineEvalGrid::orderV ( ) const [inline]

Definition at line 158 of file LRSplineEvalGrid.h.

29.275.4.12 RectDomain Go::LRSplineEvalGrid::origDom ( ) [inline]

Definition at line 274 of file LRSplineEvalGrid.h.

29.275.4.13 void Go::LRSplineEvalGrid::testCoefComputation ( )

The documentation for this class was generated from the following file:

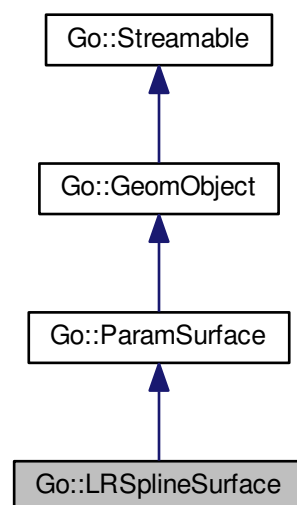
- [Irsplines2D/include/GoTools/Irsplines2D/LRSplineEvalGrid.h](#)



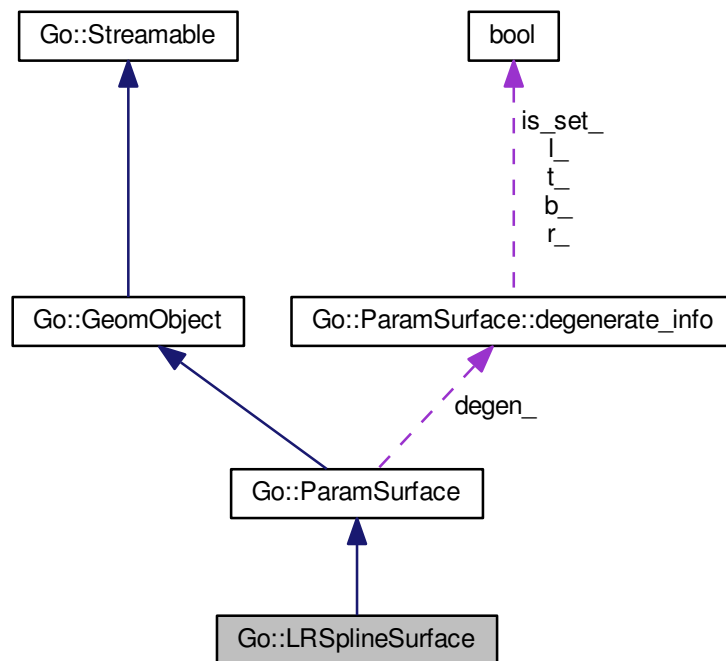
## 29.276 Go::LRSplineSurface Class Reference

```
#include <LRSplineSurface.h>
```

Inheritance diagram for Go::LRSplineSurface:



Collaboration diagram for Go::LRSplineSurface:



## Classes

- struct [BSKey](#)
- struct [double\\_pair\\_hash](#)
- struct [ElemKey](#)
- struct [Refinement2D](#)

## Public Types

- typedef `std::map< BSKey, std::unique_ptr< LRBSpline2D > >` [BSplineMap](#)
- typedef `std::map< ElemKey, std::unique_ptr< Element2D > >` [ElementMap](#)

## Public Member Functions

- `template<typename KnotIterator, typename CoeffIterator >`  
[LRSplineSurface](#) (int deg\_u, int deg\_v, int coefs\_u, int coefs\_v, int [dimension](#), KnotIterator knotvals\_u\_start, KnotIterator knotvals\_v\_start, CoeffIterator coefs\_start, [double](#) knot\_tol=1.0e-8)
- `template<typename KnotIterator >`  
[LRSplineSurface](#) (int deg\_u, int deg\_v, int coefs\_u, int coefs\_v, int [dimension](#), KnotIterator knotvals\_u\_start, KnotIterator knotvals\_v\_start, [double](#) knot\_tol=1.0e-8)
- [LRSplineSurface](#) ([SplineSurface](#) \*surf, [double](#) knot\_tol)
- [LRSplineSurface](#) ()
- [LRSplineSurface](#) (const [LRSplineSurface](#) &rhs)

- virtual `~LRSplineSurface ()`  
*Virtual destructor, enables safe inheritance.*
- `const LRSplineSurface & operator= (const LRSplineSurface &other)`
- `LRSplineSurface (std::istream &is)`
- void `swap (LRSplineSurface &rhs)`
- virtual void `read (std::istream &is)`
- virtual void `write (std::ostream &os) const`
- virtual `ClassType instanceType () const`  
*Return the class type identifier of a given, derived instance of `GeomObject`.*
- virtual `LRSplineSurface * clone () const`
- virtual `SplineSurface * asSplineSurface ()`
- virtual `BoundingBox boundingBox () const`  
*Return the object's bounding box.*
- virtual `int dimension () const`  
*Return the dimension of the space in which the object lies (usually 2 or 3)*
- virtual `const RectDomain & parameterDomain () const`
- virtual `RectDomain containingDomain () const`
- virtual `bool inDomain (double u, double v, double eps=1.0e-4) const`  
*Check if a parameter pair lies inside the domain of this surface.*
- virtual `int inDomain2 (double u, double v, double eps=1.0e-4) const`
- virtual `bool onBoundary (double u, double v, double eps=1.0e-4) const`  
*Check if a parameter pair lies at the boundary of this surface.*
- virtual `Point closestInDomain (double u, double v) const`
- virtual `CurveLoop outerBoundaryLoop (double degenerate_epsilon=DEFAULT_SPACE_EPSILON) const`
- virtual `std::vector< CurveLoop > allBoundaryLoops (double degenerate_epsilon=DEFAULT_SPACE_EPSILON) const`
- `SplineCurve * edgeCurve (int edge_num) const`
- virtual void `point (Point &pt, double upar, double vpar) const`
- void `point (Point &pt, double upar, double vpar, Element2D *elem) const`
- virtual void `point (std::vector< Point > &pts, double upar, double vpar, int derivs, bool u_from_right=true, bool v_from_right=true, double resolution=1.0e-12) const`
- void `point (std::vector< Point > &pts, double upar, double vpar, int derivs, Element2D *elem, bool u_from_right=true, bool v_from_right=true, double resolution=1.0e-12) const`
- void `closestPoint (const Point &pt, double &clo_u, double &clo_v, Point &clo_pt, double &clo_dist, double epsilon, int maxiter, Element2D *elem=NULL, const RectDomain *rd=NULL, double *seed=NULL) const`
- virtual void `evalGrid (int num_u, int num_v, double umin, double umax, double vmin, double vmax, std::vector< double > &points, double nodata_val=-9999) const`
- virtual `double startparam_u () const`
- virtual `double startparam_v () const`
- virtual `double endparam_u () const`
- virtual `double endparam_v () const`
- virtual void `normal (Point &n, double upar, double vpar) const`
- void `normal (Point &n, double upar, double vpar, Element2D *elem) const`
- virtual `DirectionCone normalCone () const`
- virtual `DirectionCone tangentCone (bool paddir_is_u) const`
- virtual `CompositeBox compositeBox () const`
- `LRSplineSurface * subSurface (double from_upar, double from_vpar, double to_upar, double to_vpar, double fuzzy) const`
- virtual `std::vector< shared_ptr< ParamSurface > > subSurfaces (double from_upar, double from_vpar, double to_upar, double to_vpar, double fuzzy=DEFAULT_PARAMETER_EPSILON) const`
- virtual `LRSplineSurface * mirrorSurface (const Point &pos, const Point &norm) const`  
*Mirror a surface around a specified plane.*
- virtual void `closestBoundaryPoint (const Point &pt, double &clo_u, double &clo_v, Point &clo_pt, double &clo_dist, double epsilon, const RectDomain *rd=NULL, double *seed=0) const`

- virtual void [getBoundaryInfo](#) ([Point](#) &pt1, [Point](#) &pt2, [double](#) epsilon, [SplineCurve](#) \*&cv, [SplineCurve](#) \*&crosscv, [double](#) knot\_tol=1e-05) const
- void [getBoundaryInfo](#) ([double](#) par1, [double](#) par2, int bindex, [SplineCurve](#) \*&cv, [SplineCurve](#) \*&crosscv, [double](#) knot\_tol) const
- void [getBoundaryIdx](#) ([Point](#) &pt1, [Point](#) &pt2, [double](#) epsilon, int &bindex, [double](#) &par1, [double](#) &par2, [double](#) knot\_tol=1e-05) const
- virtual void [turnOrientation](#) ()
  - Turns the direction of the normal of the surface.*
- virtual void [swapParameterDirection](#) ()
  - Swaps the two parameter directions.*
- virtual void [reverseParameterDirection](#) ([bool](#) direction\_is\_u)
- virtual void [setParameterDomain](#) ([double](#) u1, [double](#) u2, [double](#) v1, [double](#) v2)
- virtual [double](#) [area](#) ([double](#) tol) const
- virtual void [getCornerPoints](#) (std::vector< std::pair< [Point](#), [Point](#) > > &corners) const
- [SplineCurve](#) \* [constParamCurve](#) ([double](#) parameter, [bool](#) paddir\_is\_u) const
- void [constParamCurve](#) ([double](#) parameter, [bool](#) paddir\_is\_u, [SplineCurve](#) \*&cv, [SplineCurve](#) \*&crosscv) const
- virtual std::vector< shared\_ptr< [ParamCurve](#) > > [constParamCurves](#) ([double](#) parameter, [bool](#) paddir\_is\_u) const
- [bool](#) [isDegenerate](#) ([bool](#) &b, [bool](#) &r, [bool](#) &t, [bool](#) &l, [double](#) tolerance) const
- virtual void [getDegenerateCorners](#) (std::vector< [Point](#) > &deg\_corners, [double](#) tol) const
  - Check for parallel and anti parallel partial derivatives in surface corners.*
- virtual [double](#) [nextSegmentVal](#) (int dir, [double](#) par, [bool](#) forward, [double](#) tol) const
- [Point](#) operator() ([double](#) u, [double](#) v, int u\_deriv=0, int v\_deriv=0) const
- [Point](#) operator() ([double](#) u, [double](#) v, int u\_deriv, int v\_deriv, [Element2D](#) \*elem) const
- [double](#) [paramMin](#) ([Direction2D](#) d) const
- [double](#) [paramMax](#) ([Direction2D](#) d) const
- int [degree](#) ([Direction2D](#) d) const
- const [Mesh2D](#) & [mesh](#) () const
- int [numBasisFunctions](#) () const
- int [numElements](#) () const
- void [computeBasis](#) ([double](#) param\_u, [double](#) param\_v, [BasisPtsSf](#) &result, int iEl=-1) const
- void [computeBasis](#) ([double](#) param\_u, [double](#) param\_v, [BasisDerivsSf](#) &result, int iEl=-1) const
- void [computeBasis](#) ([double](#) param\_u, [double](#) param\_v, [BasisDerivsSf2](#) &result, int iEl=-1) const
- void [computeBasis](#) ([double](#) param\_u, [double](#) param\_v, std::vector< std::vector< [double](#) > > &result, int derivs=0, int iEl=-1) const
- int [getElementContaining](#) ([double](#) u, [double](#) v) const
- [Element2D](#) \* [coveringElement](#) ([double](#) u, [double](#) v) const
- void [constructElementMesh](#) (std::vector< [Element2D](#) \* > &elements) const
- std::vector< [LRBSpline2D](#) \* > [basisFunctionsWithSupportAt](#) ([double](#) u, [double](#) v) const
- [BSplineMap](#)::const\_iterator [basisFunctionsBegin](#) () const
- [BSplineMap](#)::const\_iterator [basisFunctionsEnd](#) () const
- [BSplineMap](#)::iterator [basisFunctionsBeginNonconst](#) ()
- [BSplineMap](#)::iterator [basisFunctionsEndNonconst](#) ()
- [BSplineMap](#)::iterator [bsplineFromDomain](#) ([double](#) start\_u, [double](#) start\_v, [double](#) end\_u, [double](#) end\_v, int startmult\_u, int startmult\_v, int endmult\_u, int endmult\_v)
- std::vector< [LRBSpline2D](#) \* > [getBoundaryBsplines](#) ([Direction2D](#) d, [bool](#) atstart)
- [bool](#) [isFullTensorProduct](#) () const
- [double](#) [getKnotTol](#) ()
  - Tolerance for equality of knots.*
- void [refine](#) ([Direction2D](#) d, [double](#) fixed\_val, [double](#) start, [double](#) end, int mult=1, [bool](#) absolute=false)
- void [refine](#) (const [Refinement2D](#) &ref, [bool](#) absolute=false)
- void [refine](#) (const std::vector< [Refinement2D](#) > &refs, [bool](#) absolute=false)
- void [refineBasisFunction](#) (int index)
- void [refineBasisFunction](#) (const std::vector< int > &indices)

- void `refineElement` (int index)
- void `refineElement` (const std::vector< int > &indices)
- void `setCoef` (const Point &value, const LRBSpline2D \*target)
- void `setCoefTimesGamma` (const Point &value, const LRBSpline2D \*target)
- void `setCoef` (const Point &value, int umin\_ix, int vmin\_ix, int umax\_ix, int vmax\_ix, int u\_mult=1, int v\_mult=1)
- void `expandToFullTensorProduct` ()
- void `addSurface` (const LRSplineSurface &other\_sf, double fac=1.0)
- void `to3D` ()
- bool `rational` () const
- void `translate` (const Point &vec)
- ElementMap::const\_iterator `elementsBegin` () const
- ElementMap::const\_iterator `elementsEnd` () const
- void `setCurrentElement` (Element2D \*curr\_el)
- LineCloud `getElementBds` (int num\_pts=5) const

### Static Public Member Functions

- static BSKey `generate_key` (const LRBSpline2D &b, const Mesh2D &m)
- static BSKey `generate_key` (const LRBSpline2D &b)
- static ElemKey `generate_key` (const double &, const double &)
- static ClassType `classType` ()

### Additional Inherited Members

#### 29.276.1 Detailed Description

Definition at line 61 of file LRSplineSurface.h.

#### 29.276.2 Member Typedef Documentation

29.276.2.1 typedef std::map<BSKey, std::unique\_ptr<LRBSpline2D> > Go::LRsplineSurface::BSplineMap

Definition at line 114 of file LRSplineSurface.h.

29.276.2.2 typedef std::map<ElemKey, std::unique\_ptr<Element2D> > Go::LRsplineSurface::ElementMap

Definition at line 143 of file LRSplineSurface.h.

#### 29.276.3 Constructor & Destructor Documentation

29.276.3.1 template<typename KnotIterator , typename CoefIterator > Go::LRsplineSurface::LRsplineSurface ( int deg\_u, int deg\_v, int coefs\_u, int coefs\_v, int dimension, KnotIterator knotvals\_u\_start, KnotIterator knotvals\_v\_start, CoefIterator coefs\_start, double knot\_tol = 1.0e-8 )

Definition at line 244 of file LRSplineSurface\_impl.h.

```
29.276.3.2 template<typename KnotIterator > Go::LRSplineSurface::LRSplineSurface (int deg_u, int deg_v, int coefs_u, int
 coefs_v, int dimension, KnotIterator knotvals_u_start, KnotIterator knotvals_v_start, double knot_tol = 1.0e-8
)
```

Definition at line 286 of file LRSplineSurface\_impl.h.

```
29.276.3.3 Go::LRSplineSurface::LRSplineSurface (SplineSurface * surf, double knot_tol)
```

```
29.276.3.4 Go::LRSplineSurface::LRSplineSurface () [inline]
```

Definition at line 184 of file LRSplineSurface.h.

```
29.276.3.5 Go::LRSplineSurface::LRSplineSurface (const LRSplineSurface & rhs)
```

```
29.276.3.6 virtual Go::LRSplineSurface::~~LRSplineSurface () [virtual]
```

Virtual destructor, enables safe inheritance.

```
29.276.3.7 Go::LRSplineSurface::LRSplineSurface (std::istream & is) [inline]
```

Definition at line 200 of file LRSplineSurface.h.

## 29.276.4 Member Function Documentation

```
29.276.4.1 void Go::LRSplineSurface::addSurface (const LRSplineSurface & other_sf, double fac = 1.0)
```

```
29.276.4.2 virtual std::vector<CurveLoop> Go::LRSplineSurface::allBoundaryLoops (double degenerate_epsilon =
 DEFAULT_SPACE_EPSILON) const [virtual]
```

Returns the anticlockwise outer boundary loop of the surface, together with clockwise loops of any interior boundaries, such that the surface always is 'to the left of' the loops.

### Parameters

<i>degenerate_epsilon</i>	edges whose length is smaller than this value are ignored.
---------------------------	------------------------------------------------------------

### Returns

a vector containing CurveLoops. The first of these describe the outer boundary of the surface (clockwise), whereas the others describe boundaries of interior holes (clockwise).

Implements [Go::ParamSurface](#).

```
29.276.4.3 virtual double Go::LRSplineSurface::area (double tol) const [virtual]
```

Compute the total area of this surface up to some tolerance

## Parameters

<i>tol</i>	the relative tolerance when approximating the area, i.e. stop iteration when error becomes smaller than $tol/(surface\ area)$
------------	-------------------------------------------------------------------------------------------------------------------------------

## Returns

the area calculated

Implements [Go::ParamSurface](#).

**29.276.4.4** virtual **SplineSurface\*** **Go::LRSplineSurface::asSplineSurface** ( ) `[virtual]`

Return the spline surface represented by this surface, if any The user may get the spline surface lying in the (refined) regular grid by calling the function [expandToFullTensorProduct\(\)](#).

Reimplemented from [Go::ParamSurface](#).

**29.276.4.5** **BSplineMap::const\_iterator** **Go::LRSplineSurface::basisFunctionsBegin** ( ) `const [inline]`

Definition at line 546 of file LRSplineSurface.h.

**29.276.4.6** **BSplineMap::iterator** **Go::LRSplineSurface::basisFunctionsBeginNonconst** ( ) `[inline]`

Definition at line 550 of file LRSplineSurface.h.

**29.276.4.7** **BSplineMap::const\_iterator** **Go::LRSplineSurface::basisFunctionsEnd** ( ) `const [inline]`

Definition at line 547 of file LRSplineSurface.h.

**29.276.4.8** **BSplineMap::iterator** **Go::LRSplineSurface::basisFunctionsEndNonconst** ( ) `[inline]`

Definition at line 551 of file LRSplineSurface.h.

**29.276.4.9** **std::vector<LRBSpline2D\*>** **Go::LRSplineSurface::basisFunctionsWithSupportAt** ( **double u**, **double v** ) `const`

**29.276.4.10** virtual **BoundingBox** **Go::LRSplineSurface::boundingBox** ( ) `const [virtual]`

Return the object's bounding box.

Implements [Go::GeomObject](#).

29.276.4.11 `BSplineMap::iterator Go::LRSplineSurface::bsplineFromDomain ( double start_u, double start_v, double end_u, double end_v, int startmult_u, int startmult_v, int endmult_u, int endmult_v )`

29.276.4.12 `static ClassType Go::LRSplineSurface::classType ( ) [inline],[static]`

Definition at line 216 of file LRSplineSurface.h.

29.276.4.13 `virtual LRSplineSurface* Go::LRSplineSurface::clone ( ) const [inline],[virtual]`

make a clone of this surface and return a pointer to it (user is responsible for clearing up memory afterwards).

#### Returns

pointer to cloned object

Implements [Go::ParamSurface](#).

Definition at line 219 of file LRSplineSurface.h.

29.276.4.14 `virtual void Go::LRSplineSurface::closestBoundaryPoint ( const Point & pt, double & clo_u, double & clo_v, Point & clo_pt, double & clo_dist, double epsilon, const RectDomain * rd = NULL, double * seed = 0 ) const [virtual]`

Iterates to the closest point to pt on the boundary of the surface.

#### See also

[closestPoint\(\)](#)

Implements [Go::ParamSurface](#).

29.276.4.15 `virtual Point Go::LRSplineSurface::closestInDomain ( double u, double v ) const [virtual]`

Fetch the parameter value in the parameter domain of the surface closest to the parameter pair (u,v)

Implements [Go::ParamSurface](#).

29.276.4.16 `void Go::LRSplineSurface::closestPoint ( const Point & pt, double & clo_u, double & clo_v, Point & clo_pt, double & clo_dist, double epsilon, int maxiter, Element2D * elem = NULL, const RectDomain * rd = NULL, double * seed = NULL ) const`

Closest point iteration taking benefit from information about an element in which to start searching



29.276.4.17 virtual **CompositeBox** Go::LRSplineSurface::compositeBox ( ) const [virtual]

Creates a composite box enclosing the surface. The composite box consists of an inner and an edge box. The inner box is supposed to be made from the interior of the surface, while the edge box is made from the boundary curves. The default implementation simply makes both boxes identical to the regular bounding box.

Returns

the [CompositeBox](#) of the surface, as specified above

Reimplemented from [Go::ParamSurface](#).

29.276.4.18 void Go::LRSplineSurface::computeBasis ( double *param\_u*, double *param\_v*, BasisPtsSf & *result*, int *iEI* = -1 ) const

29.276.4.19 void Go::LRSplineSurface::computeBasis ( double *param\_u*, double *param\_v*, BasisDerivsSf & *result*, int *iEI* = -1 ) const

29.276.4.20 void Go::LRSplineSurface::computeBasis ( double *param\_u*, double *param\_v*, BasisDerivsSf2 & *result*, int *iEI* = -1 ) const

29.276.4.21 void Go::LRSplineSurface::computeBasis ( double *param\_u*, double *param\_v*, std::vector< std::vector< double >> & *result*, int *derivs* = 0, int *iEI* = -1 ) const

29.276.4.22 **SplineCurve\*** Go::LRSplineSurface::constParamCurve ( double *parameter*, bool *pardir\_is\_u* ) const

29.276.4.23 void Go::LRSplineSurface::constParamCurve ( double *parameter*, bool *pardir\_is\_u*, **SplineCurve** \*& *cv*, **SplineCurve** \*& *crosscv* ) const

Generate and return two **SplineCurves**, representing a constant parameter curve on the surface as well as its cross-tangent curve.

Parameters

<i>parameter</i>	value of the fixed parameter
<i>pardir_is_u</i>	'true' if the first parameter is the running parameter, 'false' otherwise.
<i>cv</i>	upon function completion, 'cv' will point to a newly constructed <a href="#">SplineCurve</a> representing the specified constant parameter curve. It is the user's responsibility to delete it when it is no longer needed.
<i>crosscv</i>	upon function completion, 'crosscv' will point to a newly constructed <a href="#">SplineCurve</a> representing the cross-tangent curve of the specified constant parameter curve. It is the user's responsibility to delete it when it is no longer needed.

29.276.4.24 virtual std::vector< shared\_ptr<**ParamCurve**>> Go::LRSplineSurface::constParamCurves ( double *parameter*, bool *pardir\_is\_u* ) const [virtual]

Get the curve(s) obtained by intersecting the surface with one of its constant parameter curves. For surfaces without holes, this will be the parameter curve itself; for surfaces with interior holes this may be a collection of several, disjoint

curves.

#### Parameters

<i>parameter</i>	parameter value for the constant parameter (either u or v)
<i>pardir_is_u</i>	specify whether the <i>moving</i> parameter (as opposed to the <i>constant</i> parameter) is the first ('true') or the second ('false') one.

#### Returns

a vector containing shared pointers to the obtained, newly constructed constant-parameter curves.

Implements [Go::ParamSurface](#).

29.276.4.25 `void Go::LRSplineSurface::constructElementMesh ( std::vector< Element2D * > & elements ) const`

29.276.4.26 `virtual RectDomain Go::LRSplineSurface::containingDomain ( ) const` `[virtual]`

Get a rectangular parameter domain that is guaranteed to contain the surface's [parameterDomain\(\)](#). It may be the same. There is no guarantee that this is the smallest domain containing the actual domain.

#### Returns

a [RectDomain](#) that is guaranteed to include the surface's total parameter domain.

Implements [Go::ParamSurface](#).

29.276.4.27 `Element2D* Go::LRSplineSurface::coveringElement ( double u, double v ) const`

29.276.4.28 `int Go::LRSplineSurface::degree ( Direction2D d ) const` `[inline]`

Definition at line 84 of file `LRSplineSurface_impl.h`.

29.276.4.29 `virtual int Go::LRSplineSurface::dimension ( ) const` `[inline],[virtual]`

Return the dimension of the space in which the object lies (usually 2 or 3)

Implements [Go::GeomObject](#).

Definition at line 233 of file `LRSplineSurface.h`.

29.276.4.30 `SplineCurve* Go::LRSplineSurface::edgeCurve ( int edge_num ) const`

29.276.4.31 `ElementMap::const_iterator Go::LRSplineSurface::elementsBegin ( ) const` `[inline]`

Definition at line 638 of file `LRSplineSurface.h`.

29.276.4.32 `ElementMap::const_iterator Go::LRSplineSurface::elementsEnd ( ) const [inline]`

Definition at line 639 of file `LRSplineSurface.h`.

29.276.4.33 `virtual double Go::LRSplineSurface::endparam_u ( ) const [virtual]`

Get the end value for the u-parameter

Returns

the end value for the u-parameter

29.276.4.34 `virtual double Go::LRSplineSurface::endparam_v ( ) const [virtual]`

Get the end value for the v-parameter

Returns

the end value for the v-parameter

29.276.4.35 `virtual void Go::LRSplineSurface::evalGrid ( int num_u, int num_v, double umin, double umax, double vmin, double vmax, std::vector< double > & points, double nodata_val = -9999 ) const [virtual]`

Evaluate points in a grid The nodata value is applicable for bounded surfaces and will not be used in this context

Reimplemented from [Go::ParamSurface](#).

29.276.4.36 `void Go::LRSplineSurface::expandToFullTensorProduct ( )`

29.276.4.37 `LRSplineSurface::BSKey Go::LRSplineSurface::generate_key ( const LRBSpline2D & b, const Mesh2D & m ) [inline],[static]`

Definition at line 152 of file `LRSplineSurface_impl.h`.

29.276.4.38 `LRSplineSurface::BSKey Go::LRSplineSurface::generate_key ( const LRBSpline2D & b ) [inline],[static]`

Definition at line 176 of file `LRSplineSurface_impl.h`.

29.276.4.39 `LRSplineSurface::ElemKey Go::LRSplineSurface::generate_key ( const double & umin, const double & vmin ) [inline],[static]`

Definition at line 218 of file `LRSplineSurface_impl.h`.

29.276.4.40 `std::vector<LRBSpline2D*> Go::LRSplineSurface::getBoundaryBsplines ( Direction2D d, bool atstart )`

29.276.4.41 `void Go::LRSplineSurface::getBoundaryIdx ( Point & pt1, Point & pt2, double epsilon, int & bindex, double & par1, double & par2, double knot_tol = 1e-05 ) const`

Given two points on the surface boundary, find the number of the corresponding boundary and the curve parameter of the closest points on this surface boundary.

Ordering of boundaries:

```

/// 1
/// - - - - -
/// | |
/// 2 | | 3
/// v | |
/// ^ - - - - -
/// |-> u 0
///

```

#### Parameters

<i>pt1</i>	point in geometry space
<i>pt2</i>	point in geometry space
<i>epsilon</i>	geometric tolerance
<i>bindex</i>	if the two points are on a common boundary, the index of the boundary, otherwise -1.
<i>par1</i>	if bindex is 0 or 1, the u parameter of pt1, otherwise the v parameter.
<i>par2</i>	if bindex is 0 or 1, the u parameter of pt2, otherwise the v parameter.
<i>knot_tol</i>	tolerance used when working with the knot-vector, to specify how close a parameter value must be to a knot in order to be considered 'on' the knot.

29.276.4.42 `virtual void Go::LRSplineSurface::getBoundaryInfo ( Point & pt1, Point & pt2, double epsilon, SplineCurve *& cv, SplineCurve *& crosscv, double knot_tol = 1e-05 ) const [virtual]`

Get the boundary curve segment between two points on the boundary, as well as the cross-tangent curve. If the given points are not positioned on the same boundary (within a certain tolerance), no curves will be created.

#### Parameters

<i>pt1</i>	the first point on the boundary, given by the user
<i>pt2</i>	the second point on the boundary, given by the user
<i>epsilon</i>	the tolerance used when determining whether the given points are lying on a boundary, and if they do, whether they both lie on the <i>same</i> boundary.
<i>cv</i>	upon return, this will point to a newly created curve representing the boundary curve between 'pt1' and 'pt2'. The user assumes ownership of this object and is responsible for its deletion. No curve is created if the given points are not found to lie on the same boundary.
<i>crosscv</i>	upon return, this will point to a newly created curve representing the cross-boundary curve between 'pt1' and 'pt2'. The user assumes ownership of this object and is responsible for its deletion. The direction is outwards from the surface. No curve is created if the given points are not found to lie on the same boundary.
<i>knot_tol</i>	tolerance used when working with the knot-vector, to specify how close a parameter value must be to a knot in order to be considered 'on' the knot.

Implements [Go::ParamSurface](#).

29.276.4.43 `void Go::LRSplineSurface::getBoundaryInfo ( double par1, double par2, int bindex, SplineCurve *& cv, SplineCurve *& crosscv, double knot_tol ) const`

29.276.4.44 `virtual void Go::LRSplineSurface::getCornerPoints ( std::vector< std::pair< Point, Point > > & corners ) const` [virtual]

Return surface corners, geometric and parametric points in that sequence

Implements [Go::ParamSurface](#).

29.276.4.45 `virtual void Go::LRSplineSurface::getDegenerateCorners ( std::vector< Point > & deg_corners, double tol ) const` [virtual]

Check for parallel and anti parallel partial derivatives in surface corners.

Implements [Go::ParamSurface](#).

29.276.4.46 `LineCloud Go::LRSplineSurface::getElementBds ( int num_pts = 5 ) const`

29.276.4.47 `int Go::LRSplineSurface::getElementContaining ( double u, double v ) const`

29.276.4.48 `double Go::LRSplineSurface::getKnotTol ( )` [inline]

Tolerance for equality of knots.

Definition at line 565 of file LRSplineSurface.h.

29.276.4.49 `virtual bool Go::LRSplineSurface::inDomain ( double u, double v, double eps = 1.0e-4 ) const` [virtual]

Check if a parameter pair lies inside the domain of this surface.

Implements [Go::ParamSurface](#).

29.276.4.50 `virtual int Go::LRSplineSurface::inDomain2 ( double u, double v, double eps = 1.0e-4 ) const` [virtual]

Check if a parameter pair lies inside the domain of this surface return value = 0: outside = 1: internal = 2: at the boundary

Implements [Go::ParamSurface](#).

29.276.4.51 `virtual ClassType Go::LRSplineSurface::instanceType ( ) const` [virtual]

Return the class type identifier of a given, derived instance of [GeomObject](#).

Implements [Go::GeomObject](#).

29.276.4.52 `bool Go::LRSplineSurface::isDegenerate ( bool & b, bool & r, bool & t, bool & l, double tolerance ) const` [virtual]

The order of the edge indicators (bottom, right, top, left) matches the edge\_number of [edgeCurve\(\)](#). Query whether any of the four boundary curves are degenerate (zero length) within a certain tolerance. In the below, we refer to 'u' as the first parameter and 'v' as the second.

## Parameters

<i>b</i>	'true' upon return of function if the boundary ( $v = v_{\min}$ ) is degenerate
<i>r</i>	'true' upon return of function if the boundary ( $v = v_{\max}$ ) is degenerate
<i>t</i>	'true' upon return of function if the boundary ( $u = u_{\min}$ ) is degenerate
<i>l</i>	'true' upon return of function if the boundary ( $u = u_{\max}$ ) is degenerate
<i>tolerance</i>	boundaries are considered degenerate if their length is shorter than this value, given by the user

## Returns

'true' if at least one boundary curve was found to be degenerate, 'false' otherwise.

Reimplemented from [Go::ParamSurface](#).

29.276.4.53 **bool** `Go::LRSplineSurface::isFullTensorProduct ( ) const`

29.276.4.54 **const Mesh2D&** `Go::LRSplineSurface::mesh ( ) const` `[inline]`

Definition at line 505 of file LRSplineSurface.h.

29.276.4.55 **virtual LRSplineSurface\*** `Go::LRSplineSurface::mirrorSurface ( const Point & pos, const Point & norm ) const` `[virtual]`

Mirror a surface around a specified plane.

Reimplemented from [Go::ParamSurface](#).

29.276.4.56 **virtual double** `Go::LRSplineSurface::nextSegmentVal ( int dir, double par, bool forward, double tol ) const` `[virtual]`

Determine the parameter value of the start of the 'next segment' from a parameter value, along a given parameter direction. A 'segment' is here defined as a parameter interval in which there will be no discontinuities in derivatives or other artifacts. For spline objects, a segment will typically be the interval between two consecutive, non-coincident knots.

## Parameters

<i>dir</i>	the parameter direction in which we search for the next segment (0 or 1)
<i>par</i>	the parameter value starting from which we search for the start value of the next segment
<i>forward</i>	define whether we shall move forward ('true') or backwards when searching along this parameter
<i>tol</i>	tolerance used for determining whether the 'par' is already located <i>on</i> the next segment value

## Returns

the value of the start value of the next segment (or the end of the previous segment, if we are moving backwards...)

Implements [Go::ParamSurface](#).

29.276.4.57 `virtual void Go::LRSplineSurface::normal ( Point & n, double upar, double vpar ) const` [virtual]

Evaluates the surface normal for a given parameter pair

Parameters

<i>n</i>	the computed normal will be written to this variable
<i>upar</i>	the first parameter
<i>vpar</i>	the second parameter

Implements [Go::ParamSurface](#).

29.276.4.58 `void Go::LRSplineSurface::normal ( Point & n, double upar, double vpar, Element2D * elem ) const`

29.276.4.59 `virtual DirectionCone Go::LRSplineSurface::normalCone ( ) const` [virtual]

Function that calls `normalCone(NormalConeMethod)` with `method = SederbergMeyers`. Needed because `normalCone()` is virtual! (Inherited from [ParamSurface](#)).

Returns

a [DirectionCone](#) (not necessarily the smallest) containing all normals to this surface.

Implements [Go::ParamSurface](#).

29.276.4.60 `int Go::LRSplineSurface::numBasisFunctions ( ) const` [inline]

Definition at line 508 of file `LRSplineSurface.h`.

29.276.4.61 `int Go::LRSplineSurface::numElements ( ) const` [inline]

Definition at line 510 of file `LRSplineSurface.h`.

29.276.4.62 `virtual bool Go::LRSplineSurface::onBoundary ( double u, double v, double eps = 1.0e-4 ) const` [virtual]

Check if a parameter pair lies at the boundary of this surface.

Implements [Go::ParamSurface](#).

29.276.4.63 `Point Go::LRSplineSurface::operator() ( double u, double v, int u_deriv = 0, int v_deriv = 0 ) const`

29.276.4.64 `Point Go::LRSplineSurface::operator() ( double u, double v, int u_deriv, int v_deriv, Element2D * elem ) const`

29.276.4.65 `const LRSplineSurface& Go::LRSplineSurface::operator= ( const LRSplineSurface & other )`

29.276.4.66 `virtual CurveLoop Go::LRSplineSurface::outerBoundaryLoop ( double degenerate_epsilon = DEFAULT_SPACE_EPSILON ) const` [virtual]

Returns the anticlockwise, outer boundary loop of the surface.

## Parameters

<i>degenerate_epsilon</i>	edges whose length is smaller than this value are ignored.
---------------------------	------------------------------------------------------------

## Returns

a [CurveLoop](#) describing the anticlockwise, outer boundary loop of the surface. A negative *degenerate\_epsilon* indicates that all curves, also the degenerate ones are wanted

Implements [Go::ParamSurface](#).

29.276.4.67 `virtual const RectDomain& Go::LRSplineSurface::parameterDomain ( ) const` [virtual]

Return the parameter domain of the surface. This may be a simple rectangular domain ([RectDomain](#)) or any other subclass of [Domain](#) (such as [GoCurveBoundedDomain](#), found in the `sisl_dependent` module).

## Returns

a [Domain](#) object describing the parametric domain of the surface

Implements [Go::ParamSurface](#).

29.276.4.68 `double Go::LRSplineSurface::paramMax ( Direction2D d ) const` [inline]

Definition at line 122 of file `LRSplineSurface_impl.h`.

29.276.4.69 `double Go::LRSplineSurface::paramMin ( Direction2D d ) const` [inline]

Definition at line 92 of file `LRSplineSurface_impl.h`.

29.276.4.70 `virtual void Go::LRSplineSurface::point ( Point & pt, double upar, double vpar ) const` [virtual]

Evaluates the surface's position for a given parameter pair.

## Parameters

<i>pt</i>	the result of the evaluation is written here
<i>upar</i>	the first parameter
<i>vpar</i>	the second parameter

Implements [Go::ParamSurface](#).

29.276.4.71 `void Go::LRSplineSurface::point ( Point & pt, double upar, double vpar, Element2D * elem ) const`



29.276.4.72 `virtual void Go::LRSplineSurface::point ( std::vector< Point > & pts, double upar, double vpar, int derivs, bool u_from_right = true, bool v_from_right = true, double resolution = 1.0e-12 ) const` [virtual]

Evaluates the surface's position and a certain number of derivatives for a given parameter pair.

#### Parameters

<i>pts</i>	the vector containing the evaluated values. Its size must be set by the user prior to calling this function, and should be equal to $(derivs+1) * (derivs+2) / 2$ . Upon completion of the function, its first entry is the surface's position at the given parameter pair. Then, if 'derivs' > 0, the two next entries will be the surface tangents along the first and second parameter direction. The next three entries are the second- and cross derivatives, in the order (du2, dudv, dv2), and similar for even higher derivatives.
<i>upar</i>	the first parameter
<i>vpar</i>	the second parameter
<i>derivs</i>	number of requested derivatives
<i>u_from_right</i>	specify whether derivatives along the first parameter are to be calculated from the right ('true', default) or from the left ('false')
<i>v_from_right</i>	specify whether derivatives along the second parameter are to be calculated from the right ('true', default) or from the left ('false')
<i>resolution</i>	tolerance used when determining whether parameters are located at special values of the parameter domain (in particular; knot values in case of spline objects).

Implements [Go::ParamSurface](#).

29.276.4.73 `void Go::LRSplineSurface::point ( std::vector< Point > & pts, double upar, double vpar, int derivs, Element2D * elem, bool u_from_right = true, bool v_from_right = true, double resolution = 1.0e-12 ) const`

29.276.4.74 `bool Go::LRSplineSurface::rational ( ) const`

29.276.4.75 `virtual void Go::LRSplineSurface::read ( std::istream & is )` [virtual]

read object from stream

#### Parameters

<i>is</i>	stream from which object is read
-----------	----------------------------------

Implements [Go::Streamable](#).

29.276.4.76 `void Go::LRSplineSurface::refine ( Direction2D d, double fixed_val, double start, double end, int mult = 1, bool absolute = false )`

29.276.4.77 `void Go::LRSplineSurface::refine ( const Refinement2D & ref, bool absolute = false )`

29.276.4.78 `void Go::LRSplineSurface::refine ( const std::vector< Refinement2D > & refs, bool absolute = false )`

29.276.4.79 void Go::LRSplineSurface::refineBasisFunction ( int *index* )

29.276.4.80 void Go::LRSplineSurface::refineBasisFunction ( const std::vector< int > & *indices* )

29.276.4.81 void Go::LRSplineSurface::refineElement ( int *index* )

29.276.4.82 void Go::LRSplineSurface::refineElement ( const std::vector< int > & *indices* )

29.276.4.83 virtual void Go::LRSplineSurface::reverseParameterDirection ( bool *direction\_is\_u* ) [virtual]

Reverses the direction of the basis in input direction.

#### Parameters

<i>direction_is_u</i>	if 'true', the first parameter direction will be reversed, otherwise, the second parameter direction will be reversed
-----------------------	-----------------------------------------------------------------------------------------------------------------------

Implements [Go::ParamSurface](#).

29.276.4.84 void Go::LRSplineSurface::setCoef ( const Point & *value*, const LRBSpline2D \* *target* )

29.276.4.85 void Go::LRSplineSurface::setCoef ( const Point & *value*, int *umin\_ix*, int *vmin\_ix*, int *umax\_ix*, int *vmax\_ix*, int *u\_mult* = 1, int *v\_mult* = 1 )

29.276.4.86 void Go::LRSplineSurface::setCoefTimesGamma ( const Point & *value*, const LRBSpline2D \* *target* )

29.276.4.87 void Go::LRSplineSurface::setCurrentElement ( Element2D \* *curr\_el* ) [inline]

Definition at line 642 of file LRSplineSurface.h.

29.276.4.88 virtual void Go::LRSplineSurface::setParameterDomain ( double *u1*, double *u2*, double *v1*, double *v2* ) [virtual]

set the parameter domain to a given rectangle

#### Parameters

<i>u1</i>	new min. value of first parameter span
<i>u2</i>	new max. value of first parameter span
<i>v1</i>	new min. value of second parameter span
<i>v2</i>	new max. value of second parameter span

Reimplemented from [Go::ParamSurface](#).

29.276.4.89 `virtual double Go::LRSplineSurface::startparam_u ( ) const [virtual]`

Get the start value for the u-parameter

Returns

the start value for the u-parameter

29.276.4.90 `virtual double Go::LRSplineSurface::startparam_v ( ) const [virtual]`

Get the start value for the v-parameter

Returns

the start value for the v-parameter

29.276.4.91 `LRSplineSurface* Go::LRSplineSurface::subSurface ( double from_upar, double from_vpar, double to_upar, double to_vpar, double fuzzy ) const`

29.276.4.92 `virtual std::vector<shared_ptr<ParamSurface>> Go::LRSplineSurface::subSurfaces ( double from_upar, double from_vpar, double to_upar, double to_vpar, double fuzzy = DEFAULT_PARAMETER_EPSILON ) const [virtual]`

Get the surface(s) obtained by cropping the parameter domain of this surface between given values for the first and second parameter. In general, for surfaces with no interior holes, the result will be *one* surface; however, for surfaces with interior holes, the result might be *several disjoint* surfaces.

Parameters

<i>from_upar</i>	lower value for the first parameter in the subdomain
<i>from_vpar</i>	lower value for the second parameter in the subdomain
<i>to_upar</i>	upper value for the first parameter in the subdomain
<i>to_vpar</i>	upper value for the second parameter in the subdomain
<i>fuzzy</i>	tolerance used when determining intersection with interior boundaries

Returns

a vector contained shared pointers to the obtained, newly constructed sub-surfaces.

Implements [Go::ParamSurface](#).

29.276.4.93 `void Go::LRSplineSurface::swap ( LRSplineSurface & rhs )`

29.276.4.94 `virtual void Go::LRSplineSurface::swapParameterDirection ( ) [virtual]`

Swaps the two parameter directions.

Implements [Go::ParamSurface](#).

29.276.4.95 `virtual DirectionCone Go::LRSplineSurface::tangentCone ( bool pardir_is_u ) const` `[virtual]`

Creates a [DirectionCone](#) covering all tangents to this surface along a given parameter direction.

#### Parameters

<code><i>pardir_is_u</i></code>	if 'true', then the <a href="#">DirectionCone</a> will be defined on basis of the surface's tangents along the first parameter direction. Otherwise the second parameter direction will be used.
---------------------------------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

#### Returns

a [DirectionCone](#) (not necessarily the smallest) containing all tangents to this surface along the specified parameter direction.

Implements [Go::ParamSurface](#).

29.276.4.96 `void Go::LRSplineSurface::to3D ( )`

29.276.4.97 `void Go::LRSplineSurface::translate ( const Point & vec )`

29.276.4.98 `virtual void Go::LRSplineSurface::turnOrientation ( )` `[virtual]`

Turns the direction of the normal of the surface.

Implements [Go::ParamSurface](#).

29.276.4.99 `virtual void Go::LRSplineSurface::write ( std::ostream & os ) const` `[virtual]`

write object to stream

#### Parameters

<code><i>os</i></code>	stream to which object is written
------------------------	-----------------------------------

Implements [Go::Streamable](#).

The documentation for this class was generated from the following files:

- [Irsplines2D/include/GoTools/Irsplines2D/LRSplineSurface.h](#)
- [Irsplines2D/include/GoTools/Irsplines2D/LRSplineSurface\\_impl.h](#)

## 29.277 Go::LRSurfApprox Class Reference

```
#include <LRSurfApprox.h>
```

## Public Member Functions

- [LRSurfApprox](#) (std::vector< double > &points, int dim, double epsge, bool init\_mba=false, double mba\_level=0.0, bool closest\_dist=true, bool repar=false)
- [LRSurfApprox](#) (shared\_ptr< [SplineSurface](#) > &srf, std::vector< double > &points, double epsge, bool closest\_dist=true, bool repar=false)
- [LRSurfApprox](#) (shared\_ptr< [LRSplineSurface](#) > &srf, std::vector< double > &points, double epsge, bool closest\_dist=true, bool repar=false, bool check\_init\_accuracy=false)
- [LRSurfApprox](#) (int ncoef\_u, int order\_u, int ncoef\_v, int order\_v, std::vector< double > &points, int dim, double epsge, bool init\_mba=false, double mba\_level=0.0, bool closest\_dist=true, bool repar=false)
- [LRSurfApprox](#) (int order\_u, std::vector< double > &knots\_u, int order\_v, std::vector< double > &knots\_v, std::vector< double > &points, int dim, double epsge, bool init\_mba=false, double mba\_level=0.0, bool closest\_dist=true, bool repar=false)
- [LRSurfApprox](#) (int ncoef\_u, int order\_u, int ncoef\_v, int order\_v, std::vector< double > &points, int dim, double domain[4], double epsge, bool init\_mba=false, double mba\_level=0.0, bool closest\_dist=true, bool repar=false)
- [~LRSurfApprox](#) ()
  - Destructor.*
  - void [setSmoothingWeight](#) (double smooth)
  - void [setSmoothBoundary](#) (bool smoothbd)
  - void [setFixBoundary](#) (bool fix\_boundary)
  - void [setFixCorner](#) (bool fix\_corner)
  - void [edgeFix](#) (int edge\_fix[])
  - void [setTurn3D](#) (int iter)
  - void [setGridInfo](#) (double grid\_start[], double cell\_size[])
  - void [setMakeGhostPoints](#) (bool make\_ghost\_points)
    - Decide if ghost points should be constructed.*
  - void [setUseMBA](#) (bool useMBA)
  - void [addLowerConstraint](#) (double minval)
    - Add lower constraint. Only functional (1D surface)*
  - void [addUpperConstraint](#) (double maxval)
    - Add upper constraint. Only functional (1D surface)*
  - void [setLocalConstraint](#) (double constraint\_factor)
    - Set local constraint on coefficient. Only functional (1D surface)*
  - void [setMinimumElementSize](#) (double usize\_min, double vsize\_min)
    - Set minimum element size.*
  - void [setSwitchToMBA](#) (int iter)
    - Iteration count on which to switch to MBA (if LS)*
  - void [setInitMBA](#) (bool initMBA, double start\_coef=0.0)
  - void [setVerbose](#) (bool verbose)
  - shared\_ptr< [LRSplineSurface](#) > [getApproxSurf](#) (double &maxdist, double &avdist\_all, double &avdist, int &nmb\_out\_eps, int max\_iter=4)

### 29.277.1 Detailed Description

This class can generate a LR B-spline surface that approximates a set of points for a given accuracy.

Definition at line 57 of file LRSurfApprox.h.

### 29.277.2 Constructor & Destructor Documentation

**29.277.2.1** `Go::LRSurfApprox::LRSurfApprox ( std::vector< double > & points, int dim, double epsge, bool init_mba = false, double mba_level = 0.0, bool closest_dist = true, bool repar = false )`

Constructor given a parameterized point set

## Parameters

<i>points</i>	Parameterized point set given as (u1,v1,x1,y1,z1, u2, v2, ...) The length of the array is (2+dim)x(the number of points)
<i>dim</i>	The dimension of the geometry space
<i>epsge</i>	Requested approximation accuracy
<i>closest_dist</i>	Check accuracy in closest point or in corresponding parameter value
<i>repar</i>	Perform reparameterization during iterations

29.277.2.2 `Go::LRSurfApprox::LRSurfApprox ( shared_ptr< SplineSurface > & srf, std::vector< double > & points, double epsge, bool closest_dist = true, bool repar = false )`

Constructor given a parameterized point set and an initial spline surface

## Parameters

<i>srf</i>	Given spline surface
<i>points</i>	Parameterized point set given as (u1,v1,x1,y1,z1, u2, v2, ...) The length of the array is (2+dim)x(the number of points) Note that the sequence of the points can be changed and this class references the initial points (no copy)
<i>dim</i>	The dimension of the geometry space
<i>epsge</i>	Requested approximation accuracy
<i>closest_dist</i>	Check accuracy in closest point or in corresponding parameter value
<i>repar</i>	Perform reparameterization during iterations

29.277.2.3 `Go::LRSurfApprox::LRSurfApprox ( shared_ptr< LRBSplineSurface > & srf, std::vector< double > & points, double epsge, bool closest_dist = true, bool repar = false, bool check_init_accuracy = false )`

Constructor given a parameterized point set and an initial LR B-spline surface

## Parameters

<i>srf</i>	Given LR B-spline surface
<i>points</i>	Parameterized point set given as (u1,v1,x1,y1,z1, u2, v2, ...) The length of the array is (2+dim)x(the number of points) Note that the sequence of the points can be changed and this class references the initial points (no copy)
<i>dim</i>	The dimension of the geometry space
<i>epsge</i>	Requested approximation accuracy
<i>closest_dist</i>	Check accuracy in closest point or in corresponding parameter value
<i>repar</i>	Perform reparameterization during iterations

29.277.2.4 `Go::LRSurfApprox::LRSurfApprox ( int ncoef_u, int order_u, int ncoef_v, int order_v, std::vector< double > & points, int dim, double epsge, bool init_mba = false, double mba_level = 0.0, bool closest_dist = true, bool repar = false )`

Constructor given a parameterized point set and the size of an initial spline space

## Parameters

<i>ncoef_u</i>	Number of coefficients in the 1. parameter direction
<i>order_u</i>	Order in the 1. parameter direction
<i>ncoef_v</i>	Number of coefficients in the 2. parameter direction
<i>order_v</i>	Order in the 2. parameter direction
<i>points</i>	Parameterized point set given as (u1,v1,x1,y1,z1, u2, v2, ...) The length of the array is (2+dim)x(the number of points) Note that the sequence of the points can be changed and this class references the initial points (no copy)
<i>dim</i>	The dimension of the geometry space
<i>epsge</i>	Requested approximation accuracy
<i>closest_dist</i>	Check accuracy in closest point or in corresponding parameter value
<i>repar</i>	Perform reparameterization during iterations

**29.277.2.5** Go::LRSurfApprox::LRSurfApprox ( int *order\_u*, std::vector< double > & *knots\_u*, int *order\_v*, std::vector< double > & *knots\_v*, std::vector< double > & *points*, int *dim*, double *epsge*, bool *init\_mba* = false, double *mba\_level* = 0.0, bool *closest\_dist* = true, bool *repar* = false )

Constructor given a parameterized point set and an initial spline space

## Parameters

<i>order_u</i>	Order in the 1. parameter direction
<i>knots_u</i>	Knot vector in the 1. parameter direction
<i>order_v</i>	Order in the 2. parameter direction
<i>knots_v</i>	Knot vector in the 2. parameter direction
<i>points</i>	Parameterized point set given as (u1,v1,x1,y1,z1, u2, v2, ...) The length of the array is (2+dim)x(the number of points) Note that the sequence of the points can be changed and this class references the initial points (no copy)
<i>dim</i>	The dimension of the geometry space
<i>epsge</i>	Requested approximation accuracy
<i>closest_dist</i>	Check accuracy in closest point or in corresponding parameter value
<i>repar</i>	Perform reparameterization during iterations

**29.277.2.6** Go::LRSurfApprox::LRSurfApprox ( int *ncoef\_u*, int *order\_u*, int *ncoef\_v*, int *order\_v*, std::vector< double > & *points*, int *dim*, double *domain*[4], double *epsge*, bool *init\_mba* = false, double *mba\_level* = 0.0, bool *closest\_dist* = true, bool *repar* = false )

Constructor given a parameterized point set and the size of an initial spline space

## Parameters

<i>ncoef_u</i>	Number of coefficients in the 1. parameter direction
<i>order_u</i>	Order in the 1. parameter direction
<i>ncoef_v</i>	Number of coefficients in the 2. parameter direction
<i>order_v</i>	Order in the 2. parameter direction
<i>points</i>	Parameterized point set given as (u1,v1,x1,y1,z1, u2, v2, ...) The length of the array is (2+dim)x(the number of points) Note that the sequence of the points can be changed and this class references the initial points (no copy)

## Parameters

<i>dim</i>	The dimension of the geometry space
<i>domain</i>	A possibly extended parameter domain (umin, umax, vmin, vmax)
<i>epsge</i>	Requested approximation accuracy
<i>closest_dist</i>	Check accuracy in closest point or in corresponding parameter value
<i>repar</i>	Perform reparameterization during iterations

## 29.277.2.7 Go::LRSurfApprox::~~LRSurfApprox ( )

Destructor.

## 29.277.3 Member Function Documentation

29.277.3.1 void Go::LRSurfApprox::addLowerConstraint ( double *minval* ) [inline]

Add lower constraint. Only functional (1D surface)

Definition at line 251 of file LRSurfApprox.h.

29.277.3.2 void Go::LRSurfApprox::addUpperConstraint ( double *maxval* ) [inline]

Add upper constraint. Only functional (1D surface)

Definition at line 258 of file LRSurfApprox.h.

29.277.3.3 void Go::LRSurfApprox::edgeFix ( int *edge\_fix*[ ] ) [inline]

Decide whether specific edges of the surface's boundary should be kept fixed (i.e. unchanged by approximation process), as well as a certain number of cross-derivatives across these curves.

## Parameters

<i>edge_fix</i>	pointer to an array of four integers specifying to which extent each surface edge should be kept fixed during the approximation process. A value of 0 means that it will not be kept fixed, 1 means that its position will be kept fixed, derivatives at the boundary are currently NOT kept fixed. The integers are associated with the surface edges starting with the edge 'v=vmin' and moving counterclockwise.
-----------------	---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Definition at line 216 of file LRSurfApprox.h.

29.277.3.4 shared\_ptr<LRSplineSurface> Go::LRSurfApprox::getApproxSurf ( double & *maxdist*, double & *avdist\_all*, double & *avdist*, int & *nmb\_out\_eps*, int *max\_iter* = 4 )

When everything else is set, this function can be used to run the approximation process and fetch the approximated surface.



## Return values

<i>maxdist</i>	report the maximum distance between the approximated surface and the data points
<i>avdist</i>	report the average distance between the approximated surface and the datapoints

## Parameters

<i>nmb_out_eps</i>	report the number of data points that were found to lie outside the geometric tolerance (as specified by the 'aepsge' argument to the <a href="#">ApproxSurf</a> constructor.
<i>max_iter</i>	maximum number of iterations

## Returns

a shared pointer to the generated [SplineCurve](#), approximating the points as specified.

**29.277.3.5** void Go::LRSurfApprox::setFixBoundary ( bool *fix\_boundary* ) [inline]

Decide whether or not the total boundary of the surface should be kept fixed (i.e. unchanged by approximation process). Default is true. Cross derivatives will not be kept fixed. (If you want to keep cross derivatives fixed, use the [edgeFix\(\)](#) member function instead).

## Parameters

<i>fix_boundary</i>	if 'true' the boundary of the surface will not be modified, if 'false' it will be open to modification.
---------------------	---------------------------------------------------------------------------------------------------------

Definition at line 193 of file LRSurfApprox.h.

**29.277.3.6** void Go::LRSurfApprox::setFixCorner ( bool *fix\_corner* ) [inline]

Definition at line 200 of file LRSurfApprox.h.

**29.277.3.7** void Go::LRSurfApprox::setGridInfo ( double *grid\_start*[], double *cell\_size*[] ) [inline]

Definition at line 228 of file LRSurfApprox.h.

**29.277.3.8** void Go::LRSurfApprox::setInitMBA ( bool *initMBA*, double *start\_coef* = 0.0 ) [inline]

Set if an initial MBA surface should be used in LS approximation Default is false

Definition at line 286 of file LRSurfApprox.h.

**29.277.3.9** void Go::LRSurfApprox::setLocalConstraint ( double *constraint\_factor* ) [inline]

Set local constraint on coefficient. Only functional (1D surface)

Definition at line 265 of file LRSurfApprox.h.

29.277.3.10 void Go::LRSurfApprox::setMakeGhostPoints ( bool *make\_ghost\_points* ) [inline]

Decide if ghost points should be constructed.

Definition at line 238 of file LRSurfApprox.h.

29.277.3.11 void Go::LRSurfApprox::setMinimumElementSize ( double *usize\_min*, double *vsize\_min* ) [inline]

Set minimum element size.

Definition at line 272 of file LRSurfApprox.h.

29.277.3.12 void Go::LRSurfApprox::setSmoothBoundary ( bool *smoothbd* ) [inline]

Definition at line 182 of file LRSurfApprox.h.

29.277.3.13 void Go::LRSurfApprox::setSmoothingWeight ( double *smooth* ) [inline]

Sets the smoothing weight to something other than the default (1e-9). The value should lie in the unit interval, typically close to 0.

#### Parameters

<i>smooth</i>	the new smoothing weight.
---------------	---------------------------

Definition at line 176 of file LRSurfApprox.h.

29.277.3.14 void Go::LRSurfApprox::setSwitchToMBA ( int *iter* ) [inline]

Iteration count on which to switch to MBA (if LS)

Definition at line 279 of file LRSurfApprox.h.

29.277.3.15 void Go::LRSurfApprox::setTurn3D ( int *iter* ) [inline]

Definition at line 223 of file LRSurfApprox.h.

29.277.3.16 void Go::LRSurfApprox::setUseMBA ( bool *useMBA* ) [inline]

Decide if the MBA algorithm should be used to approximate points (default == false)

Definition at line 245 of file LRSurfApprox.h.

29.277.3.17 void Go::LRSurfApprox::setVerbose ( bool *verbose* ) [inline]

Whether or not intermediate information should be written to standard output (default is not)

Definition at line 294 of file LRSurfApprox.h.

The documentation for this class was generated from the following file:

- [Irsplines2D/include/GoTools/Irsplines2D/LRSurfApprox.h](#)

## 29.278 Go::LRSurfSmoothLS Class Reference

```
#include <LRSurfSmoothLS.h>
```

### Public Member Functions

- [LRSurfSmoothLS](#) (shared\_ptr< [LRSplineSurface](#) > surf, std::vector< int > &coef\_known)
- [LRSurfSmoothLS](#) ()  
*Empty constructor.*
- [~LRSurfSmoothLS](#) ()  
*Destructor.*
- void [setInitSf](#) (shared\_ptr< [LRSplineSurface](#) > surf, std::vector< int > &coef\_known)
- void [updateLocals](#) ()  
*Reset local arrays after changing LR B-spline surface.*
- [bool](#) [hasDataPoints](#) () const  
*Check if data points already are available.*
- void [addDataPoints](#) (std::vector< [double](#) > &points, [bool](#) is\_ghost\_points=false)
- void [setOptimize](#) (const [double](#) weight1, const [double](#) weight2, const [double](#) weight3)
- void [smoothBoundary](#) (const [double](#) weight1, const [double](#) weight2, const [double](#) weight3)
- void [setLeastSquares](#) (const [double](#) weight)
- void [setLeastSquares\\_omp](#) (const [double](#) weight)  
*OpenMP enabled version of the above function.*
- void [setLeastSquares](#) (std::vector< [double](#) > &points, const [double](#) weight)
- int [equationSolve](#) (shared\_ptr< [LRSplineSurface](#) > &surf)

### 29.278.1 Detailed Description

Definition at line 72 of file LRSurfSmoothLS.h.

### 29.278.2 Constructor & Destructor Documentation

29.278.2.1 Go::LRSurfSmoothLS::LRSurfSmoothLS ( shared\_ptr< [LRSplineSurface](#) > *surf*, std::vector< int > &*coef\_known* )

Constructor Note that the surface is modified

## Parameters

<i>coef_known</i>	indicates if the coefficient associated to the LR B-splines is known already
-------------------	------------------------------------------------------------------------------

29.278.2.2 `Go::LRSurfSmoothLS::LRSurfSmoothLS ( )`

Empty constructor.

29.278.2.3 `Go::LRSurfSmoothLS::~~LRSurfSmoothLS ( )`

Destructor.

### 29.278.3 Member Function Documentation

29.278.3.1 `void Go::LRSurfSmoothLS::addDataPoints ( std::vector< double > & points, bool is_ghost_points = false )`

Add data points for approximation. The points are parameterized and are store as follows: parameter values for point 1 (2 doubles), position for point1 (dim doubles where dim is the dimension of the associated LR spline surface), parameter values for point 2 etc.

29.278.3.2 `int Go::LRSurfSmoothLS::equationSolve ( shared_ptr< LRSplineSurface > & surf )`

Solve equation system, and produce output surface. If failing to solve the routine may throw an exception.

## Parameters

<i>surf</i>	the output surface.
-------------	---------------------

## Returns

0 = OK, negative = failed solving system.

29.278.3.3 `bool Go::LRSurfSmoothLS::hasDataPoints ( ) const`

Check if data points already are available.

29.278.3.4 `void Go::LRSurfSmoothLS::setInitSf ( shared_ptr< LRSplineSurface > surf, std::vector< int > & coef_known )`

Set initial surface Note that the surface is modified

## Parameters

<i>coef_known</i>	indicates if the coefficient associated to the LR B-splines is known already
-------------------	------------------------------------------------------------------------------

29.278.3.5 void Go::LRSurfSmoothLS::setLeastSquares ( const double *weight* )

Compute matrices for least squares approximation. The data points are expected to be added already

Parameters

<i>weight</i>	the contribution of the approximation of the pnts in the system. weight should lie in the unit interval.
---------------	----------------------------------------------------------------------------------------------------------

29.278.3.6 void Go::LRSurfSmoothLS::setLeastSquares ( std::vector< double > & *points*, const double *weight* )

Compute matrices for least squares approximation.

Parameters

<i>points</i>	Parameter values and point for each data point
<i>weight</i>	the contribution of the approximation of the pnts in the system. weight should lie in the unit interval.

29.278.3.7 void Go::LRSurfSmoothLS::setLeastSquares\_omp ( const double *weight* )

OpenMP enabled version of the above function.

29.278.3.8 void Go::LRSurfSmoothLS::setOptimize ( const double *weight1*, const double *weight2*, const double *weight3* )

Compute the smoothing part of the equation system.

Parameters

<i>weight1</i>	contribution weight with respect to the 1st derivative.
<i>weight2</i>	contribution weight with respect to the 2nd derivative.
<i>weight3</i>	contribution weight with respect to the 3rd derivative.

29.278.3.9 void Go::LRSurfSmoothLS::smoothBoundary ( const double *weight1*, const double *weight2*, const double *weight3* )

Compute the boundary smoothing part of the equation system.

Parameters

<i>weight1</i>	contribution weight with respect to the 1st derivative.
<i>weight2</i>	contribution weight with respect to the 2nd derivative.
<i>weight3</i>	contribution weight with respect to the 3rd derivative.

29.278.3.10 void Go::LRSurfSmoothLS::updateLocals ( )

Reset local arrays after changing LR B-spline surface.

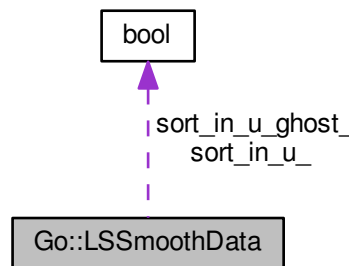
The documentation for this class was generated from the following file:

- [Irsplines2D/include/GoTools/Irsplines2D/LRSurfSmoothLS.h](#)

## 29.279 Go::LSSmoothData Struct Reference

```
#include <Element2D.h>
```

Collaboration diagram for Go::LSSmoothData:



### Public Member Functions

- [LSSmoothData](#) ( )
- [bool hasDataPoints](#) ( )
- void [eraseDataPoints](#) ( )
- void [eraseGhostPoints](#) ( )
- void [eraseDataPoints](#) (std::vector< [double](#) >::iterator start, std::vector< [double](#) >::iterator end)
- void [addDataPoints](#) (std::vector< [double](#) >::iterator start, std::vector< [double](#) >::iterator end, [bool](#) sort\_in\_u)
- void [addDataPoints](#) (std::vector< [double](#) >::iterator start, std::vector< [double](#) >::iterator end, int del, [bool](#) sort\_in\_u)
- void [addGhostPoints](#) (std::vector< [double](#) >::iterator start, std::vector< [double](#) >::iterator end, [bool](#) sort\_in\_u↔)
- void [addGhostPoints](#) (std::vector< [double](#) >::iterator start, std::vector< [double](#) >::iterator end, int del, [bool](#) sort\_in\_u)
- std::vector< [double](#) > & [getDataPoints](#) ( )
- std::vector< [double](#) > & [getGhostPoints](#) ( )
- void [getOutsidePoints](#) (std::vector< [double](#) > &points, int dim, [Direction2D](#) d, [double](#) start, [double](#) end, [bool](#) &sort\_in\_u)
- void [getOutsideGhostPoints](#) (std::vector< [double](#) > &ghost, int dim, [Direction2D](#) d, [double](#) start, [double](#) end, [bool](#) &sort\_in\_u)
- int [dataPointSize](#) ( )

- int [ghostPointSize](#) ()
- bool [hasLSMatrix](#) ()
- void [setLSMatrix](#) (int nmb, int dim)
- void [getLSMatrix](#) (double \*&LSmat, double \*&LSright, int &ncond)
- bool [hasAccuracyInfo](#) ()
- void [getAccuracyInfo](#) (double &average\_error, double &max\_error, int &nmb\_outside\_tol)
- int [getNmbOutsideTol](#) ()
- double [getAverageError](#) ()
- double [getAccumulatedError](#) ()
- double [getMaxError](#) ()
- void [setAccuracyInfo](#) (double accumulated\_error, double average\_error, double max\_error, int nmb\_outside←\_tol)
- void [resetAccuracyInfo](#) ()
- bool [getDataBoundingBox](#) (int dim, double bb[])
- void [makeDataPoints3D](#) (int dim)
- void [updateAccuracyInfo](#) (int dim)
- void [updateLSDataParDomain](#) (double u1, double u2, double v1, double v2, double u1new, double u2new, double v1new, double v2new, int dim)

## Public Attributes

- std::vector< double > [data\\_points\\_](#)
- std::vector< double > [ghost\\_points\\_](#)
- std::vector< double > [LSmat\\_](#)
- std::vector< double > [LSright\\_](#)
- int [ncond\\_](#)
- bool [sort\\_in\\_u\\_](#)
- bool [sort\\_in\\_u\\_ghost\\_](#)
- double [accumulated\\_error\\_](#)
- double [average\\_error\\_](#)
- double [max\\_error\\_](#)
- double [max\\_error\\_prev\\_](#)
- int [nmb\\_outside\\_tol\\_](#)

### 29.279.1 Detailed Description

Definition at line 52 of file Element2D.h.

### 29.279.2 Constructor & Destructor Documentation

29.279.2.1 `Go::LSSmoothData::LSSmoothData ( )` `[inline]`

Definition at line 54 of file Element2D.h.

### 29.279.3 Member Function Documentation

29.279.3.1 `void Go::LSSmoothData::addDataPoints ( std::vector< double >::iterator start, std::vector< double >::iterator end, bool sort_in_u )` `[inline]`

Definition at line 83 of file Element2D.h.

29.279.3.2 `void Go::LSSmoothData::addDataPoints ( std::vector< double >::iterator start, std::vector< double >::iterator end, int del, bool sort_in_u ) [inline]`

Definition at line 91 of file Element2D.h.

29.279.3.3 `void Go::LSSmoothData::addGhostPoints ( std::vector< double >::iterator start, std::vector< double >::iterator end, bool sort_in_u ) [inline]`

Definition at line 103 of file Element2D.h.

29.279.3.4 `void Go::LSSmoothData::addGhostPoints ( std::vector< double >::iterator start, std::vector< double >::iterator end, int del, bool sort_in_u ) [inline]`

Definition at line 111 of file Element2D.h.

29.279.3.5 `int Go::LSSmoothData::dataPointSize ( ) [inline]`

Definition at line 141 of file Element2D.h.

29.279.3.6 `void Go::LSSmoothData::eraseDataPoints ( ) [inline]`

Definition at line 67 of file Element2D.h.

29.279.3.7 `void Go::LSSmoothData::eraseDataPoints ( std::vector< double >::iterator start, std::vector< double >::iterator end ) [inline]`

Definition at line 77 of file Element2D.h.

29.279.3.8 `void Go::LSSmoothData::eraseGhostPoints ( ) [inline]`

Definition at line 72 of file Element2D.h.

29.279.3.9 `double Go::LSSmoothData::getAccumulatedError ( ) [inline]`

Definition at line 193 of file Element2D.h.

29.279.3.10 `void Go::LSSmoothData::getAccuracyInfo ( double & average_error, double & max_error, int & nmb_outside_tol ) [inline]`

Definition at line 175 of file Element2D.h.



29.279.3.11 `double Go::LSSmoothData::getAverageError ( ) [inline]`

Definition at line 188 of file Element2D.h.

29.279.3.12 `bool Go::LSSmoothData::getDataBoundingBox ( int dim, double bb[ ] )`

29.279.3.13 `std::vector<double>& Go::LSSmoothData::getDataPoints ( ) [inline]`

Definition at line 123 of file Element2D.h.

29.279.3.14 `std::vector<double>& Go::LSSmoothData::getGhostPoints ( ) [inline]`

Definition at line 128 of file Element2D.h.

29.279.3.15 `void Go::LSSmoothData::getLSMatrix ( double *& LSmat, double *& LSright, int & ncond ) [inline]`

Definition at line 163 of file Element2D.h.

29.279.3.16 `double Go::LSSmoothData::getMaxError ( ) [inline]`

Definition at line 198 of file Element2D.h.

29.279.3.17 `int Go::LSSmoothData::getNmbOutsideTol ( ) [inline]`

Definition at line 183 of file Element2D.h.

29.279.3.18 `void Go::LSSmoothData::getOutsideGhostPoints ( std::vector< double > & ghost, int dim, Direction2D d, double start, double end, bool & sort_in_u )`

29.279.3.19 `void Go::LSSmoothData::getOutsidePoints ( std::vector< double > & points, int dim, Direction2D d, double start, double end, bool & sort_in_u )`

29.279.3.20 `int Go::LSSmoothData::ghostPointSize ( ) [inline]`

Definition at line 146 of file Element2D.h.

29.279.3.21 `bool Go::LSSmoothData::hasAccuracyInfo ( ) [inline]`

Definition at line 170 of file Element2D.h.

29.279.3.22 `bool Go::LSSmoothData::hasDataPoints ( ) [inline]`

Definition at line 62 of file Element2D.h.

29.279.3.23 **bool** Go::LSSmoothData::hasLSMatrix ( ) [inline]

Definition at line 151 of file Element2D.h.

29.279.3.24 **void** Go::LSSmoothData::makeDataPoints3D ( int *dim* )

29.279.3.25 **void** Go::LSSmoothData::resetAccuracyInfo ( ) [inline]

Definition at line 214 of file Element2D.h.

29.279.3.26 **void** Go::LSSmoothData::setAccuracyInfo ( **double** *accumulated\_error*, **double** *average\_error*, **double** *max\_error*, int *nmb\_outside\_tol* ) [inline]

Definition at line 203 of file Element2D.h.

29.279.3.27 **void** Go::LSSmoothData::setLSMatrix ( int *nmb*, int *dim* ) [inline]

Definition at line 156 of file Element2D.h.

29.279.3.28 **void** Go::LSSmoothData::updateAccuracyInfo ( int *dim* )

29.279.3.29 **void** Go::LSSmoothData::updateLSDataParDomain ( **double** *u1*, **double** *u2*, **double** *v1*, **double** *v2*, **double** *u1new*, **double** *u2new*, **double** *v1new*, **double** *v2new*, int *dim* )

## 29.279.4 Member Data Documentation

29.279.4.1 **double** Go::LSSmoothData::accumulated\_error\_

Definition at line 242 of file Element2D.h.

29.279.4.2 **double** Go::LSSmoothData::average\_error\_

Definition at line 243 of file Element2D.h.

29.279.4.3 **std::vector<double>** Go::LSSmoothData::data\_points\_

Definition at line 234 of file Element2D.h.

29.279.4.4 **std::vector<double>** Go::LSSmoothData::ghost\_points\_

Definition at line 235 of file Element2D.h.

29.279.4.5 `std::vector<double>` `Go::LSSmoothData::LSmat_`

Definition at line 236 of file `Element2D.h`.

29.279.4.6 `std::vector<double>` `Go::LSSmoothData::LSright_`

Definition at line 237 of file `Element2D.h`.

29.279.4.7 `double` `Go::LSSmoothData::max_error_`

Definition at line 244 of file `Element2D.h`.

29.279.4.8 `double` `Go::LSSmoothData::max_error_prev_`

Definition at line 245 of file `Element2D.h`.

29.279.4.9 `int` `Go::LSSmoothData::ncond_`

Definition at line 238 of file `Element2D.h`.

29.279.4.10 `int` `Go::LSSmoothData::nmb_outside_tol_`

Definition at line 246 of file `Element2D.h`.

29.279.4.11 `bool` `Go::LSSmoothData::sort_in_u_`

Definition at line 239 of file `Element2D.h`.

29.279.4.12 `bool` `Go::LSSmoothData::sort_in_u_ghost_`

Definition at line 240 of file `Element2D.h`.

The documentation for this struct was generated from the following file:

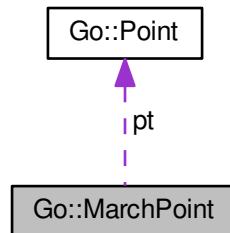
- `IrSplines2D/include/GoTools/IrSplines2D/Element2D.h`

## 29.280 Go::MarchPoint Class Reference

Helper class for marching.

```
#include <FaceAdjacency.h>
```

Collaboration diagram for Go::MarchPoint:



### Public Member Functions

- [MarchPoint](#) ([const Go::Point &p](#), [double pa](#), [double pa2](#), [double d](#), [double ca](#), [int s](#))
- [bool operator<](#) ([const MarchPoint &other](#)) [const](#)

### Public Attributes

- [Go::Point pt](#)
- [double par](#)
- [double par2](#)
- [double dist](#)
- [double cos\\_ang](#)
- [int status](#)

### 29.280.1 Detailed Description

Helper class for marching.

Definition at line 63 of file FaceAdjacency.h.

### 29.280.2 Constructor & Destructor Documentation

29.280.2.1 [Go::MarchPoint::MarchPoint](#) ( [const Go::Point & p](#), [double pa](#), [double pa2](#), [double d](#), [double ca](#), [int s](#) )  
[\[inline\]](#)

Definition at line 72 of file FaceAdjacency.h.

### 29.280.3 Member Function Documentation

29.280.3.1 `bool Go::MarchPoint::operator<( const MarchPoint & other ) const` `[inline]`

Definition at line 76 of file FaceAdjacency.h.

### 29.280.4 Member Data Documentation

29.280.4.1 `double Go::MarchPoint::cos_ang`

Definition at line 70 of file FaceAdjacency.h.

29.280.4.2 `double Go::MarchPoint::dist`

Definition at line 69 of file FaceAdjacency.h.

29.280.4.3 `double Go::MarchPoint::par`

Definition at line 67 of file FaceAdjacency.h.

29.280.4.4 `double Go::MarchPoint::par2`

Definition at line 68 of file FaceAdjacency.h.

29.280.4.5 `Go::Point Go::MarchPoint::pt`

Definition at line 66 of file FaceAdjacency.h.

29.280.4.6 `int Go::MarchPoint::status`

Definition at line 71 of file FaceAdjacency.h.

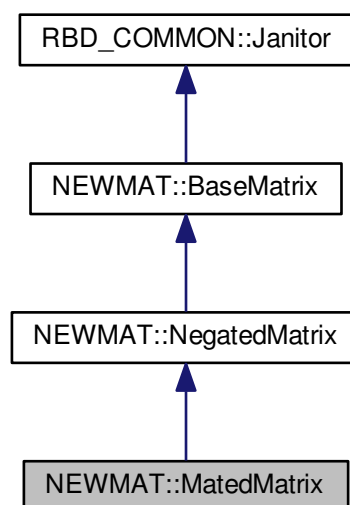
The documentation for this class was generated from the following file:

- [topology/include/GoTools/topology/FaceAdjacency.h](#)

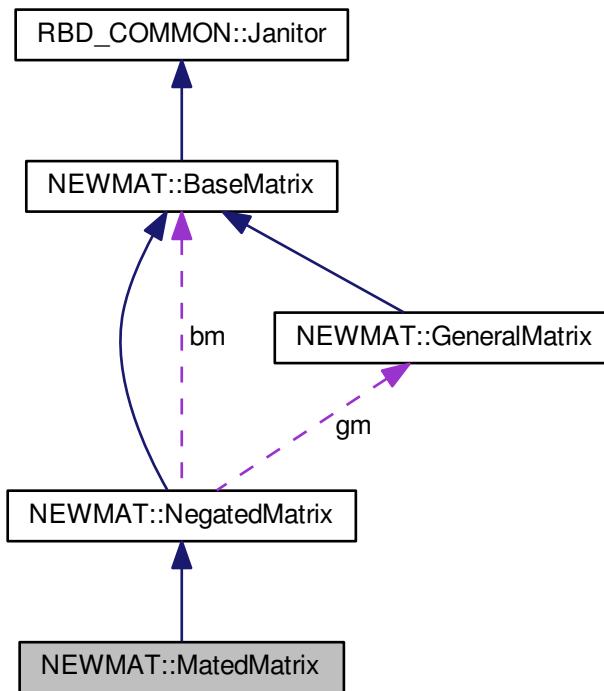
## 29.281 NEWMAT::MatedMatrix Class Reference

```
#include <newmat.h>
```

Inheritance diagram for NEWMAT::MatedMatrix:



Collaboration diagram for NEWMAT::MatedMatrix:



### Public Member Functions

- [~MatedMatrix \(\)](#)
- [GeneralMatrix \\* Evaluate \(MatrixType mt=MatrixTypeUnSp\)](#)
- [MatrixBandWidth BandWidth \(\) const](#)

### Friends

- class [BaseMatrix](#)

### Additional Inherited Members

#### 29.281.1 Detailed Description

Definition at line 1453 of file newmat.h.

#### 29.281.2 Constructor & Destructor Documentation

##### 29.281.2.1 NEWMAT::MatedMatrix::~~MatedMatrix ( ) [inline]

Definition at line 1460 of file newmat.h.

### 29.281.3 Member Function Documentation

29.281.3.1 `MatrixBandWidth MatedMatrix::BandWidth ( ) const` [virtual]

Reimplemented from [NEWMAT::NegatedMatrix](#).

Definition at line 484 of file `newmat4.cpp`.

29.281.3.2 `GeneralMatrix * MatedMatrix::Evaluate ( MatrixType mt = MatrixTypeUnSp )` [virtual]

Reimplemented from [NEWMAT::NegatedMatrix](#).

Definition at line 326 of file `newmat5.cpp`.

### 29.281.4 Friends And Related Function Documentation

29.281.4.1 `friend class BaseMatrix` [friend]

Definition at line 1458 of file `newmat.h`.

The documentation for this class was generated from the following files:

- [newmat/include/newmat.h](#)
- [newmat/src/newmat4.cpp](#)
- [newmat/src/newmat5.cpp](#)

## 29.282 material\_appearance Class Reference

```
#include <gl_aux.h>
```

### Public Member Functions

- [material\\_appearance](#) (void)
- [col\\_scheme\\_selected](#) (void) const
- [col\\_scheme\\_select](#) (const int i)
- [set\\_material](#) (void)
- [redefine\\_material\\_f](#) (const vector3t< double > &diff, const vector3t< double > &amb, const vector3t< double > &em, const vector3t< double > &spec, const double shin)
- [Key](#) (unsigned char key)

### Public Attributes

- [double ztrans\\_delta](#)



### 29.282.1 Detailed Description

Definition at line 68 of file gl\_aux.h.

### 29.282.2 Constructor & Destructor Documentation

#### 29.282.2.1 material\_appearance::material\_appearance ( void ) [inline]

Definition at line 101 of file gl\_aux.h.

### 29.282.3 Member Function Documentation

#### 29.282.3.1 void material\_appearance::col\_scheme\_select ( const int *i* ) [inline]

Definition at line 189 of file gl\_aux.h.

#### 29.282.3.2 int material\_appearance::col\_scheme\_selected ( void ) const [inline]

Definition at line 188 of file gl\_aux.h.

#### 29.282.3.3 void material\_appearance::Key ( unsigned char *key* ) [inline]

Definition at line 238 of file gl\_aux.h.

#### 29.282.3.4 void material\_appearance::redefine\_material\_f ( const vector3t< double > & *diff*, const vector3t< double > & *amb*, const vector3t< double > & *em*, const vector3t< double > & *spec*, const double *shin* ) [inline]

Definition at line 209 of file gl\_aux.h.

#### 29.282.3.5 void material\_appearance::set\_material ( void ) [inline]

Definition at line 194 of file gl\_aux.h.

### 29.282.4 Member Data Documentation

#### 29.282.4.1 double material\_appearance::ztrans\_delta

Definition at line 96 of file gl\_aux.h.

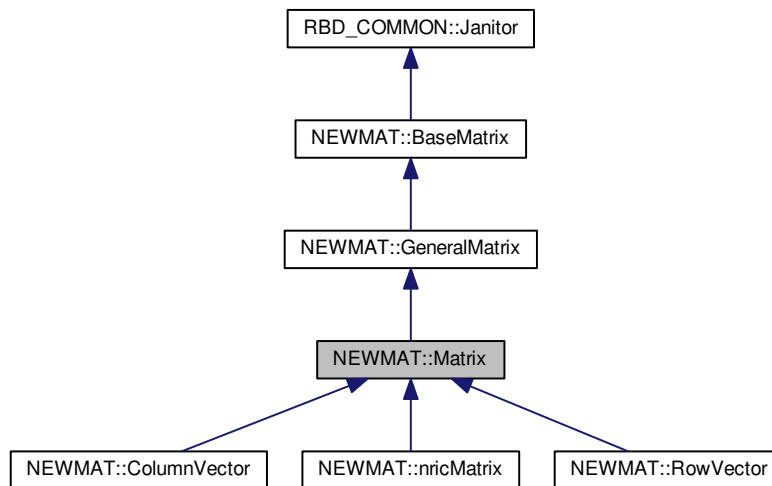
The documentation for this class was generated from the following file:

- [sisl/examples/viewer/include/gl\\_aux.h](#)

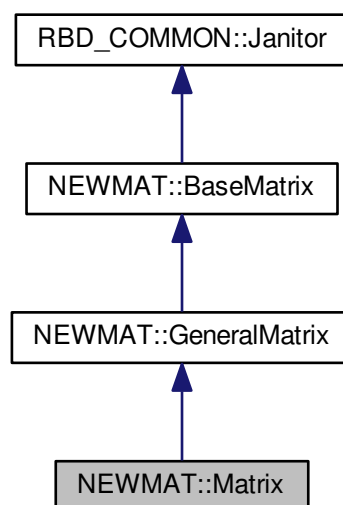
## 29.283 NEWMAT::Matrix Class Reference

```
#include <newmat.h>
```

Inheritance diagram for NEWMAT::Matrix:



Collaboration diagram for NEWMAT::Matrix:



## Public Member Functions

- [Matrix](#) ()
- [~Matrix](#) ()
- [Matrix](#) (int, int)
- [Matrix](#) (const [BaseMatrix](#) &)
- void [operator=](#) (const [BaseMatrix](#) &)
- void [operator=](#) ([Real](#) f)
- void [operator=](#) (const [Matrix](#) &m)
- [MatrixType](#) [Type](#) () const
- [Real](#) & [operator\(\)](#) (int, int)
- [Real](#) & [element](#) (int, int)
- [Real](#) [operator\(\)](#) (int, int) const
- [Real](#) [element](#) (int, int) const
- [Matrix](#) (const [Matrix](#) &gm)
- [GeneralMatrix](#) \* [MakeSolver](#) ()
- [Real](#) [Trace](#) () const
- void [GetRow](#) ([MatrixRowCol](#) &)
- void [GetCol](#) ([MatrixRowCol](#) &)
- void [GetCol](#) ([MatrixColX](#) &)
- void [RestoreCol](#) ([MatrixRowCol](#) &)
- void [RestoreCol](#) ([MatrixColX](#) &)
- void [NextRow](#) ([MatrixRowCol](#) &)
- void [NextCol](#) ([MatrixRowCol](#) &)
- void [NextCol](#) ([MatrixColX](#) &)
- virtual void [ReSize](#) (int, int)
- void [ReSize](#) (const [GeneralMatrix](#) &A)
- [Real](#) [MaximumAbsoluteValue2](#) (int &i, int &j) const
- [Real](#) [MinimumAbsoluteValue2](#) (int &i, int &j) const
- [Real](#) [Maximum2](#) (int &i, int &j) const
- [Real](#) [Minimum2](#) (int &i, int &j) const

## Friends

- [Real](#) [DotProduct](#) (const [Matrix](#) &A, const [Matrix](#) &B)

## Additional Inherited Members

### 29.283.1 Detailed Description

Definition at line 563 of file newmat.h.

### 29.283.2 Constructor & Destructor Documentation

#### 29.283.2.1 NEWMAT::Matrix::Matrix ( ) [inline]

Definition at line 567 of file newmat.h.

29.283.2.2 `NEWMAT::Matrix::~~Matrix ( )` [inline]

Definition at line 568 of file newmat.h.

29.283.2.3 `NEWMAT::Matrix::Matrix ( int , int )`

29.283.2.4 `NEWMAT::Matrix::Matrix ( const BaseMatrix & )`

29.283.2.5 `NEWMAT::Matrix::Matrix ( const Matrix & gm )` [inline]

Definition at line 583 of file newmat.h.

### 29.283.3 Member Function Documentation

29.283.3.1 `Real& NEWMAT::Matrix::element ( int , int )`

29.283.3.2 `Real NEWMAT::Matrix::element ( int , int ) const`

29.283.3.3 `void NEWMAT::Matrix::GetCol ( MatrixRowCol & )` [virtual]

Implements [NEWMAT::GeneralMatrix](#).

Reimplemented in [NEWMAT::RowVector](#).

29.283.3.4 `void NEWMAT::Matrix::GetCol ( MatrixColX & )` [virtual]

Implements [NEWMAT::GeneralMatrix](#).

Reimplemented in [NEWMAT::RowVector](#).

29.283.3.5 `void Matrix::GetRow ( MatrixRowCol & mrc )` [virtual]

Implements [NEWMAT::GeneralMatrix](#).

Definition at line 111 of file newmat3.cpp.

29.283.3.6 `GeneralMatrix * Matrix::MakeSolver ( )` [virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 31 of file newmat7.cpp.

29.283.3.7 `Real Matrix::Maximum2 ( int & i, int & j ) const` [virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 366 of file newmat8.cpp.

29.283.3.8 `Real Matrix::MaximumAbsoluteValue2 ( int & i, int & j ) const` [virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 350 of file newmat8.cpp.

29.283.3.9 `Real Matrix::Minimum2 ( int & i, int & j ) const` [virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 374 of file newmat8.cpp.

29.283.3.10 `Real Matrix::MinimumAbsoluteValue2 ( int & i, int & j ) const` [virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 358 of file newmat8.cpp.

29.283.3.11 `void NEWMAT::Matrix::NextCol ( MatrixRowCol & )` [virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Reimplemented in [NEWMAT::RowVector](#).

29.283.3.12 `void NEWMAT::Matrix::NextCol ( MatrixColIX & )` [virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Reimplemented in [NEWMAT::RowVector](#).

29.283.3.13 `void Matrix::NextRow ( MatrixRowCol & mrc )` [virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 184 of file newmat3.cpp.

29.283.3.14 **Real**& NEWMAT::Matrix::operator()( int , int )

29.283.3.15 **Real** NEWMAT::Matrix::operator()( int , int ) const

29.283.3.16 void NEWMAT::Matrix::operator= ( **const** BaseMatrix & )

29.283.3.17 void NEWMAT::Matrix::operator= ( **Real** f ) [inline]

Definition at line 572 of file newmat.h.

29.283.3.18 void NEWMAT::Matrix::operator= ( **const** Matrix & m ) [inline]

Definition at line 573 of file newmat.h.

29.283.3.19 virtual void NEWMAT::Matrix::ReSize ( int , int ) [virtual]

Reimplemented in [NEWMAT::ColumnVector](#), [NEWMAT::RowVector](#), and [NEWMAT::nricMatrix](#).

29.283.3.20 void Matrix::ReSize ( **const** GeneralMatrix & A ) [virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Reimplemented in [NEWMAT::ColumnVector](#), [NEWMAT::RowVector](#), and [NEWMAT::nricMatrix](#).

Definition at line 267 of file newmat4.cpp.

29.283.3.21 void NEWMAT::Matrix::RestoreCol ( MatrixRowCol & ) [virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Reimplemented in [NEWMAT::RowVector](#).

29.283.3.22 void NEWMAT::Matrix::RestoreCol ( MatrixColIX & ) [virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Reimplemented in [NEWMAT::RowVector](#).

29.283.3.23 **Real** Matrix::Trace ( ) const [virtual]

Reimplemented from [NEWMAT::BaseMatrix](#).

Definition at line 534 of file newmat8.cpp.

29.283.3.24 `MatrixType Matrix::Type ( ) const` [virtual]

Implements [NEWMAT::GeneralMatrix](#).

Reimplemented in [NEWMAT::ColumnVector](#), and [NEWMAT::RowVector](#).

Definition at line 385 of file `newmat4.cpp`.

## 29.283.4 Friends And Related Function Documentation

29.283.4.1 `Real DotProduct ( const Matrix & A, const Matrix & B )` [friend]

The documentation for this class was generated from the following files:

- [newmat/include/newmat.h](#)
- [newmat/src/newmat3.cpp](#)
- [newmat/src/newmat4.cpp](#)
- [newmat/src/newmat7.cpp](#)
- [newmat/src/newmat8.cpp](#)

## 29.284 NEWMAT::MatrixBandWidth Class Reference

```
#include <newmat.h>
```

### Public Member Functions

- [MatrixBandWidth \(const int l, const int u\)](#)
- [MatrixBandWidth \(const int i\)](#)
- [MatrixBandWidth operator+ \(const MatrixBandWidth &\) const](#)
- [MatrixBandWidth operator\\* \(const MatrixBandWidth &\) const](#)
- [MatrixBandWidth minimum \(const MatrixBandWidth &\) const](#)
- [MatrixBandWidth t \(\) const](#)
- [bool operator== \(const MatrixBandWidth &bw\) const](#)
- [bool operator!= \(const MatrixBandWidth &bw\) const](#)
- [int Upper \(\) const](#)
- [int Lower \(\) const](#)

### Public Attributes

- [int lower](#)
- [int upper](#)

### 29.284.1 Detailed Description

Definition at line 190 of file `newmat.h`.

## 29.284.2 Constructor & Destructor Documentation

29.284.2.1 `NEWMAT::MatrixBandWidth::MatrixBandWidth ( const int l, const int u )` `[inline]`

Definition at line 195 of file `newmat.h`.

29.284.2.2 `NEWMAT::MatrixBandWidth::MatrixBandWidth ( const int i )` `[inline]`

Definition at line 196 of file `newmat.h`.

## 29.284.3 Member Function Documentation

29.284.3.1 `int NEWMAT::MatrixBandWidth::Lower ( ) const` `[inline]`

Definition at line 205 of file `newmat.h`.

29.284.3.2 `MatrixBandWidth MatrixBandWidth::minimum ( const MatrixBandWidth & bw ) const`

Definition at line 186 of file `bandmat.cpp`.

29.284.3.3 `bool NEWMAT::MatrixBandWidth::operator!= ( const MatrixBandWidth & bw ) const` `[inline]`

Definition at line 203 of file `newmat.h`.

29.284.3.4 `MatrixBandWidth MatrixBandWidth::operator* ( const MatrixBandWidth & bw ) const`

Definition at line 177 of file `bandmat.cpp`.

29.284.3.5 `MatrixBandWidth MatrixBandWidth::operator+ ( const MatrixBandWidth & bw ) const`

Definition at line 168 of file `bandmat.cpp`.

29.284.3.6 `bool NEWMAT::MatrixBandWidth::operator== ( const MatrixBandWidth & bw ) const` `[inline]`

Definition at line 201 of file `newmat.h`.

29.284.3.7 `MatrixBandWidth NEWMAT::MatrixBandWidth::t ( ) const` `[inline]`

Definition at line 200 of file `newmat.h`.



29.284.3.8 `int NEWMAT::MatrixBandWidth::Upper ( ) const [inline]`

Definition at line 204 of file newmat.h.

## 29.284.4 Member Data Documentation

29.284.4.1 `int NEWMAT::MatrixBandWidth::lower`

Definition at line 193 of file newmat.h.

29.284.4.2 `int NEWMAT::MatrixBandWidth::upper`

Definition at line 194 of file newmat.h.

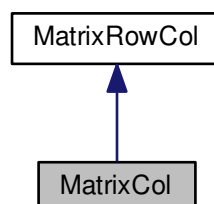
The documentation for this class was generated from the following files:

- [newmat/include/newmat.h](#)
- [newmat/src/bandmat.cpp](#)

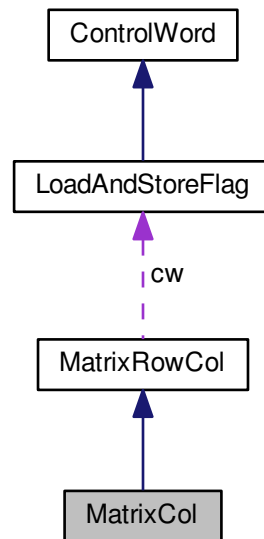
## 29.285 MatrixCol Class Reference

```
#include <newmatrc.h>
```

Inheritance diagram for MatrixCol:



Collaboration diagram for MatrixCol:



### Public Member Functions

- [MatrixCol](#) (GeneralMatrix \*, [LoadAndStoreFlag](#), int=0)
- [MatrixCol](#) (GeneralMatrix \*, Real \*, [LoadAndStoreFlag](#), int=0)
- [~MatrixCol](#) ()
- void [Next](#) ()

### Additional Inherited Members

#### 29.285.1 Detailed Description

Definition at line 114 of file newmatrc.h.

#### 29.285.2 Constructor & Destructor Documentation

29.285.2.1 `MatrixCol::MatrixCol ( GeneralMatrix * gmx, LoadAndStoreFlag cw, int col = 0 )` [inline]

Definition at line 150 of file newmatrc.h.

29.285.2.2 `MatrixCol::MatrixCol ( GeneralMatrix * gmx, Real * r, LoadAndStoreFlag cw, int col = 0 )` [inline]

Definition at line 153 of file newmatrc.h.

### 29.285.2.3 MatrixCol::~MatrixCol ( )

Definition at line 835 of file newmat3.cpp.

## 29.285.3 Member Function Documentation

### 29.285.3.1 void MatrixCol::Next ( ) [inline]

Definition at line 162 of file newmatrc.h.

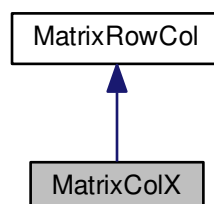
The documentation for this class was generated from the following files:

- [newmat/include/newmatrc.h](#)
- [newmat/src/newmat3.cpp](#)

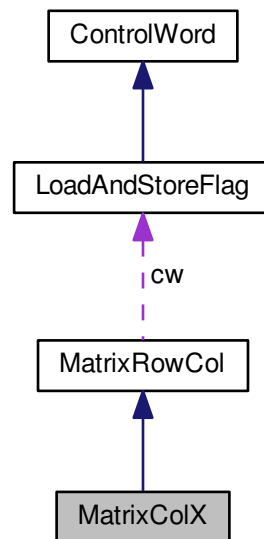
## 29.286 MatrixColX Class Reference

```
#include <newmatrc.h>
```

Inheritance diagram for MatrixColX:



Collaboration diagram for MatrixColX:



### Public Member Functions

- [MatrixColX](#) (GeneralMatrix \*, Real \*, [LoadAndStoreFlag](#), int=0)
- [~MatrixColX](#) ()
- void [Next](#) ()

### Public Attributes

- Real \* [store](#)

#### 29.286.1 Detailed Description

Definition at line 130 of file newmatrc.h.

#### 29.286.2 Constructor & Destructor Documentation

29.286.2.1 `MatrixColX::MatrixColX ( GeneralMatrix * gmx, Real * r, LoadAndStoreFlag cw, int col = 0 )` [inline]

Definition at line 157 of file newmatrc.h.

29.286.2.2 `MatrixColX::~~MatrixColX ( )`

Definition at line 837 of file newmat3.cpp.

### 29.286.3 Member Function Documentation

29.286.3.1 `void MatrixColX::Next( )` `[inline]`

Definition at line 164 of file `newmatrc.h`.

### 29.286.4 Member Data Documentation

29.286.4.1 `Real* MatrixColX::store`

Definition at line 138 of file `newmatrc.h`.

The documentation for this class was generated from the following files:

- [newmat/include/newmatrc.h](#)
- [newmat/src/newmat3.cpp](#)

## 29.287 NEWMAT::MatrixInput Class Reference

```
#include <newmat.h>
```

### Public Member Functions

- [MatrixInput](#) (`const MatrixInput &mi`)
- [MatrixInput](#) (`int nx`, `Real *rx`)
- [~MatrixInput](#) ()
- [MatrixInput operator<<](#) (`Real`)
- [MatrixInput operator<<](#) (`int f`)

### Friends

- class [GeneralMatrix](#)

### 29.287.1 Detailed Description

Definition at line 1544 of file `newmat.h`.

### 29.287.2 Constructor & Destructor Documentation

29.287.2.1 `NEWMAT::MatrixInput::MatrixInput ( const MatrixInput & mi )` `[inline]`

Definition at line 1551 of file `newmat.h`.

29.287.2.2 `NEWMAT::MatrixInput::MatrixInput ( int nx, Real * rx )` `[inline]`

Definition at line 1552 of file `newmat.h`.

29.287.2.3 `MatrixInput::~~MatrixInput ( )`

Definition at line 521 of file `newmat5.cpp`.

### 29.287.3 Member Function Documentation

29.287.3.1 `MatrixInput NEWMAT::MatrixInput::operator<< ( Real )`

29.287.3.2 `MatrixInput NEWMAT::MatrixInput::operator<< ( int f )` `[inline]`

Definition at line 1783 of file `newmat.h`.

### 29.287.4 Friends And Related Function Documentation

29.287.4.1 `friend class GeneralMatrix` `[friend]`

Definition at line 1556 of file `newmat.h`.

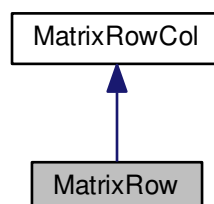
The documentation for this class was generated from the following files:

- `newmat/include/newmat.h`
- `newmat/src/newmat5.cpp`

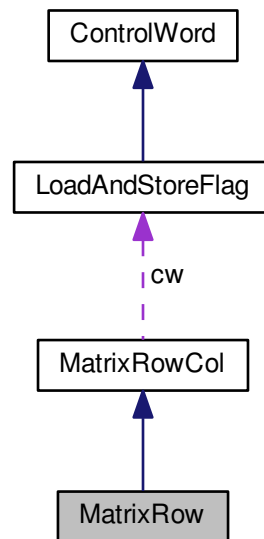
## 29.288 MatrixRow Class Reference

```
#include <newmatrc.h>
```

Inheritance diagram for `MatrixRow`:



Collaboration diagram for MatrixRow:



## Public Member Functions

- [MatrixRow](#) (GeneralMatrix \*, [LoadAndStoreFlag](#), int=0)
- [~MatrixRow](#) ()
- void [Next](#) ()

## Additional Inherited Members

### 29.288.1 Detailed Description

Definition at line 103 of file newmatrc.h.

### 29.288.2 Constructor & Destructor Documentation

29.288.2.1 [MatrixRow::MatrixRow](#) ( [GeneralMatrix](#) \* *gmx*, [LoadAndStoreFlag](#) *cwx*, int *row* = 0 ) [[inline](#)]

Definition at line 145 of file newmatrc.h.

29.288.2.2 [MatrixRow::~~MatrixRow](#) ( )

Definition at line 833 of file newmat3.cpp.

### 29.288.3 Member Function Documentation

#### 29.288.3.1 void MatrixRow::Next ( ) [inline]

Definition at line 148 of file newmatrc.h.

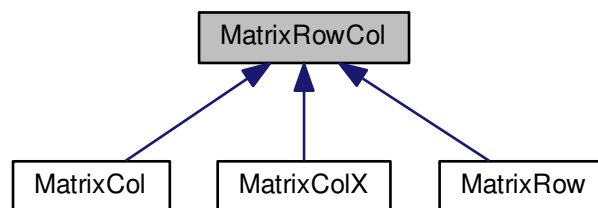
The documentation for this class was generated from the following files:

- [newmat/include/newmatrc.h](#)
- [newmat/src/newmat3.cpp](#)

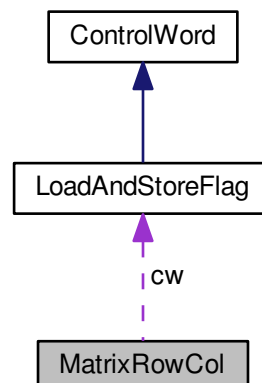
### 29.289 MatrixRowCol Class Reference

```
#include <newmatrc.h>
```

Inheritance diagram for MatrixRowCol:



Collaboration diagram for MatrixRowCol:





## Public Member Functions

- void [IncrMat](#) ()
- void [IncrDiag](#) ()
- void [IncrId](#) ()
- void [IncrUT](#) ()
- void [IncrLT](#) ()
- void [Zero](#) ()
- void [Add](#) (const [MatrixRowCol](#) &)
- void [AddScaled](#) (const [MatrixRowCol](#) &, Real)
- void [Add](#) (const [MatrixRowCol](#) &, const [MatrixRowCol](#) &)
- void [Add](#) (const [MatrixRowCol](#) &, Real)
- void [NegAdd](#) (const [MatrixRowCol](#) &, Real)
- void [Sub](#) (const [MatrixRowCol](#) &)
- void [Sub](#) (const [MatrixRowCol](#) &, const [MatrixRowCol](#) &)
- void [RevSub](#) (const [MatrixRowCol](#) &)
- void [ConCat](#) (const [MatrixRowCol](#) &, const [MatrixRowCol](#) &)
- void [Multiply](#) (const [MatrixRowCol](#) &)
- void [Multiply](#) (const [MatrixRowCol](#) &, const [MatrixRowCol](#) &)
- void [KP](#) (const [MatrixRowCol](#) &, const [MatrixRowCol](#) &)
- void [Copy](#) (const [MatrixRowCol](#) &)
- void [CopyCheck](#) (const [MatrixRowCol](#) &)
- void [Check](#) (const [MatrixRowCol](#) &)
- void [Check](#) ()
- void [Copy](#) (const Real \*&)
- void [Copy](#) (Real)
- void [Add](#) (Real)
- void [Multiply](#) (Real)
- Real [SumAbsoluteValue](#) ()
- Real [MaximumAbsoluteValue1](#) (Real r, int &i)
- Real [MinimumAbsoluteValue1](#) (Real r, int &i)
- Real [Maximum1](#) (Real r, int &i)
- Real [Minimum1](#) (Real r, int &i)
- Real [Sum](#) ()
- void [Inject](#) (const [MatrixRowCol](#) &)
- void [Negate](#) (const [MatrixRowCol](#) &)
- void [Multiply](#) (const [MatrixRowCol](#) &, Real)
- Real \* [Data](#) ()
- int [Skip](#) ()
- int [Storage](#) ()
- int [Length](#) ()
- void [Skip](#) (int i)
- void [Storage](#) (int i)
- void [Length](#) (int i)
- void [SubRowCol](#) ([MatrixRowCol](#) &, int, int) const
- [MatrixRowCol](#) ()
- [~MatrixRowCol](#) ()

## Public Attributes

- int [length](#)
- int [skip](#)
- int [storage](#)
- int [rowcol](#)
- GeneralMatrix \* [gm](#)
- Real \* [data](#)
- [LoadAndStoreFlag](#) [cw](#)

## Friends

- Real `DotProd` (`const MatrixRowCol &`, `const MatrixRowCol &`)

### 29.289.1 Detailed Description

Definition at line 33 of file `newmatrc.h`.

### 29.289.2 Constructor & Destructor Documentation

#### 29.289.2.1 `MatrixRowCol::MatrixRowCol ( )` [`inline`]

Definition at line 98 of file `newmatrc.h`.

#### 29.289.2.2 `MatrixRowCol::~~MatrixRowCol ( )`

Definition at line 824 of file `newmat3.cpp`.

### 29.289.3 Member Function Documentation

#### 29.289.3.1 `void MatrixRowCol::Add ( const MatrixRowCol & mrc )`

Definition at line 29 of file `newmat2.cpp`.

#### 29.289.3.2 `void MatrixRowCol::Add ( const MatrixRowCol & mrc1, const MatrixRowCol & mrc2 )`

Definition at line 87 of file `newmat2.cpp`.

#### 29.289.3.3 `void MatrixRowCol::Add ( const MatrixRowCol & mrc1, Real x )`

Definition at line 232 of file `newmat2.cpp`.

#### 29.289.3.4 `void MatrixRowCol::Add ( Real r )`

Definition at line 555 of file `newmat2.cpp`.

#### 29.289.3.5 `void MatrixRowCol::AddScaled ( const MatrixRowCol & mrc, Real x )`

Definition at line 40 of file `newmat2.cpp`.

29.289.3.6 void MatrixRowCol::Check ( const MatrixRowCol & *mrc1* )

Definition at line 445 of file newmat2.cpp.

29.289.3.7 void MatrixRowCol::Check ( )

Definition at line 453 of file newmat2.cpp.

29.289.3.8 void MatrixRowCol::ConCat ( const MatrixRowCol & *mrc1*, const MatrixRowCol & *mrc2* )

Definition at line 280 of file newmat2.cpp.

29.289.3.9 void MatrixRowCol::Copy ( const MatrixRowCol & *mrc1* )

Definition at line 412 of file newmat2.cpp.

29.289.3.10 void MatrixRowCol::Copy ( const Real \*& *r* )

Definition at line 529 of file newmat2.cpp.

29.289.3.11 void MatrixRowCol::Copy ( Real *r* )

Definition at line 537 of file newmat2.cpp.

29.289.3.12 void MatrixRowCol::CopyCheck ( const MatrixRowCol & *mrc1* )

Definition at line 430 of file newmat2.cpp.

29.289.3.13 Real\* MatrixRowCol::Data ( ) [inline]

Definition at line 89 of file newmatrc.h.

29.289.3.14 void MatrixRowCol::IncrDiag ( ) [inline]

Definition at line 46 of file newmatrc.h.

29.289.3.15 void MatrixRowCol::IncrId ( ) [inline]

Definition at line 47 of file newmatrc.h.

29.289.3.16 void `MatrixRowCol::IncrLT` ( ) [`inline`]

Definition at line 49 of file `newmatrc.h`.

29.289.3.17 void `MatrixRowCol::IncrMat` ( ) [`inline`]

Definition at line 45 of file `newmatrc.h`.

29.289.3.18 void `MatrixRowCol::IncrUT` ( ) [`inline`]

Definition at line 48 of file `newmatrc.h`.

29.289.3.19 void `MatrixRowCol::Inject` ( `const MatrixRowCol & mrc` )

Definition at line 62 of file `newmat2.cpp`.

29.289.3.20 void `MatrixRowCol::KP` ( `const MatrixRowCol & mrc1`, `const MatrixRowCol & mrc2` )

Definition at line 345 of file `newmat2.cpp`.

29.289.3.21 int `MatrixRowCol::Length` ( ) [`inline`]

Definition at line 92 of file `newmatrc.h`.

29.289.3.22 void `MatrixRowCol::Length` ( int *i* ) [`inline`]

Definition at line 95 of file `newmatrc.h`.

29.289.3.23 Real `MatrixRowCol::Maximum1` ( Real *r*, int & *i* )

Definition at line 593 of file `newmat2.cpp`.

29.289.3.24 Real `MatrixRowCol::MaximumAbsoluteValue1` ( Real *r*, int & *i* )

Definition at line 573 of file `newmat2.cpp`.

29.289.3.25 Real `MatrixRowCol::Minimum1` ( Real *r*, int & *i* )

Definition at line 603 of file `newmat2.cpp`.

29.289.3.26 `Real MatrixRowCol::MinimumAbsoluteValue1 ( Real r, int & i )`

Definition at line 583 of file newmat2.cpp.

29.289.3.27 `void MatrixRowCol::Multiply ( const MatrixRowCol & mrc1 )`

Definition at line 307 of file newmat2.cpp.

29.289.3.28 `void MatrixRowCol::Multiply ( const MatrixRowCol & mrc1, const MatrixRowCol & mrc2 )`

Definition at line 323 of file newmat2.cpp.

29.289.3.29 `void MatrixRowCol::Multiply ( Real r )`

Definition at line 549 of file newmat2.cpp.

29.289.3.30 `void MatrixRowCol::Multiply ( const MatrixRowCol & mrc1, Real s )`

Definition at line 479 of file newmat2.cpp.

29.289.3.31 `void MatrixRowCol::NegAdd ( const MatrixRowCol & mrc1, Real x )`

Definition at line 248 of file newmat2.cpp.

29.289.3.32 `void MatrixRowCol::Negate ( const MatrixRowCol & mrc1 )`

Definition at line 463 of file newmat2.cpp.

29.289.3.33 `void MatrixRowCol::RevSub ( const MatrixRowCol & mrc1 )`

Definition at line 264 of file newmat2.cpp.

29.289.3.34 `int MatrixRowCol::Skip ( ) [inline]`

Definition at line 90 of file newmatrc.h.

29.289.3.35 `void MatrixRowCol::Skip ( int i ) [inline]`

Definition at line 93 of file newmatrc.h.

29.289.3.36 `int MatrixRowCol::Storage ( ) [inline]`

Definition at line 91 of file newmatrc.h.

29.289.3.37 `void MatrixRowCol::Storage ( int i ) [inline]`

Definition at line 94 of file newmatrc.h.

29.289.3.38 `void MatrixRowCol::Sub ( const MatrixRowCol & mrc )`

Definition at line 51 of file newmat2.cpp.

29.289.3.39 `void MatrixRowCol::Sub ( const MatrixRowCol & mrc1, const MatrixRowCol & mrc2 )`

Definition at line 159 of file newmat2.cpp.

29.289.3.40 `void MatrixRowCol::SubRowCol ( MatrixRowCol & mrc, int skip1, int l1 ) const`

Definition at line 620 of file newmat2.cpp.

29.289.3.41 `Real MatrixRowCol::Sum ( )`

Definition at line 612 of file newmat2.cpp.

29.289.3.42 `Real MatrixRowCol::SumAbsoluteValue ( )`

Definition at line 562 of file newmat2.cpp.

29.289.3.43 `void MatrixRowCol::Zero ( )`

Definition at line 543 of file newmat2.cpp.

## 29.289.4 Friends And Related Function Documentation

29.289.4.1 `Real DotProd ( const MatrixRowCol & mrc1, const MatrixRowCol & mrc2 ) [friend]`

Definition at line 73 of file newmat2.cpp.

## 29.289.5 Member Data Documentation

### 29.289.5.1 LoadAndStoreFlag MatrixRowCol::cw

Definition at line 44 of file newmatrc.h.

### 29.289.5.2 Real\* MatrixRowCol::data

Definition at line 43 of file newmatrc.h.

### 29.289.5.3 GeneralMatrix\* MatrixRowCol::gm

Definition at line 42 of file newmatrc.h.

### 29.289.5.4 int MatrixRowCol::length

Definition at line 38 of file newmatrc.h.

### 29.289.5.5 int MatrixRowCol::rowcol

Definition at line 41 of file newmatrc.h.

### 29.289.5.6 int MatrixRowCol::skip

Definition at line 39 of file newmatrc.h.

### 29.289.5.7 int MatrixRowCol::storage

Definition at line 40 of file newmatrc.h.

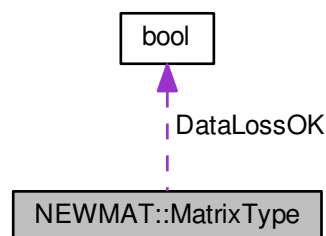
The documentation for this class was generated from the following files:

- [newmat/include/newmatrc.h](#)
- [newmat/src/newmat2.cpp](#)
- [newmat/src/newmat3.cpp](#)

## 29.290 NEWMAT::MatrixType Class Reference

```
#include <newmat.h>
```

Collaboration diagram for NEWMAT::MatrixType:



### Public Types

- enum [Attribute](#) {  
[Valid](#) = 1, [Diagonal](#) = 2, [Symmetric](#) = 4, [Band](#) = 8,  
[Lower](#) = 16, [Upper](#) = 32, [LUDeco](#) = 64, [Ones](#) = 128 }
- enum {  
[US](#) = 0, [UT](#) = Valid + Upper, [LT](#) = Valid + Lower, [Rt](#) = Valid,  
[Sm](#) = Valid + Symmetric, [Dg](#) = Valid + Diagonal + Band + Lower + Upper + Symmetric, [ld](#), [RV](#) = Valid,  
[CV](#) = Valid, [BM](#) = Valid + Band, [UB](#) = Valid + Band + Upper, [LB](#) = Valid + Band + Lower,  
[SB](#) = Valid + Band + Symmetric, [Ct](#) = Valid + LUDeco, [BC](#) = Valid + Band + LUDeco }

### Public Member Functions

- [MatrixType](#) ()
- [MatrixType](#) (int i)
- [MatrixType](#) (int i, bool dlok)
- [MatrixType](#) (const [MatrixType](#) &mt)
- void [operator=](#) (const [MatrixType](#) &mt)
- void [SetDataLossOK](#) ()
- int [operator+](#) () const
- [MatrixType](#) [operator+](#) ([MatrixType](#) mt) const
- [MatrixType](#) [operator\\*](#) (const [MatrixType](#) &) const
- [MatrixType](#) SP (const [MatrixType](#) &) const
- [MatrixType](#) KP (const [MatrixType](#) &) const
- [MatrixType](#) [operator|](#) (const [MatrixType](#) &mt) const
- [MatrixType](#) [operator&](#) (const [MatrixType](#) &mt) const
- bool [operator>=](#) ([MatrixType](#) mt) const
- bool [operator<](#) ([MatrixType](#) mt) const
- bool [operator==](#) ([MatrixType](#) t) const
- bool [operator!=](#) ([MatrixType](#) t) const
- bool [operator!](#) () const



- [MatrixType i \(\) const](#)
- [MatrixType t \(\) const](#)
- [MatrixType AddEqualEI \(\) const](#)
- [MatrixType MultRHS \(\) const](#)
- [MatrixType sub \(\) const](#)
- [MatrixType ssub \(\) const](#)
- [GeneralMatrix \\* New \(\) const](#)
- [GeneralMatrix \\* New \(int, int, BaseMatrix \\*\) const](#)
- [const char \\* Value \(\) const](#)
- [bool IsBand \(\) const](#)
- [bool IsDiagonal \(\) const](#)
- [bool IsSymmetric \(\) const](#)
- [bool CannotConvert \(\) const](#)

### Static Public Member Functions

- [static int nTypes \(\)](#)

### Public Attributes

- [int attribute](#)
- [bool DataLossOK](#)

### Friends

- [bool Rectangular \(MatrixType a, MatrixType b, MatrixType c\)](#)
- [bool Compare \(const MatrixType &, MatrixType &\)](#)

## 29.290.1 Detailed Description

Definition at line 98 of file newmat.h.

## 29.290.2 Member Enumeration Documentation

### 29.290.2.1 anonymous enum

Enumerator

***US***  
***UT***  
***LT***  
***Rt***  
***Sm***  
***Dg***  
***Id***  
***RV***  
***CV***  
***BM***  
***UB***  
***LB***  
***SB***  
***Ct***  
***BC***

Definition at line 110 of file newmat.h.

### 29.290.2.2 enum NEWMAT::MatrixType::Attribute

Enumerator

***Valid***  
***Diagonal***  
***Symmetric***  
***Band***  
***Lower***  
***Upper***  
***LUDeco***  
***Ones***

Definition at line 101 of file newmat.h.

### 29.290.3 Constructor & Destructor Documentation

#### 29.290.3.1 NEWMAT::MatrixType::MatrixType ( ) [inline]

Definition at line 136 of file newmat.h.

#### 29.290.3.2 NEWMAT::MatrixType::MatrixType ( int *i* ) [inline]

Definition at line 137 of file newmat.h.

#### 29.290.3.3 NEWMAT::MatrixType::MatrixType ( int *i*, bool *dlok* ) [inline]

Definition at line 138 of file newmat.h.

#### 29.290.3.4 NEWMAT::MatrixType::MatrixType ( const MatrixType & *mt* ) [inline]

Definition at line 139 of file newmat.h.

### 29.290.4 Member Function Documentation

#### 29.290.4.1 MatrixType NEWMAT::MatrixType::AddEqualEI ( ) const [inline]

Definition at line 165 of file newmat.h.

#### 29.290.4.2 bool NEWMAT::MatrixType::CannotConvert ( ) const [inline]

Definition at line 182 of file newmat.h.

**29.290.4.3 MatrixType MatrixType::i ( ) const**

Definition at line 63 of file newmat1.cpp.

**29.290.4.4 bool NEWMAT::MatrixType::IsBand ( ) const [inline]**

Definition at line 179 of file newmat.h.

**29.290.4.5 bool NEWMAT::MatrixType::IsDiagonal ( ) const [inline]**

Definition at line 180 of file newmat.h.

**29.290.4.6 bool NEWMAT::MatrixType::IsSymmetric ( ) const [inline]**

Definition at line 181 of file newmat.h.

**29.290.4.7 MatrixType MatrixType::KP ( const MatrixType & mt ) const**

Definition at line 52 of file newmat1.cpp.

**29.290.4.8 MatrixType MatrixType::MultRHS ( ) const**

Definition at line 81 of file newmat1.cpp.

**29.290.4.9 GeneralMatrix\* NEWMAT::MatrixType::New ( ) const****29.290.4.10 GeneralMatrix\* NEWMAT::MatrixType::New ( int, int, BaseMatrix \* ) const****29.290.4.11 static int NEWMAT::MatrixType::nTypes ( ) [inline],[static]**

Definition at line 129 of file newmat.h.

**29.290.4.12 bool NEWMAT::MatrixType::operator! ( ) const [inline]**

Definition at line 162 of file newmat.h.

**29.290.4.13 bool NEWMAT::MatrixType::operator!=( MatrixType t ) const [inline]**

Definition at line 160 of file newmat.h.

29.290.4.14 **MatrixType** NEWMAT::MatrixType::operator& ( **const MatrixType & mt** ) const [inline]

Definition at line 152 of file newmat.h.

29.290.4.15 **MatrixType** MatrixType::operator\* ( **const MatrixType & mt** ) const

Definition at line 29 of file newmat1.cpp.

29.290.4.16 **int** NEWMAT::MatrixType::operator+ ( ) const [inline]

Definition at line 144 of file newmat.h.

29.290.4.17 **MatrixType** NEWMAT::MatrixType::operator+ ( **MatrixType mt** ) const [inline]

Definition at line 145 of file newmat.h.

29.290.4.18 **bool** NEWMAT::MatrixType::operator< ( **MatrixType mt** ) const [inline]

Definition at line 156 of file newmat.h.

29.290.4.19 **void** NEWMAT::MatrixType::operator= ( **const MatrixType & mt** ) [inline]

Definition at line 141 of file newmat.h.

29.290.4.20 **bool** NEWMAT::MatrixType::operator== ( **MatrixType t** ) const [inline]

Definition at line 158 of file newmat.h.

29.290.4.21 **bool** NEWMAT::MatrixType::operator>= ( **MatrixType mt** ) const [inline]

Definition at line 154 of file newmat.h.

29.290.4.22 **MatrixType** NEWMAT::MatrixType::operator| ( **const MatrixType & mt** ) const [inline]

Definition at line 150 of file newmat.h.

29.290.4.23 **void** NEWMAT::MatrixType::SetDataLossOK ( ) [inline]

Definition at line 143 of file newmat.h.

29.290.4.24 **MatrixType** MatrixType::SP ( const MatrixType & *mt* ) const

Definition at line 37 of file newmat1.cpp.

29.290.4.25 **MatrixType** NEWMAT::MatrixType::ssub ( ) const [inline]

Definition at line 170 of file newmat.h.

29.290.4.26 **MatrixType** NEWMAT::MatrixType::sub ( ) const [inline]

Definition at line 168 of file newmat.h.

29.290.4.27 **MatrixType** MatrixType::t ( ) const

Definition at line 71 of file newmat1.cpp.

29.290.4.28 **const char \*** MatrixType::Value ( ) const

Definition at line 95 of file newmat1.cpp.

## 29.290.5 Friends And Related Function Documentation

29.290.5.1 **bool** Compare ( const MatrixType &, MatrixType & ) [friend]

29.290.5.2 **bool** Rectangular ( MatrixType *a*, MatrixType *b*, MatrixType *c* ) [friend]

## 29.290.6 Member Data Documentation

29.290.6.1 **int** NEWMAT::MatrixType::attribute

Definition at line 132 of file newmat.h.

29.290.6.2 **bool** NEWMAT::MatrixType::DataLossOK

Definition at line 133 of file newmat.h.

The documentation for this class was generated from the following files:

- [newmat/include/newmat.h](#)
- [newmat/src/newmat1.cpp](#)

## 29.291 Go::MatrixXD< T, Dim > Class Template Reference

```
#include <MatrixXD.h>
```

### Public Member Functions

- [MatrixXD \(\)](#)  
*Default constructor.*
- [~MatrixXD \(\)](#)  
*Default destructor.*
- [T & operator\(\) \(int row, int col\)](#)  
*Element access operator.*
- [T \\*\\* get \(\)](#)  
*Access data array directly.*
- [const T & operator\(\) \(int row, int col\) const](#)  
*Const element access operator.*
- [void zero \(\)](#)  
*Set all elements to zero.*
- [void identity \(\)](#)  
*Identity matrix.*
- [void transpose \(\)](#)  
*Transpose matrix.*
- [void setToRotation \(T angle, T x, T y, T z\)](#)
- [void setToRotation \(const Vector3D &p, const Vector3D &q\)](#)
- [MatrixXD operator\\* \(const MatrixXD &other\) const](#)  
*Matrix-matrix multiplication.*
- [MatrixXD & operator\\* = \(const MatrixXD &other\)](#)  
*Multiply this matrix with a matrix.*
- [MatrixXD operator\\* \(T scalar\) const](#)  
*Multiplication by a scalar.*
- [MatrixXD & operator\\* = \(T scalar\)](#)  
*Multiplication by a scalar.*
- [MatrixXD operator+ \(const MatrixXD &other\) const](#)  
*Addition.*
- [MatrixXD & operator+ = \(const MatrixXD &other\)](#)  
*Add a matrix to this matrix.*
- [MatrixXD operator- \(\) const](#)  
*Negation.*
- [template<class VectorType > VectorType operator\\* \(const VectorType &vec\) const](#)  
*Matrix-vector multiplication.*
- [template<class FromConstIteratorType, class ToIteratorType > void mult \(const FromConstIteratorType from, const ToIteratorType to\) const](#)  
*Matrix-vector multiplication.*
- [T det \(\) const](#)  
*Determinant.*
- [T trace \(\) const](#)  
*Trace.*
- [T frobeniusNorm \(\) const](#)  
*Frobenius norm.*

- [MatrixXD< T, Dim-1 > submatrix](#) (int r, int c) const  
*Submatrix with given row and column removed.*
- [template<>](#)  
void [setToRotation](#) (double angle, double x, double y, double z)
- [template<>](#)  
void [setToRotation](#) (double angle, double x, double y, double z)
- [template<>](#)  
void [setToRotation](#) (const [Vector3D](#) &p, const [Vector3D](#) &q)

### 29.291.1 Detailed Description

```
template<typename T, int Dim>
class Go::MatrixXD< T, Dim >
```

Square n-dimensional (compile time constant dim) matrix.

Definition at line 59 of file MatrixXD.h.

### 29.291.2 Constructor & Destructor Documentation

29.291.2.1 `template<typename T, int Dim> Go::MatrixXD< T, Dim >::MatrixXD ( ) [inline]`

Default constructor.

Definition at line 63 of file MatrixXD.h.

29.291.2.2 `template<typename T, int Dim> Go::MatrixXD< T, Dim >::~~MatrixXD ( ) [inline]`

Default destructor.

Definition at line 66 of file MatrixXD.h.

### 29.291.3 Member Function Documentation

29.291.3.1 `template<typename T, int Dim> T Go::MatrixXD< T, Dim >::det ( ) const`

Determinant.

29.291.3.2 `template<typename T, int Dim> T Go::MatrixXD< T, Dim >::frobeniusNorm ( ) const [inline]`

Frobenius norm.

Definition at line 245 of file MatrixXD.h.

29.291.3.3 `template<typename T, int Dim> T** Go::MatrixXD< T, Dim >::get ( ) [inline]`

Access data array directly.

Definition at line 75 of file MatrixXD.h.

29.291.3.4 `template<typename T, int Dim> void Go::MatrixXD< T, Dim >::identity ( ) [inline]`

[Identity](#) matrix.

Definition at line 94 of file MatrixXD.h.

29.291.3.5 `template<typename T, int Dim> template<class FromConstIteratorType , class ToIteratorType > void Go::MatrixXD< T, Dim >::mult ( const FromConstIteratorType from, const ToIteratorType to ) const [inline]`

Matrix-vector multiplication.

Definition at line 216 of file MatrixXD.h.

29.291.3.6 `template<typename T, int Dim> T& Go::MatrixXD< T, Dim >::operator() ( int row, int col ) [inline]`

Element access operator.

Definition at line 69 of file MatrixXD.h.

29.291.3.7 `template<typename T, int Dim> const T& Go::MatrixXD< T, Dim >::operator() ( int row, int col ) const [inline]`

Const element access operator.

Definition at line 78 of file MatrixXD.h.

29.291.3.8 `template<typename T, int Dim> MatrixXD Go::MatrixXD< T, Dim >::operator* ( const MatrixXD< T, Dim > & other ) const [inline]`

Matrix-matrix multiplication.

Definition at line 125 of file MatrixXD.h.

29.291.3.9 `template<typename T, int Dim> MatrixXD Go::MatrixXD< T, Dim >::operator* ( T scalar ) const [inline]`

Multiplication by a scalar.

Definition at line 149 of file MatrixXD.h.



```
29.291.3.10 template<typename T, int Dim> template<class VectorType > VectorType Go::MatrixXD< T, Dim
 >::operator* (const VectorType & vec) const [inline]
```

Matrix-vector multiplication.

Definition at line 200 of file MatrixXD.h.

```
29.291.3.11 template<typename T, int Dim> MatrixXD& Go::MatrixXD< T, Dim >::operator*= (const MatrixXD< T,
 Dim > & other) [inline]
```

Multiply this matrix with a matrix.

Definition at line 142 of file MatrixXD.h.

```
29.291.3.12 template<typename T, int Dim> MatrixXD& Go::MatrixXD< T, Dim >::operator*= (T scalar) [inline]
```

Multiplication by a scalar.

Definition at line 157 of file MatrixXD.h.

```
29.291.3.13 template<typename T, int Dim> MatrixXD Go::MatrixXD< T, Dim >::operator+ (const MatrixXD< T, Dim
 > & other) const [inline]
```

Addition.

Definition at line 168 of file MatrixXD.h.

```
29.291.3.14 template<typename T, int Dim> MatrixXD& Go::MatrixXD< T, Dim >::operator+= (const MatrixXD< T,
 Dim > & other) [inline]
```

Add a matrix to this matrix.

Definition at line 176 of file MatrixXD.h.

```
29.291.3.15 template<typename T, int Dim> MatrixXD Go::MatrixXD< T, Dim >::operator- () const [inline]
```

Negation.

Definition at line 187 of file MatrixXD.h.

```
29.291.3.16 template<typename T, int Dim> void Go::MatrixXD< T, Dim >::setToRotation (T angle, T x, T y, T z)
 [inline]
```

Similar to `glRotate()`, sets the matrix to be a rotation of `angle` radians about the axis given by `(x, y, z)`. `(x, y, z)` does not have to be normalized. Only makes sense for `Dim == 3` or `Dim == 4`.

Definition at line 281 of file MatrixXD.h.

29.291.3.17 `template<typename T, int Dim> void Go::MatrixXD< T, Dim >::setToRotation ( const Vector3D & p, const Vector3D & q ) [inline]`

Sets the matrix to be a rotation that takes the point p to q. p and q are assumed to lie on the 2-sphere. Only makes sense for Dim == 3.

Definition at line 343 of file MatrixXD.h.

29.291.3.18 `template<> void Go::MatrixXD< double, 3 >::setToRotation ( double angle, double x, double y, double z ) [inline]`

Definition at line 290 of file MatrixXD.h.

29.291.3.19 `template<> void Go::MatrixXD< double, 4 >::setToRotation ( double angle, double x, double y, double z ) [inline]`

Definition at line 326 of file MatrixXD.h.

29.291.3.20 `template<> void Go::MatrixXD< double, 3 >::setToRotation ( const Vector3D & p, const Vector3D & q ) [inline]`

Definition at line 352 of file MatrixXD.h.

29.291.3.21 `template<typename T, int Dim> MatrixXD<T, Dim-1> Go::MatrixXD< T, Dim >::submatrix ( int r, int c ) const [inline]`

Submatrix with given row and column removed.

Definition at line 258 of file MatrixXD.h.

29.291.3.22 `template<typename T, int Dim> T Go::MatrixXD< T, Dim >::trace ( ) const [inline]`

Trace.

Definition at line 233 of file MatrixXD.h.

29.291.3.23 `template<typename T, int Dim> void Go::MatrixXD< T, Dim >::transpose ( ) [inline]`

Transpose matrix.

Definition at line 103 of file MatrixXD.h.

29.291.3.24 `template<typename T, int Dim> void Go::MatrixXD< T, Dim >::zero ( ) [inline]`

Set all elements to zero.

Definition at line 84 of file MatrixXD.h.

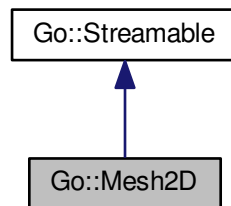
The documentation for this class was generated from the following file:

- [gotools-core/include/GoTools/Utils/MatrixXD.h](#)

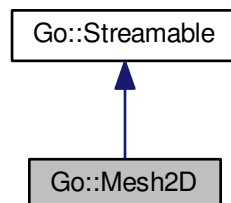
## 29.292 Go::Mesh2D Class Reference

```
#include <Mesh2D.h>
```

Inheritance diagram for Go::Mesh2D:



Collaboration diagram for Go::Mesh2D:



## Public Member Functions

- [Mesh2D](#) ()
- [Mesh2D](#) (std::istream &is)
- [template<typename Iterator > Mesh2D](#) (Iterator kx\_start, Iterator kx\_end, Iterator ky\_start, Iterator ky\_end)
- [template<typename Array > Mesh2D](#) (const Array &xknots, const Array &yknots)
- virtual void [read](#) (std::istream &is)
- virtual void [write](#) (std::ostream &os) const
- void [swap](#) (Mesh2D &rhs)
- int [nu](#) (Direction2D d, int ix, int start, int end) const
- int [numDistinctKnots](#) (Direction2D d) const
- double [kval](#) (Direction2D d, int ix) const
- double [minParam](#) (Direction2D d) const
- double [maxParam](#) (Direction2D d) const
- const double \*const [knotsBegin](#) (Direction2D d) const
- const double \*const [knotsEnd](#) (Direction2D d) const
- std::vector< double > [getKnots](#) (Direction2D d, int ix, bool right=true) const
- int [extent](#) (Direction2D d, int ix, int start, int mult) const
- int [largestMultInLine](#) (Direction2D d, int ix) const
- int [minMultInLine](#) (Direction2D d, int ix) const
- int [knotIntervalFuzzy](#) (Direction2D d, double &par, double eps) const
- int [getKnotIdx](#) (Direction2D d, double &par, double eps) const
- std::vector< std::pair< int, int > > [segments](#) (Direction2D dir, int ix, int threshold=1) const
- std::vector< std::pair< int, int > > [zeroSegments](#) (Direction2D dir, int ix) const
- [Mesh2DIterator](#) [begin](#) () const
- [Mesh2DIterator](#) [end](#) () const
- [IndexMesh2DIterator](#) [indexMeshBegin](#) () const
- [IndexMesh2DIterator](#) [indexMeshEnd](#) () const
- int [firstMeshVeclx](#) (Direction2D d) const
- int [lastMeshVeclx](#) (Direction2D d) const
- shared\_ptr< Mesh2D > [subMesh](#) (int ix1, int ix2, int iy1, int iy2) const
- void [setMult](#) (Direction2D d, int ix, int start, int end, int mult)
- void [incrementMult](#) (Direction2D d, int ix, int start, int end, int mult)
- int [insertLine](#) (Direction2D d, double kval, int mult=0)
- void [setParameterDomain](#) (double u1, double u2, double v1, double v2)
- void [swapParameterDirection](#) ()
- void [reverseParameterDirection](#) (bool dir\_is\_u)

### 29.292.1 Detailed Description

Definition at line 67 of file Mesh2D.h.

### 29.292.2 Constructor & Destructor Documentation

#### 29.292.2.1 Go::Mesh2D::Mesh2D ( ) [inline]

Definition at line 74 of file Mesh2D.h.

29.292.2.2 `Go::Mesh2D::Mesh2D ( std::istream & is )`

29.292.2.3 `template<typename Iterator > Go::Mesh2D::Mesh2D ( Iterator kx_start, Iterator kx_end, Iterator ky_start, Iterator ky_end )`

Definition at line 297 of file Mesh2D.h.

29.292.2.4 `template<typename Array > Go::Mesh2D::Mesh2D ( const Array & xknots, const Array & yknots )`

Definition at line 305 of file Mesh2D.h.

### 29.292.3 Member Function Documentation

29.292.3.1 `Mesh2DIterator Go::Mesh2D::begin ( ) const`

29.292.3.2 `Mesh2DIterator Go::Mesh2D::end ( ) const`

29.292.3.3 `int Go::Mesh2D::extent ( Direction2D d, int ix, int start, int mult ) const`

29.292.3.4 `int Go::Mesh2D::firstMeshVecIx ( Direction2D d ) const [inline]`

Definition at line 378 of file Mesh2D.h.

29.292.3.5 `int Go::Mesh2D::getKnotIdx ( Direction2D d, double & par, double eps ) const`

29.292.3.6 `std::vector<double> Go::Mesh2D::getKnots ( Direction2D d, int ix, bool right = true ) const`

29.292.3.7 `void Go::Mesh2D::incrementMult ( Direction2D d, int ix, int start, int end, int mult )`

29.292.3.8 `IndexMesh2DIterator Go::Mesh2D::indexMeshBegin ( ) const`

29.292.3.9 `IndexMesh2DIterator Go::Mesh2D::indexMeshEnd ( ) const`

29.292.3.10 `int Go::Mesh2D::insertLine ( Direction2D d, double kval, int mult = 0 )`

29.292.3.11 `int Go::Mesh2D::knotIntervalFuzzy ( Direction2D d, double & par, double eps ) const`

29.292.3.12 `const double *const Go::Mesh2D::knotsBegin ( Direction2D d ) const [inline]`

Definition at line 416 of file Mesh2D.h.

29.292.3.13 `const double *const Go::Mesh2D::knotsEnd ( Direction2D d ) const [inline]`

Definition at line 423 of file Mesh2D.h.

29.292.3.14 `double Go::Mesh2D::kval ( Direction2D d, int ix ) const [inline]`

Definition at line 391 of file Mesh2D.h.

29.292.3.15 `int Go::Mesh2D::largestMultInLine ( Direction2D d, int ix ) const`

29.292.3.16 `int Go::Mesh2D::lastMeshVeclx ( Direction2D d ) const [inline]`

Definition at line 384 of file Mesh2D.h.

29.292.3.17 `double Go::Mesh2D::maxParam ( Direction2D d ) const [inline]`

Definition at line 365 of file Mesh2D.h.

29.292.3.18 `int Go::Mesh2D::minMultInLine ( Direction2D d, int ix ) const`

29.292.3.19 `double Go::Mesh2D::minParam ( Direction2D d ) const [inline]`

Definition at line 358 of file Mesh2D.h.

29.292.3.20 `int Go::Mesh2D::nu ( Direction2D d, int ix, int start, int end ) const`

29.292.3.21 `int Go::Mesh2D::numDistinctKnots ( Direction2D d ) const [inline]`

Definition at line 372 of file Mesh2D.h.

29.292.3.22 `virtual void Go::Mesh2D::read ( std::istream & is ) [virtual]`

read object from stream

Parameters

<i>is</i>	stream from which object is read
-----------	----------------------------------

Implements [Go::Streamable](#).

29.292.3.23 `void Go::Mesh2D::reverseParameterDirection ( bool dir_is_u )`

29.292.3.24 `std::vector<std::pair<int, int> > Go::Mesh2D::segments ( Direction2D dir, int ix, int threshold = 1 ) const`

29.292.3.25 `void Go::Mesh2D::setMult ( Direction2D d, int ix, int start, int end, int mult )`

29.292.3.26 void Go::Mesh2D::setParameterDomain ( double *u1*, double *u2*, double *v1*, double *v2* )

29.292.3.27 shared\_ptr<Mesh2D> Go::Mesh2D::subMesh ( int *ix1*, int *ix2*, int *iy1*, int *iy2* ) const

29.292.3.28 void Go::Mesh2D::swap ( Mesh2D & *rhs* )

29.292.3.29 void Go::Mesh2D::swapParameterDirection ( )

29.292.3.30 virtual void Go::Mesh2D::write ( std::ostream & *os* ) const [virtual]

write object to stream

Parameters

<i>os</i>	stream to which object is written
-----------	-----------------------------------

Implements [Go::Streamable](#).

29.292.3.31 std::vector<std::pair<int, int> > Go::Mesh2D::zeroSegments ( Direction2D *dir*, int *ix* ) const

The documentation for this class was generated from the following file:

- [Irsplines2D/include/GoTools/Irsplines2D/Mesh2D.h](#)

## 29.293 Go::Mesh2DIterator Class Reference

```
#include <Mesh2DIterator.h>
```

### Public Member Functions

- [Mesh2DIterator](#) ()
- [Mesh2DIterator](#) (const [Mesh2D](#) &*m*, int *u\_ix*, int *v\_ix*)
- [Mesh2DIterator](#) & [operator++](#) ()
- [bool operator==](#) (const [Mesh2DIterator](#) &*rhs*) const
- [bool operator!=](#) (const [Mesh2DIterator](#) &*rhs*) const
- void [swap](#) ([Mesh2DIterator](#) &*rhs*)
- [const](#) std::array< int, 4 > & [operator\\*](#) () const

### 29.293.1 Detailed Description

Definition at line 52 of file [Mesh2DIterator.h](#).

## 29.293.2 Constructor & Destructor Documentation

29.293.2.1 `Go::Mesh2DIterator::Mesh2DIterator ( ) [inline]`

Definition at line 58 of file Mesh2DIterator.h.

29.293.2.2 `Go::Mesh2DIterator::Mesh2DIterator ( const Mesh2D & m, int u_ix, int v_ix )`

## 29.293.3 Member Function Documentation

29.293.3.1 `bool Go::Mesh2DIterator::operator!=( const Mesh2DIterator & rhs ) const [inline]`

Definition at line 77 of file Mesh2DIterator.h.

29.293.3.2 `const std::array<int, 4>& Go::Mesh2DIterator::operator*( ) const [inline]`

Definition at line 92 of file Mesh2DIterator.h.

29.293.3.3 `Mesh2DIterator& Go::Mesh2DIterator::operator++ ( )`

29.293.3.4 `bool Go::Mesh2DIterator::operator==( const Mesh2DIterator & rhs ) const [inline]`

Definition at line 72 of file Mesh2DIterator.h.

29.293.3.5 `void Go::Mesh2DIterator::swap ( Mesh2DIterator & rhs ) [inline]`

Definition at line 82 of file Mesh2DIterator.h.

The documentation for this class was generated from the following file:

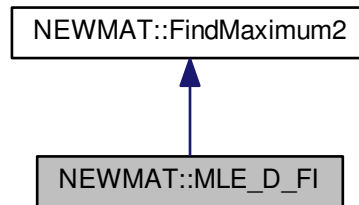
- [Irsplines2D/include/GoTools/Irsplines2D/Mesh2DIterator.h](#)



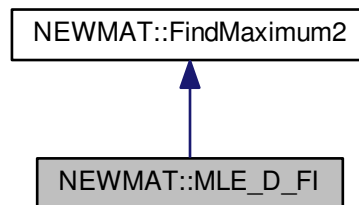
## 29.294 NEWMAT::MLE\_D\_FI Class Reference

```
#include <newmatnl.h>
```

Inheritance diagram for NEWMAT::MLE\_D\_FI:



Collaboration diagram for NEWMAT::MLE\_D\_FI:



### Public Member Functions

- [MLE\\_D\\_FI](#) ([LL\\_D\\_FI](#) &ll, int lim=1000, [Real](#) criterion=0.0001)
- void [Fit](#) ([ColumnVector](#) &Parameters)
- void [GetStandardErrors](#) ([ColumnVector](#) &)
- void [GetCorrelations](#) ([SymmetricMatrix](#) &)

### 29.294.1 Detailed Description

Definition at line 282 of file newmatnl.h.

### 29.294.2 Constructor & Destructor Documentation

29.294.2.1 [NEWMAT::MLE\\_D\\_FI::MLE\\_D\\_FI](#) ([LL\\_D\\_FI](#) &ll, int *lim* = 1000, [Real](#) *criterion* = 0.0001 ) [inline]

Definition at line 299 of file newmatnl.h.

### 29.294.3 Member Function Documentation

29.294.3.1 void MLE\_D\_Fl::Fit ( *ColumnVector* & *Parameters* )

Definition at line 228 of file newmatnl.cpp.

29.294.3.2 void MLE\_D\_Fl::GetCorrelations ( *SymmetricMatrix* & *Corr* )

Definition at line 250 of file newmatnl.cpp.

29.294.3.3 void MLE\_D\_Fl::GetStandardErrors ( *ColumnVector* & *SEX* )

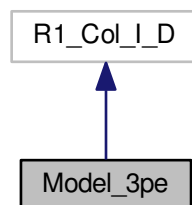
Definition at line 247 of file newmatnl.cpp.

The documentation for this class was generated from the following files:

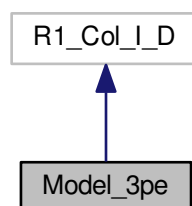
- newmat/include/newmatnl.h
- newmat/src/newmatnl.cpp

### 29.295 Model\_3pe Class Reference

Inheritance diagram for Model\_3pe:



Collaboration diagram for Model\_3pe:



## Public Member Functions

- [Model\\_3pe](#) (`const` ColumnVector &X\_Values)
- Real `operator()` (`int`)
- `bool` `IsValid` ()
- [ReturnMatrix Derivatives](#) ()

### 29.295.1 Detailed Description

Definition at line 20 of file `nl_ex.cpp`.

### 29.295.2 Constructor & Destructor Documentation

29.295.2.1 `Model_3pe::Model_3pe ( const ColumnVector & X_Values )` [`inline`]

Definition at line 25 of file `nl_ex.cpp`.

### 29.295.3 Member Function Documentation

29.295.3.1 `ReturnMatrix Model_3pe::Derivatives ( )` [`inline`]

Definition at line 31 of file `nl_ex.cpp`.

29.295.3.2 `bool Model_3pe::IsValid ( )` [`inline`]

Definition at line 29 of file `nl_ex.cpp`.

29.295.3.3 `Real Model_3pe::operator() ( int i )`

Definition at line 34 of file `nl_ex.cpp`.

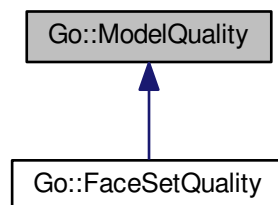
The documentation for this class was generated from the following file:

- [newmat/app/nl\\_ex.cpp](#)

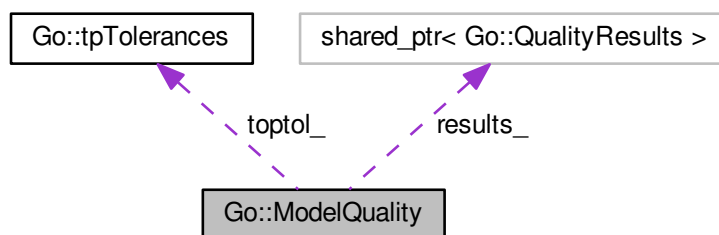
## 29.296 Go::ModelQuality Class Reference

```
#include <ModelQuality.h>
```

Inheritance diagram for Go::ModelQuality:



Collaboration diagram for Go::ModelQuality:



### Public Member Functions

- `ModelQuality` (`double` gap, `double` kink, `double` approx)
- `ModelQuality` (`const tpTolerances` &toptol, `double` approx)
- `virtual ~ModelQuality` ()
- `void setMiniElementSize` (`double` element\_size)
- `void setCurvatureRadius` (`double` radius)
- `virtual void degenSurfaces` (`std::vector< shared_ptr< ParamSurface > >` &deg\_sfs)  
*Does not generate new geometry.*
- `virtual void degenerateSfCorners` (`std::vector< shared_ptr< fitPoint > >` &deg\_corners)  
*Generates new points, pointers to existing surfaces.*
- `virtual void identicalVertices` (`std::vector< std::pair< shared_ptr< Vertex >, shared_ptr< Vertex > >` > &identical\_vertices)  
*Uses existing topological entity.*

- virtual void `identicalOrEmbeddedEdges` (std::vector< std::pair< shared\_ptr< [ftEdge](#) >, shared\_ptr< [ftEdge](#) > > > &identical\_edges, std::vector< std::pair< shared\_ptr< [ftEdge](#) >, shared\_ptr< [ftEdge](#) > > > &embedded\_edges)
 

*Uses existing topological entities. May contain generated curves.*
- virtual void `identicalOrEmbeddedFaces` (std::vector< std::pair< shared\_ptr< [ftSurface](#) >, shared\_ptr< [ftSurface](#) > > > &identical\_faces, std::vector< std::pair< shared\_ptr< [ftSurface](#) >, shared\_ptr< [ftSurface](#) > > > &embedded\_faces)
 

*Uses existing topological entities. Points to existing surfaces.*
- virtual void `miniEdges` (std::vector< shared\_ptr< [ftEdge](#) > > &mini\_edges)
 

*Uses existing topological entities.*
- virtual void `miniSurfaces` (std::vector< shared\_ptr< [ftSurface](#) > > &mini\_surfaces)
 

*Uses existing topological entities. Points to existing surfaces.*
- virtual void `vanishingSurfaceNormal` (std::vector< shared\_ptr< [ftPoint](#) > > &singular\_points, std::vector< shared\_ptr< [ftCurve](#) > > &singular\_curves)
 

*Returns ftPoints and ftCurves that will contain generated geometry.*
- virtual void `vanishingCurveTangent` (std::vector< shared\_ptr< [PointOnCurve](#) > > &sing\_points, std::vector< std::pair< shared\_ptr< [PointOnCurve](#) >, shared\_ptr< [PointOnCurve](#) > > > &sing\_curves)
 

*Contains generated points and curves fetched from topological entities.*
- virtual void `sliverSurfaces` (std::vector< shared\_ptr< [ParamSurface](#) > > &sliver\_sfs, [double](#) thickness, [double](#) factor=2.0)
 

*Uses existing geometry.*
- virtual void `narrowRegion` (std::vector< std::pair< shared\_ptr< [PointOnEdge](#) >, shared\_ptr< [PointOnEdge](#) > > > &narrow\_regions)
 

*Uses existing topological entities. Contains generated geometry.*
- virtual void `edgeVertexDistance` (std::vector< std::pair< [ftEdge](#) \*, shared\_ptr< [Vertex](#) > > > &edge\_↔\_vertices)
 

*Uses existing topological entities.*
- virtual void `faceVertexDistance` (std::vector< std::pair< [ftSurface](#) \*, shared\_ptr< [Vertex](#) > > > &face\_↔\_vertices)
 

*Existing topology. Partly generated geometry, existing surfaces.*
- virtual void `faceEdgeDistance` (std::vector< std::pair< [ftSurface](#) \*, [ftEdge](#) \* > > &face\_edges)
 

*Existing topology. Partly generated geometry, existing surfaces.*
- virtual void `edgePosAndTangDiscontinuity` (std::vector< std::pair< [ftEdge](#) \*, [ftEdge](#) \* > > &pos\_disconts, std::vector< std::pair< [ftEdge](#) \*, [ftEdge](#) \* > > &tang\_disconts)
 

*Uses existing topological entities.*
- virtual void `facePositionDiscontinuity` (std::vector< std::pair< [ftEdge](#) \*, [ftEdge](#) \* > > &pos\_disconts)
 

*Uses existing topological entities. Existing surfaces available.*
- virtual void `faceTangentDiscontinuity` (std::vector< std::pair< [ftEdge](#) \*, [ftEdge](#) \* > > &tangent\_disconts)
 

*Uses existing topological entities. Existing surfaces available.*
- virtual void `loopOrientationConsistency` (std::vector< shared\_ptr< [Loop](#) > > &inconsistent\_loops)
 

*Uses existing topological entities.*
- virtual void `faceNormalConsistency` (std::vector< shared\_ptr< [ftSurface](#) > > &inconsistent\_faces)
 

*Uses existing topological entities. Existing surfaces available.*
- virtual void `sFG1Discontinuity` (std::vector< shared\_ptr< [ftSurface](#) > > &discont\_sfs)
 

*Uses existing topological entities. Existing surfaces available.*
- virtual void `sFC1Discontinuity` (std::vector< shared\_ptr< [ftSurface](#) > > &discont\_sfs)
 

*Uses existing topological entities. Existing surfaces available.*
- virtual void `cvC1G1Discontinuity` (std::vector< shared\_ptr< [ParamCurve](#) > > &c1\_discont, std::vector< shared\_ptr< [ParamCurve](#) > > &g1\_discont)
 

*Contains curves fetched from topological entities.*
- virtual void `cvCurvatureRadius` (std::vector< std::pair< shared\_ptr< [PointOnCurve](#) >, [double](#) > > &small\_↔\_curv\_rad, std::pair< shared\_ptr< [PointOnCurve](#) >, [double](#) > &minimum\_curv\_rad)

*Contains generated points and curves fetched from topological entities.*

- virtual void [sfCurvatureRadius](#) (std::vector< std::pair< shared\_ptr< [ftPoint](#) >, double > > &small\_curv\_rad, std::pair< shared\_ptr< [ftPoint](#) >, double > &minimum\_curv\_rad)

*Generates new points, pointers to existing surfaces.*

- virtual void [acuteEdgeAngle](#) (std::vector< std::pair< [ftEdge](#) \*, [ftEdge](#) \* > > &edge\_acute)

*Uses existing topological entities.*

- virtual void [acuteFaceAngle](#) (std::vector< std::pair< [ftSurface](#) \*, [ftSurface](#) \* > > &face\_acute)

*Uses existing topological entities. Existing surfaces available.*

- virtual void [loopIntersection](#) (std::vector< std::pair< shared\_ptr< [PointOnEdge](#) >, shared\_ptr< [PointOnEdge](#) > > &loop\_intersection)

*Uses existing topological entities. Contains generated geometry.*

- virtual void [loopSelfIntersection](#) (std::vector< std::pair< shared\_ptr< [PointOnEdge](#) >, shared\_ptr< [PointOnEdge](#) > > &loop\_self\_intersection)

*Uses existing topological entities. Contains generated geometry.*

- virtual void [indistinctKnots](#) (std::vector< shared\_ptr< [ParamCurve](#) > > &cv\_knots, std::vector< shared\_ptr< [ParamSurface](#) > > &sf\_knots, double tol=1.0e-8)

*Uses existing surface and possibly generated curves.*

- shared\_ptr< [QualityResults](#) > [getResults](#) ()

## Protected Attributes

- [tpTolerances](#) [toptol\\_](#)
- [double](#) [approx\\_](#)
- [double](#) [small\\_size\\_](#)
- [double](#) [curvature\\_radius\\_](#)
- [shared\\_ptr](#)< [QualityResults](#) > [results\\_](#)

### 29.296.1 Detailed Description

Definition at line 53 of file ModelQuality.h.

### 29.296.2 Constructor & Destructor Documentation

29.296.2.1 [Go::ModelQuality::ModelQuality](#) ( [double](#) *gap*, [double](#) *kink*, [double](#) *approx* )

29.296.2.2 [Go::ModelQuality::ModelQuality](#) ( [const](#) [tpTolerances](#) & *toptol*, [double](#) *approx* )

29.296.2.3 [virtual](#) [Go::ModelQuality::~ModelQuality](#) ( ) [[virtual](#)]

### 29.296.3 Member Function Documentation

29.296.3.1 [virtual](#) void [Go::ModelQuality::acuteEdgeAngle](#) ( [std::vector](#)< [std::pair](#)< [ftEdge](#) \*, [ftEdge](#) \* > > & *edge\_acute* ) [[virtual](#)]

Uses existing topological entities.

Reimplemented in [Go::FaceSetQuality](#).

```
29.296.3.2 virtual void Go::ModelQuality::acuteFaceAngle (std::vector< std::pair< ftSurface *, ftSurface * > > &
 face_acute) [virtual]
```

Uses existing topological entities. Existing surfaces available.

Reimplemented in [Go::FaceSetQuality](#).

```
29.296.3.3 virtual void Go::ModelQuality::cvC1G1Discontinuity (std::vector< shared_ptr< ParamCurve > > & c1_discont,
 std::vector< shared_ptr< ParamCurve > > & g1_discont) [virtual]
```

Contains curves fetched from topological entities.

Reimplemented in [Go::FaceSetQuality](#).

```
29.296.3.4 virtual void Go::ModelQuality::cvCurvatureRadius (std::vector< std::pair< shared_ptr< PointOnCurve >,
 double > > & small_curv_rad, std::pair< shared_ptr< PointOnCurve >, double > & minimum_curv_rad)
 [virtual]
```

Contains generated points and curves fetched from topological entities.

Reimplemented in [Go::FaceSetQuality](#).

```
29.296.3.5 virtual void Go::ModelQuality::degenerateSfCorners (std::vector< shared_ptr< ftPoint > > & deg_corners)
 [virtual]
```

Generates new points, pointers to existing surfaces.

Reimplemented in [Go::FaceSetQuality](#).

```
29.296.3.6 virtual void Go::ModelQuality::degenSurfaces (std::vector< shared_ptr< ParamSurface > > & deg_sfs)
 [virtual]
```

Does not generate new geometry.

Reimplemented in [Go::FaceSetQuality](#).

```
29.296.3.7 virtual void Go::ModelQuality::edgePosAndTangDiscontinuity (std::vector< std::pair< ftEdge *, ftEdge * > >
 & pos_disconts, std::vector< std::pair< ftEdge *, ftEdge * > > & tang_disconts) [virtual]
```

Uses existing topological entities.

Reimplemented in [Go::FaceSetQuality](#).

```
29.296.3.8 virtual void Go::ModelQuality::edgeVertexDistance (std::vector< std::pair< ftEdge *, shared_ptr< Vertex > >
 > & edge_vertices) [virtual]
```

Uses existing topological entities.

Reimplemented in [Go::FaceSetQuality](#).

29.296.3.9 `virtual void Go::ModelQuality::faceEdgeDistance ( std::vector< std::pair< ftSurface *, ftEdge * > > & face_edges ) [virtual]`

Existing topology. Partly generated geometry, existing surfaces.

Reimplemented in [Go::FaceSetQuality](#).

29.296.3.10 `virtual void Go::ModelQuality::faceNormalConsistency ( std::vector< shared_ptr< ftSurface > > & inconsistent_faces ) [virtual]`

Uses existing topological entities. Existing surfaces available.

Reimplemented in [Go::FaceSetQuality](#).

29.296.3.11 `virtual void Go::ModelQuality::facePositionDiscontinuity ( std::vector< std::pair< ftEdge *, ftEdge * > > & pos_disconts ) [virtual]`

Uses existing topological entities. Existing surfaces available.

Reimplemented in [Go::FaceSetQuality](#).

29.296.3.12 `virtual void Go::ModelQuality::faceTangentDiscontinuity ( std::vector< std::pair< ftEdge *, ftEdge * > > & tangent_disconts ) [virtual]`

Uses existing topological entities. Existing surfaces available.

Reimplemented in [Go::FaceSetQuality](#).

29.296.3.13 `virtual void Go::ModelQuality::faceVertexDistance ( std::vector< std::pair< ftSurface *, shared_ptr< Vertex > > > & face_vertices ) [virtual]`

Existing topology. Partly generated geometry, existing surfaces.

Reimplemented in [Go::FaceSetQuality](#).

29.296.3.14 `shared_ptr<QualityResults> Go::ModelQuality::getResults ( ) [inline]`

Definition at line 215 of file ModelQuality.h.

29.296.3.15 `virtual void Go::ModelQuality::identicalOrEmbeddedEdges ( std::vector< std::pair< shared_ptr< ftEdge >, shared_ptr< ftEdge > > > & identical_edges, std::vector< std::pair< shared_ptr< ftEdge >, shared_ptr< ftEdge > > > & embedded_edges ) [virtual]`

Uses existing topological entities. May contain generated curves.

Reimplemented in [Go::FaceSetQuality](#).



```
29.296.3.16 virtual void Go::ModelQuality::identicalOrEmbeddedFaces (std::vector< std::pair< shared_ptr< ftSurface >, shared_ptr< ftSurface > > > & identical_faces, std::vector< std::pair< shared_ptr< ftSurface >, shared_ptr< ftSurface > > > & embedded_faces) [virtual]
```

Uses existing topological entities. Points to existing surfaces.

Reimplemented in [Go::FaceSetQuality](#).

```
29.296.3.17 virtual void Go::ModelQuality::identicalVertices (std::vector< std::pair< shared_ptr< Vertex >, shared_ptr< Vertex > > > & identical_vertices) [virtual]
```

Uses existing topological entity.

Reimplemented in [Go::FaceSetQuality](#).

```
29.296.3.18 virtual void Go::ModelQuality::indistinctKnots (std::vector< shared_ptr< ParamCurve > > & cv_knots, std::vector< shared_ptr< ParamSurface > > & sf_knots, double tol = 1.0e-8) [virtual]
```

Uses existing surface and possibly generated curves.

Reimplemented in [Go::FaceSetQuality](#).

```
29.296.3.19 virtual void Go::ModelQuality::loopIntersection (std::vector< std::pair< shared_ptr< PointOnEdge >, shared_ptr< PointOnEdge > > > & loop_intersection) [virtual]
```

Uses existing topological entities. Contains generated geometry.

Reimplemented in [Go::FaceSetQuality](#).

```
29.296.3.20 virtual void Go::ModelQuality::loopOrientationConsistency (std::vector< shared_ptr< Loop > > & inconsistent_loops) [virtual]
```

Uses existing topological entities.

Reimplemented in [Go::FaceSetQuality](#).

```
29.296.3.21 virtual void Go::ModelQuality::loopSelfIntersection (std::vector< std::pair< shared_ptr< PointOnEdge >, shared_ptr< PointOnEdge > > > & loop_self_intersection) [virtual]
```

Uses existing topological entities. Contains generated geometry.

Reimplemented in [Go::FaceSetQuality](#).

```
29.296.3.22 virtual void Go::ModelQuality::miniEdges (std::vector< shared_ptr< ftEdge > > & mini_edges) [virtual]
```

Uses existing topological entities.

Reimplemented in [Go::FaceSetQuality](#).

```
29.296.3.23 virtual void Go::ModelQuality::miniSurfaces (std::vector< shared_ptr< ftSurface > > & mini_surfaces)
 [virtual]
```

Uses existing topological entities. Points to existing surfaces.

Reimplemented in [Go::FaceSetQuality](#).

```
29.296.3.24 virtual void Go::ModelQuality::narrowRegion (std::vector< std::pair< shared_ptr< PointOnEdge > ,
 shared_ptr< PointOnEdge > > > & narrow_regions) [virtual]
```

Uses existing topological entities. Contains generated geometry.

Reimplemented in [Go::FaceSetQuality](#).

```
29.296.3.25 void Go::ModelQuality::setCurvatureRadius (double radius) [inline]
```

Definition at line 70 of file ModelQuality.h.

```
29.296.3.26 void Go::ModelQuality::setMiniElementSize (double element_size) [inline]
```

Definition at line 65 of file ModelQuality.h.

```
29.296.3.27 virtual void Go::ModelQuality::sfC1Discontinuity (std::vector< shared_ptr< ftSurface > > & scont_sfs)
 [virtual]
```

Uses existing topological entities. Existing surfaces available.

Reimplemented in [Go::FaceSetQuality](#).

```
29.296.3.28 virtual void Go::ModelQuality::sfCurvatureRadius (std::vector< std::pair< shared_ptr< ftPoint > , double >
 > & small_curv_rad, std::pair< shared_ptr< ftPoint > , double > & minimum_curv_rad) [virtual]
```

Generates new points, pointers to existing surfaces.

Reimplemented in [Go::FaceSetQuality](#).

```
29.296.3.29 virtual void Go::ModelQuality::sfG1Discontinuity (std::vector< shared_ptr< ftSurface > > & scont_sfs)
 [virtual]
```

Uses existing topological entities. Existing surfaces available.

Reimplemented in [Go::FaceSetQuality](#).

29.296.3.30 `virtual void Go::ModelQuality::sliverSurfaces ( std::vector< shared_ptr< ParamSurface > > & sliver_sfs, double thickness, double factor = 2.0 ) [virtual]`

Uses existing geometry.

Reimplemented in [Go::FaceSetQuality](#).

29.296.3.31 `virtual void Go::ModelQuality::vanishingCurveTangent ( std::vector< shared_ptr< PointOnCurve > > & sing_points, std::vector< std::pair< shared_ptr< PointOnCurve >, shared_ptr< PointOnCurve > > > & sing_curves ) [virtual]`

Contains generated points and curves fetched from topological entities.

Reimplemented in [Go::FaceSetQuality](#).

29.296.3.32 `virtual void Go::ModelQuality::vanishingSurfaceNormal ( std::vector< shared_ptr< ftPoint > > & singular_points, std::vector< shared_ptr< ftCurve > > & singular_curves ) [virtual]`

Returns ftPoints and ftCurves that will contain generated geometry.

Reimplemented in [Go::FaceSetQuality](#).

## 29.296.4 Member Data Documentation

29.296.4.1 `double Go::ModelQuality::approx_ [protected]`

Definition at line 222 of file ModelQuality.h.

29.296.4.2 `double Go::ModelQuality::curvature_radius_ [protected]`

Definition at line 224 of file ModelQuality.h.

29.296.4.3 `shared_ptr<QualityResults> Go::ModelQuality::results_ [protected]`

Definition at line 225 of file ModelQuality.h.

29.296.4.4 `double Go::ModelQuality::small_size_ [protected]`

Definition at line 223 of file ModelQuality.h.

29.296.4.5 `tpTolerances Go::ModelQuality::toptol_ [protected]`

Definition at line 221 of file ModelQuality.h.

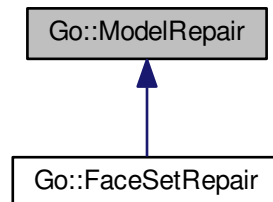
The documentation for this class was generated from the following file:

- [qualitymodule/include/GoTools/qualitymodule/ModelQuality.h](#)

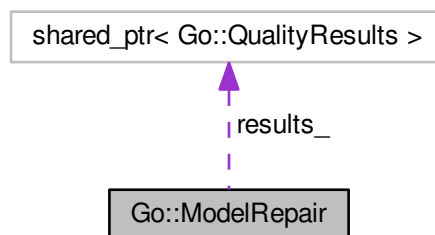
## 29.297 Go::ModelRepair Class Reference

```
#include <ModelRepair.h>
```

Inheritance diagram for Go::ModelRepair:



Collaboration diagram for Go::ModelRepair:



### Public Member Functions

- [ModelRepair](#) ()  
*Constructor.*
- [ModelRepair](#) (shared\_ptr< [QualityResults](#) > results)
- virtual void [mendGaps](#) ()=0  
*Remove gaps in the model.*
- virtual void [identicalAndEmbeddedFaces](#) ()=0  
*Remove identical and embedded faces.*
- virtual void [identicalVertices](#) ()=0  
*Handle identical vertices.*
- virtual void [consistentFaceNormal](#) ()=0  
*Handle inconsistent face normals.*
- virtual void [optimizeVertexPosition](#) ()=0
- virtual void [mendEdgeDistance](#) ()=0

## Protected Attributes

- `shared_ptr< QualityResults > results_`

### 29.297.1 Detailed Description

Definition at line 49 of file `ModelRepair.h`.

### 29.297.2 Constructor & Destructor Documentation

#### 29.297.2.1 `Go::ModelRepair::ModelRepair ( )`

Constructor.

#### 29.297.2.2 `Go::ModelRepair::ModelRepair ( shared_ptr< QualityResults > results )`

### 29.297.3 Member Function Documentation

#### 29.297.3.1 `virtual void Go::ModelRepair::consistentFaceNormal ( ) [pure virtual]`

[Handle](#) inconsistent face normals.

Implemented in [Go::FaceSetRepair](#).

#### 29.297.3.2 `virtual void Go::ModelRepair::identicalAndEmbeddedFaces ( ) [pure virtual]`

Remove identical and embedded faces.

Implemented in [Go::FaceSetRepair](#).

#### 29.297.3.3 `virtual void Go::ModelRepair::identicalVertices ( ) [pure virtual]`

[Handle](#) identical vertices.

Implemented in [Go::FaceSetRepair](#).

#### 29.297.3.4 `virtual void Go::ModelRepair::mendEdgeDistance ( ) [pure virtual]`

Implemented in [Go::FaceSetRepair](#).

#### 29.297.3.5 `virtual void Go::ModelRepair::mendGaps ( ) [pure virtual]`

Remove gaps in the model.

Implemented in [Go::FaceSetRepair](#).

29.297.3.6 `virtual void Go::ModelRepair::optimizeVertexPosition ( ) [pure virtual]`

Implemented in [Go::FaceSetRepair](#).

## 29.297.4 Member Data Documentation

29.297.4.1 `shared_ptr<QualityResults> Go::ModelRepair::results_ [protected]`

Definition at line 75 of file ModelRepair.h.

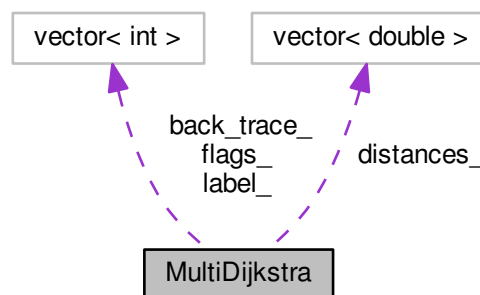
The documentation for this class was generated from the following file:

- [qualitymodule/include/GoTools/qualitymodule/ModelRepair.h](#)

## 29.298 MultiDijkstra Class Reference

```
#include <PrMultiDijkstra.h>
```

Collaboration diagram for MultiDijkstra:



### Public Member Functions

- [MultiDijkstra \( \)](#)  
*Constructor.*
- [~MultiDijkstra \( \)](#)  
*Destructor.*
- void [setGraph \(GraphType \\*tri\)](#)  
*Set the graph to run the algorithm on. Must be run before 'initialize()'.*
- void [initialize \( \)](#)
- void [setSource \(int node\\_idx, int node\\_label, double dist=0\)](#)
- void [run \( \)](#)
- void [run \(double radius\)](#)
- void [run \(int target\)](#)
- int [getLabel \(int node\\_idx\)](#)
- double [getDistance \(int node\\_idx\)](#)
- double [getDistance \(int node\\_idx, int neighbour\)](#)
- int [closestNeighbour \(int node\\_idx\)](#)

## Protected Member Functions

- void [setFinished](#) (int node\_idx)
- int [isFinished](#) (int node\_idx)
- void [insertInCandidates](#) (int node\_idx)

## Protected Attributes

- [GraphType](#) \* [graph\\_](#)
- [HeapType2](#) \* [queue\\_](#)
- vector< [double](#) > \* [distances\\_](#)
- vector< int > \* [back\\_trace\\_](#)
- vector< int > \* [label\\_](#)
- vector< int > \* [flags\\_](#)
- [double](#) [large\\_distance\\_](#)

### 29.298.1 Detailed Description

[MultiDijkstra](#) Class implementing the [Dijkstra](#) algorithm, where each node is labeled according to which "candidate node" its path ends up in.

Definition at line 76 of file PrMultiDijkstra.h.

### 29.298.2 Constructor & Destructor Documentation

#### 29.298.2.1 [MultiDijkstra::MultiDijkstra \( \)](#)

Constructor.

#### 29.298.2.2 [MultiDijkstra::~~MultiDijkstra \( \)](#)

Destructor.

### 29.298.3 Member Function Documentation

#### 29.298.3.1 [int MultiDijkstra::closestNeighbour \( int node\\_idx \)](#)

Get the index of the closest neighbour to the node 'node\_idx'. Does not give a valid answer until the [Dijkstra](#) algorithm has been [run\(\)](#).

#### 29.298.3.2 [double MultiDijkstra::getDistance \( int node\\_idx \)](#) `[inline]`

Get the distance associated with the node 'node\_ix'. Does not give a valid answer until the [Dijkstra](#) algorithm has been [run\(\)](#).

Definition at line 129 of file PrMultiDijkstra.h.

**29.298.3.3** `double MultiDijkstra::getDistance ( int node_idx, int neighbour )` `[inline]`

Compute the euclidean distance between two nodes, supposedly neighbours. If they are boundary nodes, their mutual distance will be set to a very large number.

Definition at line 166 of file PrMultiDijkstra.h.

**29.298.3.4** `int MultiDijkstra::getLabel ( int node_idx )` `[inline]`

After the algorithm has been run (using the `run()` function *without* arguments), this function can be used to find which label each node has been assigned.

Definition at line 125 of file PrMultiDijkstra.h.

**29.298.3.5** `void MultiDijkstra::initialize ( )`

Initialize internal data structures. Must be done after '`setGraph()`' has been called.

**29.298.3.6** `void MultiDijkstra::insertInCandidates ( int node_idx )` `[inline]`, `[protected]`

Definition at line 158 of file PrMultiDijkstra.h.

**29.298.3.7** `int MultiDijkstra::isFinished ( int node_idx )` `[inline]`, `[protected]`

Definition at line 150 of file PrMultiDijkstra.h.

**29.298.3.8** `void MultiDijkstra::run ( )`

Run the algorithm. Compute the associated distance for all nodes in the graph.

**29.298.3.9** `void MultiDijkstra::run ( double radius )`

Run the algorithm. Compute the associated distance for all nodes in the graph. Edges longer than '`radius`' will be ignored.

**29.298.3.10** `void MultiDijkstra::run ( int target )`

Run the algorithm. Compute the associated distance for nodes in the graph until the distance for the '`target`' node has been found. Then skip further computations.

**29.298.3.11** `void MultiDijkstra::setFinished ( int node_idx )` `[inline]`, `[protected]`

Definition at line 143 of file PrMultiDijkstra.h.

**29.298.3.12** `void MultiDijkstra::setGraph ( GraphType * tri )` `[inline]`

Set the graph to run the algorithm on. Must be run before '`initialize()`'.

Definition at line 99 of file PrMultiDijkstra.h.

**29.298.3.13** `void MultiDijkstra::setSource ( int node_idx, int node_label, double dist = 0 )`

Set a '`source`' for the algorithm. This is a node for which the associated minimum distance is known. At least one source must be set for the algorithm to work properly.



## Parameters

<i>node_idx</i>	the index of the node to be designated as a source
<i>node_label</i>	set a label for this source (makes it possibly to identify which nodes in the graph whose solution depends on this node).
<i>dist</i>	the distance at this node (typically 0).

## 29.298.4 Member Data Documentation

29.298.4.1 `vector<int>* MultiDijkstra::back_trace_` [protected]

Definition at line 83 of file PrMultiDijkstra.h.

29.298.4.2 `vector<double>* MultiDijkstra::distances_` [protected]

Definition at line 82 of file PrMultiDijkstra.h.

29.298.4.3 `vector<int>* MultiDijkstra::flags_` [protected]

Definition at line 85 of file PrMultiDijkstra.h.

29.298.4.4 `GraphType* MultiDijkstra::graph_` [protected]

Definition at line 80 of file PrMultiDijkstra.h.

29.298.4.5 `vector<int>* MultiDijkstra::label_` [protected]

Definition at line 84 of file PrMultiDijkstra.h.

29.298.4.6 `double MultiDijkstra::large_distance_` [protected]

Definition at line 86 of file PrMultiDijkstra.h.

29.298.4.7 `HeapType2* MultiDijkstra::queue_` [protected]

Definition at line 81 of file PrMultiDijkstra.h.

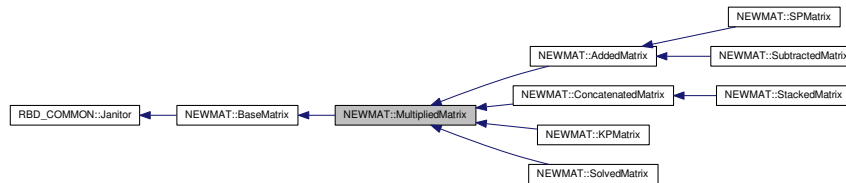
The documentation for this class was generated from the following file:

- parametrization/include/GoTools/parametrization/[PrMultiDijkstra.h](#)

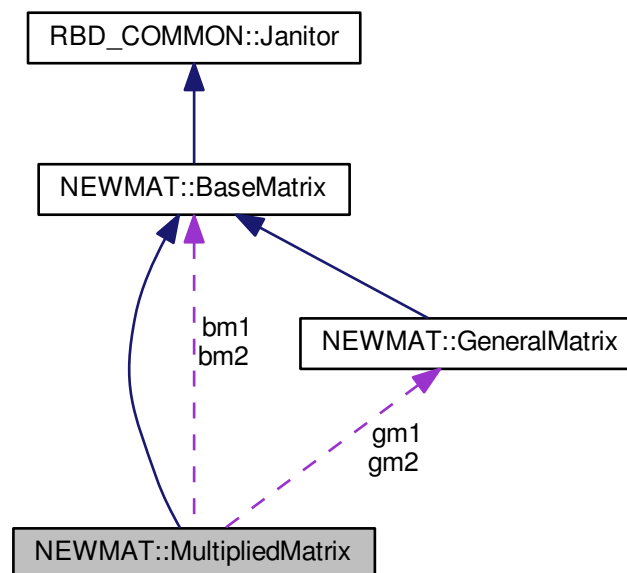
## 29.299 NEWMAT::MultipliedMatrix Class Reference

```
#include <newmat.h>
```

Inheritance diagram for NEWMAT::MultipliedMatrix:



Collaboration diagram for NEWMAT::MultipliedMatrix:



### Public Member Functions

- `~MultipliedMatrix ()`
- `GeneralMatrix * Evaluate (MatrixType mt=MatrixTypeUnSp)`
- `MatrixBandWidth BandWidth () const`

### Protected Member Functions

- `MultipliedMatrix (const BaseMatrix *bm1x, const BaseMatrix *bm2x)`
- `int search (const BaseMatrix *) const`

## Protected Attributes

- union {  
    [const BaseMatrix](#) \* *bm1*  
    [GeneralMatrix](#) \* *gm1*  
};
- union {  
    [const BaseMatrix](#) \* *bm2*  
    [GeneralMatrix](#) \* *gm2*  
};

## Friends

- class [BaseMatrix](#)
- class [GeneralMatrix](#)
- class [GenericMatrix](#)

### 29.299.1 Detailed Description

Definition at line 1173 of file newmat.h.

### 29.299.2 Constructor & Destructor Documentation

29.299.2.1 [NEWMAT::MultipliedMatrix::MultipliedMatrix](#) ( [const BaseMatrix](#) \* *bm1x*, [const BaseMatrix](#) \* *bm2x* )  
    [[inline](#)], [[protected](#)]

Definition at line 1181 of file newmat.h.

29.299.2.2 [NEWMAT::MultipliedMatrix::~~MultipliedMatrix](#) ( ) [[inline](#)]

Definition at line 1188 of file newmat.h.

### 29.299.3 Member Function Documentation

29.299.3.1 [MatrixBandWidth](#) [MultipliedMatrix::BandWidth](#) ( ) [const](#) [[virtual](#)]

Reimplemented from [NEWMAT::BaseMatrix](#).

Reimplemented in [NEWMAT::SolvedMatrix](#), [NEWMAT::ConcatenatedMatrix](#), [NEWMAT::KPMatrix](#), [NEWMAT::S←PMatrix](#), and [NEWMAT::AddedMatrix](#).

Definition at line 453 of file newmat4.cpp.

29.299.3.2 **GeneralMatrix \* MultipliedMatrix::Evaluate ( MatrixType mt = MatrixTypeUnSp )** [virtual]

Implements [NEWMAT::BaseMatrix](#).

Reimplemented in [NEWMAT::SubtractedMatrix](#), [NEWMAT::SolvedMatrix](#), [NEWMAT::StackedMatrix](#), [NEWMAT::ConcatenatedMatrix](#), [NEWMAT::KPMatrix](#), [NEWMAT::SPMatrix](#), and [NEWMAT::AddedMatrix](#).

Definition at line 99 of file newmat7.cpp.

29.299.3.3 **int MultipliedMatrix::search ( const BaseMatrix \* s ) const** [protected],[virtual]

Implements [NEWMAT::BaseMatrix](#).

Definition at line 373 of file newmat4.cpp.

## 29.299.4 Friends And Related Function Documentation

29.299.4.1 **friend class BaseMatrix** [friend]

Definition at line 1184 of file newmat.h.

29.299.4.2 **friend class GeneralMatrix** [friend]

Definition at line 1185 of file newmat.h.

29.299.4.3 **friend class GenericMatrix** [friend]

Definition at line 1186 of file newmat.h.

## 29.299.5 Member Data Documentation

29.299.5.1 **union { ... }** [protected]

29.299.5.2 **union { ... }** [protected]

29.299.5.3 **const BaseMatrix\* NEWMAT::MultipliedMatrix::bm1**

Definition at line 1178 of file newmat.h.

29.299.5.4 **const BaseMatrix\* NEWMAT::MultipliedMatrix::bm2**

Definition at line 1180 of file newmat.h.

#### 29.299.5.5 GeneralMatrix\* NEWMAT::MultipliedMatrix::gm1

Definition at line 1178 of file newmat.h.

#### 29.299.5.6 GeneralMatrix\* NEWMAT::MultipliedMatrix::gm2

Definition at line 1180 of file newmat.h.

The documentation for this class was generated from the following files:

- [newmat/include/newmat.h](#)
- [newmat/src/newmat4.cpp](#)
- [newmat/src/newmat7.cpp](#)

## 29.300 NEWMAT::MultiRadixCounter Class Reference

```
#include <newmatap.h>
```

### Public Member Functions

- [MultiRadixCounter](#) (int nx, [const SimpleIntArray](#) &rx, [SimpleIntArray](#) &vx)
- void [operator++](#) ()
- [bool Swap](#) () const
- [bool Finish](#) () const
- int [Reverse](#) () const
- int [Counter](#) () const

#### 29.300.1 Detailed Description

Definition at line 129 of file newmatap.h.

#### 29.300.2 Constructor & Destructor Documentation

##### 29.300.2.1 MultiRadixCounter::MultiRadixCounter ( int nx, const SimpleIntArray & rx, SimpleIntArray & vx )

Definition at line 1014 of file newfft.cpp.

#### 29.300.3 Member Function Documentation

##### 29.300.3.1 int NEWMAT::MultiRadixCounter::Counter ( ) const `[inline]`

Definition at line 147 of file newmatap.h.

29.300.3.2 `bool NEWMAT::MultiRadixCounter::Finish ( ) const [inline]`

Definition at line 145 of file newmatap.h.

29.300.3.3 `void MultiRadixCounter::operator++ ( )`

Definition at line 1022 of file newfft.cpp.

29.300.3.4 `int NEWMAT::MultiRadixCounter::Reverse ( ) const [inline]`

Definition at line 146 of file newmatap.h.

29.300.3.5 `bool NEWMAT::MultiRadixCounter::Swap ( ) const [inline]`

Definition at line 144 of file newmatap.h.

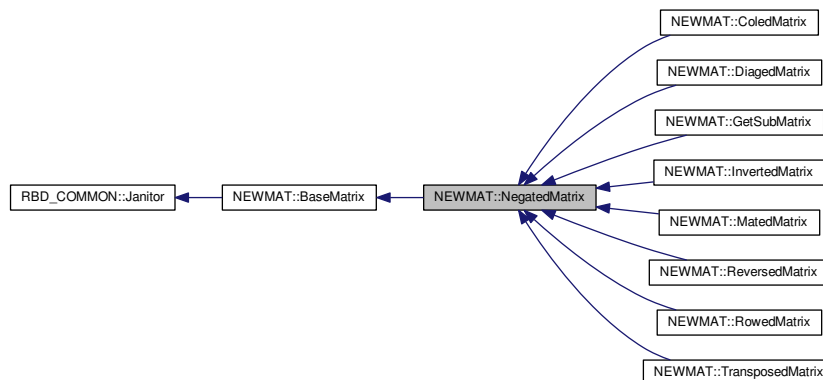
The documentation for this class was generated from the following files:

- [newmat/include/newmatap.h](#)
- [newmat/src/newfft.cpp](#)

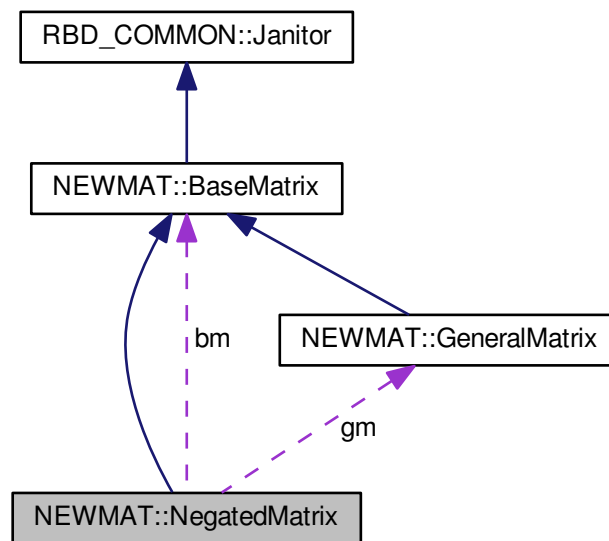
## 29.301 NEWMAT::NegatedMatrix Class Reference

```
#include <newmat.h>
```

Inheritance diagram for NEWMAT::NegatedMatrix:



Collaboration diagram for NEWMAT::NegatedMatrix:



### Public Member Functions

- `~NegatedMatrix ()`
- `GeneralMatrix * Evaluate (MatrixType mt=MatrixTypeUnSp)`
- `MatrixBandWidth BandWidth () const`

### Protected Member Functions

- `NegatedMatrix (const BaseMatrix *bm)`
- `int search (const BaseMatrix *) const`

### Protected Attributes

- union {
  - `const BaseMatrix * bm`
  - `GeneralMatrix * gm`
- };

### Friends

- class `BaseMatrix`

### 29.301.1 Detailed Description

Definition at line 1366 of file newmat.h.

### 29.301.2 Constructor & Destructor Documentation

29.301.2.1 **NEWMAT::NegatedMatrix::NegatedMatrix ( const **BaseMatrix** \* *bm*x )** [inline],[protected]

Definition at line 1370 of file newmat.h.

29.301.2.2 **NEWMAT::NegatedMatrix::~~NegatedMatrix ( )** [inline]

Definition at line 1375 of file newmat.h.

### 29.301.3 Member Function Documentation

29.301.3.1 **MatrixBandWidth NegatedMatrix::BandWidth ( ) const** [virtual]

Reimplemented from [NEWMAT::BaseMatrix](#).

Reimplemented in [NEWMAT::GetSubMatrix](#), [NEWMAT::MatedMatrix](#), [NEWMAT::DiagedMatrix](#), [NEWMAT::ColedMatrix](#), [NEWMAT::RowedMatrix](#), [NEWMAT::InvertedMatrix](#), and [NEWMAT::TransposedMatrix](#).

Definition at line 468 of file newmat4.cpp.

29.301.3.2 **GeneralMatrix \* NegatedMatrix::Evaluate ( **MatrixType** *mt* = **MatrixTypeUnSp** )** [virtual]

Implements [NEWMAT::BaseMatrix](#).

Reimplemented in [NEWMAT::GetSubMatrix](#), [NEWMAT::MatedMatrix](#), [NEWMAT::DiagedMatrix](#), [NEWMAT::ColedMatrix](#), [NEWMAT::RowedMatrix](#), [NEWMAT::InvertedMatrix](#), [NEWMAT::ReversedMatrix](#), and [NEWMAT::TransposedMatrix](#).

Definition at line 210 of file newmat5.cpp.

29.301.3.3 **int NegatedMatrix::search ( const **BaseMatrix** \* *s* ) const** [protected],[virtual]

Implements [NEWMAT::BaseMatrix](#).

Definition at line 379 of file newmat4.cpp.

### 29.301.4 Friends And Related Function Documentation

29.301.4.1 **friend class **BaseMatrix**** [friend]

Definition at line 1373 of file newmat.h.



### 29.301.5 Member Data Documentation

29.301.5.1 `union { ... } [protected]`

29.301.5.2 `const BaseMatrix* NEWMAT::NegatedMatrix::bm`

Definition at line 1369 of file newmat.h.

29.301.5.3 `GeneralMatrix* NEWMAT::NegatedMatrix::gm`

Definition at line 1369 of file newmat.h.

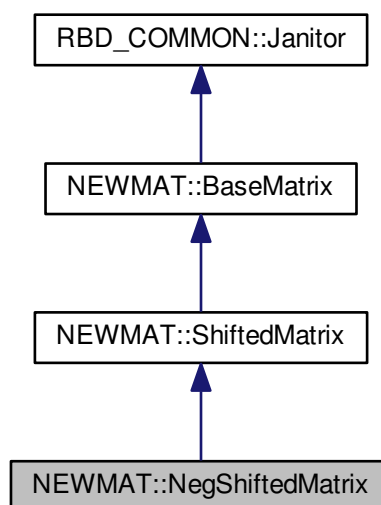
The documentation for this class was generated from the following files:

- [newmat/include/newmat.h](#)
- [newmat/src/newmat4.cpp](#)
- [newmat/src/newmat5.cpp](#)

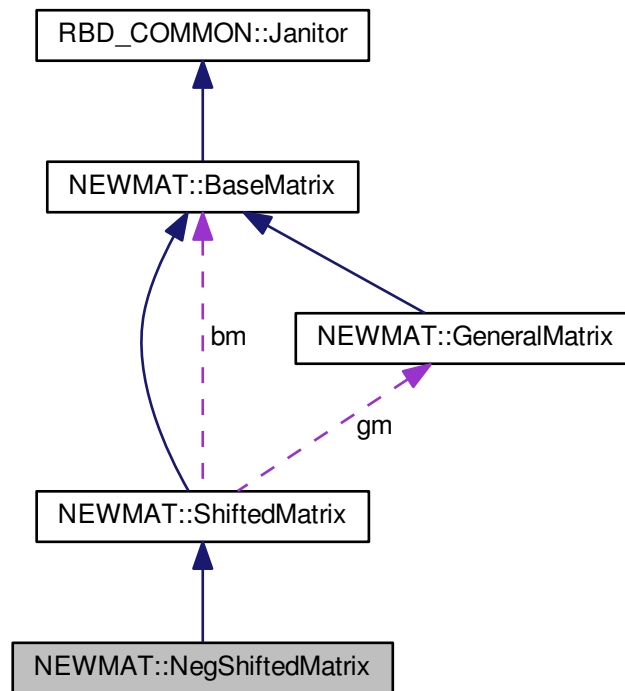
## 29.302 NEWMAT::NegShiftedMatrix Class Reference

```
#include <newmat.h>
```

Inheritance diagram for NEWMAT::NegShiftedMatrix:



Collaboration diagram for NEWMAT::NegShiftedMatrix:



### Public Member Functions

- [~NegShiftedMatrix](#) ()
- [GeneralMatrix \\* Evaluate](#) ([MatrixType](#) mt=[MatrixTypeUnSp](#))

### Protected Member Functions

- [NegShiftedMatrix](#) ([Real](#) fx, [const BaseMatrix](#) \*bm)

### Friends

- class [BaseMatrix](#)
- class [GeneralMatrix](#)
- class [GenericMatrix](#)
- [NegShiftedMatrix operator-](#) ([Real](#), [const BaseMatrix](#) &)

### Additional Inherited Members

#### 29.302.1 Detailed Description

Definition at line 1331 of file newmat.h.

## 29.302.2 Constructor & Destructor Documentation

29.302.2.1 `NEWMAT::NegShiftedMatrix::NegShiftedMatrix ( Real fx, const BaseMatrix * bmx ) [inline], [protected]`

Definition at line 1334 of file newmat.h.

29.302.2.2 `NEWMAT::NegShiftedMatrix::~~NegShiftedMatrix ( ) [inline]`

Definition at line 1339 of file newmat.h.

## 29.302.3 Member Function Documentation

29.302.3.1 `GeneralMatrix * NegShiftedMatrix::Evaluate ( MatrixType mt = MatrixTypeUnSp ) [virtual]`

Reimplemented from [NEWMAT::ShiftedMatrix](#).

Definition at line 132 of file newmat5.cpp.

## 29.302.4 Friends And Related Function Documentation

29.302.4.1 `friend class BaseMatrix [friend]`

Definition at line 1335 of file newmat.h.

29.302.4.2 `friend class GeneralMatrix [friend]`

Definition at line 1336 of file newmat.h.

29.302.4.3 `friend class GenericMatrix [friend]`

Definition at line 1337 of file newmat.h.

29.302.4.4 `NegShiftedMatrix operator- ( Real , const BaseMatrix & ) [friend]`

The documentation for this class was generated from the following files:

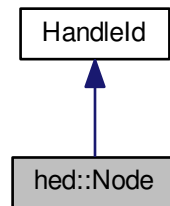
- [newmat/include/newmat.h](#)
- [newmat/src/newmat5.cpp](#)

## 29.303 hed::Node Class Reference

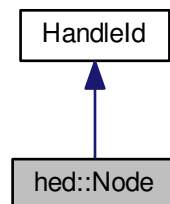
**Node** class for data structures (Inherits from [HandleId](#))

```
#include <HeTriang.h>
```

Inheritance diagram for hed::Node:



Collaboration diagram for hed::Node:



### Public Member Functions

- [Node](#) ()  
*Default constructor.*
- [Node](#) (double x, double y, double z=0.0)  
*Constructor.*
- [~Node](#) ()  
*Destructor.*
- void [init](#) (double x, double y, double z)  
*Initialize the position.*
- [const double & x](#) () const  
*Returns the x-coordinate.*
- [const double & y](#) () const

- Returns the y-coordinate.*
- `const double & z () const`  
*Returns the z-coordinate.*
- `const int & id () const`  
*Returns the id (TTL\_USE\_NODE\_ID must be defined)*
- `void setFlag (bool flag)`  
*Sets the flag (TTL\_USE\_NODE\_FLAG must be defined)*
- `const bool & getFlag () const`  
*Returns the flag (TTL\_USE\_NODE\_FLAG must be defined)*

## Additional Inherited Members

### 29.303.1 Detailed Description

**Node** class for data structures (Inherits from [HandleId](#))

#### Note

- To enable node IDs, TTL\_USE\_NODE\_ID must be defined.
- To enable node flags, TTL\_USE\_NODE\_FLAG must be defined.
- TTL\_USE\_NODE\_ID and TTL\_USE\_NODE\_FLAG should only be enabled if this functionality is required by the application, because they increase the memory usage for each [Node](#) object.

Definition at line 80 of file HeTriang.h.

### 29.303.2 Constructor & Destructor Documentation

#### 29.303.2.1 hed::Node::Node ( ) [inline]

Default constructor.

Definition at line 101 of file HeTriang.h.

#### 29.303.2.2 hed::Node::Node ( double x, double y, double z = 0.0 ) [inline]

Constructor.

Definition at line 104 of file HeTriang.h.

#### 29.303.2.3 hed::Node::~~Node ( ) [inline]

Destructor.

Definition at line 107 of file HeTriang.h.

### 29.303.3 Member Function Documentation

**29.303.3.1** `const bool& hed::Node::getFlag ( ) const` `[inline]`

Returns the flag (TTL\_USE\_NODE\_FLAG must be defined)

Definition at line 138 of file HeTriang.h.

**29.303.3.2** `const int& hed::Node::id ( ) const` `[inline]`

Returns the id (TTL\_USE\_NODE\_ID must be defined)

Definition at line 130 of file HeTriang.h.

**29.303.3.3** `void hed::Node::init ( double x, double y, double z )` `[inline]`

Initialize the position.

Definition at line 110 of file HeTriang.h.

**29.303.3.4** `void hed::Node::setFlag ( bool flag )` `[inline]`

Sets the flag (TTL\_USE\_NODE\_FLAG must be defined)

Definition at line 135 of file HeTriang.h.

**29.303.3.5** `const double& hed::Node::x ( void ) const` `[inline]`

Returns the x-coordinate.

Definition at line 120 of file HeTriang.h.

**29.303.3.6** `const double& hed::Node::y ( void ) const` `[inline]`

Returns the y-coordinate.

Definition at line 123 of file HeTriang.h.

**29.303.3.7** `const double& hed::Node::z ( void ) const` `[inline]`

Returns the z-coordinate.

Definition at line 126 of file HeTriang.h.

The documentation for this class was generated from the following file:

- [ttl/include/ttl/halfedge/HeTriang.h](#)

## 29.304 hetriang::Node Class Reference

**Node** class in the half-edge data structure

```
#include <ttlTriang.h>
```

### Public Member Functions

- [Node](#) ()
- [Node](#) (double x, double y, double z=0.0)
- [Node](#) (Go::PointIter iter, double x, double y, double z=0)
- [~Node](#) ()
- void [init](#) (double x, double y, double z)
- double [x](#) () const
- double [y](#) () const
- double [z](#) () const
- const Go::PointIter & [pointIter](#) () const

### 29.304.1 Detailed Description

**Node** class in the half-edge data structure

Definition at line 64 of file ttlTriang.h.

### 29.304.2 Constructor & Destructor Documentation

29.304.2.1 `hetriang::Node::Node ( )` [\[inline\]](#)

Definition at line 72 of file ttlTriang.h.

29.304.2.2 `hetriang::Node::Node ( double x, double y, double z = 0.0 )` [\[inline\]](#)

Definition at line 73 of file ttlTriang.h.

29.304.2.3 `hetriang::Node::Node ( Go::PointIter iter, double x, double y, double z = 0 )` [\[inline\]](#)

Definition at line 75 of file ttlTriang.h.

29.304.2.4 `hetriang::Node::~~Node ( )` [\[inline\]](#)

Definition at line 78 of file ttlTriang.h.

### 29.304.3 Member Function Documentation

29.304.3.1 `void hetriang::Node::init ( double x, double y, double z ) [inline]`

Definition at line 79 of file `tllTriang.h`.

29.304.3.2 `const Go::PointIter& hetriang::Node::pointIter ( ) const [inline]`

Definition at line 84 of file `tllTriang.h`.

29.304.3.3 `double hetriang::Node::x ( ) const [inline]`

Definition at line 81 of file `tllTriang.h`.

29.304.3.4 `double hetriang::Node::y ( ) const [inline]`

Definition at line 82 of file `tllTriang.h`.

29.304.3.5 `double hetriang::Node::z ( ) const [inline]`

Definition at line 83 of file `tllTriang.h`.

The documentation for this class was generated from the following file:

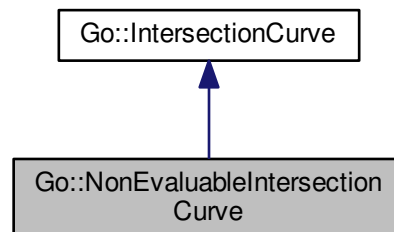
- [compositemodel/include/GoTools/compositemodel/tllTriang.h](#)

### 29.305 Go::NonEvaluableIntersectionCurve Class Reference

[IntersectionCurve](#) that cannot be evaluated.

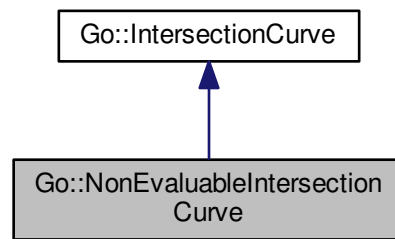
```
#include <IntersectionCurve.h>
```

Inheritance diagram for `Go::NonEvaluableIntersectionCurve`:





Collaboration diagram for Go::NonEvaluableIntersectionCurve:



## Public Member Functions

- virtual `~NonEvaluableIntersectionCurve ()`
- virtual `shared_ptr< ParamCurve > getCurve () const`
- virtual `shared_ptr< ParamCurve > getParamCurve (int obj_nmb) const`
- virtual `void getParamSpan (double &start, double &end) const`
- virtual `void evaluateAt (double pval, Point &pos, Point &tan)`
- virtual `void refine (const double &pos_tol, const double &angle_tol)`
- virtual `bool isIsocurve () const`
- virtual `bool isDegenerated () const`

## Friends

- `template<class iterator > shared_ptr< IntersectionCurve > constructIntersectionCurve (const iterator begin, const iterator end)`

## Additional Inherited Members

### 29.305.1 Detailed Description

[IntersectionCurve](#) that cannot be evaluated.

Definition at line 275 of file `IntersectionCurve.h`.

### 29.305.2 Constructor & Destructor Documentation

29.305.2.1 `virtual Go::NonEvaluableIntersectionCurve::~~NonEvaluableIntersectionCurve ( ) [virtual]`

### 29.305.3 Member Function Documentation

29.305.3.1 `virtual void Go::NonEvaluableIntersectionCurve::evaluateAt ( double pval, Point & pos, Point & tan ) [inline], [virtual]`

Evaluate the [IntersectionCurve](#) at parameter value *pval*. Its position at this parameter value will be returned in *pos* and its tangent direction will be returned in *tan*.

## Parameters

<i>pval</i>	the parameter value for which to evaluate the <a href="#">IntersectionCurve</a> . It should be within the parametric span (which can be obtained from <code>GetParamSpan()</code> ).
-------------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

## Return values

<i>pos</i>	the calculated position of the point on the curve at parameter <i>pval</i> .
------------	------------------------------------------------------------------------------

## Parameters

<i>tan</i>	the calculated tangent of the curve at parameter <i>pval</i> .
------------	----------------------------------------------------------------

Implements [Go::IntersectionCurve](#).

Definition at line 290 of file `IntersectionCurve.h`.

```
29.305.3.2 virtual shared_ptr<ParamCurve> Go::NonEvaluableIntersectionCurve::getCurve () const [inline],
 [virtual]
```

Get out a (shared) pointer to a parametric curve that approximates this [IntersectionCurve](#).

Implements [Go::IntersectionCurve](#).

Definition at line 281 of file `IntersectionCurve.h`.

```
29.305.3.3 virtual shared_ptr<ParamCurve> Go::NonEvaluableIntersectionCurve::getParamCurve (int obj_nmb) const
 [inline], [virtual]
```

Get out a (shared) pointer to the curve in the parametric plane of the first or second object. Should only be called when the concerned object is 2-parametric.

## Parameters

<i>obj_nmb</i>	should be either 1 or 2, depending on whether you want to get the parametric curve in object 1 or object 2.
----------------	-------------------------------------------------------------------------------------------------------------

## Returns

a shared pointer to a curve that approximates the intersection curve in the parametric domain of the specified object.

Implements [Go::IntersectionCurve](#).

Definition at line 284 of file `IntersectionCurve.h`.

```
29.305.3.4 virtual void Go::NonEvaluableIntersectionCurve::getParamSpan (double & start, double & end) const
 [inline], [virtual]
```

Get the start and end value for the parametric span of the [IntersectionCurve](#).

## Return values

<i>start</i>	upon function return this variable will contain the start value of the parametric span.
<i>end</i>	upon function return this variable will contain the end value of the parametric span.

Implements [Go::IntersectionCurve](#).

Definition at line 287 of file IntersectionCurve.h.

**29.305.3.5** `virtual bool Go::NonEvaluableIntersectionCurve::isDegenerated ( ) const [inline], [virtual]`

Determine if the [IntersectionCurve](#) is in fact a degenerated curve (a single point in space).

## Returns

'true' if the [IntersectionCurve](#) is degenerated. 'false' otherwise.

Implements [Go::IntersectionCurve](#).

Definition at line 305 of file IntersectionCurve.h.

**29.305.3.6** `virtual bool Go::NonEvaluableIntersectionCurve::isIsocurve ( ) const [inline], [virtual]`

Determine if the [IntersectionCurve](#) is in fact representing an isoparametric intersection.

## Returns

'true' if the [IntersectionCurve](#) is part of an isocurve for one of the objects, 'false' otherwise.

Implements [Go::IntersectionCurve](#).

Definition at line 302 of file IntersectionCurve.h.

**29.305.3.7** `virtual void Go::NonEvaluableIntersectionCurve::refine ( const double & pos_tol, const double & angle_tol ) [inline], [virtual]`

Refine the curve (which means to increase the number of [IntersectionPoint](#) s defining it), so that the curve matches a specific positional and angle tolerance.

## Parameters

<i>pos_tol</i>	the positional tolerance that the curve should match after refinement.
<i>angle_tol</i>	the angular tolerance that the tangent of the curve should match after refinement.

Implements [Go::IntersectionCurve](#).

Definition at line 296 of file IntersectionCurve.h.

### 29.305.4 Friends And Related Function Documentation

29.305.4.1 `template<class iterator > shared_ptr<IntersectionCurve> constructIntersectionCurve ( const iterator begin, const iterator end ) [friend]`

Definition at line 601 of file IntersectionCurve.h.

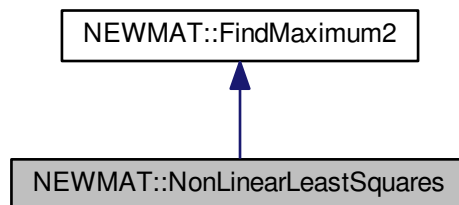
The documentation for this class was generated from the following file:

- [intersections/include/GoTools/intersections/IntersectionCurve.h](#)

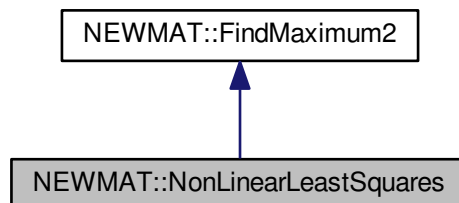
### 29.306 NEWMAT::NonLinearLeastSquares Class Reference

```
#include <newmatnl.h>
```

Inheritance diagram for NEWMAT::NonLinearLeastSquares:



Collaboration diagram for NEWMAT::NonLinearLeastSquares:



## Public Member Functions

- [NonLinearLeastSquares](#) ([R1\\_Col\\_I\\_D](#) &pred, int lim=1000, [Real](#) crit=0.0001)
- void [Fit](#) ([const ColumnVector](#) &, [ColumnVector](#) &)
- [Real ResidualVariance](#) () const
- void [GetResiduals](#) ([ColumnVector](#) &Z) const
- void [GetStandardErrors](#) ([ColumnVector](#) &)
- void [GetCorrelations](#) ([SymmetricMatrix](#) &)
- void [GetHatDiagonal](#) ([DiagonalMatrix](#) &) const

### 29.306.1 Detailed Description

Definition at line 211 of file newmatnl.h.

### 29.306.2 Constructor & Destructor Documentation

**29.306.2.1** `NEWMAT::NonLinearLeastSquares::NonLinearLeastSquares ( R1\_Col\_I\_D & pred, int lim = 1000, Real crit = 0.0001 ) [inline]`

Definition at line 232 of file newmatnl.h.

### 29.306.3 Member Function Documentation

**29.306.3.1** `void NonLinearLeastSquares::Fit ( const ColumnVector & Data, ColumnVector & Parameters )`

Definition at line 165 of file newmatnl.cpp.

**29.306.3.2** `void NonLinearLeastSquares::GetCorrelations ( SymmetricMatrix & Corr )`

Definition at line 189 of file newmatnl.cpp.

**29.306.3.3** `void NonLinearLeastSquares::GetHatDiagonal ( DiagonalMatrix & Hat ) const`

Definition at line 192 of file newmatnl.cpp.

**29.306.3.4** `void NEWMAT::NonLinearLeastSquares::GetResiduals ( ColumnVector & Z ) const [inline]`

Definition at line 236 of file newmatnl.h.

**29.306.3.5** `void NonLinearLeastSquares::GetStandardErrors ( ColumnVector & SEX )`

Definition at line 186 of file newmatnl.cpp.

29.306.3.6 Real NEWMAT::NonLinearLeastSquares::ResidualVariance ( ) const [inline]

Definition at line 235 of file newmatnl.h.

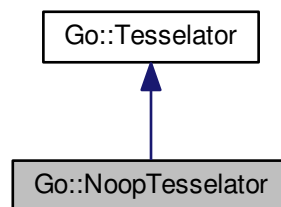
The documentation for this class was generated from the following files:

- newmat/include/newmatnl.h
- newmat/src/newmatnl.cpp

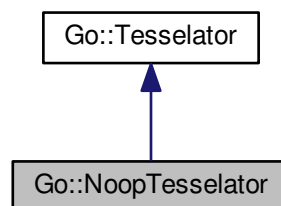
## 29.307 Go::NoopTesselator Class Reference

```
#include <NoopTesselator.h>
```

Inheritance diagram for Go::NoopTesselator:



Collaboration diagram for Go::NoopTesselator:



### Public Member Functions

- virtual [~NoopTesselator](#) ()
- virtual void [tesselate](#) ()  
*Perform tessellation.*

### 29.307.1 Detailed Description

Not operative tessellator

Definition at line 50 of file NoopTessellator.h.

### 29.307.2 Constructor & Destructor Documentation

29.307.2.1 virtual Go::NoopTessellator::~~NoopTessellator ( ) [virtual]

### 29.307.3 Member Function Documentation

29.307.3.1 virtual void Go::NoopTessellator::tessellate ( ) [virtual]

Perform tessellation.

Implements [Go::Tessellator](#).

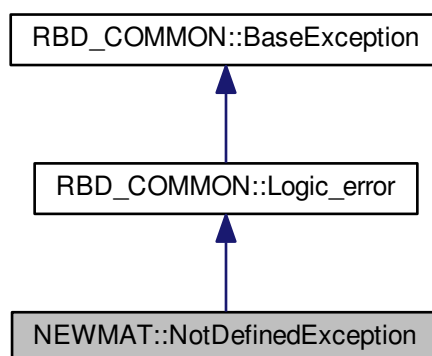
The documentation for this class was generated from the following file:

- [gotools-core/include/GoTools/tessellator/NoopTessellator.h](#)

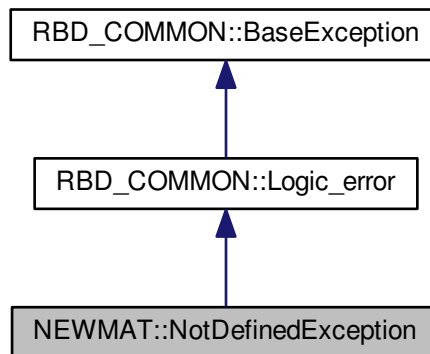
## 29.308 NEWMAT::NotDefinedException Class Reference

```
#include <newmat.h>
```

Inheritance diagram for NEWMAT::NotDefinedException:



Collaboration diagram for NEWMAT::NotDefinedException:



### Public Member Functions

- [NotDefinedException](#) (`const char *op`, `const char *matrix`)

### Static Public Attributes

- static unsigned long [Select](#)

### Additional Inherited Members

#### 29.308.1 Detailed Description

Definition at line 1677 of file newmat.h.

#### 29.308.2 Constructor & Destructor Documentation

##### 29.308.2.1 NotDefinedException::NotDefinedException ( `const char * op`, `const char * matrix` )

Definition at line 178 of file newmatex.cpp.

#### 29.308.3 Member Data Documentation

##### 29.308.3.1 unsigned long NotDefinedException::Select [static]

Definition at line 1680 of file newmat.h.

The documentation for this class was generated from the following files:

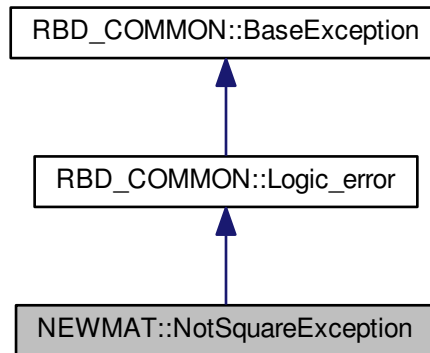
- newmat/include/newmat.h
- newmat/src/newmatex.cpp



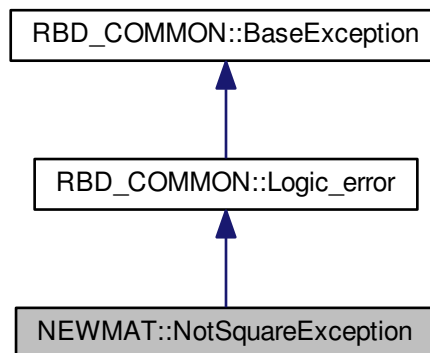
## 29.309 NEWMAT::NotSquareException Class Reference

```
#include <newmat.h>
```

Inheritance diagram for NEWMAT::NotSquareException:



Collaboration diagram for NEWMAT::NotSquareException:



### Public Member Functions

- [NotSquareException](#) (const [GeneralMatrix](#) &A)

### Static Public Attributes

- static unsigned long [Select](#)

## Additional Inherited Members

### 29.309.1 Detailed Description

Definition at line 1655 of file newmat.h.

### 29.309.2 Constructor & Destructor Documentation

#### 29.309.2.1 NotSquareException::NotSquareException ( const GeneralMatrix & A )

Definition at line 143 of file newmatex.cpp.

### 29.309.3 Member Data Documentation

#### 29.309.3.1 unsigned long NotSquareException::Select [static]

Definition at line 1658 of file newmat.h.

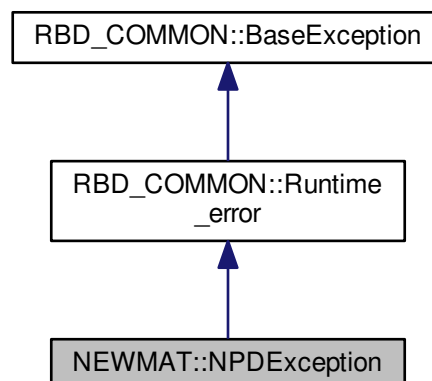
The documentation for this class was generated from the following files:

- [newmat/include/newmat.h](#)
- [newmat/src/newmatex.cpp](#)

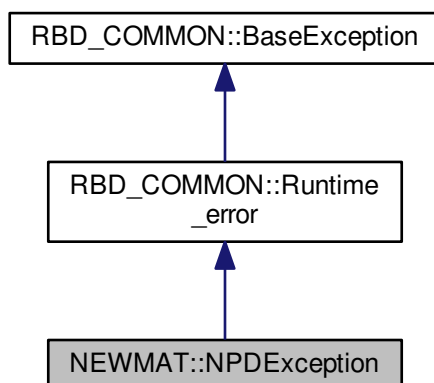
## 29.310 NEWMAT::NPDException Class Reference

```
#include <newmat.h>
```

Inheritance diagram for NEWMAT::NPDException:



Collaboration diagram for NEWMAT::NPDException:



### Public Member Functions

- [NPDException](#) (`const GeneralMatrix &`)

### Static Public Attributes

- static unsigned long [Select](#)

### Additional Inherited Members

#### 29.310.1 Detailed Description

Definition at line 1595 of file `newmat.h`.

#### 29.310.2 Constructor & Destructor Documentation

##### 29.310.2.1 `NPDException::NPDException ( const GeneralMatrix & A )`

Definition at line 45 of file `newmatex.cpp`.

#### 29.310.3 Member Data Documentation

##### 29.310.3.1 `unsigned long NPDException::Select [static]`

Definition at line 1598 of file `newmat.h`.

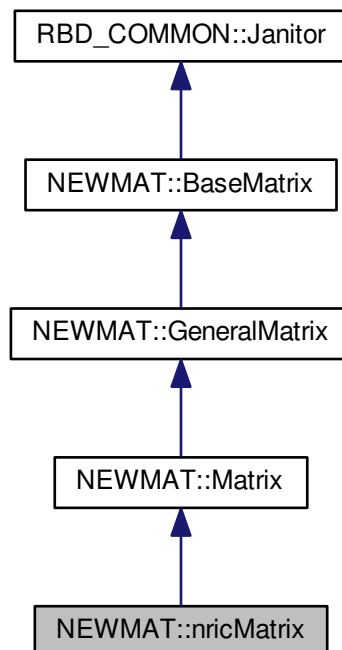
The documentation for this class was generated from the following files:

- `newmat/include/newmat.h`
- `newmat/src/newmatex.cpp`

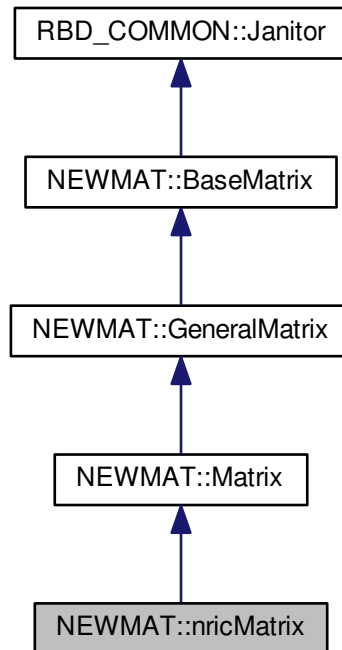
## 29.311 NEWMAT::nricMatrix Class Reference

```
#include <newmat.h>
```

Inheritance diagram for NEWMAT::nricMatrix:



Collaboration diagram for NEWMAT::nricMatrix:



## Public Member Functions

- [nricMatrix \(\)](#)
- [nricMatrix \(int m, int n\)](#)
- [nricMatrix \(const BaseMatrix &bm\)](#)
- [void operator= \(const BaseMatrix &bm\)](#)
- [void operator= \(Real f\)](#)
- [void operator= \(const nricMatrix &m\)](#)
- [void operator<< \(const BaseMatrix &X\)](#)
- [nricMatrix \(const nricMatrix &gm\)](#)
- [void ReSize \(int m, int n\)](#)
- [void ReSize \(const GeneralMatrix &A\)](#)
- [~nricMatrix \(\)](#)
- [Real \\*\\* nric \(\) const](#)
- [void CleanUp \(\)](#)

## Additional Inherited Members

### 29.311.1 Detailed Description

Definition at line 605 of file newmat.h.

## 29.311.2 Constructor & Destructor Documentation

29.311.2.1 `NEWMAT::nricMatrix::nricMatrix ( )` [inline]

Definition at line 613 of file newmat.h.

29.311.2.2 `NEWMAT::nricMatrix::nricMatrix ( int m, int n )` [inline]

Definition at line 614 of file newmat.h.

29.311.2.3 `NEWMAT::nricMatrix::nricMatrix ( const BaseMatrix & bm )` [inline]

Definition at line 616 of file newmat.h.

29.311.2.4 `NEWMAT::nricMatrix::nricMatrix ( const nricMatrix & gm )` [inline]

Definition at line 624 of file newmat.h.

29.311.2.5 `NEWMAT::nricMatrix::~~nricMatrix ( )` [inline]

Definition at line 628 of file newmat.h.

## 29.311.3 Member Function Documentation

29.311.3.1 `void nricMatrix::CleanUp ( )` [virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 829 of file newmat4.cpp.

29.311.3.2 `Real** NEWMAT::nricMatrix::nric ( ) const` [inline]

Definition at line 629 of file newmat.h.

29.311.3.3 `void NEWMAT::nricMatrix::operator<< ( const BaseMatrix & X )` [inline]

Definition at line 622 of file newmat.h.

29.311.3.4 `void NEWMAT::nricMatrix::operator= ( const BaseMatrix & bm )` [inline]

Definition at line 618 of file newmat.h.

29.311.3.5 void NEWMAT::nricMatrix::operator=( Real *f* ) [inline]

Definition at line 620 of file newmat.h.

29.311.3.6 void NEWMAT::nricMatrix::operator=( const nricMatrix & *m* ) [inline]

Definition at line 621 of file newmat.h.

29.311.3.7 void NEWMAT::nricMatrix::ReSize ( int *m*, int *n* ) [inline],[virtual]

Reimplemented from [NEWMAT::Matrix](#).

Definition at line 625 of file newmat.h.

29.311.3.8 void nricMatrix::ReSize ( const GeneralMatrix & *A* ) [virtual]

Reimplemented from [NEWMAT::Matrix](#).

Definition at line 270 of file newmat4.cpp.

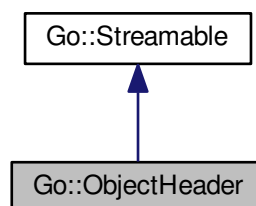
The documentation for this class was generated from the following files:

- [newmat/include/newmat.h](#)
- [newmat/src/newmat4.cpp](#)

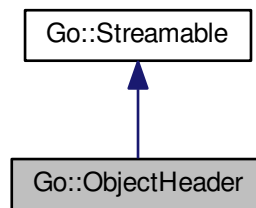
## 29.312 Go::ObjectHeader Class Reference

```
#include <ObjectHeader.h>
```

Inheritance diagram for Go::ObjectHeader:



Collaboration diagram for Go::ObjectHeader:



## Public Member Functions

- [ObjectHeader](#) ()  
*Default constructor (uninitialized header)*
- [ObjectHeader](#) ([ClassType](#) t, int major, int minor)
- [ObjectHeader](#) ([ClassType](#) t, int major, int minor, [const](#) std::vector< int > &auxdata)
- virtual [~ObjectHeader](#) ()  
*Virtual destructor, allowing safe destruction of derived objects.*
- virtual void [read](#) (std::istream &is)
- virtual void [write](#) (std::ostream &os) [const](#)
- [ClassType](#) [classType](#) () [const](#)  
*Get the ClassType stored in this [ObjectHeader](#).*
- int [majorVersion](#) () [const](#)  
*Get the major version number stored in this [ObjectHeader](#).*
- int [minorVersion](#) () [const](#)  
*Get the minor version number stored in this [ObjectHeader](#).*
- int [auxdataSize](#) () [const](#)
- int [auxdata](#) (int i) [const](#)

### 29.312.1 Detailed Description

An object representing the "header" usually preceding a [GeomObject](#) in a stream. This header contains information about the [GeomObject](#), and this information can be read by [ObjectHeader](#) and accessed by its member functions.

Definition at line 58 of file `ObjectHeader.h`.

### 29.312.2 Constructor & Destructor Documentation

#### 29.312.2.1 `Go::ObjectHeader::ObjectHeader ( )` `[inline]`

Default constructor (uninitialized header)

Definition at line 62 of file `ObjectHeader.h`.

#### 29.312.2.2 `Go::ObjectHeader::ObjectHeader ( ClassType t, int major, int minor )` `[inline]`

Constructor creating an [ObjectHeader](#) that is initialized with a given `ClassType`, major and minor version.



## Parameters

<i>t</i>	the ClassType of the <a href="#">GeomObject</a> that this <a href="#">ObjectHeader</a> shall represent.
<i>major</i>	major version number
<i>minor</i>	minor version number

Definition at line 74 of file ObjectHeader.h.

**29.312.2.3** `Go::ObjectHeader::ObjectHeader ( ClassType t, int major, int minor, const std::vector< int > & auxdata )`  
`[inline]`

Constructor creating an [ObjectHeader](#) that is initialized with a given ClassType, major and minor version and auxiliary, class- specific data.

## Parameters

<i>t</i>	the ClassType of the <a href="#">GeomObject</a> that this <a href="#">ObjectHeader</a> shall represent.
<i>major</i>	major version number
<i>minor</i>	minor version number
<i>auxdata</i>	auxiliary data, specific to the ClassType.

Definition at line 88 of file ObjectHeader.h.

**29.312.2.4** `virtual Go::ObjectHeader::~~ObjectHeader ( )` `[virtual]`

Virtual destructor, allowing safe destruction of derived objects.

**29.312.3 Member Function Documentation**

**29.312.3.1** `int Go::ObjectHeader::auxdata ( int i ) const` `[inline]`

Get a certain piece of auxiliary data (an integer).

## Parameters

<i>i</i>	the requested integer's position in the auxiliary data vector
----------	---------------------------------------------------------------

Definition at line 122 of file ObjectHeader.h.

**29.312.3.2** `int Go::ObjectHeader::auxdataSize ( ) const` `[inline]`

Get the size of the auxiliary data stored in this [ObjectHeader](#) (size measured in number of ints).

Definition at line 118 of file ObjectHeader.h.

29.312.3.3 `ClassType Go::ObjectHeader::classType ( ) const [inline]`

Get the `ClassType` stored in this [ObjectHeader](#).

Definition at line 108 of file `ObjectHeader.h`.

29.312.3.4 `int Go::ObjectHeader::majorVersion ( ) const [inline]`

Get the major version number stored in this [ObjectHeader](#).

Definition at line 111 of file `ObjectHeader.h`.

29.312.3.5 `int Go::ObjectHeader::minorVersion ( ) const [inline]`

Get the minor version number stored in this [ObjectHeader](#).

Definition at line 114 of file `ObjectHeader.h`.

29.312.3.6 `virtual void Go::ObjectHeader::read ( std::istream & is ) [virtual]`

Read the [ObjectHeader](#) from an input stream

Parameters

<code>is</code>	the input stream from which the <a href="#">ObjectHeader</a> is read
-----------------	----------------------------------------------------------------------

Implements [Go::Streamable](#).

29.312.3.7 `virtual void Go::ObjectHeader::write ( std::ostream & os ) const [virtual]`

Write the [ObjectHeader](#) to an output stream

Parameters

<code>os</code>	the output stream to which the <a href="#">ObjectHeader</a> is written
-----------------	------------------------------------------------------------------------

Implements [Go::Streamable](#).

The documentation for this class was generated from the following file:

- `gotools-core/include/GoTools/geometry/ObjectHeader.h`

## 29.313 RBD\_COMMON::OneDimSolve Class Reference

```
#include <solution.h>
```

## Public Member Functions

- [OneDimSolve](#) ([R1\\_R1](#) &*f*, [Real](#) *AccY*=0.0001, [Real](#) *AccX*=0.0)
- [Real Solve](#) ([Real](#) *Y*, [Real](#) *X*, [Real](#) *Dev*, int *Lim*=100)

### 29.313.1 Detailed Description

Definition at line 52 of file `solution.h`.

### 29.313.2 Constructor & Destructor Documentation

**29.313.2.1** `RBD_COMMON::OneDimSolve::OneDimSolve ( R1_R1 & f, Real AccY = 0.0001, Real AccX = 0.0 )`  
`[inline]`

Definition at line 60 of file `solution.h`.

### 29.313.3 Member Function Documentation

**29.313.3.1** `Real OneDimSolve::Solve ( Real Y, Real X, Real Dev, int Lim = 100 )`

Definition at line 96 of file `solution.cpp`.

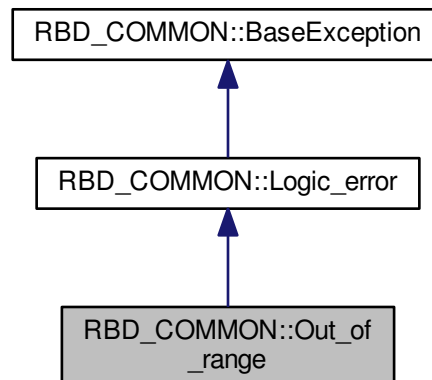
The documentation for this class was generated from the following files:

- [newmat/include/solution.h](#)
- [newmat/src/solution.cpp](#)

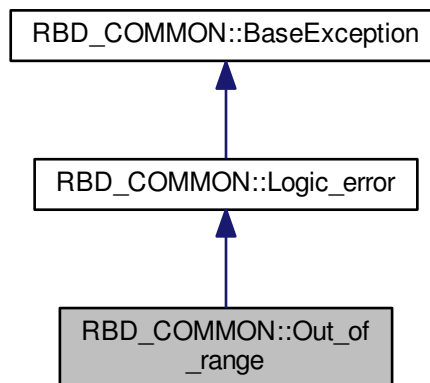
## 29.314 RBD\_COMMON::Out\_of\_range Class Reference

```
#include <myexcept.h>
```

Inheritance diagram for `RBD_COMMON::Out_of_range`:



Collaboration diagram for RBD\_COMMON::Out\_of\_range:



- static unsigned long [Select](#)
- [Out\\_of\\_range](#) (`const char *a_what=0`)

### Additional Inherited Members

#### 29.314.1 Detailed Description

Definition at line 392 of file `myexcept.h`.

#### 29.314.2 Constructor & Destructor Documentation

##### 29.314.2.1 `Out_of_range::Out_of_range ( const char * a_what = 0 )`

Definition at line 433 of file `myexcept.cpp`.

#### 29.314.3 Member Data Documentation

##### 29.314.3.1 unsigned long `Out_of_range::Select` [static]

Definition at line 395 of file `myexcept.h`.

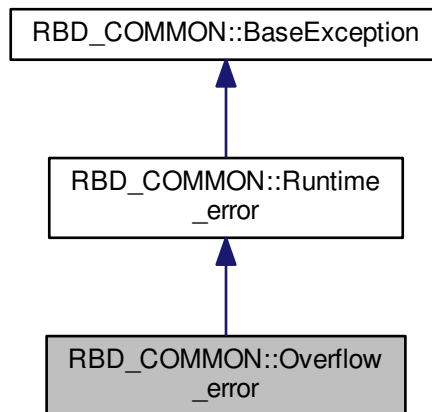
The documentation for this class was generated from the following files:

- `newmat/include/myexcept.h`
- `newmat/src/myexcept.cpp`

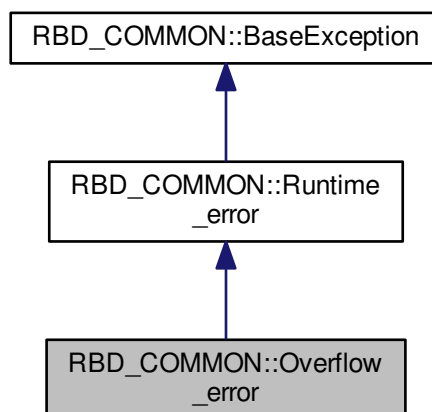
## 29.315 RBD\_COMMON::Overflow\_error Class Reference

```
#include <myexcept.h>
```

Inheritance diagram for RBD\_COMMON::Overflow\_error:



Collaboration diagram for RBD\_COMMON::Overflow\_error:



- static unsigned long [Select](#)
- [Overflow\\_error](#) (const char \*a\_what=0)

## Additional Inherited Members

### 29.315.1 Detailed Description

Definition at line 420 of file myexcept.h.

### 29.315.2 Constructor & Destructor Documentation

#### 29.315.2.1 `Overflow_error::Overflow_error ( const char * a_what = 0 )`

Definition at line 461 of file myexcept.cpp.

### 29.315.3 Member Data Documentation

#### 29.315.3.1 `unsigned long Overflow_error::Select` [static]

Definition at line 423 of file myexcept.h.

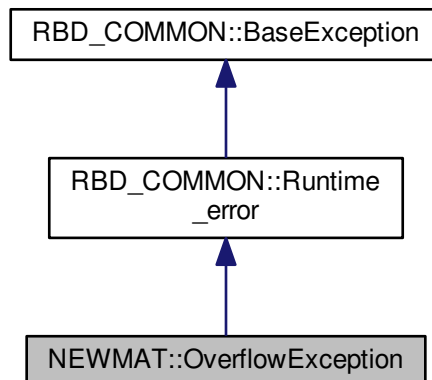
The documentation for this class was generated from the following files:

- [newmat/include/myexcept.h](#)
- [newmat/src/myexcept.cpp](#)

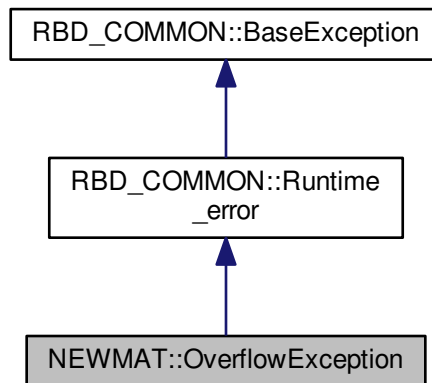
## 29.316 NEWMAT::OverflowException Class Reference

```
#include <newmat.h>
```

Inheritance diagram for NEWMAT::OverflowException:



Collaboration diagram for NEWMAT::OverflowException:



### Public Member Functions

- [OverflowException](#) (`const char *c`)

### Static Public Attributes

- static unsigned long [Select](#)

### Additional Inherited Members

#### 29.316.1 Detailed Description

Definition at line 1617 of file `newmat.h`.

#### 29.316.2 Constructor & Destructor Documentation

##### 29.316.2.1 `OverflowException::OverflowException ( const char * c )`

Definition at line 80 of file `newmatex.cpp`.

#### 29.316.3 Member Data Documentation

##### 29.316.3.1 `unsigned long OverflowException::Select [static]`

Definition at line 1620 of file `newmat.h`.

The documentation for this class was generated from the following files:

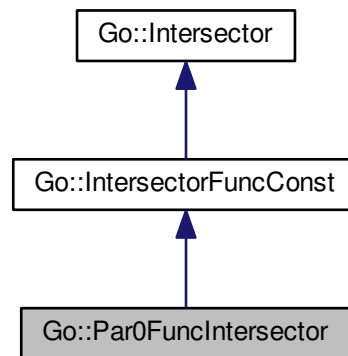
- `newmat/include/newmat.h`
- `newmat/src/newmatex.cpp`

## 29.317 Go::Par0FuncIntersector Class Reference

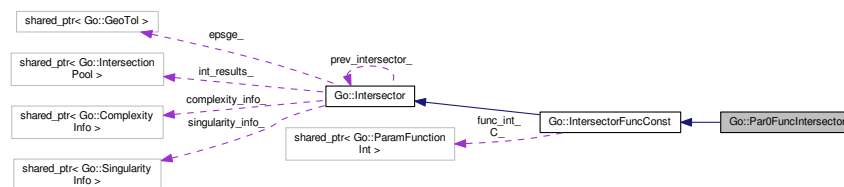
This class is performing intersections between two constants.

```
#include <Par0FuncIntersector.h>
```

Inheritance diagram for Go::Par0FuncIntersector:



Collaboration diagram for Go::Par0FuncIntersector:



### Public Member Functions

- `Par0FuncIntersector` (`shared_ptr< ParamFunctionInt > func`, `shared_ptr< ParamFunctionInt > C`, `shared_ptr< GeoTol > epsge`, `Intersector *prev=0`, `int eliminated_parameter=-1`, `double eliminated_value=0`)
- `virtual ~Par0FuncIntersector ()`  
*Destructor.*
- `virtual int numParams () const`

### Protected Member Functions

- `virtual shared_ptr< Intersector > lowerOrderIntersector` (`shared_ptr< ParamFunctionInt > obj1`, `shared_ptr< ParamFunctionInt > obj2`, `Intersector *prev=0`, `int eliminated_parameter=-1`, `double eliminated_value=0`)
- `virtual int checkCoincidence ()`
- `virtual void microCase ()`
- `virtual int updateIntersections ()`
- `virtual int repairIntersections ()`
- `virtual int doSubdivide ()`



## Additional Inherited Members

### 29.317.1 Detailed Description

This class is performing intersections between two constants.

Definition at line 52 of file Par0FuncIntersector.h.

### 29.317.2 Constructor & Destructor Documentation

29.317.2.1 `Go::Par0FuncIntersector::Par0FuncIntersector ( shared_ptr< ParamFunctionInt > func, shared_ptr< ParamFunctionInt > C, shared_ptr< GeoTol > epsge, Intersector * prev = 0, int eliminated_parameter = -1, double eliminated_value = 0 )`

Constructor. Both objects should refer to constants (this is not checked compile-time, so we rely on the user to obey this rule). The last two variables are relevant only if the parent has one more parameter than the [Intersector](#) to be constructed.

#### Parameters

<i>func</i>	of type <a href="#">Param0FunctionInt</a> .
<i>C</i>	of type <a href="#">Param0FunctionInt</a> .
<i>epsge</i>	the associated tolerance.
<i>prev</i>	the "parent" <a href="#">Intersector</a> (0 if there is no parent).
<i>eliminated_parameter</i>	the index (0) of the parameter that was removed from the parent <i>prev</i> .
<i>eliminated_value</i>	the value of the parameter that was removed from the parent <i>prev</i> .

29.317.2.2 `virtual Go::Par0FuncIntersector::~~Par0FuncIntersector ( ) [virtual]`

Destructor.

### 29.317.3 Member Function Documentation

29.317.3.1 `virtual int Go::Par0FuncIntersector::checkCoincidence ( ) [protected],[virtual]`

Implements [Go::IntersectorFuncConst](#).

29.317.3.2 `virtual int Go::Par0FuncIntersector::doSubdivide ( ) [protected],[virtual]`

Implements [Go::IntersectorFuncConst](#).

29.317.3.3 `virtual shared_ptr< Intersector > Go::Par0FuncIntersector::lowerOrderIntersector ( shared_ptr< ParamFunctionInt > obj1, shared_ptr< ParamFunctionInt > obj2, Intersector * prev = 0, int eliminated_parameter = -1, double eliminated_value = 0 ) [protected],[virtual]`

Implements [Go::IntersectorFuncConst](#).

29.317.3.4 virtual void Go::Par0FuncIntersector::microCase ( ) [protected],[virtual]

Implements [Go::IntersectorFuncConst](#).

29.317.3.5 virtual int Go::Par0FuncIntersector::numParams ( ) const [inline],[virtual]

Return the number of parameter directions for the object.

Returns

the number of parameter directions

Implements [Go::Intersector](#).

Definition at line 91 of file Par0FuncIntersector.h.

29.317.3.6 virtual int Go::Par0FuncIntersector::repairIntersections ( ) [inline],[protected],[virtual]

Implements [Go::Intersector](#).

Definition at line 110 of file Par0FuncIntersector.h.

29.317.3.7 virtual int Go::Par0FuncIntersector::updateIntersections ( ) [protected],[virtual]

Implements [Go::IntersectorFuncConst](#).

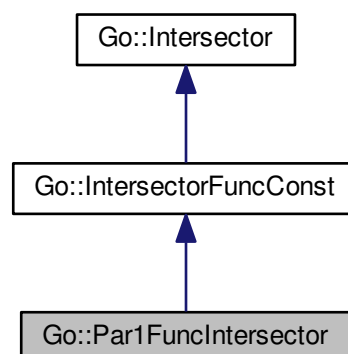
The documentation for this class was generated from the following file:

- [intersections/include/GoTools/intersections/Par0FuncIntersector.h](#)

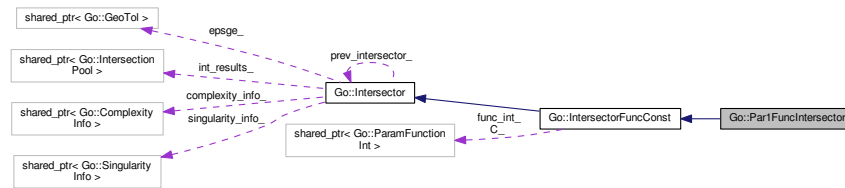
## 29.318 Go::Par1FuncIntersector Class Reference

```
#include <Par1FuncIntersector.h>
```

Inheritance diagram for Go::Par1FuncIntersector:



Collaboration diagram for Go::Par1FuncIntersector:



## Public Member Functions

- [Par1FuncIntersector](#) (shared\_ptr< [ParamFunctionInt](#) > func, shared\_ptr< [ParamFunctionInt](#) > C, shared\_ptr< [GeoTol](#) > epsge, [Intersector](#) \*prev=0, int eliminated\_parameter=-1, double eliminated\_value=0)
- virtual [~Par1FuncIntersector](#) ()  
*Destructor.*
- virtual int [numParams](#) () const

## Protected Member Functions

- virtual shared\_ptr< [Intersector](#) > [lowerOrderIntersector](#) (shared\_ptr< [ParamFunctionInt](#) > obj1, shared\_ptr< [ParamFunctionInt](#) > obj2, [Intersector](#) \*prev=0, int eliminated\_parameter=-1, double eliminated\_value=0)
- virtual int [checkCoincidence](#) ()
- virtual void [microCase](#) ()
- virtual int [updateIntersections](#) ()
- virtual int [repairIntersections](#) ()
- virtual int [doSubdivide](#) ()

## Additional Inherited Members

### 29.318.1 Detailed Description

This class is performing intersections between a 1-dimensional parametric curve and a constant.

Definition at line 53 of file Par1FuncIntersector.h.

### 29.318.2 Constructor & Destructor Documentation

**29.318.2.1** `Go::Par1FuncIntersector::Par1FuncIntersector ( shared_ptr< ParamFunctionInt > func, shared_ptr< ParamFunctionInt > C, shared_ptr< GeoTol > epsge, Intersector * prev = 0, int eliminated_parameter = -1, double eliminated_value = 0 )`

Constructor. One of the objects should refer to a 1D-curve, the other a constant (this is not checked compile-time, so we rely on the user to obey this rule). The last two variables are relevant only if the parent has one more parameter than the [Intersector](#) to be constructed.

## Parameters

<i>func</i>	of type <a href="#">Param1FunctionInt</a> .
<i>C</i>	of type <a href="#">Param0FunctionInt</a> .
<i>epsge</i>	the associated tolerance.
<i>prev</i>	the "parent" <a href="#">Intersector</a> (0 if there is no parent).
<i>eliminated_parameter</i>	the index (0 or 1) of the parameter that was removed from the parent <i>prev</i> .
<i>eliminated_value</i>	the value of the parameter that was removed from the parent <i>prev</i> .

29.318.2.2 `virtual Go::Par1FuncIntersector::~~Par1FuncIntersector ( ) [virtual]`

Destructor.

### 29.318.3 Member Function Documentation

29.318.3.1 `virtual int Go::Par1FuncIntersector::checkCoincidence ( ) [protected],[virtual]`

Implements [Go::IntersectorFuncConst](#).

29.318.3.2 `virtual int Go::Par1FuncIntersector::doSubdivide ( ) [protected],[virtual]`

Implements [Go::IntersectorFuncConst](#).

29.318.3.3 `virtual shared_ptr<Intersector> Go::Par1FuncIntersector::lowerOrderIntersector ( shared_ptr<ParamFunctionInt > obj1, shared_ptr< ParamFunctionInt > obj2, Intersector * prev = 0, int eliminated_parameter = -1, double eliminated_value = 0 ) [protected],[virtual]`

Implements [Go::IntersectorFuncConst](#).

29.318.3.4 `virtual void Go::Par1FuncIntersector::microCase ( ) [protected],[virtual]`

Implements [Go::IntersectorFuncConst](#).

29.318.3.5 `virtual int Go::Par1FuncIntersector::numParams ( ) const [inline],[virtual]`

Return the number of parameter directions for the object.

#### Returns

the number of parameter directions

Implements [Go::Intersector](#).

Definition at line 93 of file [Par1FuncIntersector.h](#).

29.318.3.6 `virtual int Go::Par1FuncIntersector::repairIntersections ( ) [inline],[protected],[virtual]`

Implements [Go::Intersector](#).

Definition at line 112 of file `Par1FuncIntersector.h`.

29.318.3.7 `virtual int Go::Par1FuncIntersector::updateIntersections ( ) [protected],[virtual]`

Implements [Go::IntersectorFuncConst](#).

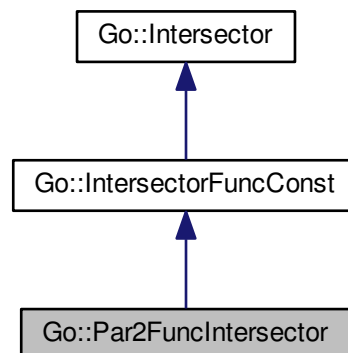
The documentation for this class was generated from the following file:

- `intersections/include/GoTools/intersections/Par1FuncIntersector.h`

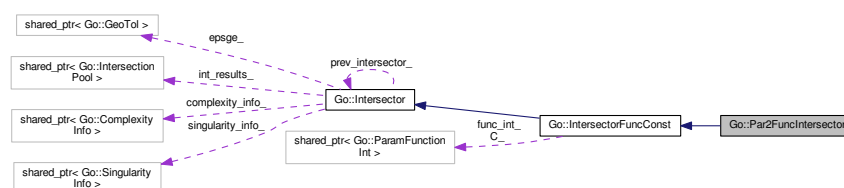
## 29.319 Go::Par2FuncIntersector Class Reference

```
#include <Par2FuncIntersector.h>
```

Inheritance diagram for `Go::Par2FuncIntersector`:



Collaboration diagram for `Go::Par2FuncIntersector`:



## Public Member Functions

- [Par2FuncIntersector](#) (shared\_ptr< [ParamFunctionInt](#) > func, shared\_ptr< [ParamFunctionInt](#) > C, shared\_ptr< [GeoTol](#) > epsge, [Intersector](#) \*prev=0, int eliminated\_parameter=-1, double eliminated\_value=0)
- virtual [~Par2FuncIntersector](#) ()
  - Destructor.*
- virtual int [numParams](#) () const

## Protected Member Functions

- virtual shared\_ptr< [Intersector](#) > [lowerOrderIntersector](#) (shared\_ptr< [ParamFunctionInt](#) > obj1, shared\_ptr< [ParamFunctionInt](#) > obj2, [Intersector](#) \*prev=0, int eliminated\_parameter=-1, double eliminated\_value=0)
- virtual int [checkCoincidence](#) ()
- virtual void [microCase](#) ()
- virtual int [updateIntersections](#) ()
- virtual int [repairIntersections](#) ()
- [bool](#) [isConnected](#) (std::vector< shared\_ptr< [IntersectionPoint](#) > > bd\_ints, int nmbbd)
- [bool](#) [isConnected](#) (std::vector< std::pair< shared\_ptr< [IntersectionPoint](#) >, [IntPtClassification](#) > > &bd\_ints, int nmb\_nottouch)
- [bool](#) [connectDirected](#) (std::vector< std::pair< shared\_ptr< [IntersectionPoint](#) >, [IntPtClassification](#) > > &bd\_ints, int nmbbd)
- [bool](#) [canConnect](#) (shared\_ptr< [IntersectionPoint](#) > pt1, shared\_ptr< [IntersectionPoint](#) > pt2)
- virtual int [doSubdivide](#) ()

## Additional Inherited Members

### 29.319.1 Detailed Description

This class is performing intersections between a 1-dimensional parametric surface and a constant.

Definition at line 54 of file [Par2FuncIntersector.h](#).

### 29.319.2 Constructor & Destructor Documentation

**29.319.2.1** `Go::Par2FuncIntersector::Par2FuncIntersector ( shared_ptr< ParamFunctionInt > func, shared_ptr< ParamFunctionInt > C, shared_ptr< GeoTol > epsge, Intersector * prev = 0, int eliminated_parameter = -1, double eliminated_value = 0 )`

Constructor. One of the objects should refer to a 1D-surface, the other a constant (this is not checked compile-time, so we rely on the user to obey this rule). The last two variables are relevant only if the parent has one more parameter than the [Intersector](#) to be constructed.

#### Parameters

<i>func</i>	of type <a href="#">Param2FunctionInt</a> .
<i>C</i>	of type <a href="#">Param0FunctionInt</a> .
<i>epsg</i>	the associated tolerance.
<i>prev</i>	the "parent" <a href="#">Intersector</a> (0 if there is no parent).
<i>eliminated_parameter</i>	the index of the parameter that was removed from the parent <i>prev</i> .
<i>eliminated_value</i>	the value of the parameter that was removed from the parent <i>prev</i> .

29.319.2.2 `virtual Go::Par2FuncIntersector::~~Par2FuncIntersector ( ) [virtual]`

Destructor.

### 29.319.3 Member Function Documentation

29.319.3.1 `bool Go::Par2FuncIntersector::canConnect ( shared_ptr< IntersectionPoint > pt1, shared_ptr< IntersectionPoint > pt2 ) [protected]`

29.319.3.2 `virtual int Go::Par2FuncIntersector::checkCoincidence ( ) [protected],[virtual]`

Implements [Go::IntersectorFuncConst](#).

29.319.3.3 `bool Go::Par2FuncIntersector::connectDirected ( std::vector< std::pair< shared_ptr< IntersectionPoint >, IntPtClassification >> & bd_ints, int nmbbd ) [protected]`

29.319.3.4 `virtual int Go::Par2FuncIntersector::doSubdivide ( ) [protected],[virtual]`

Implements [Go::IntersectorFuncConst](#).

29.319.3.5 `bool Go::Par2FuncIntersector::isConnected ( std::vector< shared_ptr< IntersectionPoint >> bd_ints, int nmbbd ) [protected]`

29.319.3.6 `bool Go::Par2FuncIntersector::isConnected ( std::vector< std::pair< shared_ptr< IntersectionPoint >, IntPtClassification >> & bd_ints, int nmb_nottouch ) [protected]`

29.319.3.7 `virtual shared_ptr<Intersector> Go::Par2FuncIntersector::lowerOrderIntersector ( shared_ptr< ParamFunctionInt > obj1, shared_ptr< ParamFunctionInt > obj2, Intersector * prev = 0, int eliminated_parameter = -1, double eliminated_value = 0 ) [protected],[virtual]`

Implements [Go::IntersectorFuncConst](#).

29.319.3.8 `virtual void Go::Par2FuncIntersector::microCase ( ) [protected],[virtual]`

Implements [Go::IntersectorFuncConst](#).

29.319.3.9 `virtual int Go::Par2FuncIntersector::numParams ( ) const [inline],[virtual]`

Return the number of parameter directions for the object.

Returns

the number of parameter directions

Implements [Go::Intersector](#).

Definition at line 92 of file `Par2FuncIntersector.h`.

29.319.3.10 `virtual int Go::Par2FuncIntersector::repairIntersections ( ) [inline],[protected],[virtual]`

Implements [Go::Intersector](#).

Definition at line 111 of file `Par2FuncIntersector.h`.

29.319.3.11 `virtual int Go::Par2FuncIntersector::updateIntersections ( ) [protected],[virtual]`

Implements [Go::IntersectorFuncConst](#).

The documentation for this class was generated from the following file:

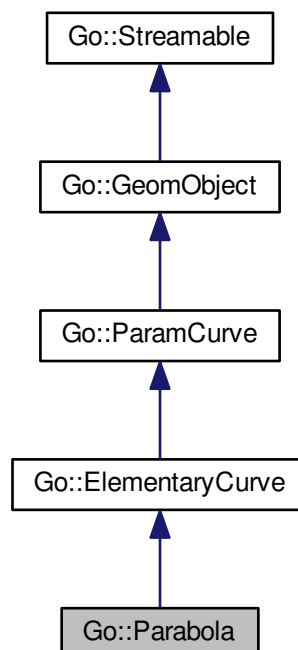
- `intersections/include/GoTools/intersections/Par2FuncIntersector.h`

## 29.320 [Go::Parabola](#) Class Reference

Class that represents a parabola. It is a subclass of [ElementaryCurve](#) and thus has a parametrization.

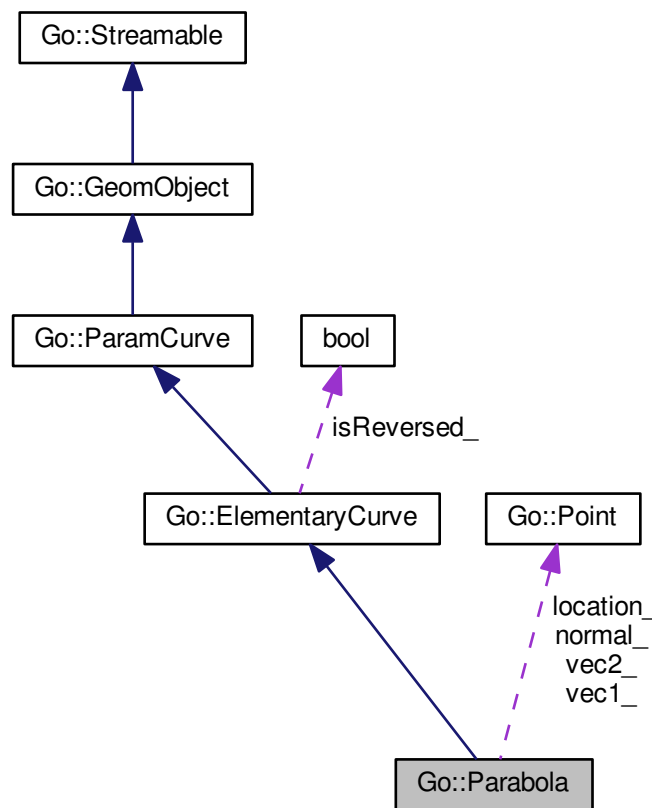
```
#include <Parabola.h>
```

Inheritance diagram for `Go::Parabola`:





Collaboration diagram for Go::Parabola:



## Public Member Functions

- [Parabola](#) ()
- [Parabola](#) ([Point](#) location, [Point](#) direction, [Point](#) normal, [double](#) focal\_dist, [bool](#) isReversed=false)
- virtual [~Parabola](#) ()
  - virtual destructor - ensures safe inheritance*
- virtual void [read](#) (std::istream &is)
- virtual void [write](#) (std::ostream &os) **const**
- virtual [BoundingBox](#) boundingBox () **const**
  - Return the object's bounding box.*
- virtual int [dimension](#) () **const**
  - Return the dimension of the space in which the object lies (usually 2 or 3)*
- virtual [ClassType](#) instanceType () **const**
  - Return the class type identifier of a given, derived instance of [GeomObject](#).*
- virtual [Parabola](#) \* [clone](#) () **const**
- virtual void [point](#) ([Point](#) &pt, [double](#) tpar) **const**
- virtual void [point](#) (std::vector< [Point](#) > &pts, [double](#) tpar, int derives, [bool](#) from\_right=true) **const**
- virtual [double](#) startparam () **const**
- virtual [double](#) endparam () **const**

- virtual void [setParameterInterval](#) (double t1, double t2)  
*Limit the curve by limiting the parameter interval.*
- virtual [SplineCurve](#) \* [geometryCurve](#) ()
- virtual [SplineCurve](#) \* [createSplineCurve](#) () const  
*Fetch spline representation of curve.*
- virtual [bool](#) [isDegenerate](#) (double degenerate\_epsilon)
- virtual [Parabola](#) \* [subCurve](#) (double from\_par, double to\_par, double fuzzy=DEFAULT\_PARAMETER\_EPSILON) const
- virtual [DirectionCone](#) [directionCone](#) () const
- virtual void [appendCurve](#) ([ParamCurve](#) \*cv, [bool](#) reparam=true)
- virtual void [appendCurve](#) ([ParamCurve](#) \*cv, int continuity, [double](#) &dist, [bool](#) reparam=true)
- virtual void [closestPoint](#) (const [Point](#) &pt, [double](#) tmin, [double](#) tmax, [double](#) &clo\_t, [Point](#) &clo\_pt, [double](#) &clo\_dist, [double](#) const \*seed=0) const
- virtual [double](#) [length](#) ([double](#) tol)
- virtual void [setParamBounds](#) ([double](#) startpar, [double](#) endpar)
- [bool](#) [isBounded](#) () const
- virtual void [translateCurve](#) (const [Point](#) &dir)
- virtual void [swapParameters2D](#) ()
- virtual [bool](#) [isInPlane](#) (const [Point](#) &norm, [double](#) eps, [Point](#) &pos) const  
*Check if the parabola lies in a plane with a given normal.*

### Static Public Member Functions

- static [ClassType](#) [classType](#) ()

### Protected Member Functions

- void [setSpanningVectors](#) ()

### Protected Attributes

- [Point](#) [location\\_](#)
- [Point](#) [vec1\\_](#)
- [Point](#) [vec2\\_](#)
- [Point](#) [normal\\_](#)
- [double](#) [f\\_](#)
- [double](#) [startparam\\_](#)
- [double](#) [endparam\\_](#)

## 29.320.1 Detailed Description

Class that represents a parabola. It is a subclass of [ElementaryCurve](#) and thus has a parametrization.

An parabola has a natural parametrization in terms of its values:  $f(t) = C + F*(t^2 * x + 2 * t * y)$ . The parametrization is unbounded:  $-\infty < t < \infty$ .

Definition at line 59 of file [Parabola.h](#).

## 29.320.2 Constructor & Destructor Documentation

### 29.320.2.1 Go::Parabola::Parabola ( ) [inline]

Default constructor. Constructs an uninitialized [Parabola](#) which can only be assigned to or read into.

Definition at line 64 of file Parabola.h.

### 29.320.2.2 Go::Parabola::Parabola ( Point location, Point direction, Point normal, double focal\_dist, bool isReversed = false )

Constructor. Input is location, direction (x-axis), normal and focal dist. The default [Parabola](#) is unbounded in its parametrization. To bound it, use [setParamBounds\(\)](#).

### 29.320.2.3 virtual Go::Parabola::~~Parabola ( ) [virtual]

virtual destructor - ensures safe inheritance

## 29.320.3 Member Function Documentation

### 29.320.3.1 virtual void Go::Parabola::appendCurve ( ParamCurve \* cv, bool reparam = true ) [virtual]

append a curve to this curve, with eventual reparametrization NB: This virtual member function currently only works for [SplineCurves](#) and [CurveOnSurfaces](#). Moreover, 'this' curve and the 'cv' curve must be of the same type.

#### Parameters

<i>cv</i>	the curve to append to 'this' curve.
<i>reparam</i>	specify whether or not there should be reparametrization

Implements [Go::ParamCurve](#).

### 29.320.3.2 virtual void Go::Parabola::appendCurve ( ParamCurve \* cv, int continuity, double & dist, bool reparam = true ) [virtual]

append a curve to this curve, with eventual reparametrization

#### Parameters

<i>cv</i>	the curve to append to 'this' curve.
<i>continuity</i>	the required continuity at the transition. Can be $G^{(-1)}$ and upwards.
<i>dist</i>	a measure of the local distortion around the transition in order to achieve the specified continuity.
<i>reparam</i>	specify whether or not there should be reparametrization

Implements [Go::ParamCurve](#).

29.320.3.3 virtual **BoundingBox** Go::Parabola::boundingBox ( ) const [virtual]

Return the object's bounding box.

Implements [Go::GeomObject](#).

29.320.3.4 static **ClassType** Go::Parabola::classType ( ) [static]

29.320.3.5 virtual **Parabola\*** Go::Parabola::clone ( ) const [virtual]

The clone-function is herited from [GeomObject](#), but overridden here to get a covariant return type (for those compilers that allow this).

Implements [Go::ElementaryCurve](#).

29.320.3.6 virtual void Go::Parabola::closestPoint ( const **Point** & *pt*, double *tmin*, double *tmax*, double & *clo\_t*, **Point** & *clo\_pt*, double & *clo\_dist*, double const \* *seed* = 0 ) const [virtual]

Compute the closest point from an interval of this curve to a specified point.

Parameters

<i>pt</i>	point we want to find the closest point to
<i>tmin</i>	start parameter of search interval
<i>tmax</i>	end parameter of search interval
<i>clo_t</i>	upon function return, 'clo_t' will contain the parameter value of the closest point found.
<i>clo_pt</i>	upon function return, 'clo_pt' will contain the position of the closest point found.
<i>clo_dist</i>	upon function return, 'clo_dist' will contain the distance between 'pt' and the closest point found.
<i>seed</i>	pointer to initial guess value, provided by the user (can be 0, for which the algorithm will determine a (hopefully) reasonable choice).

Implements [Go::ParamCurve](#).

29.320.3.7 virtual **SplineCurve\*** Go::Parabola::createSplineCurve ( ) const [virtual]

Fetch spline representation of curve.

Implements [Go::ElementaryCurve](#).

29.320.3.8 virtual int Go::Parabola::dimension ( ) const [virtual]

Return the dimension of the space in which the object lies (usually 2 or 3)

Implements [Go::GeomObject](#).

29.320.3.9 virtual [DirectionCone](#) Go::Parabola::directionCone ( ) const [virtual]

Creates a [DirectionCone](#) which covers all tangent directions of this curve.

Returns

the smallest [DirectionCone](#) containing all tangent directions of this curve.

Implements [Go::ParamCurve](#).

29.320.3.10 virtual double Go::Parabola::endparam ( ) const [virtual]

Query the end parameter of the curve

Returns

the curve's end parameter

Implements [Go::ParamCurve](#).

29.320.3.11 virtual [SplineCurve\\*](#) Go::Parabola::geometryCurve ( ) [virtual]

If the definition of this [ParamCurve](#) contains a [SplineCurve](#) describing its spatial shape, then this function will return a pointer to this [SplineCurve](#). Otherwise it will return a null pointer. The returned curve is NEWed, so the user is responsible for deleting it. This function may have side-effects.

Returns

a pointer to a [SplineCurve](#) representation of the [ParamCurve](#), if it exists. Null pointer otherwise.

Implements [Go::ParamCurve](#).

29.320.3.12 virtual [ClassType](#) Go::Parabola::instanceType ( ) const [virtual]

Return the class type identifier of a given, derived instance of [GeomObject](#).

Implements [Go::GeomObject](#).

29.320.3.13 bool Go::Parabola::isBounded ( ) const

Query if parametrization is bounded. Both upper and lower parameter bounds must be finite for this to be true.

Returns

*true* if bounded, *false* otherwise

29.320.3.14 virtual bool Go::Parabola::isDegenerate ( double *degenerate\_epsilon* ) [virtual]

Query whether the curve is degenerate (collapsed into a single point).

## Parameters

<i>degenerate_epsilon</i>	the tolerance used in determine whether the curve is degenerate. A curve is considered degenerate if its total length is shorter than this value.
---------------------------	---------------------------------------------------------------------------------------------------------------------------------------------------

## Returns

`true` if the curve is degenerate, `false` otherwise.

Implements [Go::ParamCurve](#).

29.320.3.15 `virtual bool Go::Parabola::isInPlane ( const Point & norm, double eps, Point & pos ) const` [virtual]

Check if the parabola lies in a plane with a given normal.

Reimplemented from [Go::ParamCurve](#).

29.320.3.16 `virtual double Go::Parabola::length ( double tol )` [virtual]

Compute the total length of this curve up to some tolerance

## Parameters

<i>tol</i>	the relative tolerance when approximating the length, i.e. stop iteration when error becomes smaller than $tol/(curve\ length)$
------------	---------------------------------------------------------------------------------------------------------------------------------

## Returns

the length calculated

Implements [Go::ParamCurve](#).

29.320.3.17 `virtual void Go::Parabola::point ( Point & pt, double tpar ) const` [virtual]

Evaluate the curve's position at a given parameter

## Parameters

<i>pt</i>	the evaluated position will be written to this <a href="#">Point</a>
<i>tpar</i>	the parameter for which we wish to evaluate the curve

Implements [Go::ParamCurve](#).

29.320.3.18 `virtual void Go::Parabola::point ( std::vector< Point > & pts, double tpar, int derivs, bool from_right = true ) const [virtual]`

Evaluate the curve's position and a certain number of derivatives at a given parameter.

#### Parameters

<i>pts</i>	the evaluated position and derivatives (tangent, curvature vector, etc.) will be written to this vector. The first entry will be the position, the second entry will be the first derivative, etc. The size of this vector must be set to 'derivs'+ 1 prior to calling this function.
<i>tpar</i>	the parameter for which we want to evaluate the curve
<i>derivs</i>	the number of derivatives we want to have calculated
<i>from_right</i>	specify whether we should calculate derivatives 'from the right' or 'from the left' (default is from the right). This matters only when the curve presents discontinuities in its derivatives.

Implements [Go::ParamCurve](#).

29.320.3.19 `virtual void Go::Parabola::read ( std::istream & is ) [virtual]`

Read object from stream

#### Parameters

<i>is</i>	stream from which object is read
-----------	----------------------------------

Implements [Go::Streamable](#).

29.320.3.20 `virtual void Go::Parabola::setParamBounds ( double startpar, double endpar ) [virtual]`

Set bounds for the parametrization of the [Parabola](#).

#### Parameters

<i>startpar</i>	start parameter
<i>endpar</i>	end parameter

Implements [Go::ElementaryCurve](#).

29.320.3.21 `virtual void Go::Parabola::setParameterInterval ( double t1, double t2 ) [virtual]`

Limit the curve by limiting the parameter interval.

Implements [Go::ParamCurve](#).

29.320.3.22 `void Go::Parabola::setSpanningVectors ( ) [protected]`

29.320.3.23 `virtual double Go::Parabola::startparam ( ) const [virtual]`

Query the start parameter of the curve

#### Returns

the curve's start parameter

Implements [Go::ParamCurve](#).

29.320.3.24 `virtual Parabola* Go::Parabola::subCurve ( double from_par, double to_par, double fuzzy = DEFAULT_PARAMETER_EPSILON ) const [virtual]`

Returns a curve which is a part of this curve. The result is NEWed, so the user is responsible for deleting it. NB: It is not guaranteed that the [ParamCurve](#) that is returned is of the same type as the curve itself. Thus, the returned curve might be a [SplineCurve](#).

#### Parameters

<i>from_par</i>	start value of parameter interval that will define the subcurve
<i>to_par</i>	end value of parameter interval that will define the subcurve
<i>fuzzy</i>	since subCurve works on those curves who are spline-based, this tolerance defines how close the start and end parameter must be to an existing knot in order to be considered <i>on</i> the knot.

#### Returns

a pointer to a new subcurve which represents the part of the curve between 'from\_par' and 'to\_par'. It will be spline-based and have a k-regular knotvector. The user is responsible for deleting this subcurve when it is no longer needed.

Implements [Go::ElementaryCurve](#).

29.320.3.25 `virtual void Go::Parabola::swapParameters2D ( ) [virtual]`

If the curve is 2 dimensional, x and y coordinates will be swapped. Used when curve is a parameter curve.

Implements [Go::ElementaryCurve](#).

29.320.3.26 `virtual void Go::Parabola::translateCurve ( const Point & dir ) [virtual]`

Implements [Go::ElementaryCurve](#).

29.320.3.27 `virtual void Go::Parabola::write ( std::ostream & os ) const [virtual]`

Write object to stream



## Parameters

<i>os</i>	stream to which object is written
-----------	-----------------------------------

Implements [Go::Streamable](#).

## 29.320.4 Member Data Documentation

### 29.320.4.1 `double Go::Parabola::endparam_` [protected]

Definition at line 174 of file Parabola.h.

### 29.320.4.2 `double Go::Parabola::f_` [protected]

Definition at line 171 of file Parabola.h.

### 29.320.4.3 `Point Go::Parabola::location_` [protected]

Definition at line 166 of file Parabola.h.

### 29.320.4.4 `Point Go::Parabola::normal_` [protected]

Definition at line 169 of file Parabola.h.

### 29.320.4.5 `double Go::Parabola::startparam_` [protected]

Definition at line 173 of file Parabola.h.

### 29.320.4.6 `Point Go::Parabola::vec1_` [protected]

Definition at line 167 of file Parabola.h.

### 29.320.4.7 `Point Go::Parabola::vec2_` [protected]

Definition at line 168 of file Parabola.h.

The documentation for this class was generated from the following file:

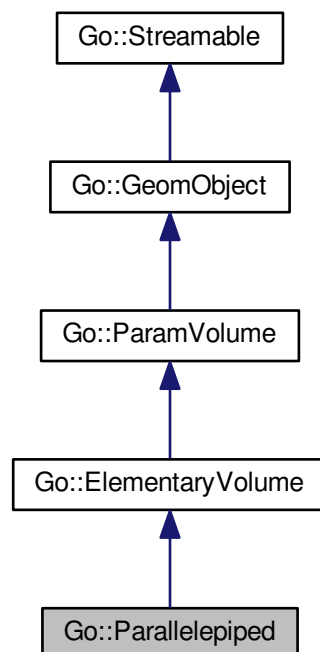
- [gotools-core/include/GoTools/geometry/Parabola.h](#)

## 29.321 Go::Parallelepiped Class Reference

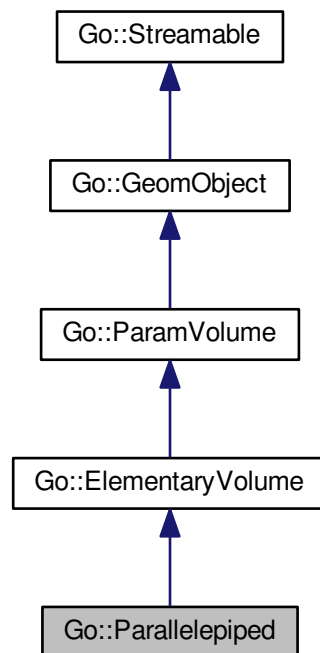
Class that represents a solid parallelepiped. It is a subclass of [ElementaryVolume](#), and thus has a parametrization.

```
#include <Parallelepiped.h>
```

Inheritance diagram for Go::Parallelepiped:



Collaboration diagram for Go::Parallelepiped:



## Public Member Functions

- [Parallelepiped](#) ()
- [Parallelepiped](#) ([Point](#) corner, [Point](#) dir\_u, [Point](#) dir\_v, [Point](#) dir\_w, [double](#) len\_u, [double](#) len\_v, [double](#) len\_w)  
*Constructor.*
- virtual [~Parallelepiped](#) ()  
*virtual destructor - ensures safe inheritance*
- virtual void [read](#) ([std::istream](#) &is)
- virtual void [write](#) ([std::ostream](#) &os) [const](#)
- virtual int [dimension](#) () [const](#)  
*Return the dimension of the space in which the object lies (usually 2 or 3)*
- virtual [ClassType](#) [instanceType](#) () [const](#)  
*Return the class type identifier of a given, derived instance of [GeomObject](#).*
- virtual [BoundingBox](#) [boundingBox](#) () [const](#)  
*Return the object's bounding box.*
- virtual [Parallelepiped](#) \* [clone](#) () [const](#)
- [DirectionCone](#) [tangentCone](#) (int pardir) [const](#)
- [const](#) [Array](#)< [double](#), 6 > [parameterSpan](#) () [const](#)
- void [point](#) ([Point](#) &pt, [double](#) upar, [double](#) vpar, [double](#) wpar) [const](#)
- void [point](#) ([std::vector](#)< [Point](#) > &pts, [double](#) upar, [double](#) vpar, [double](#) wpar, int derivs, [bool](#) u\_from\_↔right=true, [bool](#) v\_from\_right=true, [bool](#) w\_from\_right=true, [double](#) resolution=1.0e-12) [const](#)
- [double](#) [nextSegmentVal](#) (int dir, [double](#) par, [bool](#) forward, [double](#) tol) [const](#)
- void [closestPoint](#) ([const](#) [Point](#) &pt, [double](#) &clo\_u, [double](#) &clo\_v, [double](#) &clo\_w, [Point](#) &clo\_pt, [double](#) &clo\_↔\_dist, [double](#) epsilon, [double](#) \*seed=0) [const](#)

- void [reverseParameterDirection](#) (int paddir)
- void [swapParameterDirection](#) (int paddir1, int paddir2)
- virtual std::vector< shared\_ptr< [ParamSurface](#) > > [getAllBoundarySurfaces](#) () const  
*Fetch all boundary surfaces corresponding to the volume.*
- virtual void [translate](#) (const [Point](#) &vec)  
*Translate.*
- [SplineVolume](#) \* [geometryVolume](#) () const  
*Make a NURBS representation of the object.*

### Static Public Member Functions

- static [ClassType](#) [classType](#) ()

#### 29.321.1 Detailed Description

Class that represents a solid parallelepiped. It is a subclass of [ElementaryVolume](#), and thus has a parametrization.

A [Parallelepiped](#) has a natural parametrization in terms of the distances  $u$ ,  $v$  and  $w$  along the edges:  $\mathbf{p}(u, v, w) = \mathbf{C} + u \mathbf{x} + v \mathbf{y} + w \mathbf{z}$ . The parametrization is bounded by:  $0 \leq u \leq a$ ,  $0 \leq v \leq b$ ,  $0 \leq w \leq c$ , where  $a$ ,  $b$ ,  $c$  are the edge lengths. The dimension is 3.

Definition at line 64 of file [Parallelepiped.h](#).

#### 29.321.2 Constructor & Destructor Documentation

##### 29.321.2.1 [Go::Parallelepiped::Parallelepiped](#) ( ) [inline]

Default constructor. Constructs an uninitialized [Parallelepiped](#) which can only be assigned to or read into.

Definition at line 69 of file [Parallelepiped.h](#).

##### 29.321.2.2 [Go::Parallelepiped::Parallelepiped](#) ( [Point](#) *corner*, [Point](#) *dir\_u*, [Point](#) *dir\_v*, [Point](#) *dir\_w*, double *len\_u*, double *len\_v*, double *len\_w* )

Constructor.

##### 29.321.2.3 [virtual Go::Parallelepiped::~~Parallelepiped](#) ( ) [virtual]

virtual destructor - ensures safe inheritance

#### 29.321.3 Member Function Documentation

##### 29.321.3.1 [virtual BoundingBox](#) [Go::Parallelepiped::boundingBox](#) ( ) const [virtual]

Return the object's bounding box.

Implements [Go::GeomObject](#).

29.321.3.2 `static ClassType Go::Parallelepiped::classType ( ) [inline],[static]`

Definition at line 94 of file `Parallelepiped.h`.

29.321.3.3 `virtual Parallelepiped* Go::Parallelepiped::clone ( ) const [inline],[virtual]`

make a clone of this volume and return a pointer to it (user is responsible for clearing up memory afterwards).

Returns

pointer to cloned object

Implements [Go::ElementaryVolume](#).

Definition at line 101 of file `Parallelepiped.h`.

29.321.3.4 `void Go::Parallelepiped::closestPoint ( const Point & pt, double & clo_u, double & clo_v, double & clo_w, Point & clo_pt, double & clo_dist, double epsilon, double * seed = 0 ) const [virtual]`

Iterates to the closest point to `pt` on the volume.

Parameters

<code>pt</code>	the point to find the closest point to
<code>clo_u</code>	u parameter of the closest point
<code>clo_v</code>	v parameter of the closest point
<code>clo_w</code>	w parameter of the closest point
<code>clo_pt</code>	the geometric position of the closest point
<code>clo_dist</code>	the distance between <code>pt</code> and <code>clo_pt</code>
<code>epsilon</code>	parameter tolerance (will in any case not be higher than $\sqrt{\text{machine\_precision}}$ x magnitude of solution)
<code>seed</code>	pointer to parameter values where iteration starts.

Implements [Go::ParamVolume](#).

29.321.3.5 `virtual int Go::Parallelepiped::dimension ( ) const [virtual]`

Return the dimension of the space in which the object lies (usually 2 or 3)

Implements [Go::GeomObject](#).

29.321.3.6 `SplineVolume* Go::Parallelepiped::geometryVolume ( ) const [virtual]`

Make a NURBS representation of the object.

Implements [Go::ElementaryVolume](#).

29.321.3.7 `virtual std::vector<shared_ptr<ParamSurface> > Go::Parallelepiped::getAllBoundarySurfaces ( ) const` [virtual]

Fetch all boundary surfaces corresponding to the volume.

Implements [Go::ParamVolume](#).

29.321.3.8 `virtual ClassType Go::Parallelepiped::instanceType ( ) const` [virtual]

Return the class type identifier of a given, derived instance of [GeomObject](#).

Implements [Go::GeomObject](#).

29.321.3.9 `double Go::Parallelepiped::nextSegmentVal ( int dir, double par, bool forward, double tol ) const` [virtual]

Determine the parameter value of the start of the 'next segment' from a parameter value, along a given parameter direction. A 'segment' is here defined as a parameter interval in which there will be no discontinuities in derivatives or other artifacts. For spline objects, a segment will typically be the interval between two consecutive, non-coincident knots.

#### Parameters

<i>dir</i>	the parameter direction in which we search for the next segment (0, 1 or 2)
<i>par</i>	the parameter value starting from which we search for the start value of the next segment
<i>forward</i>	define whether we shall move forward ('true') or backwards when searching along this parameter
<i>tol</i>	tolerance used for determining whether the 'par' is already located <i>on</i> the next segment value

#### Returns

the value of the start value of the next segment (or the end of the previous segment, if we are moving backwards...)

Implements [Go::ParamVolume](#).

29.321.3.10 `const Array<double,6> Go::Parallelepiped::parameterSpan ( ) const` [virtual]

Return the parameter domain of the volume. This is an array containing the start and end parameter values. The spline's parameter *i* has its start value at the array position (2*i*) and its end parameter value at the array position (2*i*+1)

#### Returns

An array describing the parametric domain of the volume

Implements [Go::ParamVolume](#).

29.321.3.11 `void Go::Parallelepiped::point ( Point & pt, double upar, double vpar, double wpar ) const` [virtual]

Evaluates the volume's position for a given parameter triple.

## Parameters

<i>pt</i>	the result of the evaluation is written here
<i>upar</i>	the first parameter
<i>vpar</i>	the second parameter
<i>wpar</i>	the third parameter

Implements [Go::ParamVolume](#).

```
29.321.3.12 void Go::Parallelepiped::point (std::vector< Point > & pts, double upar, double vpar, double wpar, int
derivs, bool u_from_right = true, bool v_from_right = true, bool w_from_right = true, double resolution =
1.0e-12) const [virtual]
```

Evaluates the volume's position and a certain number of derivatives for a given parameter triple.

## Parameters

<i>pts</i>	the vector containing the evaluated values. Its size must be set by the user prior to calling this function. Upon completion of the function, its first entry is the volume's position at the given parameter pair. Then, if 'derivs' > 0, the two next entries will be the volume tangents along the first, second and third parameter direction. The next three entries are the second- and cross derivatives, in the order (du2, dudv, dudw, dv2, dvdw, dw2), and similar for even higher derivatives.
<i>upar</i>	the first parameter
<i>vpar</i>	the second parameter
<i>wpar</i>	the third parameter
<i>derivs</i>	number of requested derivatives
<i>u_from_right</i>	specify whether derivatives along the first parameter are to be calculated from the right ('true', default) or from the left ('false')
<i>v_from_right</i>	specify whether derivatives along the second parameter are to be calculated from the right ('true', default) or from the left ('false')
<i>w_from_right</i>	specify whether derivatives along the third parameter are to be calculated from the right ('true', default) or from the left ('false')
<i>resolution</i>	tolerance used when determining whether parameters are located at special values of the parameter domain (in particular; knot values in case of spline objects).

Implements [Go::ParamVolume](#).

```
29.321.3.13 virtual void Go::Parallelepiped::read (std::istream & is) [virtual]
```

read object from stream

## Parameters

<i>is</i>	stream from which object is read
-----------	----------------------------------

Implements [Go::Streamable](#).

29.321.3.14 `void Go::Parallelepiped::reverseParameterDirection ( int pardir )` [virtual]

Reverses the direction of the basis in input direction.

#### Parameters

<i>pardir</i>	which parameter direction to reverse
---------------	--------------------------------------

Implements [Go::ParamVolume](#).

29.321.3.15 `void Go::Parallelepiped::swapParameterDirection ( int pardir1, int pardir2 )` [virtual]

Swaps two parameter directions

#### Parameters

<i>pardir1</i>	One of the parameter directions (0, 1 or 2) to be swapped
<i>pardir2</i>	The other parameter direction (0, 1 or 2) to be swapped

Implements [Go::ParamVolume](#).

29.321.3.16 `DirectionCone Go::Parallelepiped::tangentCone ( int pardir ) const` [virtual]

Creates a [DirectionCone](#) covering all tangents to this volume along a given parameter direction.

#### Parameters

<i>pardir</i>	if 1, the <a href="#">DirectionCone</a> will be defined on basis of the volume's tangents along the first parameter direction. If 2, the second parameter direction will be used. If 3, the third parameter direction will be used.
---------------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

#### Returns

a [DirectionCone](#) (not necessarily the smallest) containing all tangents to this volume along the specified parameter direction.

Implements [Go::ParamVolume](#).

29.321.3.17 `virtual void Go::Parallelepiped::translate ( const Point & vec )` [virtual]

Translate.

Implements [Go::ParamVolume](#).

29.321.3.18 `virtual void Go::Parallelepiped::write ( std::ostream & os ) const` [virtual]

write object to stream



## Parameters

<i>os</i>	stream to which object is written
-----------	-----------------------------------

Implements [Go::Streamable](#).

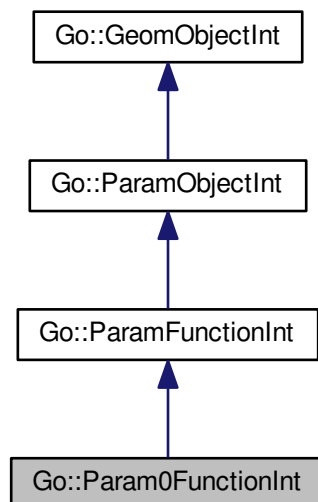
The documentation for this class was generated from the following file:

- [trivariate/include/GoTools/trivariate/Parallelepiped.h](#)

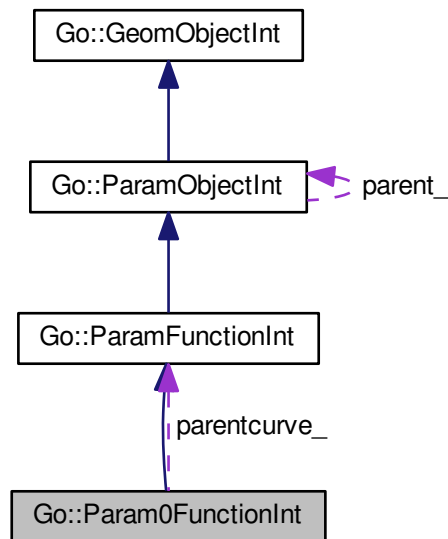
## 29.322 Go::Param0FunctionInt Class Reference

```
#include <Param0FunctionInt.h>
```

Inheritance diagram for Go::Param0FunctionInt:



Collaboration diagram for Go::Param0FunctionInt:



## Public Member Functions

- [Param0FunctionInt](#) (double C)
- [Param0FunctionInt](#) (double C, [ParamFunctionInt](#) \*parent)
- virtual [~Param0FunctionInt](#) ()
  - Destructor.*
- virtual void [point](#) ([Point](#) &res, const double \*par) const
- virtual void [point](#) (std::vector< [Point](#) > &pt, const double \*tpar, int derivs, const bool \*from\_right=0, double resolution=1.0e-12) const
- virtual [Param0FunctionInt](#) \* [getParam0FunctionInt](#) ()
  - Return pointer to this object.*
- virtual int [numParams](#) () const
  - The number of parameters in the object.*
- virtual void [getLengthAndWiggle](#) (double \*length, double \*wiggle)
- virtual bool [hasInnerKnots](#) (int paddir) const
- virtual std::vector< double > [getInnerKnotVals](#) (int paddir, bool sort=false) const
- virtual bool [hasCriticalVals](#) (int paddir) const
- virtual std::vector< double > [getCriticalVals](#) (int paddir) const
- virtual bool [hasCriticalValsOrKnots](#) (int paddir) const
- virtual std::vector< double > [getCriticalValsAndKnots](#) (int paddir) const
- virtual bool [canDivide](#) (int paddir)
- virtual int [getMeshSize](#) (int dir)
- virtual double [paramFromMesh](#) (int dir, int idx)
- virtual std::vector< double >::iterator [getMesh](#) ()
  - Return the geometric sample mesh for the parametric function.*
- virtual double [startParam](#) (int paddir) const
- virtual double [endParam](#) (int paddir) const

- virtual `bool boundaryPoint (const double *par, double eps) const`
- virtual `void subdivide (int pdir, double par, std::vector< shared_ptr< ParamFunctionInt > > &subdiv_objs, std::vector< shared_ptr< ParamFunctionInt > > &bd_objs)`
- virtual `CompositeBox compositeBox () const`  
*Return the [CompositeBox](#) for the parametric object.*
- virtual `bool monotone (Point &dir, double tol=1.0e-15) const`
- virtual `void getBoundaryObjects (std::vector< shared_ptr< BoundaryFunctionInt > > &bd_objs)`
- `int dimension ()`  
*Return the dimension of the geometric space.*
- virtual `int knotIntervalFuzzy (double &t, double tol) const`
- virtual `double nextSegmentVal (double par, bool forward) const`
- `double getValue () const`  
*Return the scalar value of this object.*

## Protected Attributes

- `double C_`
- `int dim_`
- `ParamFunctionInt * parentcurve_`

### 29.322.1 Detailed Description

This is a class that represents the "intersection object" of a scalar (a "0-variate" function).

Definition at line 53 of file Param0FunctionInt.h.

### 29.322.2 Constructor & Destructor Documentation

#### 29.322.2.1 Go::Param0FunctionInt::Param0FunctionInt ( double C ) [explicit]

Constructor.

Parameters

<code>C</code>	the constant defining the object.
----------------	-----------------------------------

#### 29.322.2.2 Go::Param0FunctionInt::Param0FunctionInt ( double C, ParamFunctionInt \* parent ) [explicit]

Constructor.

Parameters

<code>C</code>	the constant defining the object.
<code>parent</code>	the parent object to this object.

29.322.2.3 virtual Go::Param0FunctionInt::~~Param0FunctionInt ( ) [virtual]

Destructor.

### 29.322.3 Member Function Documentation

29.322.3.1 virtual bool Go::Param0FunctionInt::boundaryPoint ( const double \* *par*, double *eps* ) const [virtual]

Return true if the specified point lies within eps from the boundary.

#### Parameters

<i>par</i>	the parameter in which to evaluate. Size of array should be equal to <a href="#">numParams()</a> .
<i>eps</i>	the tolerance defining boundary neighbourhood.

Implements [Go::ParamObjectInt](#).

29.322.3.2 virtual bool Go::Param0FunctionInt::canDivide ( int *pardir* ) [virtual]

Return true if we are allowed to divide in the specified parameter direction.

#### Parameters

<i>pardir</i>	the parameter direction in question.
---------------	--------------------------------------

Implements [Go::ParamObjectInt](#).

29.322.3.3 virtual CompositeBox Go::Param0FunctionInt::compositeBox ( ) const [virtual]

Return the [CompositeBox](#) for the parametric object.

Implements [Go::ParamObjectInt](#).

29.322.3.4 int Go::Param0FunctionInt::dimension ( )

Return the dimension of the geometric space.

29.322.3.5 virtual double Go::Param0FunctionInt::endParam ( int *pardir* ) const [virtual]

Return the end parameter in the specified direction.

#### Parameters

<i>pardir</i>	the parameter direction in question. Indexing starts at 0.
---------------	------------------------------------------------------------

Implements [Go::ParamObjectInt](#).

29.322.3.6 `virtual void Go::Param0FunctionInt::getBoundaryObjects ( std::vector< shared_ptr< BoundaryFunctionInt > & bd_objs ) [virtual]`

Return the boundary objects of this object.

Parameters

<i>bd_objs</i>	the boundary objects of this object.
----------------	--------------------------------------

Implements [Go::ParamFunctionInt](#).

29.322.3.7 `virtual std::vector<double> Go::Param0FunctionInt::getCriticalVals ( int padir ) const [virtual]`

Return the critical parameter values in the specified direction.

Parameters

<i>padir</i>	the parameter direction in question. Indexing starts at 0.
--------------	------------------------------------------------------------

Implements [Go::ParamObjectInt](#).

29.322.3.8 `virtual std::vector<double> Go::Param0FunctionInt::getCriticalValsAndKnots ( int padir ) const [virtual]`

Return the critical parameter values and inner knots for object.

Parameters

<i>padir</i>	the parameter direction in question. Indexing starts at 0.
--------------	------------------------------------------------------------

Implements [Go::ParamObjectInt](#).

29.322.3.9 `virtual std::vector<double> Go::Param0FunctionInt::getInnerKnotVals ( int padir, bool sort = false ) const [virtual]`

Return the inner knot values in the specified direction.

Parameters

<i>padir</i>	the parameter direction in question. Indexing starts at 0.
<i>sort</i>	the returned values may be sorted by the function.

Implements [Go::ParamObjectInt](#).

29.322.3.10 `virtual void Go::Param0FunctionInt::getLengthAndWiggle ( double * length, double * wiggle )` [virtual]

Return an estimate on the size and wiggle of the object.

#### Parameters

<i>length</i>	the approximative length of the object in the corresponding parameter directions. The size of the array should be equal to <code>numParams()</code> .
<i>wiggle</i>	a scalar representing the wiggle of the object in the corresponding parameter directions. The size of the array should be equal to <code>numParams()</code> .

Implements [Go::ParamObjectInt](#).

29.322.3.11 `virtual std::vector<double>::iterator Go::Param0FunctionInt::getMesh ( )` [virtual]

Return the geometric sample mesh for the parametric function.

Implements [Go::ParamFunctionInt](#).

29.322.3.12 `virtual int Go::Param0FunctionInt::getMeshSize ( int dir )` [virtual]

Return the size of the geometric sample mesh in the specified direction.

#### Parameters

<i>dir</i>	the parameter direction in question. Indexing starts at 0.
------------	------------------------------------------------------------

Implements [Go::ParamFunctionInt](#).

29.322.3.13 `virtual Param0FunctionInt* Go::Param0FunctionInt::getParam0FunctionInt ( )` [virtual]

Return pointer to this object.

Reimplemented from [Go::ParamFunctionInt](#).

29.322.3.14 `double Go::Param0FunctionInt::getValue ( ) const` [inline]

Return the scalar value of this object.

Definition at line 238 of file `Param0FunctionInt.h`.

29.322.3.15 `virtual bool Go::Param0FunctionInt::hasCriticalVals ( int pardir ) const` [virtual]

Return true if the object has any critical parameter values in the specified parameter direction.

## Parameters

<i>pardir</i>	the parameter direction in question. Indexing starts at 0.
---------------	------------------------------------------------------------

Implements [Go::ParamObjectInt](#).

**29.322.3.16** `virtual bool Go::Param0FunctionInt::hasCriticalValsOrKnots ( int pardir ) const` [virtual]

Return true if the object has any critical parameter values or inner knots in the specified parameter direction.

## Parameters

<i>pardir</i>	the parameter direction in question. Indexing starts at 0.
---------------	------------------------------------------------------------

Implements [Go::ParamObjectInt](#).

**29.322.3.17** `virtual bool Go::Param0FunctionInt::hasInnerKnots ( int pardir ) const` [virtual]

Return true if the object has any inner knots in the specified parameter direction.

## Parameters

<i>pardir</i>	the parameter direction in question. Indexing starts at 0.
---------------	------------------------------------------------------------

Implements [Go::ParamObjectInt](#).

**29.322.3.18** `virtual int Go::Param0FunctionInt::knotIntervalFuzzy ( double & t, double tol ) const` [virtual]

Return the knot interval for which *t* lies inside, possibly moving the value *t* if it lies close to a knot.

## Parameters

<i>t</i>	the parameter value
<i>tol</i>	the parametric tolerance deciding if the input parameter should be moved.

**29.322.3.19** `virtual bool Go::Param0FunctionInt::monotone ( Point & dir, double tol = 1.0e-15 ) const` [virtual]

Return true if the object is monotone in any direction.

## Parameters

<i>dir</i>	the direction in which the object is monotone. Relevant only if object is monotone.
------------	-------------------------------------------------------------------------------------

Implements [Go::ParamFunctionInt](#).

29.322.3.20 `virtual double Go::Param0FunctionInt::nextSegmentVal ( double par, bool forward ) const` [virtual]

Return the value of the knot closest to the input parameter.

Parameters

<i>par</i>	the parameter value
<i>forward</i>	if true we return the closest knot to the right, otherwise the closest knot to the left.

29.322.3.21 `virtual int Go::Param0FunctionInt::numParams ( ) const` [virtual]

The number of parameters in the object.

Implements [Go::ParamObjectInt](#).

29.322.3.22 `virtual double Go::Param0FunctionInt::paramFromMesh ( int dir, int idx )` [virtual]

Return the corresponding mesh parameter.

Parameters

<i>dir</i>	the parameter direction in question. Indexing starts at 0.
<i>idx</i>	the mesh idx in the specified direction. Indexing starts at 0.

Implements [Go::ParamFunctionInt](#).

29.322.3.23 `virtual void Go::Param0FunctionInt::point ( Point & res, const double * par ) const` [virtual]

Evaluate the object in the input parameter.

Parameters

<i>res</i>	the <a href="#">Point</a> to be returned.
<i>par</i>	the parameter in which to evaluate. The size of the array should be equal to <a href="#">numParams()</a> .

Implements [Go::ParamObjectInt](#).

29.322.3.24 `virtual void Go::Param0FunctionInt::point ( std::vector< Point > & pt, const double * tpar, int derivs, const bool * from_right = 0, double resolution = 1.0e-12 ) const` [inline],[virtual]

Evaluate the object in the input parameter, with the specified number of derivatives.

Parameters

<i>pt</i>	the <a href="#">Point</a> to be returned.
-----------	-------------------------------------------



## Parameters

<i>tpar</i>	the parameter in which to evaluate. The size of the array should be equal to <code>numParams()</code> .
<i>derivs</i>	the number of derivatives to calculate.
<i>from_right</i>	if true the evaluation is to be performed from the right side of the parameter value.
<i>resolution</i>	tolerance used when determining whether parameters are located at special values of the parameter domain (in particular: knot values in case of spline objects).

Implements [Go::ParamObjectInt](#).

Definition at line 85 of file `Param0FunctionInt.h`.

**29.322.3.25** `virtual double Go::Param0FunctionInt::startParam ( int parDir ) const` `[virtual]`

Return the start parameter value in the specified direction.

## Parameters

<i>parDir</i>	the parameter direction in question. Indexing starts at 0.
---------------	------------------------------------------------------------

Implements [Go::ParamObjectInt](#).

**29.322.3.26** `virtual void Go::Param0FunctionInt::subdivide ( int parDir, double par, std::vector< shared_ptr< ParamFunctionInt > > & subdiv_objs, std::vector< shared_ptr< ParamFunctionInt > > & bd_objs )` `[virtual]`

Subdivide the object in the specified parameter direction and parameter value.

## Parameters

<i>parDir</i>	direction in which to subdivide. Indexing starts at 0.
<i>par</i>	parameter in which to subdivide.
<i>subdiv_objs</i>	the subparts of this object. Of the same geometric dimension as this object.
<i>bd_objs</i>	the boundaries between the returned <i>subdiv_objs</i> . Of geometric dimension 1 less than this object.

Implements [Go::ParamFunctionInt](#).

## 29.322.4 Member Data Documentation

**29.322.4.1** `double Go::Param0FunctionInt::C_` `[protected]`

Definition at line 243 of file `Param0FunctionInt.h`.

**29.322.4.2** `int Go::Param0FunctionInt::dim_` `[protected]`

Definition at line 245 of file `Param0FunctionInt.h`.

29.322.4.3 `ParamFunctionInt* Go::Param0FunctionInt::parentcurve_` [protected]

Definition at line 247 of file Param0FunctionInt.h.

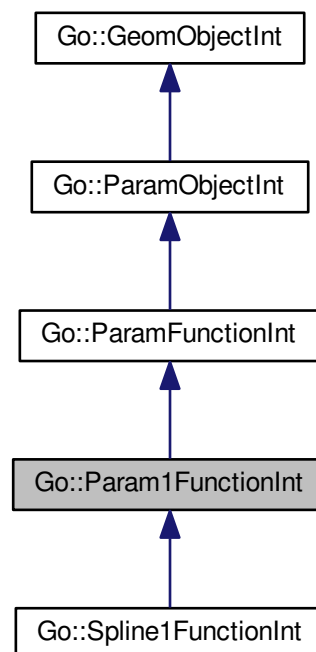
The documentation for this class was generated from the following file:

- [intersections/include/GoTools/intersections/Param0FunctionInt.h](#)

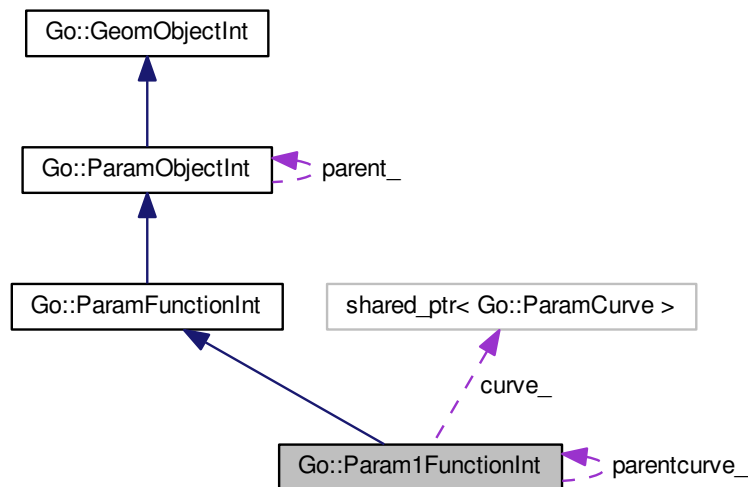
### 29.323 Go::Param1FunctionInt Class Reference

```
#include <Param1FunctionInt.h>
```

Inheritance diagram for Go::Param1FunctionInt:



Collaboration diagram for Go::Param1FunctionInt:



## Public Member Functions

- [Param1FunctionInt](#) (shared\_ptr< [ParamCurve](#) > curve)
- [Param1FunctionInt](#) (shared\_ptr< [ParamCurve](#) > curve, [ParamFunctionInt](#) \*parent)
- virtual [~Param1FunctionInt](#) ()
  - Destructor.*
- virtual void [point](#) ([Point](#) &res, const double \*par) const
- virtual void [point](#) (std::vector< [Point](#) > &res, double par, int der)
- virtual void [point](#) (std::vector< [Point](#) > &pt, const double \*tpar, int derivs, const bool \*from\_right=0, double resolution=1.0e-12) const
- virtual [Param1FunctionInt](#) \* [getParam1FunctionInt](#) ()
  - Return pointer to this object.*
- shared\_ptr< [ParamCurve](#) > [getParamCurve](#) ()
  - Return pointer to the parametric curve.*
- shared\_ptr< const [ParamCurve](#) > [getParamCurve](#) () const
  - Return pointer to the parametric curve.*
- shared\_ptr< [ParamCurve](#) > [getParentParamCurve](#) (double &start, double &end)
- shared\_ptr< [ParamCurve](#) > [getParentParamCurve](#) ()
- virtual shared\_ptr< [Param1FunctionInt](#) > [makeIntFunction](#) (shared\_ptr< [ParamCurve](#) > curve)
- virtual int [numParams](#) () const
  - The number of parameters in the object.*
- virtual void [getLengthAndWiggle](#) (double \*length, double \*wiggle)
- virtual bool [hasInnerKnots](#) (int paddir) const
- virtual bool [hasCriticalVals](#) (int paddir) const
- virtual bool [hasCriticalValsOrKnots](#) (int paddir) const
- virtual bool [canDivide](#) (int paddir)
- virtual std::vector< double > [getCriticalVals](#) (int paddir) const
- virtual std::vector< double > [getInnerKnotVals](#) (int paddir, bool sort=false) const
- virtual std::vector< double > [getCriticalValsAndKnots](#) (int paddir) const

- virtual int [getMeshSize](#) (int dir)
- virtual [double paramFromMesh](#) (int dir, int idx)
- virtual std::vector< [double](#) >::iterator [getMesh](#) ()  
*Return the geometric sample mesh for the parametric function.*
- virtual [double startParam](#) (int paddir) [const](#)
- virtual [double endParam](#) (int paddir) [const](#)
- virtual [bool boundaryPoint](#) (const [double](#) \*par, [double](#) eps) [const](#)
- virtual void [subdivide](#) (int paddir, [double](#) par, std::vector< shared\_ptr< [ParamFunctionInt](#) > > &subdiv\_objs, std::vector< shared\_ptr< [ParamFunctionInt](#) > > &bd\_objs)
- virtual [bool monotone](#) (Point &dir, [double](#) tol=1.0e-15) [const](#)
- virtual [CompositeBox compositeBox](#) () [const](#)  
*Return the [CompositeBox](#) for the parametric object.*
- virtual void [getBoundaryObjects](#) (std::vector< shared\_ptr< [BoundaryFunctionInt](#) > > &bd\_objs)
- int [dimension](#) ()  
*Return the dimension of the geometric space.*
- [double startparam](#) () [const](#)  
*Return the start parameter of the curve.*
- [double endparam](#) () [const](#)  
*Return the end parameter of the curve.*
- void [assureInRange](#) ([double](#) &t)
- virtual int [knotIntervalFuzzy](#) ([double](#) &t, [double](#) tol) [const](#)
- virtual [double nextSegmentVal](#) ([double](#) par, [bool](#) forward) [const](#)
- virtual [bool isDegenerate](#) ([double](#) epsge, int dir, [double](#) \*par)

## Protected Attributes

- shared\_ptr< [ParamCurve](#) > [curve\\_](#)
- int [dim\\_](#)
- [Param1FunctionInt](#) \* [parentcurve\\_](#)
- std::vector< std::pair< [double](#), int > > [segment\\_](#)
- std::vector< [double](#) > [mesh\\_](#)
- [double](#) [deg\\_tol\\_](#)

### 29.323.1 Detailed Description

Class that represents an "intersection object" of a parametric curve of dimension 1.

Definition at line 54 of file [Param1FunctionInt.h](#).

### 29.323.2 Constructor & Destructor Documentation

**29.323.2.1** [Go::Param1FunctionInt::Param1FunctionInt](#) ( shared\_ptr< [ParamCurve](#) > [curve](#) ) [\[explicit\]](#)

Constructor.

Parameters

<i>curve</i>	the parametric 1-dimensional curve defining the intersection object.
--------------	----------------------------------------------------------------------

29.323.2.2 `Go::Param1FunctionInt::Param1FunctionInt ( shared_ptr< ParamCurve > curve, ParamFunctionInt * parent ) [explicit]`

Constructor.

Parameters

<i>curve</i>	the parametric 1-dimensional curve defining the intersection object.
<i>parent</i>	the parent object to this object. Can be either a curve or a surface.

29.323.2.3 `virtual Go::Param1FunctionInt::~~Param1FunctionInt ( ) [virtual]`

Destructor.

### 29.323.3 Member Function Documentation

29.323.3.1 `void Go::Param1FunctionInt::assureInRange ( double & t )`

Make sure that the input parameter lies inside the range of the parametric curve. Set *t* equal to *tmin* if it lies below *tmin*, or *tmax* if it lies above *tmax*.

Parameters

<i>t</i>	the input parameter
----------	---------------------

29.323.3.2 `virtual bool Go::Param1FunctionInt::boundaryPoint ( const double * par, double eps ) const [virtual]`

Return true if the specified point lies within *eps* from the boundary.

Parameters

<i>par</i>	the parameter in which to evaluate. Size of array should be equal to <a href="#">numParams()</a> .
<i>eps</i>	the tolerance defining boundary neighbourhood.

Implements [Go::ParamObjectInt](#).

29.323.3.3 `virtual bool Go::Param1FunctionInt::canDivide ( int pardir ) [virtual]`

Return true if we are allowed to divide in the specified parameter direction.

Parameters

<i>pardir</i>	the parameter direction in question.
---------------	--------------------------------------

Implements [Go::ParamObjectInt](#).

29.323.3.4 `virtual CompositeBox Go::Param1FunctionInt::compositeBox ( ) const [virtual]`

Return the [CompositeBox](#) for the parametric object.

Implements [Go::ParamObjectInt](#).

29.323.3.5 `int Go::Param1FunctionInt::dimension ( ) [inline]`

Return the dimension of the geometric space.

Definition at line 256 of file Param1FunctionInt.h.

29.323.3.6 `virtual double Go::Param1FunctionInt::endParam ( int pardir ) const [inline],[virtual]`

Return the end parameter in the specified direction.

#### Parameters

<i>pardir</i>	the parameter direction in question. Indexing starts at 0.
---------------	------------------------------------------------------------

Implements [Go::ParamObjectInt](#).

Definition at line 215 of file Param1FunctionInt.h.

29.323.3.7 `double Go::Param1FunctionInt::endparam ( ) const [inline]`

Return the end parameter of the curve.

Definition at line 263 of file Param1FunctionInt.h.

29.323.3.8 `virtual void Go::Param1FunctionInt::getBoundaryObjects ( std::vector< shared_ptr< BoundaryFunctionInt > & bd_objs ) [virtual]`

Return the boundary objects of this object.

#### Parameters

<i>bd_objs</i>	the boundary objects of this object.
----------------	--------------------------------------

Implements [Go::ParamFunctionInt](#).

29.323.3.9 `virtual std::vector<double> Go::Param1FunctionInt::getCriticalVals ( int pardir ) const [virtual]`

Return the critical parameter values in the specified direction.

## Parameters

<i>parDir</i>	the parameter direction in question. Indexing starts at 0.
---------------	------------------------------------------------------------

Implements [Go::ParamObjectInt](#).

29.323.3.10 `virtual std::vector<double> Go::Param1FunctionInt::getCriticalValsAndKnots ( int parDir ) const`  
[virtual]

Return the critical parameter values and inner knots for object.

## Parameters

<i>parDir</i>	the parameter direction in question. Indexing starts at 0.
---------------	------------------------------------------------------------

Implements [Go::ParamObjectInt](#).

Reimplemented in [Go::Spline1FunctionInt](#).

29.323.3.11 `virtual std::vector<double> Go::Param1FunctionInt::getInnerKnotVals ( int parDir, bool sort = false ) const`  
[virtual]

Return the inner knot values in the specified direction.

## Parameters

<i>parDir</i>	the parameter direction in question. Indexing starts at 0.
<i>sort</i>	the returned values may be sorted by the function.

Implements [Go::ParamObjectInt](#).

Reimplemented in [Go::Spline1FunctionInt](#).

29.323.3.12 `virtual void Go::Param1FunctionInt::getLengthAndWiggle ( double * length, double * wiggle )` [virtual]

Return an estimate on the size and wiggle of the object.

## Parameters

<i>length</i>	the approximative length of the object in the corresponding parameter directions. The size of the array should be equal to <a href="#">numParams()</a> .
<i>wiggle</i>	a scalar representing the wiggle of the object in the corresponding parameter directions. The size of the array should be equal to <a href="#">numParams()</a> .

Implements [Go::ParamObjectInt](#).

29.323.3.13 `virtual std::vector<double>::iterator Go::Param1FunctionInt::getMesh ( ) [virtual]`

Return the geometric sample mesh for the parametric function.

Implements [Go::ParamFunctionInt](#).

Reimplemented in [Go::Spline1FunctionInt](#).

29.323.3.14 `virtual int Go::Param1FunctionInt::getMeshSize ( int dir ) [virtual]`

Return the size of the geometric sample mesh in the specified direction.

Parameters

<i>dir</i>	the parameter direction in question. Indexing starts at 0.
------------	------------------------------------------------------------

Implements [Go::ParamFunctionInt](#).

Reimplemented in [Go::Spline1FunctionInt](#).

29.323.3.15 `virtual Param1FunctionInt* Go::Param1FunctionInt::getParam1FunctionInt ( ) [virtual]`

Return pointer to this object.

Reimplemented from [Go::ParamFunctionInt](#).

29.323.3.16 `shared_ptr<ParamCurve> Go::Param1FunctionInt::getParamCurve ( )`

Return pointer to the parametric curve.

29.323.3.17 `shared_ptr<const ParamCurve> Go::Param1FunctionInt::getParamCurve ( ) const`

Return pointer to the parametric curve.

29.323.3.18 `shared_ptr<ParamCurve> Go::Param1FunctionInt::getParentParamCurve ( double & start, double & end )`

Return pointer to a subpart of the parent curve of this object. If a parent curve does not exist, return pointer to curve in this object. To reduce numerical noise we go straight to the source (undivided) curve.

29.323.3.19 `shared_ptr<ParamCurve> Go::Param1FunctionInt::getParentParamCurve ( )`

Return pointer to the parent curve of this object. If a parent curve does not exist, return pointer to curve in this object. To reduce numerical noise we go straight to the source (undivided) curve.

29.323.3.20 `virtual bool Go::Param1FunctionInt::hasCriticalVals ( int pardir ) const [virtual]`

Return true if the object has any critical parameter values in the specified parameter direction.



## Parameters

<i>pardir</i>	the parameter direction in question. Indexing starts at 0.
---------------	------------------------------------------------------------

Implements [Go::ParamObjectInt](#).

**29.323.3.21** `virtual bool Go::Param1FunctionInt::hasCriticalValsOrKnots ( int pardir ) const` [virtual]

Return true if the object has any critical parameter values or inner knots in the specified parameter direction.

## Parameters

<i>pardir</i>	the parameter direction in question. Indexing starts at 0.
---------------	------------------------------------------------------------

Implements [Go::ParamObjectInt](#).

Reimplemented in [Go::Spline1FunctionInt](#).

**29.323.3.22** `virtual bool Go::Param1FunctionInt::hasInnerKnots ( int pardir ) const` [virtual]

Return true if the object has any inner knots in the specified parameter direction.

## Parameters

<i>pardir</i>	the parameter direction in question. Indexing starts at 0.
---------------	------------------------------------------------------------

Implements [Go::ParamObjectInt](#).

Reimplemented in [Go::Spline1FunctionInt](#).

**29.323.3.23** `virtual bool Go::Param1FunctionInt::isDegenerate ( double epsge, int dir, double * par )` [virtual]

Return true if the object is degenerate in the specified direction and parameter.

## Parameters

<i>epsge</i>	the geometric tolerance defining degeneracy.
<i>dir</i>	the parameter direction in question.
<i>par</i>	the parameter in which to evaluate. Size of array should be equal to <a href="#">numParams()</a> .

Reimplemented from [Go::ParamObjectInt](#).

**29.323.3.24** `virtual int Go::Param1FunctionInt::knotIntervalFuzzy ( double & t, double tol ) const` [virtual]

Return the knot interval for which *t* lies inside, moving the value *t* if it lies close to a knot.

## Parameters

<i>t</i>	the parameter value
<i>tol</i>	the parametric tolerance deciding if the input parameter <i>t</i> should be moved.

Reimplemented in [Go::Spline1FunctionInt](#).

```
29.323.3.25 virtual shared_ptr<Param1FunctionInt> Go::Param1FunctionInt::makeIntFunction (shared_ptr<
ParamCurve > curve) [virtual]
```

Return an intersection object for the input curve. This object is used as the parent for the intersection object.

## Parameters

<i>curve</i>	the parametric curve defining the intersection object.
--------------	--------------------------------------------------------

Reimplemented in [Go::Spline1FunctionInt](#).

```
29.323.3.26 virtual bool Go::Param1FunctionInt::monotone (Point & dir, double tol=1.0e-15) const [virtual]
```

Return true if the curve is monotone.

## Parameters

<i>dir</i>	the direction in which the object is monotone. Is not of interest here as the curve has only 1 parameter direction.
------------	---------------------------------------------------------------------------------------------------------------------

## Returns

Whether or not the curve is monotone.

Implements [Go::ParamFunctionInt](#).

Reimplemented in [Go::Spline1FunctionInt](#).

```
29.323.3.27 virtual double Go::Param1FunctionInt::nextSegmentVal (double par, bool forward) const [virtual]
```

Return the value of the knot next to the input parameter *par*.

## Parameters

<i>par</i>	the parameter value
<i>forward</i>	if true we return the closest knot to the right, otherwise the closest knot to the left.

**Returns**

The knot closest to the input parameter.

Reimplemented in [Go::Spline1FunctionInt](#).

**29.323.3.28** `virtual int Go::Param1FunctionInt::numParams ( ) const [virtual]`

The number of parameters in the object.

Implements [Go::ParamObjectInt](#).

**29.323.3.29** `virtual double Go::Param1FunctionInt::paramFromMesh ( int dir, int idx ) [virtual]`

Return the corresponding mesh parameter.

**Parameters**

<i>dir</i>	the parameter direction in question. Indexing starts at 0.
<i>idx</i>	the mesh idx in the specified direction. Indexing starts at 0.

Implements [Go::ParamFunctionInt](#).

Reimplemented in [Go::Spline1FunctionInt](#).

**29.323.3.30** `virtual void Go::Param1FunctionInt::point ( Point & res, const double * par ) const [inline], [virtual]`

Evaluate the object in the input parameter.

**Parameters**

<i>res</i>	the <a href="#">Point</a> to be returned.
<i>par</i>	the parameter in which to evaluate. The size of the array should be equal to <a href="#">numParams()</a> .

Implements [Go::ParamObjectInt](#).

Definition at line 76 of file Param1FunctionInt.h.

**29.323.3.31** `virtual void Go::Param1FunctionInt::point ( std::vector< Point > & res, double par, int der ) [inline], [virtual]`

Evaluate the object in the input parameter, with the specified number of derivatives.

**Parameters**

<i>res</i>	the <a href="#">Point</a> to be returned.
<i>par</i>	the parameter in which to evaluate. The size of the array should be equal to <a href="#">numParams()</a> .
<i>der</i>	the number of derivatives to calculate.

Definition at line 85 of file Param1FunctionInt.h.

```
29.323.3.32 virtual void Go::Param1FunctionInt::point (std::vector< Point > & pt, const double * tpar, int derivs, const
bool * from_right = 0, double resolution = 1.0e-12) const [inline],[virtual]
```

Evaluate the object in the input parameter, with the specified number of derivatives.

#### Parameters

<i>pt</i>	the <a href="#">Point</a> to be returned.
<i>tpar</i>	the parameter in which to evaluate. The size of the array should be equal to <a href="#">numParams()</a> .
<i>derivs</i>	the number of derivatives to calculate.
<i>from_right</i>	if true the evaluation is to be performed from the right side of the parameter value.
<i>resolution</i>	tolerance used when determining whether parameters are located at special values of the parameter domain (in particular: knot values in case of spline objects).

Implements [Go::ParamObjectInt](#).

Definition at line 99 of file Param1FunctionInt.h.

```
29.323.3.33 virtual double Go::Param1FunctionInt::startParam (int pardir) const [inline],[virtual]
```

Return the start parameter value in the specified direction.

#### Parameters

<i>pardir</i>	the parameter direction in question. Indexing starts at 0.
---------------	------------------------------------------------------------

Implements [Go::ParamObjectInt](#).

Definition at line 209 of file Param1FunctionInt.h.

```
29.323.3.34 double Go::Param1FunctionInt::startparam () const [inline]
```

Return the start parameter of the curve.

Definition at line 260 of file Param1FunctionInt.h.

```
29.323.3.35 virtual void Go::Param1FunctionInt::subdivide (int pardir, double par, std::vector< shared_ptr<
ParamFunctionInt > > & subdiv_objs, std::vector< shared_ptr< ParamFunctionInt > > & bd_objs)
[virtual]
```

Subdivide the object in the specified parameter direction and parameter value.

#### Parameters

<i>pardir</i>	direction in which to subdivide. Indexing starts at 0.
<i>par</i>	parameter in which to subdivide.
<i>subdiv_objs</i>	The subparts of this object. Of the same geometric dimension as this object. <span style="float: right;">Generated by Doxygen</span>
<i>bd_objs</i>	the boundaries between the returned <i>subdiv_objs</i> . Of geometric dimension 1 less than this object.

Implements [Go::ParamFunctionInt](#).

#### 29.323.4 Member Data Documentation

**29.323.4.1** `shared_ptr<ParamCurve> Go::Param1FunctionInt::curve_` `[protected]`

Definition at line 304 of file Param1FunctionInt.h.

**29.323.4.2** `double Go::Param1FunctionInt::deg_tol_` `[protected]`

Definition at line 315 of file Param1FunctionInt.h.

**29.323.4.3** `int Go::Param1FunctionInt::dim_` `[protected]`

Definition at line 306 of file Param1FunctionInt.h.

**29.323.4.4** `std::vector<double> Go::Param1FunctionInt::mesh_` `[mutable],[protected]`

Definition at line 313 of file Param1FunctionInt.h.

**29.323.4.5** `Param1FunctionInt* Go::Param1FunctionInt::parentcurve_` `[protected]`

Definition at line 308 of file Param1FunctionInt.h.

**29.323.4.6** `std::vector<std::pair<double, int> > Go::Param1FunctionInt::segment_` `[protected]`

Definition at line 310 of file Param1FunctionInt.h.

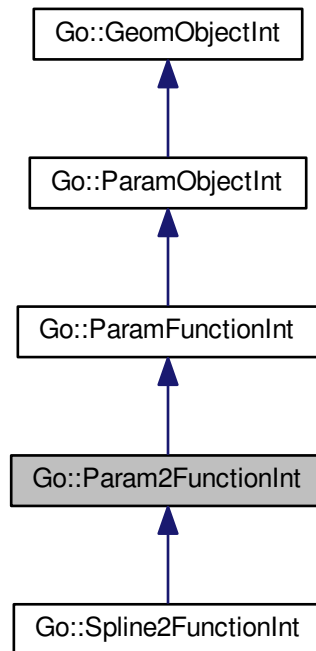
The documentation for this class was generated from the following file:

- [intersections/include/GoTools/intersections/Param1FunctionInt.h](#)

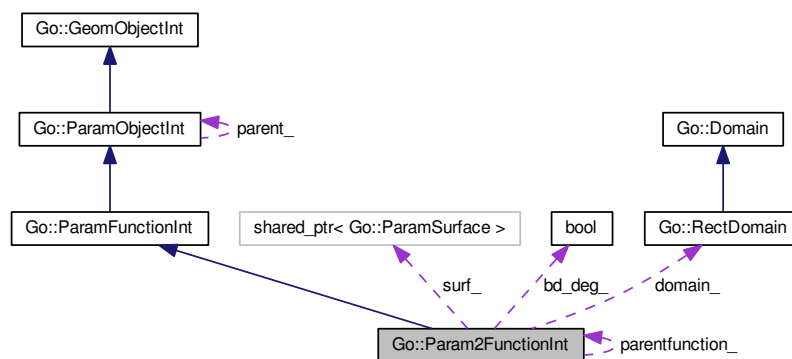
## 29.324 Go::Param2FunctionInt Class Reference

```
#include <Param2FunctionInt.h>
```

Inheritance diagram for Go::Param2FunctionInt:



Collaboration diagram for Go::Param2FunctionInt:



## Public Member Functions

- [Param2FunctionInt](#) (shared\_ptr< [ParamSurface](#) > surf)
- [Param2FunctionInt](#) (shared\_ptr< [ParamSurface](#) > surf, [Param2FunctionInt](#) \*parent)
- virtual [~Param2FunctionInt](#) ()
  - Destructor.*
- virtual void [point](#) ([Point](#) &res, const double \*par) const
- virtual void [point](#) (std::vector< [Point](#) > &pt, const double \*tpar, int der, const bool \*from\_right=0, double resolution=1.0e-12) const
- virtual [Param2FunctionInt](#) \* [getParam2FunctionInt](#) ()
  - Return pointer to this object.*
- shared\_ptr< [ParamSurface](#) > [getParamSurface](#) ()
  - Return pointer to the parametric surface.*
- shared\_ptr< const [ParamSurface](#) > [getParamSurface](#) () const
  - Return pointer to the parametric surface.*
- shared\_ptr< [ParamSurface](#) > [getParentParamSurface](#) ([RectDomain](#) &domain)
- shared\_ptr< [ParamSurface](#) > [getParentParamSurface](#) ()
- virtual shared\_ptr< [Param2FunctionInt](#) > [makeIntFunction](#) (shared\_ptr< [ParamSurface](#) > surf)
- virtual int [numParams](#) () const
  - The number of parameters in the object.*
- shared\_ptr< [ParamCurve](#) > [getIsoCurve](#) (double param\_start, double param\_end, double isoval, bool pardir\_is\_u) const
- shared\_ptr< [ParamCurve](#) > [getConstantParameterCurve](#) (int dir, double par)
- virtual void [getLengthAndWiggle](#) (double \*length, double \*wiggle)
- virtual bool [hasInnerKnots](#) (int pardir) const
- virtual bool [hasCriticalVals](#) (int pardir) const
- virtual bool [hasCriticalValsOrKnots](#) (int pardir) const
- virtual bool [canDivide](#) (int pardir)
- virtual std::vector< double > [getCriticalVals](#) (int pardir) const
- virtual std::vector< double > [getInnerKnotVals](#) (int pardir, bool sort=false) const
- virtual std::vector< double > [getCriticalValsAndKnots](#) (int pardir) const
- virtual int [getMeshSize](#) (int dir)
- virtual double [paramFromMesh](#) (int dir, int idx)
- virtual std::vector< double >::iterator [getMesh](#) ()
  - Return the geometric sample mesh for the parametric function.*
- virtual double [startParam](#) (int pardir) const
- virtual double [endParam](#) (int pardir) const
- virtual bool [boundaryPoint](#) (const double \*par, double eps) const
- std::vector< double > [getMima](#) () const
- virtual void [subdivide](#) (int pardir, double par, std::vector< shared\_ptr< [ParamFunctionInt](#) > > &subdiv\_objs, std::vector< shared\_ptr< [ParamFunctionInt](#) > > &bd\_objs)
- virtual bool [monotone](#) ([Point](#) &dir, double tol=1.0e-15) const
- virtual [CompositeBox](#) [compositeBox](#) () const
- virtual void [getBoundaryObjects](#) (std::vector< shared\_ptr< [BoundaryFunctionInt](#) > > &bd\_objs)
- int [dimension](#) ()
  - Return the dimension of the geometric space.*
- double [isolateDegPar](#) (int dir, int deg\_edge, double threshold)
- void [assureInRange](#) (int pardir, double &t)
- virtual int [knotIntervalFuzzy](#) (int pardir, double &t, double tol) const
- virtual double [nextSegmentVal](#) (int pardir, double par, bool forward) const
- bool [isDegenerate](#) (double epsge, int pardir)
- virtual bool [isDegenerate](#) (double epsge, int pardir, double \*par)
- shared\_ptr< [ParamSurface](#) > [getSurface](#) ()
- shared\_ptr< const [ParamSurface](#) > [getSurface](#) () const
- void [derivs](#) (double u, double v, [Point](#) &deriv\_u, [Point](#) &deriv\_v) const

## Protected Member Functions

- `Go::SplineCurve * extractBdCurve (const Go::SplineSurface &, int bddix, int &pardir, double &tpar) const`

## Protected Attributes

- `shared_ptr< ParamSurface > surf_`
- `int dim_`
- `Param2FunctionInt * parentfunction_`
- `std::vector< std::pair< double, int > > segment_ [2]`
- `bool bd_deg_ [4]`
- `double deg_tol_`
- `RectDomain domain_`
- `int nmesh_ [2]`
- `std::vector< double > mesh_`
- `std::vector< Point > temp_point_array_`

### 29.324.1 Detailed Description

Class that represents the "intersection object" of a parametric surface of dimension 1.

Definition at line 57 of file Param2FunctionInt.h.

### 29.324.2 Constructor & Destructor Documentation

**29.324.2.1** `Go::Param2FunctionInt::Param2FunctionInt ( shared_ptr< ParamSurface > surf ) [explicit]`

Constructor.

#### Parameters

<i>surf</i>	the parametric 1-dimensional surface defining the object.
-------------	-----------------------------------------------------------

**29.324.2.2** `Go::Param2FunctionInt::Param2FunctionInt ( shared_ptr< ParamSurface > surf, Param2FunctionInt * parent ) [explicit]`

Constructor.

#### Parameters

<i>surf</i>	the parametric 1-dimensional surface defining the object.
<i>parent</i>	the parent object to this object.

**29.324.2.3** `virtual Go::Param2FunctionInt::~~Param2FunctionInt ( ) [inline],[virtual]`

Destructor.



Definition at line 72 of file Param2FunctionInt.h.

### 29.324.3 Member Function Documentation

29.324.3.1 void Go::Param2FunctionInt::assureInRange ( int *pardir*, double & *t* )

Make sure that the input parameter lies inside the range of the parametric surface. Set *t* equal to *umin/vmin* if it lies below *umin/vmin*, or *umax/vmax* if it lies above *umax/vmax*.

#### Parameters

<i>pardir</i>	the parameter direction in question. Indexing starts at 0.
<i>t</i>	the input parameter

29.324.3.2 virtual bool Go::Param2FunctionInt::boundaryPoint ( const double \* *par*, double *eps* ) const [virtual]

Return true if the specified point lies within *eps* from the boundary.

#### Parameters

<i>par</i>	the parameter in which to evaluate. Size of array should be equal to <a href="#">numParams()</a> .
<i>eps</i>	the tolerance defining boundary neighbourhood.

Implements [Go::ParamObjectInt](#).

29.324.3.3 virtual bool Go::Param2FunctionInt::canDivide ( int *pardir* ) [virtual]

Return true if we are allowed to divide in the specified parameter direction.

#### Parameters

<i>pardir</i>	the parameter direction in question.
---------------	--------------------------------------

Implements [Go::ParamObjectInt](#).

29.324.3.4 virtual CompositeBox Go::Param2FunctionInt::compositeBox ( ) const [virtual]

Return the [CompositeBox](#) for the parametric object.

#### Returns

The [compositeBox](#) for the parametric object.

Implements [Go::ParamObjectInt](#).

29.324.3.5 `void Go::Param2FunctionInt::derivs ( double u, double v, Point & deriv_u, Point & deriv_v ) const`  
`[inline]`

Return the partial derivatives in the input parameter point.

#### Parameters

<i>u</i>	the u-parameter
<i>v</i>	the v-parameter
<i>deriv</i> <sub>↔<i>u</i></sub>	the partial derivative in the u-direction.
<i>deriv</i> <sub>↔<i>v</i></sub>	the partial derivative in the v-direction.

Definition at line 351 of file Param2FunctionInt.h.

29.324.3.6 `int Go::Param2FunctionInt::dimension ( )` `[inline]`

Return the dimension of the geometric space.

Definition at line 278 of file Param2FunctionInt.h.

29.324.3.7 `virtual double Go::Param2FunctionInt::endParam ( int par_dir ) const` `[virtual]`

Return the end parameter in the specified direction.

#### Parameters

<i>par_dir</i>	the parameter direction in question. Indexing starts at 0.
----------------	------------------------------------------------------------

Implements [Go::ParamObjectInt](#).

Reimplemented in [Go::Spline2FunctionInt](#).

29.324.3.8 `Go::SplineCurve* Go::Param2FunctionInt::extractBdCurve ( const Go::SplineSurface &, int bdidx, int & par_dir, double & tpar ) const` `[protected]`

29.324.3.9 `virtual void Go::Param2FunctionInt::getBoundaryObjects ( std::vector< shared_ptr< BoundaryFunctionInt > & bd_objs )` `[virtual]`

Return the boundary objects of this object.

#### Parameters

<i>bd_objs</i>	the boundary objects of this object.
----------------	--------------------------------------

Implements [Go::ParamFunctionInt](#).

Reimplemented in [Go::Spline2FunctionInt](#).

29.324.3.10 `shared_ptr<ParamCurve> Go::Param2FunctionInt::getConstantParameterCurve ( int dir, double par )`

Return a curveOnSurface along the current surface in the given direction and parameter value

29.324.3.11 `virtual std::vector<double> Go::Param2FunctionInt::getCriticalVals ( int pardir ) const [virtual]`

Return the critical parameter values in the specified direction.

#### Parameters

<i>pardir</i>	the parameter direction in question. Indexing starts at 0.
---------------	------------------------------------------------------------

Implements [Go::ParamObjectInt](#).

29.324.3.12 `virtual std::vector<double> Go::Param2FunctionInt::getCriticalValsAndKnots ( int pardir ) const [virtual]`

Return the critical parameter values and inner knots for the object.

#### Parameters

<i>pardir</i>	the parameter direction in question. Indexing starts at 0.
---------------	------------------------------------------------------------

Implements [Go::ParamObjectInt](#).

Reimplemented in [Go::Spline2FunctionInt](#).

29.324.3.13 `virtual std::vector<double> Go::Param2FunctionInt::getInnerKnotVals ( int pardir, bool sort = false ) const [virtual]`

Return the inner knot values in the specified direction.

#### Parameters

<i>pardir</i>	the parameter direction in question. Indexing starts at 0.
<i>sort</i>	the returned values may be sorted by the function.

Implements [Go::ParamObjectInt](#).

Reimplemented in [Go::Spline2FunctionInt](#).

29.324.3.14 `shared_ptr<ParamCurve> Go::Param2FunctionInt::getIsoCurve ( double param_start, double param_end, double isoval, bool pardir_is_u ) const`

Returns the specified isocurve.

## Parameters

<i>param_start</i>	start parameter for the isocurve.
<i>param_end</i>	end parameter for the isocurve.
<i>isoval</i>	the value for the isoparameter.
<i>pardir_is_u</i> ↔ <i>_u</i>	if 'pardir_is_u' is 'true', then the first parameter is the running direction and the second parameter is the isoparameter; vice versa for 'pardir_is_u' equal to 'false'.

29.324.3.15 `virtual void Go::Param2FunctionInt::getLengthAndWiggle ( double * length, double * wiggle ) [virtual]`

Return an estimate on the size and wiggle of the object.

## Parameters

<i>length</i>	the approximative length of the object in the corresponding parameter directions. The size of the array should be equal to <a href="#">numParams()</a> .
<i>wiggle</i>	a scalar representing the wiggle of the object in the corresponding parameter directions. The size of the array should be equal to <a href="#">numParams()</a> .

Implements [Go::ParamObjectInt](#).

29.324.3.16 `virtual std::vector<double>::iterator Go::Param2FunctionInt::getMesh ( ) [virtual]`

Return the geometric sample mesh for the parametric function.

Implements [Go::ParamFunctionInt](#).

29.324.3.17 `virtual int Go::Param2FunctionInt::getMeshSize ( int dir ) [virtual]`

Return the size of the geometric sample mesh in the specified direction.

## Parameters

<i>dir</i>	the parameter direction in question. Indexing starts at 0.
------------	------------------------------------------------------------

Implements [Go::ParamFunctionInt](#).

29.324.3.18 `std::vector<double> Go::Param2FunctionInt::getMima ( ) const`

Return the domain of the surface.

## Returns

The returned vector consists of (umin, umax, vmin, vmax).

29.324.3.19 `virtual Param2FunctionInt* Go::Param2FunctionInt::getParam2FunctionInt ( ) [virtual]`

Return pointer to this object.

Reimplemented from [Go::ParamFunctionInt](#).

29.324.3.20 `shared_ptr<ParamSurface> Go::Param2FunctionInt::getParamSurface ( )`

Return pointer to the parametric surface.

29.324.3.21 `shared_ptr<const ParamSurface> Go::Param2FunctionInt::getParamSurface ( ) const`

Return pointer to the parametric surface.

29.324.3.22 `shared_ptr<ParamSurface> Go::Param2FunctionInt::getParentParamSurface ( RectDomain & domain )`

Return pointer to a subpart of the parent surface of this object. If a parent surface does not exist, return pointer to surface in this object. To reduce numerical noise we go straight to the source (undivided) surface.

29.324.3.23 `shared_ptr<ParamSurface> Go::Param2FunctionInt::getParentParamSurface ( )`

Return pointer to the parent-curve of this object. If a parent surface does not exist, return pointer to surface in this object. To reduce numerical noise we go straight to the source (undivided) surface.

29.324.3.24 `shared_ptr<ParamSurface> Go::Param2FunctionInt::getSurface ( ) [inline]`

Return pointer to the parametric surface.

#### Returns

The pointer to the parametric surface.

Definition at line 338 of file Param2FunctionInt.h.

29.324.3.25 `shared_ptr<const ParamSurface> Go::Param2FunctionInt::getSurface ( ) const [inline]`

Return pointer to the parametric surface.

#### Returns

The pointer to the parametric surface.

Definition at line 343 of file Param2FunctionInt.h.

29.324.3.26 `virtual bool Go::Param2FunctionInt::hasCriticalVals ( int pdir ) const [virtual]`

Return true if the object has any critical parameter values in the specified parameter direction.

## Parameters

<i>pardir</i>	the parameter direction in question. Indexing starts at 0.
---------------	------------------------------------------------------------

Implements [Go::ParamObjectInt](#).

**29.324.3.27** `virtual bool Go::Param2FunctionInt::hasCriticalValsOrKnots ( int pardir ) const` [virtual]

Return true if the object has any critical parameter values or inner knots in the specified parameter direction.

## Parameters

<i>pardir</i>	the parameter direction in question. Indexing starts at 0.
---------------	------------------------------------------------------------

Implements [Go::ParamObjectInt](#).

Reimplemented in [Go::Spline2FunctionInt](#).

**29.324.3.28** `virtual bool Go::Param2FunctionInt::hasInnerKnots ( int pardir ) const` [virtual]

Return true if the object has any inner knots in the specified parameter direction.

## Parameters

<i>pardir</i>	the parameter direction in question. Indexing starts at 0.
---------------	------------------------------------------------------------

Implements [Go::ParamObjectInt](#).

Reimplemented in [Go::Spline2FunctionInt](#).

**29.324.3.29** `bool Go::Param2FunctionInt::isDegenerate ( double epsge, int pardir )`

Return true if the object is degenerate in the specified direction.

## Parameters

<i>epsge</i>	the geometric tolerance defining degeneracy.
<i>pardir</i>	the parameter direction in question.

**29.324.3.30** `virtual bool Go::Param2FunctionInt::isDegenerate ( double epsge, int pardir, double * par )` [virtual]

Return true if the object is degenerate in the specified direction and parameter.

## Parameters

<i>epsge</i>	the geometric tolerance defining degeneracy.
<i>pardir</i>	the parameter direction in question.
<i>par</i>	the parameter in which to evaluate. Size of array should be equal to <a href="#">numParams()</a> .

Reimplemented from [Go::ParamObjectInt](#).

**29.324.3.31** `double Go::Param2FunctionInt::isolateDegPar ( int dir, int deg_edge, double threshold )`

Return info on parameter domain which needs special treatment near a degenerated edge.

## Parameters

<i>dir</i>	the parameter direction in question. Indexing starts at 0.
<i>deg_edge</i>	the degenerate edge
<i>threshold</i>	a tolerance

## Returns

a safe parameter value that is isolated from the degenerate value

**29.324.3.32** `virtual int Go::Param2FunctionInt::knotIntervalFuzzy ( int pardir, double & t, double tol ) const` `[virtual]`

Return the knot interval for which t lies inside, moving the value t if it lies close to a knot.

## Parameters

<i>pardir</i>	the parameter direction in question. Indexing starts at 0.
<i>t</i>	the parameter value.
<i>tol</i>	the parametric tolerance deciding if the input parameter t should be moved.

Reimplemented in [Go::Spline2FunctionInt](#).

**29.324.3.33** `virtual shared_ptr<Param2FunctionInt> Go::Param2FunctionInt::makeIntFunction ( shared_ptr<ParamSurface> surf )` `[virtual]`

Return an intersection object for the input surface. This object is used as the parent for the intersection object.

## Parameters

<i>surf</i>	the parametric surface defining the intersection object.
-------------	----------------------------------------------------------

Reimplemented in [Go::Spline2FunctionInt](#).

29.324.3.34 `virtual bool Go::Param2FunctionInt::monotone ( Point & dir, double tol = 1.0e-15 ) const` [virtual]

Return true if the surface is monotone in any direction.

#### Parameters

<i>dir</i>	the direction in which the surface is monotone. Pertains only if surface is monotone.
------------	---------------------------------------------------------------------------------------

#### Returns

Whether or not the surface is monotone.

Implements [Go::ParamFunctionInt](#).

Reimplemented in [Go::Spline2FunctionInt](#).

29.324.3.35 `virtual double Go::Param2FunctionInt::nextSegmentVal ( int pdir, double par, bool forward ) const`  
[virtual]

Return the value of the knot next to the input parameter par.

#### Parameters

<i>pdir</i>	the parameter direction in question. Indexing starts at 0.
<i>par</i>	the parameter value
<i>forward</i>	if true we return the closest knot to the right, otherwise the closest knot to the left.

#### Returns

The knot closest to the input parameter.

Reimplemented in [Go::Spline2FunctionInt](#).

29.324.3.36 `virtual int Go::Param2FunctionInt::numParams ( ) const` [virtual]

The number of parameters in the object.

Implements [Go::ParamObjectInt](#).

29.324.3.37 `virtual double Go::Param2FunctionInt::paramFromMesh ( int dir, int idx )` [virtual]

Return the corresponding mesh parameter.

#### Parameters

<i>dir</i>	the parameter direction in question. Indexing starts at 0.
<i>idx</i>	the mesh idx in the specified direction. Indexing starts at 0.



Implements [Go::ParamFunctionInt](#).

**29.324.3.38** `virtual void Go::Param2FunctionInt::point ( Point & res, const double * par ) const [inline], [virtual]`

Evaluate the object in the input parameter.

#### Parameters

<i>res</i>	the <a href="#">Point</a> to be returned.
<i>par</i>	the parameter in which to evaluate. The size of the array should be equal to <a href="#">numParams()</a> .

Implements [Go::ParamObjectInt](#).

Definition at line 78 of file Param2FunctionInt.h.

**29.324.3.39** `virtual void Go::Param2FunctionInt::point ( std::vector< Point > & pt, const double * tpar, int der, const bool * from_right = 0, double resolution = 1.0e-12 ) const [inline], [virtual]`

Evaluate the object in the input parameter, with the specified number of derivatives.

#### Parameters

<i>pt</i>	the vector of points to be returned.
<i>tpar</i>	the parameter in which to evaluate. The size of the array should be equal to <a href="#">numParams()</a> .
<i>der</i>	the number of derivatives to calculate.
<i>from_right</i>	if true the evaluation is to be performed from the right side of the parameter value.
<i>resolution</i>	tolerance used when determining whether parameters are located at special values of the parameter domain (in particular: knot values in case of spline objects).

Implements [Go::ParamObjectInt](#).

Definition at line 92 of file Param2FunctionInt.h.

**29.324.3.40** `virtual double Go::Param2FunctionInt::startParam ( int paddir ) const [virtual]`

Return the start parameter value in the specified direction.

#### Parameters

<i>paddir</i>	the parameter direction in question. Indexing starts at 0.
---------------	------------------------------------------------------------

Implements [Go::ParamObjectInt](#).

Reimplemented in [Go::Spline2FunctionInt](#).

29.324.3.41 `virtual void Go::Param2FunctionInt::subdivide ( int pardir, double par, std::vector< shared_ptr< ParamFunctionInt > > & subdiv_objs, std::vector< shared_ptr< ParamFunctionInt > > & bd_objs )`  
`[virtual]`

Subdivide the object in the specified parameter direction and parameter value.

#### Parameters

<i>pardir</i>	direction in which to subdivide. Indexing starts at 0.
<i>par</i>	parameter in which to subdivide.
<i>subdiv_objs</i>	The subparts of this object. Of the same geometric dimension as this object.
<i>bd_objs</i>	the boundaries between the returned <i>subdiv_objs</i> . Of geometric dimension 1 less than this object.

Implements [Go::ParamFunctionInt](#).

### 29.324.4 Member Data Documentation

29.324.4.1 `bool Go::Param2FunctionInt::bd_deg_[4]` `[protected]`

Definition at line 380 of file Param2FunctionInt.h.

29.324.4.2 `double Go::Param2FunctionInt::deg_tol_` `[protected]`

Definition at line 381 of file Param2FunctionInt.h.

29.324.4.3 `int Go::Param2FunctionInt::dim_` `[protected]`

Definition at line 365 of file Param2FunctionInt.h.

29.324.4.4 `RectDomain Go::Param2FunctionInt::domain_` `[protected]`

Definition at line 382 of file Param2FunctionInt.h.

29.324.4.5 `std::vector<double> Go::Param2FunctionInt::mesh_` `[mutable],[protected]`

Definition at line 385 of file Param2FunctionInt.h.

29.324.4.6 `int Go::Param2FunctionInt::nmesh_[2]` `[mutable],[protected]`

Definition at line 384 of file Param2FunctionInt.h.

29.324.4.7 `Param2FunctionInt* Go::Param2FunctionInt::parentfunction_` `[protected]`

Definition at line 367 of file Param2FunctionInt.h.

29.324.4.8 `std::vector<std::pair<double, int> > Go::Param2FunctionInt::segment_[2]` [protected]

Definition at line 370 of file Param2FunctionInt.h.

29.324.4.9 `shared_ptr<ParamSurface> Go::Param2FunctionInt::surf_` [protected]

Definition at line 362 of file Param2FunctionInt.h.

29.324.4.10 `std::vector<Point> Go::Param2FunctionInt::temp_point_array_` [mutable], [protected]

Definition at line 387 of file Param2FunctionInt.h.

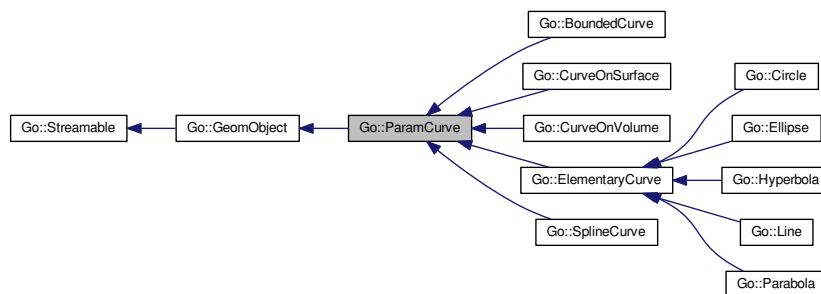
The documentation for this class was generated from the following file:

- intersections/include/GoTools/intersections/[Param2FunctionInt.h](#)

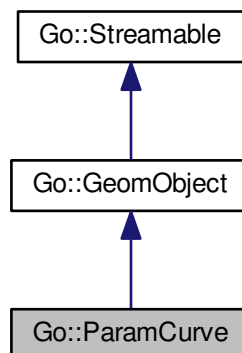
## 29.325 Go::ParamCurve Class Reference

```
#include <ParamCurve.h>
```

Inheritance diagram for Go::ParamCurve:



Collaboration diagram for Go::ParamCurve:



## Public Member Functions

- virtual `~ParamCurve ()`  
*virtual destructor - ensures safe inheritance*
- virtual void `point (Point &pt, double tpar) const =0`
- virtual void `point (std::vector< Point > &pts, double tpar, int derivs, bool from_right=true) const =0`
- `Point point (double tpar) const`
- `std::vector< Point > point (double tpar, int derivs, bool from_right=true) const`
- virtual void `uniformEvaluator (int num, std::vector< Point > &points, std::vector< double > &param) const`
- virtual `double startparam () const =0`
- virtual `double endparam () const =0`
- virtual void `reverseParameterDirection (bool switchparam=false)=0`
- virtual void `setParameterInterval (double t1, double t2)=0`  
*Linear reparametrization. The meaning is changed for elementary curves.*
- virtual `SplineCurve * geometryCurve ()=0`
- virtual `bool isDegenerate (double degenerate_epsilon)=0`
- virtual `bool isClosed ()`
- virtual `ParamCurve * subCurve (double from_par, double to_par, double fuzzy=DEFAULT_PARAMETER←_EPSILON) const =0`
- virtual `std::vector< shared_ptr< ParamCurve > > split (double param, double fuzzy=DEFAULT_PARAM←ETER_EPSILON) const`  
*Split curve in a specified parameter value.*
- virtual `ParamCurve * clone () const =0`
- virtual `DirectionCone directionCone () const =0`
- virtual `CompositeBox compositeBox () const`
- virtual void `appendCurve (ParamCurve *cv, bool reparam=true)=0`
- virtual void `appendCurve (ParamCurve *cv, int continuity, double &dist, bool reparam=true)=0`
- `double estimatedCurveLength (int numpts=4) const`
- `double estimatedCurveLength (double tmin, double tmax, int numpts=4) const`
- virtual void `closestPoint (const Point &pt, double tmin, double tmax, double &clo_t, Point &clo_pt, double &clo_dist, double const *seed=0) const =0`
- void `closestPoint (const Point &pt, double &clo_t, Point &clo_pt, double &clo_dist) const`
- virtual `double nextSegmentVal (double par, bool forward, double tol) const`
- virtual `double length (double tol)=0`
- virtual `double length (double tol, double tstart, double tend)`
- virtual `bool isAxisRotational (Point &centre, Point &axis, Point &vec, double &angle)`
- virtual `bool isLinear (Point &dir, double tol)`  
*Check if the curve is linear.*
- virtual `bool isInPlane (const Point &loc, const Point &axis, double eps, Point &normal) const`  
*Check if the curves lies in a plane passing through a given axis.*
- virtual `bool isInPlane (const Point &norm, double eps, Point &pos) const`  
*Check if the curve lies in a plane with a given normal.*

## Protected Member Functions

- void `closestPointGeneric (const Point &pt, double tmin, double tmax, double guess_param, double &clo_t, Point &clo_pt, double &clo_dist) const`
- void `s1771 (Point pt, double aepsge, double astart, double aend, double anext, double &cpos, int *jstat) const`
- void `s1771_s9point (Point pt, std::vector< Point > val, Point diff, double astart, double aend, int max_it, double *cnext, double *ad, double adel, double *cdist, double aprev, int *jstat) const`
- `double s1771_s9del (double *eco, double *eco1, double *eco2, int idim) const`

## Additional Inherited Members

### 29.325.1 Detailed Description

Base class for parametric curves in [Go](#)

Definition at line 62 of file ParamCurve.h.

### 29.325.2 Constructor & Destructor Documentation

29.325.2.1 `virtual Go::ParamCurve::~~ParamCurve ( ) [virtual]`

virtual destructor - ensures safe inheritance

### 29.325.3 Member Function Documentation

29.325.3.1 `virtual void Go::ParamCurve::appendCurve ( ParamCurve * cv, bool reparam = true ) [pure virtual]`

append a curve to this curve, with eventual reparametrization NB: This virtual member function currently only works for SplineCurves and CurveOnSurfaces. Moreover, 'this' curve and the 'cv' curve must be of the same type.

#### Parameters

<i>cv</i>	the curve to append to 'this' curve.
<i>reparam</i>	specify whether or not there should be reparametrization

Implemented in [Go::SplineCurve](#), [Go::CurveOnSurface](#), [Go::CurveOnVolume](#), [Go::BoundedCurve](#), [Go::Circle](#), [Go::Line](#), [Go::Hyperbola](#), [Go::Parabola](#), and [Go::Ellipse](#).

29.325.3.2 `virtual void Go::ParamCurve::appendCurve ( ParamCurve * cv, int continuity, double & dist, bool reparam = true ) [pure virtual]`

append a curve to this curve, with eventual reparametrization

#### Parameters

<i>cv</i>	the curve to append to 'this' curve.
<i>continuity</i>	the required continuity at the transition. Can be $G^{(-1)}$ and upwards.
<i>dist</i>	a measure of the local distortion around the transition in order to achieve the specified continuity.
<i>reparam</i>	specify whether or not there should be reparametrization

Implemented in [Go::SplineCurve](#), [Go::CurveOnSurface](#), [Go::CurveOnVolume](#), [Go::BoundedCurve](#), [Go::Circle](#), [Go::Line](#), [Go::Hyperbola](#), [Go::Parabola](#), and [Go::Ellipse](#).

29.325.3.3 `virtual ParamCurve* Go::ParamCurve::clone ( ) const` `[pure virtual]`

The clone-function is inherited from [GeomObject](#), but overridden here to get a covariant return type (for those compilers that allow this).

Implements [Go::GeomObject](#).

Implemented in [Go::SplineCurve](#), [Go::CurveOnSurface](#), [Go::CurveOnVolume](#), [Go::BoundedCurve](#), [Go::Line](#), [Go::Circle](#), [Go::Hyperbola](#), [Go::Parabola](#), [Go::Ellipse](#), and [Go::ElementaryCurve](#).

29.325.3.4 `virtual void Go::ParamCurve::closestPoint ( const Point & pt, double tmin, double tmax, double & clo_t, Point & clo_pt, double & clo_dist, double const * seed = 0 ) const` `[pure virtual]`

Compute the closest point from an interval of this curve to a specified point.

Parameters

<i>pt</i>	point we want to find the closest point to
<i>tmin</i>	start parameter of search interval
<i>tmax</i>	end parameter of search interval
<i>clo_t</i>	upon function return, 'clo_t' will contain the parameter value of the closest point found.
<i>clo_pt</i>	upon function return, 'clo_pt' will contain the position of the closest point found.
<i>clo_dist</i>	upon function return, 'clo_dist' will contain the distance between 'pt' and the closest point found.
<i>seed</i>	pointer to initial guess value, provided by the user (can be 0, for which the algorithm will determine a (hopefully) reasonable choice).

Implemented in [Go::SplineCurve](#), [Go::CurveOnSurface](#), [Go::CurveOnVolume](#), [Go::BoundedCurve](#), [Go::Circle](#), [Go::Line](#), [Go::Hyperbola](#), [Go::Parabola](#), and [Go::Ellipse](#).

29.325.3.5 `void Go::ParamCurve::closestPoint ( const Point & pt, double & clo_t, Point & clo_pt, double & clo_dist ) const`

Compute the closest point from this curve to a specified point, taking the whole curve into account (not just an interval of it).

Parameters

<i>pt</i>	point we want to find the closest point to
<i>clo_t</i>	upon function return, 'clo_t' will contain the parameter value of the closest point found.
<i>clo_pt</i>	upon function return, 'clo_pt' will contain the position of the closest point found.
<i>clo_dist</i>	upon function return, 'clo_dist' will contain the distance between 'pt' and the closest point found.

29.325.3.6 `void Go::ParamCurve::closestPointGeneric ( const Point & pt, double tmin, double tmax, double guess_param, double & clo_t, Point & clo_pt, double & clo_dist ) const` `[protected]`

29.325.3.7 `virtual CompositeBox Go::ParamCurve::compositeBox ( ) const` `[virtual]`

Creates a composite box enclosing the curve. The composite box consists of an inner and an edge box. The inner box is supposed to be made from the interior of the curve, while the edge box is made from the start and end points.

The default implementation simply makes both boxes identical to the regular bounding box.

#### Returns

the [CompositeBox](#) enclosing the curve.

Reimplemented in [Go::SplineCurve](#).

**29.325.3.8** `virtual DirectionCone Go::ParamCurve::directionCone ( ) const [pure virtual]`

Creates a [DirectionCone](#) which covers all tangent directions of this curve.

#### Returns

the smallest [DirectionCone](#) containing all tangent directions of this curve.

Implemented in [Go::SplineCurve](#), [Go::CurveOnSurface](#), [Go::BoundedCurve](#), [Go::Circle](#), [Go::Line](#), [Go::Hyperbola](#), [Go::Parabola](#), [Go::Ellipse](#), and [Go::CurveOnVolume](#).

**29.325.3.9** `virtual double Go::ParamCurve::endparam ( ) const [pure virtual]`

Query the end parameter of the curve

#### Returns

the curve's end parameter

Implemented in [Go::SplineCurve](#), [Go::CurveOnSurface](#), [Go::CurveOnVolume](#), [Go::BoundedCurve](#), [Go::Line](#), [Go::Circle](#), [Go::Hyperbola](#), [Go::Parabola](#), and [Go::Ellipse](#).

**29.325.3.10** `double Go::ParamCurve::estimatedCurveLength ( int numpts = 4 ) const`

Estimate the length of the curve, by sampling it at a certain number of points and calculating the linear approximation to the curve through these points.

#### Parameters

<i>numpts</i>	number of sample points used
---------------	------------------------------

#### Returns

the estimated curve length

**29.325.3.11** `double Go::ParamCurve::estimatedCurveLength ( double tmin, double tmax, int numpts = 4 ) const`

Estimate the length of an interval of the curve, by sampling it at a certain number of points in the interval and calculating the linear approximation through these points.

## Parameters

<i>tmin</i>	parameter at start of interval
<i>tmax</i>	parameter at end of interval
<i>numpts</i>	number of sample points used

## Returns

the estimated curve length

**29.325.3.12** `virtual SplineCurve* Go::ParamCurve::geometryCurve ( )` [pure virtual]

If the definition of this [ParamCurve](#) contains a [SplineCurve](#) describing its spatial shape, then this function will return a pointer to this [SplineCurve](#). Otherwise it will return a null pointer. The returned curve is NEWed, so the user is responsible for deleting it. This function may have side-effects.

## Returns

a pointer to a [SplineCurve](#) representation of the [ParamCurve](#), if it exists. Null pointer otherwise.

Implemented in [Go::SplineCurve](#), [Go::CurveOnSurface](#), [Go::CurveOnVolume](#), [Go::BoundedCurve](#), [Go::Line](#), [Go::Circle](#), [Go::Hyperbola](#), [Go::Parabola](#), and [Go::Ellipse](#).

**29.325.3.13** `virtual bool Go::ParamCurve::isAxisRotational ( Point & centre, Point & axis, Point & vec, double & angle )`  
[inline], [virtual]

Check if the curve is axis rotational. Only true if a connection to an axis rotational elementary curve exist The axis and rotational angle is only specified if the curve is actually rotational

Reimplemented in [Go::SplineCurve](#), [Go::CurveOnSurface](#), [Go::Circle](#), and [Go::BoundedCurve](#).

Definition at line 313 of file ParamCurve.h.

**29.325.3.14** `virtual bool Go::ParamCurve::isClosed ( )` [virtual]

Query whether the curve is closed. Periodic curves like circles and ellipses are closed.

## Returns

`true` if the curve is closed, `false` otherwise.

**29.325.3.15** `virtual bool Go::ParamCurve::isDegenerate ( double degenerate_epsilon )` [pure virtual]

Query whether the curve is degenerate (collapsed into a single point).



## Parameters

<i>degenerate_epsilon</i>	the tolerance used in determine whether the curve is degenerate. A curve is considered degenerate if its total length is shorter than this value.
---------------------------	---------------------------------------------------------------------------------------------------------------------------------------------------

## Returns

`true` if the curve is degenerate, `false` otherwise.

Implemented in [Go::SplineCurve](#), [Go::CurveOnSurface](#), [Go::CurveOnVolume](#), [Go::BoundedCurve](#), [Go::Circle](#), [Go::Line](#), [Go::Hyperbola](#), [Go::Parabola](#), and [Go::Ellipse](#).

**29.325.3.16** `virtual bool Go::ParamCurve::isInPlane ( const Point & loc, const Point & axis, double eps, Point & normal ) const` [virtual]

Check if the curves lies in a plane passing through a given axis.

Reimplemented in [Go::SplineCurve](#), [Go::CurveOnSurface](#), [Go::Circle](#), [Go::Line](#), and [Go::BoundedCurve](#).

**29.325.3.17** `virtual bool Go::ParamCurve::isInPlane ( const Point & norm, double eps, Point & pos ) const` [virtual]

Check if the curve lies in a plane with a given normal.

Reimplemented in [Go::SplineCurve](#), [Go::CurveOnSurface](#), [Go::Circle](#), [Go::Line](#), [Go::Ellipse](#), [Go::Hyperbola](#), and [Go::Parabola](#).

**29.325.3.18** `virtual bool Go::ParamCurve::isLinear ( Point & dir, double tol )` [virtual]

Check if the curve is linear.

Reimplemented in [Go::SplineCurve](#), [Go::CurveOnSurface](#), [Go::Line](#), and [Go::BoundedCurve](#).

**29.325.3.19** `virtual double Go::ParamCurve::length ( double tol )` [pure virtual]

Compute the total length of this curve up to some tolerance

## Parameters

<i>tol</i>	the relative tolerance when approximating the length, i.e. stop iteration when error becomes smaller than <code>tol/(curve length)</code>
------------	-------------------------------------------------------------------------------------------------------------------------------------------

## Returns

the length calculated

Implemented in [Go::SplineCurve](#), [Go::CurveOnSurface](#), [Go::CurveOnVolume](#), [Go::BoundedCurve](#), [Go::Circle](#), [Go::Line](#), [Go::Hyperbola](#), [Go::Parabola](#), and [Go::Ellipse](#).

29.325.3.20 `virtual double Go::ParamCurve::length ( double tol, double tstart, double tend ) [virtual]`

Compute the length of a segment of this curve up to some tolerance

#### Parameters

<i>tol</i>	the relative tolerance when approximating the length, i.e. stop iteration when error becomes smaller than $tol/(curve\ length)$
<i>tstart</i>	the parameter value for the start point of the segment
<i>tend</i>	the parameter value for the end point of the segment

#### Returns

the length calculated

29.325.3.21 `virtual double Go::ParamCurve::nextSegmentVal ( double par, bool forward, double tol ) const [virtual]`

If the [ParamCurve](#) is divided up into logical segments, this function will return the parameter value of the "next segment", starting from a parameter given by the user. If no division into logical segments exist, then it is the start- or end parameter that is returned.

#### Parameters

<i>par</i>	the parameter from which we start the search for the next segment.
<i>forward</i>	whether to search forwards or backwards along the parameter domain.
<i>tol</i>	the tolerance to determine or not 'par' is already located on the start of the next segment.

#### Returns

the parameter value of the next segment.

Reimplemented in [Go::SplineCurve](#), [Go::CurveOnSurface](#), and [Go::CurveOnVolume](#).

29.325.3.22 `virtual void Go::ParamCurve::point ( Point & pt, double tpar ) const [pure virtual]`

Evaluate the curve's position at a given parameter

#### Parameters

<i>pt</i>	the evaluated position will be written to this <a href="#">Point</a>
<i>tpar</i>	the parameter for which we wish to evaluate the curve

Implemented in [Go::SplineCurve](#), [Go::CurveOnSurface](#), [Go::CurveOnVolume](#), [Go::BoundedCurve](#), [Go::Line](#), [Go::Circle](#), [Go::Hyperbola](#), [Go::Parabola](#), and [Go::Ellipse](#).

29.325.3.23 `virtual void Go::ParamCurve::point ( std::vector< Point > & pts, double tpar, int derivs, bool from_right = true ) const` [pure virtual]

Evaluate the curve's position and a certain number of derivatives at a given parameter.

#### Parameters

<i>pts</i>	the evaluated position and derivatives (tangent, curvature vector, etc.) will be written to this vector. The first entry will be the position, the second entry will be the first derivative, etc. The size of this vector must be set to 'derivs'+ 1 prior to calling this function.
<i>tpar</i>	the parameter for which we want to evaluate the curve
<i>derivs</i>	the number of derivatives we want to have calculated
<i>from_right</i>	specify whether we should calculate derivatives 'from the right' or 'from the left' (default is from the right). This matters only when the curve presents discontinuities in its derivatives.

Implemented in [Go::SplineCurve](#), [Go::CurveOnSurface](#), [Go::CurveOnVolume](#), [Go::BoundedCurve](#), [Go::Line](#), [Go::Circle](#), [Go::Hyperbola](#), [Go::Parabola](#), and [Go::Ellipse](#).

29.325.3.24 `Point Go::ParamCurve::point ( double tpar ) const`

Evaluate the curve's position at a certain parameter

#### Parameters

<i>tpar</i>	the parameter for which we want to evaluate the curve's position
-------------	------------------------------------------------------------------

#### Returns

the curve's position for the parameter 'tpar'. NB: This function is implemented in terms of the [ParamCurve](#)'s virtual 'point(...)' functions, but is itself not virtual. If you make a concrete subclass and wish to make this function visible to the user, you must put a 'using [ParamCurve::point](#)' statement in the class definition.

29.325.3.25 `std::vector<Point> Go::ParamCurve::point ( double tpar, int derivs, bool from_right = true ) const`

Evaluate the curve's position and a certain number of derivatives at a given parameter.

#### Parameters

<i>tpar</i>	the parameter for which we want to evaluate the curve
<i>derivs</i>	the number of derivatives we want to have calculated
<i>from_right</i>	specify whether we should calculate derivatives 'from the right' or 'from the left' (default is from the right). This matters only when the curve presents discontinuities in its derivatives.

#### Returns

a STL vector containing the evaluated position and the specified number of derivatives. The first entry will be the position, the second entry will be the first derivative, etc. The total size of the returned vector will be 'derivs' + 1. NB: This function is implemented in terms of the [ParamCurve](#)'s virtual 'point(...)' functions, but is

itself not virtual. If you make a concrete subclass and wish to make this function visible to the user, you must put a 'using [ParamCurve::point](#)' in the class definition.

**29.325.3.26** `virtual void Go::ParamCurve::reverseParameterDirection ( bool switchparam = false )` [pure virtual]

Set the parameter direction of the curve. The curve's parameter interval will always remain constant, but by flipping the parameter direction, the curve will be traced the opposite way when moving a parameter over the parameter interval.

#### Parameters

<i>switchparam</i>	if true, and the curve is 2D, the x and y coordinates should be swapped. This is used when turning the orientation of bounded (trimmed) surfaces.
--------------------	---------------------------------------------------------------------------------------------------------------------------------------------------

Implemented in [Go::SplineCurve](#), [Go::CurveOnSurface](#), [Go::CurveOnVolume](#), [Go::BoundedCurve](#), and [Go::ElementaryCurve](#).

**29.325.3.27** `void Go::ParamCurve::s1771 ( Point pt, double aepsge, double astart, double aend, double anext, double & cpos, int * jstat ) const` [protected]

**29.325.3.28** `double Go::ParamCurve::s1771_s9del ( double * eco, double * eco1, double * eco2, int idim ) const` [protected]

**29.325.3.29** `void Go::ParamCurve::s1771_s9point ( Point pt, std::vector< Point > val, Point diff, double astart, double aend, int max_it, double * cnext, double * ad, double adel, double * cdist, double aprev, int * jstat ) const` [protected]

**29.325.3.30** `virtual void Go::ParamCurve::setParameterInterval ( double t1, double t2 )` [pure virtual]

Linear reparametrization. The meaning is changed for elementary curves.

Implemented in [Go::SplineCurve](#), [Go::CurveOnSurface](#), [Go::CurveOnVolume](#), [Go::BoundedCurve](#), [Go::Line](#), [Go::Circle](#), [Go::Hyperbola](#), [Go::Parabola](#), and [Go::Ellipse](#).

**29.325.3.31** `virtual std::vector<shared_ptr<ParamCurve>> > Go::ParamCurve::split ( double param, double fuzzy = DEFAULT_PARAMETER_EPSILON ) const` [virtual]

Split curve in a specified parameter value.

Reimplemented in [Go::SplineCurve](#), [Go::CurveOnSurface](#), and [Go::CurveOnVolume](#).

**29.325.3.32** `virtual double Go::ParamCurve::startparam ( ) const` [pure virtual]

Query the start parameter of the curve

#### Returns

the curve's start parameter

Implemented in [Go::SplineCurve](#), [Go::CurveOnSurface](#), [Go::CurveOnVolume](#), [Go::BoundedCurve](#), [Go::Line](#), [Go::Circle](#), [Go::Hyperbola](#), [Go::Parabola](#), and [Go::Ellipse](#).

29.325.3.33 `virtual ParamCurve* Go::ParamCurve::subCurve ( double from_par, double to_par, double fuzzy = DEFAULT_PARAMETER_EPSILON ) const [pure virtual]`

Returns a curve which is a part of this curve. The result is NEWed, so the user is responsible for deleting it. NB: It is not guaranteed that the [ParamCurve](#) that is returned is of the same type as the curve itself. Thus, the returned curve might be a [SplineCurve](#).

#### Parameters

<i>from_par</i>	start value of parameter interval that will define the subcurve
<i>to_par</i>	end value of parameter interval that will define the subcurve
<i>fuzzy</i>	since subCurve works on those curves who are spline-based, this tolerance defines how close the start and end parameter must be to an existing knot in order to be considered <i>on</i> the knot.

#### Returns

a pointer to a new subcurve which represents the part of the curve between 'from\_par' and 'to\_par'. It will be spline-based and have a k-regular knotvector. The user is responsible for deleting this subcurve when it is no longer needed.

Implemented in [Go::SplineCurve](#), [Go::CurveOnSurface](#), [Go::CurveOnVolume](#), [Go::BoundedCurve](#), [Go::Circle](#), [Go::Line](#), [Go::Hyperbola](#), [Go::Parabola](#), [Go::Ellipse](#), and [Go::ElementaryCurve](#).

29.325.3.34 `virtual void Go::ParamCurve::uniformEvaluator ( int num, std::vector< Point > & points, std::vector< double > & param ) const [virtual]`

Evaluate points on a regular set of parameter values

#### Parameters

<i>num</i>	number of values to evaluate
<i>points</i>	upon function return, this vector holds all the evaluated points
<i>param</i>	upon function return, this vector holds all the numerical values where evaluation has taken place

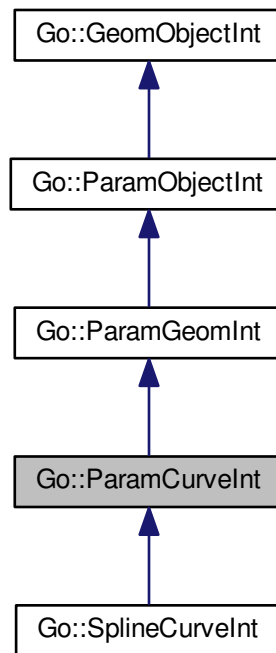
The documentation for this class was generated from the following file:

- [gotools-core/include/GoTools/geometry/ParamCurve.h](#)

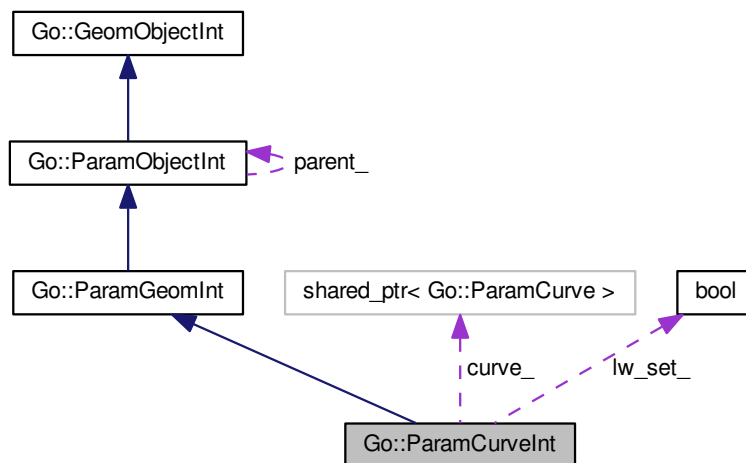
## 29.326 Go::ParamCurveInt Class Reference

```
#include <ParamCurveInt.h>
```

Inheritance diagram for Go::ParamCurveInt:



Collaboration diagram for Go::ParamCurveInt:



### Public Member Functions

- `ParamCurveInt` (`shared_ptr< ParamCurve > curve`, `ParamGeomInt *parent=0`)

- virtual `~ParamCurveInt ()`
  - Destructor.*
- virtual void `point (Point &res, const double *par) const`
- virtual void `point (std::vector< Point > &res, const double *par, int der, const bool *from_right=0, double resolution=1.0e-12) const`
- virtual `ParamCurveInt * getParamCurveInt ()`
- virtual `ParamSurfaceInt * getParamSurfaceInt ()`
- `shared_ptr< ParamCurve > getParamCurve ()`
- `shared_ptr< const ParamCurve > getParamCurve () const`
- `shared_ptr< ParamCurve > getParentParamCurve (double &start, double &end)`
- `shared_ptr< ParamCurve > getParentParamCurve ()`
- virtual `shared_ptr< ParamCurveInt > makeIntObject (shared_ptr< ParamCurve > curve)`
- virtual int `numParams () const`
  - The number of parameters in the object.*
- virtual void `getLengthAndWiggle (double *length, double *wiggle)`
- virtual `bool hasInnerKnots (int pdir) const`
- virtual `bool hasCriticalVals (int pdir) const`
- virtual void `setCriticalVal (int pdir, double par)`
  - Set info about subdivision value.*
- virtual `bool hasCriticalValsOrKnots (int pdir) const`
- virtual `bool canDivide (int pdir)`
- virtual `std::vector< double > getCriticalVals (int pdir) const`
- virtual `std::vector< double > getInnerKnotVals (int pdir, bool sort=false) const`
- virtual `std::vector< double > getCriticalValsAndKnots (int pdir) const`
- virtual int `getMeshSize (int dir)`
- virtual `double paramFromMesh (int dir, int idx)`
- virtual `std::vector< double >::iterator getMesh ()`
  - Return the geometric sample mesh for the parametric function.*
- virtual `double startParam (int pdir) const`
- virtual `double endParam (int pdir) const`
- virtual `bool boundaryPoint (const double *par, double eps) const`
- virtual void `subdivide (int pdir, double par, std::vector< shared_ptr< ParamGeomInt > > &subdiv_objs, std::vector< shared_ptr< ParamGeomInt > > &bd_objs)`
- virtual `CompositeBox compositeBox () const`
- virtual `DirectionCone directionCone () const`
- virtual void `getBoundaryObjects (std::vector< shared_ptr< BoundaryGeomInt > > &bd_objs)`
- virtual int `checkPeriodicity (int pdir=0) const`
- virtual int `dimension () const`
  - The dimension of the geometric space.*
- `double startparam () const`
- `double endparam () const`
- void `assureInRange (double &t)`
- virtual int `knotIntervalFuzzy (double &t, double tol) const`
- virtual `double nextSegmentVal (double par, bool forward, double tol) const`
- virtual `bool isDegenerate (double epsge, int dir, double *par)`
- virtual `bool isSpline ()`
- virtual `const SplineCurve * getSpline ()`
- virtual `double getOptimizedConeAngle (Point &axis1, Point &axis2)`
- virtual `RotatedBox getRotatedBox (std::vector< Point > &axis) const`
- void `axisFromEndpts (Point &axis) const`

## Protected Attributes

- `shared_ptr< ParamCurve > curve_`
- `int dim_`
- `std::vector< std::pair< double, int > > segment_`
- `std::vector< double > mesh_`
- `bool lw_set_`
- `double length_`
- `double wiggle_`

### 29.326.1 Detailed Description

This class represents the "intersection object" of a parametric curve.

Definition at line 56 of file ParamCurveInt.h.

### 29.326.2 Constructor & Destructor Documentation

29.326.2.1 `Go::ParamCurveInt::ParamCurveInt ( shared_ptr< ParamCurve > curve, ParamGeomInt * parent = 0 )`  
`[explicit]`

Constructor.

Parameters

<i>curve</i>	the parametric curve defining the intersection object.
<i>parent</i>	the parent object to this object. Can be either a curve or a surface.

29.326.2.2 `virtual Go::ParamCurveInt::~~ParamCurveInt ( )` `[inline],[virtual]`

Destructor.

Definition at line 67 of file ParamCurveInt.h.

### 29.326.3 Member Function Documentation

29.326.3.1 `void Go::ParamCurveInt::assureInRange ( double & t )`

Make sure that the input parameter lies inside the range of the parametric curve. Set `t` equal to `tmin` if it lies below `tmin`, or `tmax` if it lies above `tmax`.

Parameters

<i>t</i>	the input parameter
----------	---------------------



29.326.3.2 void Go::ParamCurveInt::axisFromEndpts ( Point & axis ) const

Create an axis by interpolating the end points.

#### Returns

The created axis.

29.326.3.3 virtual bool Go::ParamCurveInt::boundaryPoint ( const double \* par, double eps ) const [virtual]

Return true if the specified point lies within eps from the boundary.

#### Parameters

<i>par</i>	the parameter in which to evaluate. Size of array should be equal to <a href="#">numParams()</a> .
<i>eps</i>	the tolerance defining the boundary neighbourhood.

Implements [Go::ParamObjectInt](#).

29.326.3.4 virtual bool Go::ParamCurveInt::canDivide ( int pdir ) [virtual]

Return true if we are allowed to divide in the specified parameter direction.

#### Parameters

<i>pdir</i>	the parameter direction in question.
-------------	--------------------------------------

Implements [Go::ParamObjectInt](#).

29.326.3.5 virtual int Go::ParamCurveInt::checkPeriodicity ( int pdir = 0 ) const [virtual]

Check if the object is periodic. Analyze periodicity of curve based on number of repeating knots and control points. The return value is -1 if the curve ends are disjoint, otherwise k if cv is  $C^k$  continuous. These are sufficient but not necessary conditions for periodicity, so it is possible that a higher degree of periodicity exists. Should not be called on this layer, should be overruled by inherited class.

#### Parameters

<i>pdir</i>	the parameter direction in question. Does not pertain to for a curve.
-------------	-----------------------------------------------------------------------

#### Returns

-1 if the curve ends are disjoint, or k if the curve is proven to be  $C^k$  continuous.

Reimplemented from [Go::ParamGeomInt](#).

Reimplemented in [Go::SplineCurveInt](#).

29.326.3.6 virtual **CompositeBox** Go::ParamCurveInt::compositeBox ( ) const [virtual]

Create the [CompositeBox](#) for the parametric object.

#### Returns

The [CompositeBox](#) for the parametric object.

Implements [Go::ParamGeomInt](#).

29.326.3.7 virtual int Go::ParamCurveInt::dimension ( ) const [inline],[virtual]

The dimension of the geometric space.

Implements [Go::ParamGeomInt](#).

Definition at line 273 of file ParamCurveInt.h.

29.326.3.8 virtual **DirectionCone** Go::ParamCurveInt::directionCone ( ) const [virtual]

A cone which contains all normals of the object.

#### Returns

A cone which contains all normals of the object.

Implements [Go::ParamGeomInt](#).

29.326.3.9 virtual double Go::ParamCurveInt::endParam ( int *pardir* ) const [inline],[virtual]

Return the end parameter in the specified direction.

#### Parameters

<i>pardir</i>	the parameter direction in question. Indexing starts at 0.
---------------	------------------------------------------------------------

Implements [Go::ParamObjectInt](#).

Definition at line 222 of file ParamCurveInt.h.

29.326.3.10 double Go::ParamCurveInt::endparam ( ) const [inline]

The end parameter of the curve.

#### Returns

The end parameter of the curve.

Definition at line 283 of file ParamCurveInt.h.

29.326.3.11 `virtual void Go::ParamCurveInt::getBoundaryObjects ( std::vector< shared_ptr< BoundaryGeomInt > > & bd_objs ) [virtual]`

Return the boundary objects of this object.

#### Parameters

<i>bd_objs</i>	the boundary objects of this object.
----------------	--------------------------------------

Implements [Go::ParamGeomInt](#).

29.326.3.12 `virtual std::vector<double> Go::ParamCurveInt::getCriticalVals ( int pardir ) const [virtual]`

Return the critical parameter values in the specified direction.

#### Parameters

<i>pardir</i>	the parameter direction in question. Indexing starts at 0.
---------------	------------------------------------------------------------

Implements [Go::ParamObjectInt](#).

29.326.3.13 `virtual std::vector<double> Go::ParamCurveInt::getCriticalValsAndKnots ( int pardir ) const [virtual]`

Return the critical parameter values and inner knots for object.

#### Parameters

<i>pardir</i>	the parameter direction in question. Indexing starts at 0.
---------------	------------------------------------------------------------

Implements [Go::ParamObjectInt](#).

Reimplemented in [Go::SplineCurveInt](#).

29.326.3.14 `virtual std::vector<double> Go::ParamCurveInt::getInnerKnotVals ( int pardir, bool sort = false ) const [virtual]`

Return the inner knot values in the specified direction.

#### Parameters

<i>pardir</i>	the parameter direction in question. Indexing starts at 0.
<i>sort</i>	the returned values may be sorted by the function.

Implements [Go::ParamObjectInt](#).

Reimplemented in [Go::SplineCurveInt](#).

29.326.3.15 `virtual void Go::ParamCurveInt::getLengthAndWiggle ( double * length, double * wiggle ) [virtual]`

Return an estimate on the size and wiggle of the object.

#### Parameters

<i>length</i>	the approximative length of the object in the corresponding parameter directions. The size of the array should be equal to <code>numParams()</code> .
<i>wiggle</i>	a scalar representing the wiggle of the object in the corresponding parameter directions. The size of the array should be equal to <code>numParams()</code> .

Implements [Go::ParamObjectInt](#).

29.326.3.16 `virtual std::vector<double>::iterator Go::ParamCurveInt::getMesh ( ) [virtual]`

Return the geometric sample mesh for the parametric function.

Implements [Go::ParamGeomInt](#).

Reimplemented in [Go::SplineCurveInt](#).

29.326.3.17 `virtual int Go::ParamCurveInt::getMeshSize ( int dir ) [virtual]`

Return the size of the geometric sample mesh in the specified direction.

#### Parameters

<i>dir</i>	the parameter direction in question. Indexing starts at 0.
------------	------------------------------------------------------------

Implements [Go::ParamGeomInt](#).

Reimplemented in [Go::SplineCurveInt](#).

29.326.3.18 `virtual double Go::ParamCurveInt::getOptimizedConeAngle ( Point & axis1, Point & axis2 ) [virtual]`

We try to treat problems which will never result in a simple case by shrinking the domain slightly, resulting in smaller cones. This is useful for scenarios where the normals are parallel in a boundary point.

#### Parameters

<i>axis1</i>	first vector defining a projection plane
<i>axis2</i>	second vector defining a projection plane

#### Returns

The optimized cone angle

Implements [Go::ParamGeomInt](#).

Reimplemented in [Go::SplineCurveInt](#).

**29.326.3.19** `shared_ptr<ParamCurve> Go::ParamCurveInt::getParamCurve ( )`

Return pointer to the parametric curve defining this object.

**Returns**

Pointer to the parametric curve defining this object.

**29.326.3.20** `shared_ptr<const ParamCurve> Go::ParamCurveInt::getParamCurve ( ) const`

Return pointer to the parametric curve defining this object.

**Returns**

Pointer to the parametric curve defining this object.

**29.326.3.21** `virtual ParamCurveInt* Go::ParamCurveInt::getParamCurveInt ( ) [virtual]`

Return a pointer to this object.

**Returns**

Pointer to this object.

Reimplemented from [Go::ParamObjectInt](#).

**29.326.3.22** `virtual ParamSurfaceInt* Go::ParamCurveInt::getParamSurfaceInt ( ) [inline],[virtual]`

Return a NULL pointer (as this object is not of the correct type).

**Returns**

NULL pointer (as this object is not of the correct type).

Reimplemented from [Go::ParamObjectInt](#).

Definition at line 106 of file ParamCurveInt.h.

**29.326.3.23** `shared_ptr<ParamCurve> Go::ParamCurveInt::getParentParamCurve ( double & start, double & end )`

Return pointer to a subcurve of the parent curve for this object. If no such curve exist, we use the curve of this object instead.

## Parameters

<i>start</i>	the start parameter of the subcurve.
<i>end</i>	the end parameter of the subcurve.

## Returns

Pointer to a subcurve of the parent curve for this object.

29.326.3.24 `shared_ptr<ParamCurve> Go::ParamCurveInt::getParentParamCurve ( )`

Return pointer to the parent curve for this object. If no such curve exist, we use the curve of this object instead.

## Returns

Pointer to a subcurve of the parent curve for this object.

29.326.3.25 `virtual RotatedBox Go::ParamCurveInt::getRotatedBox ( std::vector< Point > & axis ) const [virtual]`

Create a box containing the geometric sample mesh in the input coordinate system.

## Parameters

<i>axis</i>	the axis defining the coordinate system. The size of vector axis is 2, the last point is given as the cross product axis[0]axis[1].
-------------	-------------------------------------------------------------------------------------------------------------------------------------

## Returns

The rotated box

Reimplemented in [Go::SplineCurveInt](#).

29.326.3.26 `virtual const SplineCurve* Go::ParamCurveInt::getSpline ( ) [inline],[virtual]`

Reimplemented in [Go::SplineCurveInt](#).

Definition at line 324 of file ParamCurveInt.h.

29.326.3.27 `virtual bool Go::ParamCurveInt::hasCriticalVals ( int pdir ) const [virtual]`

Return true if the object has any critical parameter values in the specified parameter direction.

## Parameters

<i>pdir</i>	the parameter direction in question. Indexing starts at 0.
-------------	------------------------------------------------------------

Implements [Go::ParamObjectInt](#).

**29.326.3.28** `virtual bool Go::ParamCurveInt::hasCriticalValsOrKnots ( int pardir ) const` [virtual]

Return true if the object has any critical parameter values or inner knots in the specified parameter direction.

#### Parameters

<i>pardir</i>	the parameter direction in question. Indexing starts at 0.
---------------	------------------------------------------------------------

Implements [Go::ParamObjectInt](#).

Reimplemented in [Go::SplineCurveInt](#).

**29.326.3.29** `virtual bool Go::ParamCurveInt::hasInnerKnots ( int pardir ) const` [virtual]

Return true if the object has any inner knots in the specified parameter direction.

#### Parameters

<i>pardir</i>	the parameter direction in question. Indexing starts at 0.
---------------	------------------------------------------------------------

Implements [Go::ParamObjectInt](#).

Reimplemented in [Go::SplineCurveInt](#).

**29.326.3.30** `virtual bool Go::ParamCurveInt::isDegenerate ( double epsge, int dir, double * par )` [virtual]

Verify whether the object is degenerate in the specified direction and parameter.

#### Parameters

<i>epsge</i>	the geometric tolerance defining degeneracy.
<i>dir</i>	the parameter direction in question.
<i>par</i>	the parameter in which to evaluate. Size of array should be equal to <a href="#">numParams()</a> .

#### Returns

True if the object is degenerate.

Implements [Go::ParamGeomInt](#).

**29.326.3.31** `virtual bool Go::ParamCurveInt::isSpline ( )` [virtual]

Verify whether the object is a spline.

**Returns**

True if the object is a spline.

Implements [Go::ParamGeomInt](#).

Reimplemented in [Go::SplineCurveInt](#).

29.326.3.32 `virtual int Go::ParamCurveInt::knotIntervalFuzzy ( double & t, double tol ) const` [virtual]

Find the knot interval for which t lies inside, moving the value t if it lies close to a knot.

**Parameters**

<i>t</i>	the parameter value
<i>tol</i>	the parametric tolerance deciding if the input parameter t should be moved.

Reimplemented in [Go::SplineCurveInt](#).

29.326.3.33 `virtual shared_ptr<ParamCurveInt> Go::ParamCurveInt::makeIntObject ( shared_ptr< ParamCurve > curve )` [virtual]

Return an intersection object for the input curve. This object is used as the parent for the intersection object.

**Parameters**

<i>curve</i>	the parametric curve defining the intersection object.
--------------	--------------------------------------------------------

Reimplemented in [Go::SplineCurveInt](#).

29.326.3.34 `virtual double Go::ParamCurveInt::nextSegmentVal ( double par, bool forward, double tol ) const` [virtual]

Return the value of the knot next to the input parameter par.

**Parameters**

<i>par</i>	the parameter value
<i>forward</i>	if true we return the closest knot to the right, otherwise the closest knot to the left.
<i>tol</i>	the tolerance to determine if <i>par</i> is already located on the start of the next segment.

**Returns**

The knot closest to the input parameter.

Reimplemented in [Go::SplineCurveInt](#).



29.326.3.35 `virtual int Go::ParamCurveInt::numParams ( ) const [virtual]`

The number of parameters in the object.

Implements [Go::ParamObjectInt](#).

29.326.3.36 `virtual double Go::ParamCurveInt::paramFromMesh ( int dir, int idx ) [virtual]`

Return the corresponding mesh parameter.

Parameters

<i>dir</i>	the parameter direction in question. Indexing starts at 0.
<i>idx</i>	the mesh idx in the specified direction. Indexing starts at 0.

Implements [Go::ParamGeomInt](#).

Reimplemented in [Go::SplineCurveInt](#).

29.326.3.37 `virtual void Go::ParamCurveInt::point ( Point & res, const double * par ) const [inline],[virtual]`

Evaluate the object in the input parameter.

Parameters

<i>res</i>	the <a href="#">Point</a> to be returned.
<i>par</i>	the parameter in which to evaluate. The size of the array should be equal to <a href="#">numParams()</a> .

Implements [Go::ParamObjectInt](#).

Definition at line 73 of file [ParamCurveInt.h](#).

29.326.3.38 `virtual void Go::ParamCurveInt::point ( std::vector< Point > & res, const double * par, int der, const bool * from_right = 0, double resolution = 1.0e-12 ) const [inline],[virtual]`

Evaluate the object in the input parameter, with the specified number of derivatives.

Parameters

<i>res</i>	the <a href="#">Point</a> to be returned.
<i>par</i>	the parameter in which to evaluate. The size of the array should be equal to <a href="#">numParams()</a> .
<i>der</i>	the number of derivatives to calculate.
<i>from_right</i>	if true the evaluation is to be performed from the right side of the parameter value.
<i>resolution</i>	tolerance used when determining whether parameters are located at special values of the parameter domain (in particular: knot values in case of spline objects).

Implements [Go::ParamObjectInt](#).

Definition at line 87 of file ParamCurveInt.h.

29.326.3.39 `virtual void Go::ParamCurveInt::setCriticalVal ( int pardir, double par )` [virtual]

Set info about subdivision value.

Reimplemented from [Go::ParamObjectInt](#).

29.326.3.40 `virtual double Go::ParamCurveInt::startParam ( int pardir ) const` [inline],[virtual]

Return the start parameter value in the specified direction.

#### Parameters

<i>pardir</i>	the parameter direction in question. Indexing starts at 0.
---------------	------------------------------------------------------------

Implements [Go::ParamObjectInt](#).

Definition at line 216 of file ParamCurveInt.h.

29.326.3.41 `double Go::ParamCurveInt::startparam ( ) const` [inline]

The start parameter of the curve.

#### Returns

The start parameter of the curve.

Definition at line 278 of file ParamCurveInt.h.

29.326.3.42 `virtual void Go::ParamCurveInt::subdivide ( int pardir, double par, std::vector< shared_ptr< ParamGeomInt >> & subdiv_objs, std::vector< shared_ptr< ParamGeomInt >> & bd_objs )` [virtual]

Subdivide the object in the specified parameter direction and parameter value.

#### Parameters

<i>pardir</i>	direction in which to subdivide. Indexing starts at 0.
<i>par</i>	parameter in which to subdivide.
<i>subdiv_objs</i>	The subparts of this object. Of the same geometric dimension as this object.
<i>bd_objs</i>	the boundaries between the returned <i>subdiv_objs</i> . Of geometric dimension 1 less than this object.

Implements [Go::ParamGeomInt](#).

## 29.326.4 Member Data Documentation

29.326.4.1 `shared_ptr<ParamCurve> Go::ParamCurveInt::curve_` [protected]

Definition at line 350 of file ParamCurveInt.h.

29.326.4.2 `int Go::ParamCurveInt::dim_` [protected]

Definition at line 352 of file ParamCurveInt.h.

29.326.4.3 `double Go::ParamCurveInt::length_` [mutable], [protected]

Definition at line 360 of file ParamCurveInt.h.

29.326.4.4 `bool Go::ParamCurveInt::lw_set_` [mutable], [protected]

Definition at line 359 of file ParamCurveInt.h.

29.326.4.5 `std::vector<double> Go::ParamCurveInt::mesh_` [mutable], [protected]

Definition at line 357 of file ParamCurveInt.h.

29.326.4.6 `std::vector<std::pair<double, int>> Go::ParamCurveInt::segment_` [protected]

Definition at line 354 of file ParamCurveInt.h.

29.326.4.7 `double Go::ParamCurveInt::wiggle_` [mutable], [protected]

Definition at line 360 of file ParamCurveInt.h.

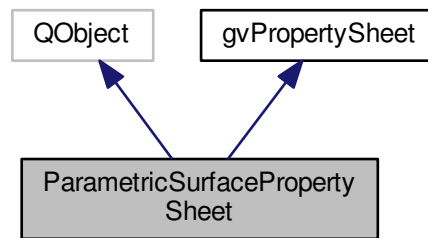
The documentation for this class was generated from the following file:

- [intersections/include/GoTools/intersections/ParamCurveInt.h](#)

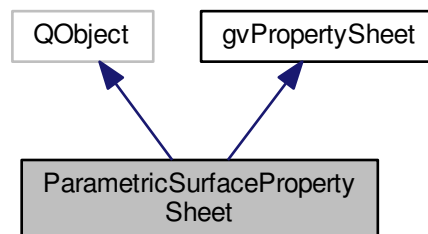
## 29.327 ParametricSurfacePropertySheet Class Reference

```
#include <ParametricSurfacePropertySheet.h>
```

Inheritance diagram for ParametricSurfacePropertySheet:



Collaboration diagram for ParametricSurfacePropertySheet:



### Public Slots

- void [apply](#) ()

### Public Member Functions

- [ParametricSurfacePropertySheet](#) ()
- [ParametricSurfacePropertySheet](#) (Go::ParametricSurfaceTesselator \*tess, [gvParametricSurfacePaintable](#) \*pable, shared\_ptr< Go::ParamSurface > &surf)
- virtual [~ParametricSurfacePropertySheet](#) ()
- virtual void [createSheet](#) (QWidget \*parent, [gvObserver](#) \*obs)

### 29.327.1 Detailed Description

Documentation ... etc

Definition at line 61 of file ParametricSurfacePropertySheet.h.

### 29.327.2 Constructor & Destructor Documentation

29.327.2.1 ParametricSurfacePropertySheet::ParametricSurfacePropertySheet ( ) [inline]

Definition at line 67 of file ParametricSurfacePropertySheet.h.

29.327.2.2 ParametricSurfacePropertySheet::ParametricSurfacePropertySheet ( Go::ParametricSurfaceTesselator \* tess, gvParametricSurfacePaintable \* pable, shared\_ptr< Go::ParamSurface > & surf ) [inline]

Definition at line 70 of file ParametricSurfacePropertySheet.h.

29.327.2.3 virtual ParametricSurfacePropertySheet::~~ParametricSurfacePropertySheet ( ) [virtual]

### 29.327.3 Member Function Documentation

29.327.3.1 void ParametricSurfacePropertySheet::apply ( ) [slot]

29.327.3.2 virtual void ParametricSurfacePropertySheet::createSheet ( QWidget \* parent, gvObserver \* obs ) [virtual]

Implements [gvPropertySheet](#).

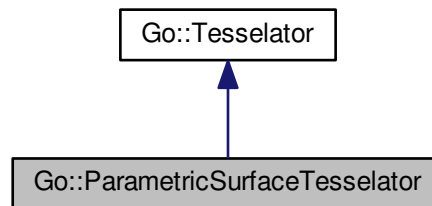
The documentation for this class was generated from the following file:

- [viewlib/include/GoTools/viewlib/ParametricSurfacePropertySheet.h](#)

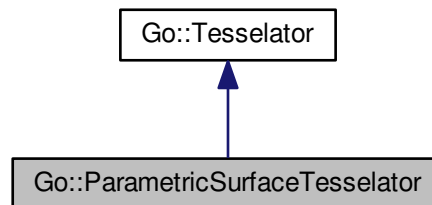
## 29.328 Go::ParametricSurfaceTesselator Class Reference

```
#include <ParametricSurfaceTesselator.h>
```

Inheritance diagram for Go::ParametricSurfaceTesselator:



Collaboration diagram for Go::ParametricSurfaceTesselator:



### Public Member Functions

- [ParametricSurfaceTesselator](#) (`const ParamSurface &surf`)
- virtual `~ParametricSurfaceTesselator` ()
- virtual void `tesselate` ()  
*Perform tessellation.*
- `shared_ptr< GenericTriMesh > getMesh` ()  
*Fetch the resulting mesh.*
- void `changeRes` (`int n`, `int m`)  
*Change mesh size.*
- void `getRes` (`int &n`, `int &m`)  
*Fetch info about mesh size.*

### 29.328.1 Detailed Description

[ParametricSurfaceTesselator](#): create a mesh for a possibly trimmed surface with a suitable triangulation.

Definition at line 57 of file `ParametricSurfaceTesselator.h`.

### 29.328.2 Constructor & Destructor Documentation

29.328.2.1 `Go::ParametricSurfaceTesselator::ParametricSurfaceTesselator ( const ParamSurface & surf ) [inline]`

Constructor. Surface and mesh size are given. The mesh size relates to the underlying surface in the case of bounded surfaces.

Definition at line 62 of file `ParametricSurfaceTesselator.h`.

29.328.2.2 `virtual Go::ParametricSurfaceTesselator::~~ParametricSurfaceTesselator ( ) [virtual]`

### 29.328.3 Member Function Documentation

29.328.3.1 `void Go::ParametricSurfaceTesselator::changeRes ( int n, int m )`

Change mesh size.

29.328.3.2 `shared_ptr<GenericTriMesh> Go::ParametricSurfaceTesselator::getMesh ( ) [inline]`

Fetch the resulting mesh.

Definition at line 78 of file `ParametricSurfaceTesselator.h`.

29.328.3.3 `void Go::ParametricSurfaceTesselator::getRes ( int & n, int & m ) [inline]`

Fetch info about mesh size.

Definition at line 87 of file `ParametricSurfaceTesselator.h`.

29.328.3.4 `virtual void Go::ParametricSurfaceTesselator::tesselate ( ) [virtual]`

Perform tessellation.

Implements [Go::Tesselator](#).

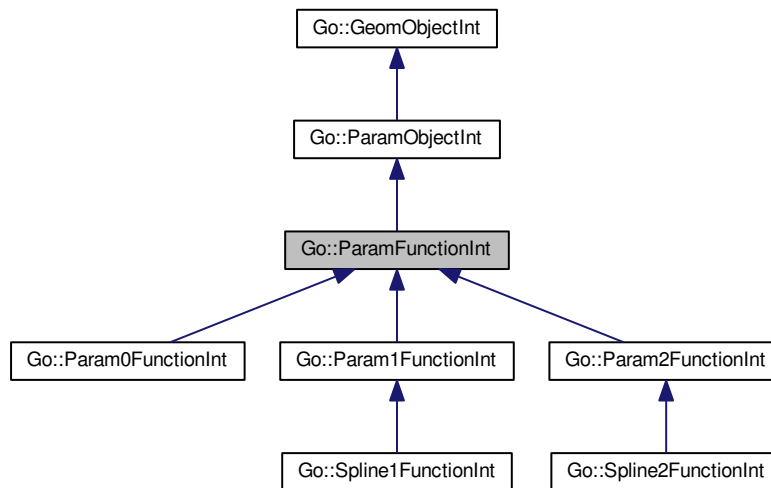
The documentation for this class was generated from the following file:

- `gotools-core/include/GoTools/tesselator/ParametricSurfaceTesselator.h`

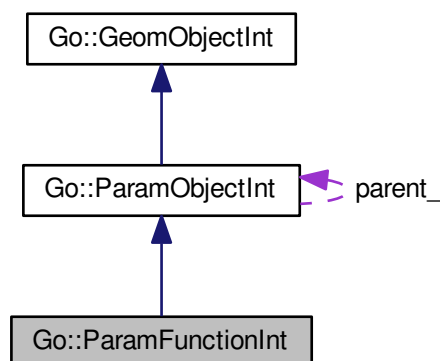
## 29.329 Go::ParamFunctionInt Class Reference

```
#include <ParamFunctionInt.h>
```

Inheritance diagram for Go::ParamFunctionInt:



Collaboration diagram for Go::ParamFunctionInt:



### Public Member Functions

- virtual `~ParamFunctionInt` ()  
*Destructor.*
- virtual `Param0FunctionInt * getParam0FunctionInt` ()



- virtual [Param1FunctionInt](#) \* [getParam1FunctionInt](#) ()
- virtual [Param2FunctionInt](#) \* [getParam2FunctionInt](#) ()
- virtual void [subdivide](#) (int pardir, double par, std::vector< shared\_ptr< [ParamFunctionInt](#) > > &subdiv\_objs, std::vector< shared\_ptr< [ParamFunctionInt](#) > > &bd\_objs)=0
- virtual bool [monotone](#) ([Point](#) &dir, double tol=1.0e-15) const =0
- virtual void [getBoundaryObjects](#) (std::vector< shared\_ptr< [BoundaryFunctionInt](#) > > &bd\_objs)=0
- virtual int [getMeshSize](#) (int dir)=0
- virtual double [paramFromMesh](#) (int dir, int idx)=0
- virtual std::vector< double >::iterator [getMesh](#) ()=0

*Return the geometric sample mesh for the parametric function.*

## Protected Attributes

- std::vector< shared\_ptr< [BoundaryFunctionInt](#) > > [boundary\\_obj\\_](#)

### 29.329.1 Detailed Description

This class is a base class providing an interface to the parametric "intersection objects" with 1-dimensional range.

Definition at line 61 of file [ParamFunctionInt.h](#).

### 29.329.2 Constructor & Destructor Documentation

**29.329.2.1** virtual [Go::ParamFunctionInt::~ParamFunctionInt](#) ( ) `[inline]`, `[virtual]`

Destructor.

Definition at line 64 of file [ParamFunctionInt.h](#).

### 29.329.3 Member Function Documentation

**29.329.3.1** virtual void [Go::ParamFunctionInt::getBoundaryObjects](#) ( std::vector< shared\_ptr< [BoundaryFunctionInt](#) > > & *bd\_objs* ) `[pure virtual]`

Return the boundary objects of this object.

Parameters

<i>bd_objs</i>	the boundary objects of this object.
----------------	--------------------------------------

Implemented in [Go::Param2FunctionInt](#), [Go::Param1FunctionInt](#), [Go::Param0FunctionInt](#), and [Go::Spline2FunctionInt](#).

**29.329.3.2** virtual std::vector<double>::iterator [Go::ParamFunctionInt::getMesh](#) ( ) `[pure virtual]`

Return the geometric sample mesh for the parametric function.

Implemented in [Go::Param2FunctionInt](#), [Go::Param1FunctionInt](#), [Go::Param0FunctionInt](#), and [Go::Spline1↔FunctionInt](#).

**29.329.3.3** `virtual int Go::ParamFunctionInt::getMeshSize ( int dir ) [pure virtual]`

Return the size of the geometric sample mesh in the specified direction.

#### Parameters

<i>dir</i>	the parameter direction in question. Indexing starts at 0.
------------	------------------------------------------------------------

Implemented in [Go::Param2FunctionInt](#), [Go::Param1FunctionInt](#), [Go::Param0FunctionInt](#), and [Go::Spline1↔FunctionInt](#).

**29.329.3.4** `virtual Param0FunctionInt* Go::ParamFunctionInt::getParam0FunctionInt ( ) [inline], [virtual]`

If the object is not of type [Param0FunctionInt](#), return NULL, otherwise return the object.

Reimplemented in [Go::Param0FunctionInt](#).

Definition at line 68 of file ParamFunctionInt.h.

**29.329.3.5** `virtual Param1FunctionInt* Go::ParamFunctionInt::getParam1FunctionInt ( ) [inline], [virtual]`

If the object is not of type [Param1FunctionInt](#), return NULL, otherwise return the object.

Reimplemented in [Go::Param1FunctionInt](#).

Definition at line 73 of file ParamFunctionInt.h.

**29.329.3.6** `virtual Param2FunctionInt* Go::ParamFunctionInt::getParam2FunctionInt ( ) [inline], [virtual]`

If the object is not of type [Param2FunctionInt](#), return NULL, otherwise return the object.

Reimplemented in [Go::Param2FunctionInt](#).

Definition at line 78 of file ParamFunctionInt.h.

**29.329.3.7** `virtual bool Go::ParamFunctionInt::monotone ( Point & dir, double tol = 1.0e-15 ) const [pure virtual]`

Returns true if the object is monotone in any direction.

#### Parameters

<i>dir</i>	the direction in which the object is monotone. Relevant only if object is monotone.
------------	-------------------------------------------------------------------------------------

Implemented in [Go::Param2FunctionInt](#), [Go::Param1FunctionInt](#), [Go::Param0FunctionInt](#), [Go::Spline1FunctionInt](#), and [Go::Spline2FunctionInt](#).

**29.329.3.8** `virtual double Go::ParamFunctionInt::paramFromMesh ( int dir, int idx ) [pure virtual]`

Return the corresponding mesh parameter.

#### Parameters

<i>dir</i>	the parameter direction in question. Indexing starts at 0.
<i>idx</i>	the mesh idx in the specified direction. Indexing starts at 0.

Implemented in [Go::Param2FunctionInt](#), [Go::Param1FunctionInt](#), [Go::Param0FunctionInt](#), and [Go::Spline1↔FunctionInt](#).

**29.329.3.9** `virtual void Go::ParamFunctionInt::subdivide ( int pardir, double par, std::vector< shared_ptr< ParamFunctionInt > > & subdiv_objs, std::vector< shared_ptr< ParamFunctionInt > > & bd_objs ) [pure virtual]`

Subdivide the object in the specified parameter direction and parameter value.

#### Parameters

<i>pardir</i>	direction in which to subdivide. Indexing starts at 0.
<i>par</i>	parameter in which to subdivide.
<i>subdiv_objs</i>	The subparts of this object. Of the same geometric dimension as this object.
<i>bd_objs</i>	the boundaries between the returned <i>subdiv_objs</i> . Of geometric dimension 1 less than this object.

Implemented in [Go::Param2FunctionInt](#), [Go::Param1FunctionInt](#), and [Go::Param0FunctionInt](#).

## 29.329.4 Member Data Documentation

**29.329.4.1** `std::vector<shared_ptr<BoundaryFunctionInt> > Go::ParamFunctionInt::boundary_obj_ [protected]`

Definition at line 125 of file ParamFunctionInt.h.

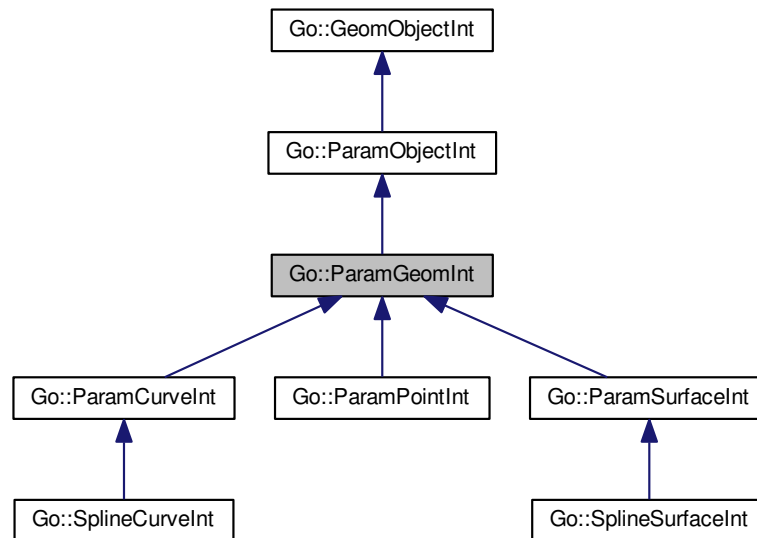
The documentation for this class was generated from the following file:

- intersections/include/GoTools/intersections/[ParamFunctionInt.h](#)

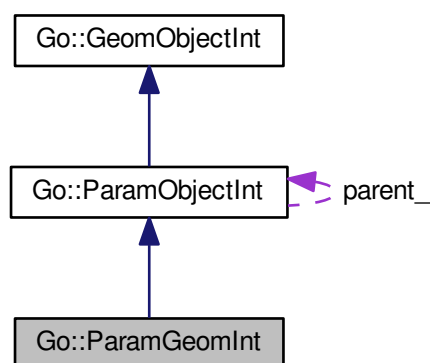
## 29.330 Go::ParamGeomInt Class Reference

```
#include <ParamGeomInt.h>
```

Inheritance diagram for Go::ParamGeomInt:



Collaboration diagram for Go::ParamGeomInt:



### Public Member Functions

- [ParamGeomInt](#) ([ParamGeomInt](#) \*parent=0)

- virtual `~ParamGeomInt ()`  
*Destructor.*
- virtual `int dimension () const =0`  
*The dimension of the geometric space.*
- virtual `int checkPeriodicity (int paddir) const`
- virtual `void subdivide (int paddir, double par, std::vector< shared_ptr< ParamGeomInt > > &subdiv_objs, std::vector< shared_ptr< ParamGeomInt > > &bd_objs)=0`
- virtual `CompositeBox compositeBox () const =0`
- virtual `DirectionCone directionCone () const =0`
- `bool coneLargerThanPi ()`  
*Check if the associated direction cone has an angle larger than pi.*
- virtual `DirectionCone reducedDirectionCone (bool reduce_at_bd[4], double epsge) const`
- virtual `void getBoundaryObjects (std::vector< shared_ptr< BoundaryGeomInt > > &bd_objs)=0`
- virtual `int getMeshSize (int dir)=0`
- virtual `double paramFromMesh (int dir, int idx)=0`
- virtual `std::vector< double >::iterator getMesh ()=0`  
*Return the geometric sample mesh for the parametric function.*
- `int nmbBdObj () const`
- `BoundaryGeomInt * getBoundaryObject (int bd_idx) const`
- virtual `int isDegenerate (double epsge, int dir)`
- virtual `bool isDegenerate (double epsge, int dir, double *par)=0`
- virtual `bool isLinear (double epsge)`  
*Check the linearity of the object.*
- virtual `bool isSpline ()=0`
- virtual `double getOptimizedConeAngle (Point &axis1, Point &axis2)=0`

## Protected Attributes

- `std::vector< shared_ptr< BoundaryGeomInt > > boundary_obj_`

### 29.330.1 Detailed Description

This class is a base class providing an interface to the parametric "intersection objects".

Definition at line 62 of file ParamGeomInt.h.

### 29.330.2 Constructor & Destructor Documentation

29.330.2.1 `Go::ParamGeomInt::ParamGeomInt ( ParamGeomInt * parent = 0 ) [inline]`

Constructor

Parameters

<code>parent</code>	the parent to this intersection object.
---------------------	-----------------------------------------

Definition at line 66 of file ParamGeomInt.h.

29.330.2.2 `virtual Go::ParamGeomInt::~~ParamGeomInt ( ) [inline],[virtual]`

Destructor.

Definition at line 69 of file ParamGeomInt.h.

### 29.330.3 Member Function Documentation

29.330.3.1 `virtual int Go::ParamGeomInt::checkPeriodicity ( int pardir ) const [inline],[virtual]`

Check if the object is periodic. Analyze periodicity of curve based on number of repeating knots and control points. The return value is -1 if the curve ends are disjoint, otherwise k if cv is  $C^k$  continuous. These are sufficient but not necessary conditions for periodicity, so it is possible that a higher degree of periodicity exists. Should not be called on this layer, should be overruled by inherited class.

#### Parameters

<i>pardir</i>	the parameter direction in question.
---------------	--------------------------------------

#### Returns

-1 if the curve ends are disjoint, or k if the curve is proven to be  $C^k$  continuous.

Reimplemented in [Go::ParamSurfaceInt](#), [Go::ParamCurveInt](#), [Go::SplineSurfaceInt](#), and [Go::SplineCurveInt](#).

Definition at line 84 of file ParamGeomInt.h.

29.330.3.2 `virtual CompositeBox Go::ParamGeomInt::compositeBox ( ) const [pure virtual]`

Return the [CompositeBox](#) for the parametric object.

#### Returns

The compositeBox for the parametric object.

Implements [Go::ParamObjectInt](#).

Implemented in [Go::ParamSurfaceInt](#), [Go::ParamCurveInt](#), and [Go::ParamPointInt](#).

29.330.3.3 `bool Go::ParamGeomInt::coneLargerThanPi ( )`

Check if the associated direction cone has an angle larger than pi.

29.330.3.4 `virtual int Go::ParamGeomInt::dimension ( ) const [pure virtual]`

The dimension of the geometric space.

Implemented in [Go::ParamSurfaceInt](#), [Go::ParamCurveInt](#), and [Go::ParamPointInt](#).

29.330.3.5 virtual **DirectionCone** Go::ParamGeomInt::directionCone ( ) const [pure virtual]

A cone which contains all normals of the object.

#### Returns

A cone which contains all normals of the object.

Implemented in [Go::ParamSurfaceInt](#), [Go::ParamCurveInt](#), [Go::ParamPointInt](#), and [Go::SplineSurfaceInt](#).

29.330.3.6 **BoundaryGeomInt\*** Go::ParamGeomInt::getBoundaryObject ( int *bd\_idx* ) const [inline]

Return the specified boundary object.

#### Parameters

<i>bd_idx</i>	index of the boundary object.
---------------	-------------------------------

#### Returns

The boundary object.

Definition at line 150 of file ParamGeomInt.h.

29.330.3.7 virtual void Go::ParamGeomInt::getBoundaryObjects ( std::vector< shared\_ptr< **BoundaryGeomInt** > > & *bd\_objs* ) [pure virtual]

Return the boundary objects of this object.

#### Parameters

<i>bd_objs</i>	the boundary objects of this object.
----------------	--------------------------------------

Implemented in [Go::ParamSurfaceInt](#), [Go::ParamCurveInt](#), [Go::ParamPointInt](#), and [Go::SplineSurfaceInt](#).

29.330.3.8 virtual std::vector<double>::iterator Go::ParamGeomInt::getMesh ( ) [pure virtual]

Return the geometric sample mesh for the parametric function.

Implemented in [Go::ParamSurfaceInt](#), [Go::ParamCurveInt](#), [Go::ParamPointInt](#), [Go::SplineSurfaceInt](#), and [Go::SplineCurveInt](#).

29.330.3.9 virtual int Go::ParamGeomInt::getMeshSize ( int *dir* ) [pure virtual]

Return the size of the geometric sample mesh in the specified direction.

## Parameters

<i>dir</i>	the parameter direction in question. Indexing starts at 0.
------------	------------------------------------------------------------

Implemented in [Go::ParamSurfaceInt](#), [Go::ParamCurveInt](#), [Go::ParamPointInt](#), [Go::SplineSurfaceInt](#), and [Go::SplineCurveInt](#).

29.330.3.10 `virtual double Go::ParamGeomInt::getOptimizedConeAngle ( Point & axis1, Point & axis2 ) [pure virtual]`

We try to treat problems which will never result in a simple case by shrinking the domain slightly, resulting in smaller cones. This is useful for scenarios where the normals are parallel in a boundary point.

Implemented in [Go::ParamSurfaceInt](#), [Go::ParamCurveInt](#), [Go::ParamPointInt](#), [Go::SplineSurfaceInt](#), and [Go::SplineCurveInt](#).

29.330.3.11 `virtual int Go::ParamGeomInt::isDegenerate ( double epsge, int dir ) [inline],[virtual]`

Return true if the object is degenerate in the specified direction. Specialized for surface

## Parameters

<i>epsge</i>	the geometric tolerance defining degeneracy.
<i>dir</i>	the parameter direction in question.

Reimplemented in [Go::ParamSurfaceInt](#).

Definition at line 160 of file ParamGeomInt.h.

29.330.3.12 `virtual bool Go::ParamGeomInt::isDegenerate ( double epsge, int dir, double * par ) [pure virtual]`

Return true if the object is degenerate in the specified direction and parameter.

## Parameters

<i>epsge</i>	the geometric tolerance defining degeneracy.
<i>dir</i>	the parameter direction in question.
<i>par</i>	the parameter in which to evaluate. Size of array should be equal to <a href="#">numParams()</a> .

Reimplemented from [Go::ParamObjectInt](#).

Implemented in [Go::ParamSurfaceInt](#), [Go::ParamCurveInt](#), and [Go::ParamPointInt](#).

29.330.3.13 `virtual bool Go::ParamGeomInt::isLinear ( double epsge ) [virtual]`

Check the linearity of the object.



29.330.3.14 `virtual bool Go::ParamGeomInt::isSpline ( ) [pure virtual]`

Return whether or not the object is a spline.

#### Returns

True if the object is a spline.

Implemented in [Go::ParamSurfaceInt](#), [Go::ParamCurveInt](#), [Go::ParamPointInt](#), [Go::SplineSurfaceInt](#), and [Go::SplineCurveInt](#).

29.330.3.15 `int Go::ParamGeomInt::nmbBdObj ( ) const [inline]`

Return the number of boundary objects.

#### Returns

The number of boundary objects.

Definition at line 144 of file `ParamGeomInt.h`.

29.330.3.16 `virtual double Go::ParamGeomInt::paramFromMesh ( int dir, int idx ) [pure virtual]`

Return the corresponding mesh parameter.

#### Parameters

<i>dir</i>	the parameter direction in question. Indexing starts at 0.
<i>idx</i>	the mesh idx in the specified direction. Indexing starts at 0.

Implemented in [Go::ParamSurfaceInt](#), [Go::ParamCurveInt](#), [Go::ParamPointInt](#), [Go::SplineSurfaceInt](#), and [Go::SplineCurveInt](#).

29.330.3.17 `virtual DirectionCone Go::ParamGeomInt::reducedDirectionCone ( bool reduce_at_bd[4], double epsge ) const [virtual]`

Try to make a smaller direction cone.

#### Parameters

<i>reduce_at_bd</i>	indicates which boundaries the reduction attempt should be made
<i>epsge</i>	the geometric tolerancedefining degeneracy

#### Returns

A direction cone. This is either the same cone or a smaller one.

Reimplemented in [Go::SplineSurfaceInt](#).

29.330.3.18 `virtual void Go::ParamGeomInt::subdivide ( int pardir, double par, std::vector< shared_ptr< ParamGeomInt >> & subdiv_objs, std::vector< shared_ptr< ParamGeomInt >> & bd_objs )` [pure virtual]

Subdivide the object in the specified parameter direction and parameter value.

#### Parameters

<i>pardir</i>	direction in which to subdivide. Indexing starts at 0.
<i>par</i>	parameter in which to subdivide.
<i>subdiv_objs</i>	The subparts of this object. Of the same geometric dimension as this object.
<i>bd_objs</i>	the boundaries between the returned <i>subdiv_objs</i> . Of geometric dimension 1 less than this object.

Implemented in [Go::ParamSurfaceInt](#), [Go::ParamCurveInt](#), and [Go::ParamPointInt](#).

### 29.330.4 Member Data Documentation

29.330.4.1 `std::vector<shared_ptr<BoundaryGeomInt>> Go::ParamGeomInt::boundary_obj_` [protected]

Definition at line 185 of file `ParamGeomInt.h`.

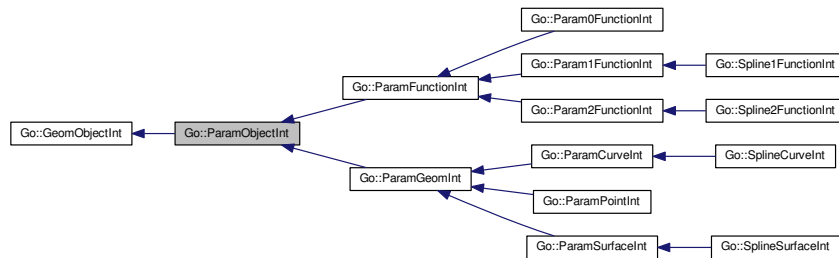
The documentation for this class was generated from the following file:

- [intersections/include/GoTools/intersections/ParamGeomInt.h](#)

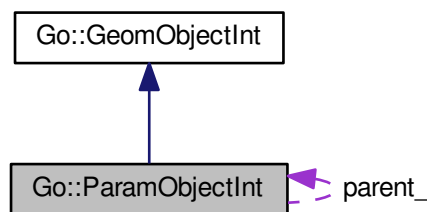
### 29.331 Go::ParamObjectInt Class Reference

```
#include <ParamObjectInt.h>
```

Inheritance diagram for Go::ParamObjectInt:



Collaboration diagram for Go::ParamObjectInt:



## Public Member Functions

- [ParamObjectInt](#) ([ParamObjectInt](#) \*parent=0)
- virtual [~ParamObjectInt](#) ()
  - Destructor.*
- virtual [ParamPointInt](#) \* [getParamPointInt](#) ()
- virtual [ParamCurveInt](#) \* [getParamCurveInt](#) ()
- virtual [ParamSurfaceInt](#) \* [getParamSurfaceInt](#) ()
- virtual void [point](#) ([Point](#) &pt, const double \*tpar) const =0
- virtual void [point](#) (std::vector< [Point](#) > &pt, const double \*tpar, int derivs, const bool \*from\_right=0, double resolution=1.0e-12) const =0
- virtual int [numParams](#) () const =0
  - The number of parameters in the object.*
- virtual void [getLengthAndWiggle](#) (double \*length, double \*wiggle)=0
- virtual bool [hasInnerKnots](#) (int paddir) const =0
- virtual bool [hasCriticalVals](#) (int paddir) const =0
- virtual void [setCriticalVal](#) (int paddir, double par)
  - Set info about subdivision value.*
- virtual bool [hasCriticalValsOrKnots](#) (int paddir) const =0
- virtual bool [canDivide](#) (int paddir)=0

- virtual `bool canDivideTinyTriang` (int paddir)
- virtual `std::vector< double > getCriticalVals` (int paddir) `const =0`
- virtual `std::vector< double > getInnerKnotVals` (int paddir, `bool sort=false`) `const =0`
- virtual `std::vector< double > getCriticalValsAndKnots` (int paddir) `const =0`
- virtual `double startParam` (int paddir) `const =0`
- virtual `double endParam` (int paddir) `const =0`
- virtual `bool isDegenerate` (double epsge, int dir, `double *par`)
- virtual `bool boundaryPoint` (`const double *par`, `double eps`) `const =0`
- void `setParent` (`ParamObjectInt *parent`)
- `ParamObjectInt * getParent` () `const`  
*Return the parent of this object.*
- virtual `CompositeBox compositeBox` () `const =0`  
*Return the `CompositeBox` for the parametric object.*
- `const ParamObjectInt * getSameTypeAncestor` () `const`
- virtual `bool inCorner` (`const double *par`, `double epspar`) `const`

## Protected Attributes

- `ParamObjectInt * parent_`

### 29.331.1 Detailed Description

This class is a base class providing an interface to the "intersection objects". The object is parametric.

Definition at line 61 of file `ParamObjectInt.h`.

### 29.331.2 Constructor & Destructor Documentation

29.331.2.1 `Go::ParamObjectInt::ParamObjectInt ( ParamObjectInt * parent = 0 )` `[inline]`

Constructor.

#### Parameters

<code>parent</code>	is a parametric object for which this object constitutes only a subpart (0 if there is no parent).
---------------------	----------------------------------------------------------------------------------------------------

Definition at line 66 of file `ParamObjectInt.h`.

29.331.2.2 `virtual Go::ParamObjectInt::~~ParamObjectInt ( )` `[inline]`, `[virtual]`

Destructor.

Definition at line 69 of file `ParamObjectInt.h`.

### 29.331.3 Member Function Documentation

29.331.3.1 `virtual bool Go::ParamObjectInt::boundaryPoint ( const double * par, double eps ) const` [pure virtual]

Return true if the specified point lies within *eps* from the boundary.

#### Parameters

<i>par</i>	the parameter in which to evaluate. Size of array should be equal to <a href="#">numParams()</a> .
<i>eps</i>	the tolerance defining boundary neighbourhood.

Implemented in [Go::ParamSurfaceInt](#), [Go::Param2FunctionInt](#), [Go::ParamCurveInt](#), [Go::Param1FunctionInt](#), [Go::ParamPointInt](#), and [Go::Param0FunctionInt](#).

29.331.3.2 `virtual bool Go::ParamObjectInt::canDivide ( int pardir )` [pure virtual]

Return true if we are allowed to divide in the specified parameter direction.

#### Parameters

<i>pardir</i>	the parameter direction in question.
---------------	--------------------------------------

Implemented in [Go::ParamSurfaceInt](#), [Go::Param2FunctionInt](#), [Go::ParamCurveInt](#), [Go::Param1FunctionInt](#), [Go::Param0FunctionInt](#), and [Go::ParamPointInt](#).

29.331.3.3 `virtual bool Go::ParamObjectInt::canDivideTinyTriang ( int pardir )` [inline],[virtual]

Reimplemented in [Go::ParamSurfaceInt](#).

Definition at line 153 of file [ParamObjectInt.h](#).

29.331.3.4 `virtual CompositeBox Go::ParamObjectInt::compositeBox ( ) const` [pure virtual]

Return the [CompositeBox](#) for the parametric object.

Implemented in [Go::ParamSurfaceInt](#), [Go::Param2FunctionInt](#), [Go::Param1FunctionInt](#), [Go::ParamCurveInt](#), [Go::ParamPointInt](#), [Go::Param0FunctionInt](#), and [Go::ParamGeomInt](#).

29.331.3.5 `virtual double Go::ParamObjectInt::endParam ( int pardir ) const` [pure virtual]

Return the end parameter in the specified direction.

#### Parameters

<i>pardir</i>	the parameter direction in question. Indexing starts at 0.
---------------	------------------------------------------------------------

Implemented in [Go::ParamSurfaceInt](#), [Go::Param2FunctionInt](#), [Go::ParamCurveInt](#), [Go::Param1FunctionInt](#), [Go::ParamPointInt](#), [Go::Param0FunctionInt](#), and [Go::Spline2FunctionInt](#).

29.331.3.6 `virtual std::vector<double> Go::ParamObjectInt::getCriticalVals ( int pardir ) const` [pure virtual]

Return the critical parameter values in the specified direction.

#### Parameters

<i>pardir</i>	the parameter direction in question. Indexing starts at 0.
---------------	------------------------------------------------------------

Implemented in [Go::ParamSurfaceInt](#), [Go::Param2FunctionInt](#), [Go::ParamCurveInt](#), [Go::Param1FunctionInt](#), [Go::ParamPointInt](#), and [Go::Param0FunctionInt](#).

29.331.3.7 `virtual std::vector<double> Go::ParamObjectInt::getCriticalValsAndKnots ( int pardir ) const` [pure virtual]

Return the critical parameter values and inner knots for object.

#### Parameters

<i>pardir</i>	the parameter direction in question. Indexing starts at 0.
---------------	------------------------------------------------------------

Implemented in [Go::ParamSurfaceInt](#), [Go::Param2FunctionInt](#), [Go::ParamCurveInt](#), [Go::Param1FunctionInt](#), [Go::ParamPointInt](#), [Go::Param0FunctionInt](#), [Go::SplineSurfaceInt](#), [Go::Spline1FunctionInt](#), [Go::SplineCurveInt](#), and [Go::Spline2FunctionInt](#).

29.331.3.8 `virtual std::vector<double> Go::ParamObjectInt::getInnerKnotVals ( int pardir, bool sort = false ) const` [pure virtual]

Return the inner knot values in the specified direction.

#### Parameters

<i>pardir</i>	the parameter direction in question. Indexing starts at 0.
<i>sort</i>	the returned values may be sorted by the function.

Implemented in [Go::ParamSurfaceInt](#), [Go::Param2FunctionInt](#), [Go::ParamCurveInt](#), [Go::Param1FunctionInt](#), [Go::ParamPointInt](#), [Go::Param0FunctionInt](#), [Go::SplineSurfaceInt](#), [Go::Spline1FunctionInt](#), [Go::SplineCurveInt](#), and [Go::Spline2FunctionInt](#).

29.331.3.9 `virtual void Go::ParamObjectInt::getLengthAndWiggle ( double * length, double * wiggle )` [pure virtual]

Return an estimate on the size and wiggle of the object.

## Parameters

<i>length</i>	the approximative length of the object in the corresponding parameter directions. The size of the array should be equal to <a href="#">numParams()</a> .
<i>wiggle</i>	a scalar representing the wiggle of the object in the corresponding parameter directions. The size of the array should be equal to <a href="#">numParams()</a> .

Implemented in [Go::ParamSurfaceInt](#), [Go::Param2FunctionInt](#), [Go::ParamCurveInt](#), [Go::Param1FunctionInt](#), [Go::ParamPointInt](#), and [Go::Param0FunctionInt](#).

**29.331.3.10** `virtual ParamCurveInt* Go::ParamObjectInt::getParamCurveInt ( ) [inline],[virtual]`

Return pointer to a parametric intersection curve.

## Returns

Pointer to a parametric intersection curve. If that is not the class type for this object a NULL pointer is returned.

Reimplemented in [Go::ParamCurveInt](#).

Definition at line 80 of file ParamObjectInt.h.

**29.331.3.11** `virtual ParamPointInt* Go::ParamObjectInt::getParamPointInt ( ) [inline],[virtual]`

Return pointer to a parametric intersection point.

## Returns

Pointer to a parametric intersection point. If that is not the class type for this object a NULL pointer is returned.

Reimplemented in [Go::ParamPointInt](#).

Definition at line 74 of file ParamObjectInt.h.

**29.331.3.12** `virtual ParamSurfaceInt* Go::ParamObjectInt::getParamSurfaceInt ( ) [inline],[virtual]`

Return pointer to a parametric intersection surface.

## Returns

Pointer to a parametric intersection surface. If that is not the class type for this object a NULL pointer is returned.

Reimplemented in [Go::ParamCurveInt](#), and [Go::ParamSurfaceInt](#).

Definition at line 87 of file ParamObjectInt.h.

29.331.3.13 **ParamObjectInt\*** `Go::ParamObjectInt::getParent ( ) const` `[inline]`

Return the parent of this object.

Definition at line 208 of file ParamObjectInt.h.

29.331.3.14 **const ParamObjectInt\*** `Go::ParamObjectInt::getSameTypeAncestor ( ) const` `[inline]`

Return an ancestor of the same type. If the object has no parent, or if its parent has a higher number of parameters, return a pointer to itself. Else, call this function recursively on the parent.

Definition at line 218 of file ParamObjectInt.h.

29.331.3.15 **virtual bool** `Go::ParamObjectInt::hasCriticalVals ( int pardir ) const` `[pure virtual]`

Return true if the object has any critical parameter values in the specified parameter direction.

Parameters

<i>pardir</i>	the parameter direction in question. Indexing starts at 0.
---------------	------------------------------------------------------------

Implemented in [Go::ParamSurfaceInt](#), [Go::Param2FunctionInt](#), [Go::ParamCurveInt](#), [Go::Param1FunctionInt](#), [Go::ParamPointInt](#), and [Go::Param0FunctionInt](#).

29.331.3.16 **virtual bool** `Go::ParamObjectInt::hasCriticalValsOrKnots ( int pardir ) const` `[pure virtual]`

Return true if the object has any critical parameter values or inner knots in the specified parameter direction.

Parameters

<i>pardir</i>	the parameter direction in question. Indexing starts at 0.
---------------	------------------------------------------------------------

Implemented in [Go::ParamSurfaceInt](#), [Go::Param2FunctionInt](#), [Go::ParamCurveInt](#), [Go::Param1FunctionInt](#), [Go::ParamPointInt](#), [Go::Param0FunctionInt](#), [Go::SplineSurfaceInt](#), [Go::Spline1FunctionInt](#), [Go::SplineCurveInt](#), and [Go::Spline2FunctionInt](#).

29.331.3.17 **virtual bool** `Go::ParamObjectInt::hasInnerKnots ( int pardir ) const` `[pure virtual]`

Return true if the object has any inner knots in the specified parameter direction.

Parameters

<i>pardir</i>	the parameter direction in question. Indexing starts at 0.
---------------	------------------------------------------------------------

Implemented in [Go::ParamSurfaceInt](#), [Go::Param2FunctionInt](#), [Go::ParamCurveInt](#), [Go::Param1FunctionInt](#), [Go::ParamPointInt](#), [Go::Param0FunctionInt](#), [Go::SplineSurfaceInt](#), [Go::Spline1FunctionInt](#), [Go::SplineCurveInt](#), and [Go::Spline2FunctionInt](#).



[Go::Spline2FunctionInt](#).

29.331.3.18 `virtual bool Go::ParamObjectInt::inCorner ( const double * par, double epspar ) const [inline], [virtual]`

Check if parameter point lies close to a corner of the parameter domain.

#### Parameters

<i>par</i>	the input parameter point.
<i>epspar</i>	the parametric tolerance defining the neighbourhood.

Reimplemented in [Go::ParamSurfaceInt](#).

Definition at line 232 of file ParamObjectInt.h.

29.331.3.19 `virtual bool Go::ParamObjectInt::isDegenerate ( double epsge, int dir, double * par ) [inline], [virtual]`

Return true if the object is degenerate in the specified direction and parameter.

#### Parameters

<i>epsge</i>	the geometric tolerance defining degeneracy.
<i>dir</i>	the parameter direction in question.
<i>par</i>	the parameter in which to evaluate. Size of array should be equal to <a href="#">numParams()</a> .

Reimplemented in [Go::ParamSurfaceInt](#), [Go::Param2FunctionInt](#), [Go::ParamCurveInt](#), [Go::Param1FunctionInt](#), [Go::ParamPointInt](#), and [Go::ParamGeomInt](#).

Definition at line 192 of file ParamObjectInt.h.

29.331.3.20 `virtual int Go::ParamObjectInt::numParams ( ) const [pure virtual]`

The number of parameters in the object.

Implemented in [Go::ParamSurfaceInt](#), [Go::ParamCurveInt](#), [Go::Param1FunctionInt](#), [Go::Param2FunctionInt](#), [Go::ParamPointInt](#), and [Go::Param0FunctionInt](#).

29.331.3.21 `virtual void Go::ParamObjectInt::point ( Point & pt, const double * tpar ) const [pure virtual]`

Evaluate the object in the input parameter.

#### Parameters

<i>pt</i>	the <a href="#">Point</a> to be returned.
<i>tpar</i>	the parameter in which to evaluate. The size of the array should be equal to <a href="#">numParams()</a> .

Implemented in [Go::Param2FunctionInt](#), [Go::Param1FunctionInt](#), [Go::ParamPointInt](#), [Go::ParamSurfaceInt](#), [Go::ParamCurveInt](#), and [Go::Param0FunctionInt](#).

**29.331.3.22** `virtual void Go::ParamObjectInt::point ( std::vector< Point > & pt, const double * tpar, int derivs, const bool * from_right = 0, double resolution = 1.0e-12 ) const` [pure virtual]

Evaluate the object in the input parameter, with the specified number of derivatives.

#### Parameters

<i>pt</i>	the <a href="#">Point</a> to be returned.
<i>tpar</i>	the parameter in which to evaluate. The size of the array should be equal to <a href="#">numParams()</a> .
<i>derivs</i>	the number of derivatives to calculate.
<i>from_right</i>	if true the evaluation is to be performed from the right of the parameter value.
<i>resolution</i>	tolerance used when determining whether parameters are located at special values of the parameter domain (in particular: knot values in case of spline objects).

Implemented in [Go::Param1FunctionInt](#), [Go::Param2FunctionInt](#), [Go::ParamPointInt](#), [Go::ParamSurfaceInt](#), [Go::ParamCurveInt](#), and [Go::Param0FunctionInt](#).

**29.331.3.23** `virtual void Go::ParamObjectInt::setCriticalVal ( int paddir, double par )` [inline],[virtual]

Set info about subdivision value.

Reimplemented in [Go::ParamCurveInt](#).

Definition at line 139 of file [ParamObjectInt.h](#).

**29.331.3.24** `void Go::ParamObjectInt::setParent ( ParamObjectInt * parent )` [inline]

Set the parent of this object.

#### Parameters

<i>parent</i>	the parent of this object.
---------------	----------------------------

Definition at line 204 of file [ParamObjectInt.h](#).

**29.331.3.25** `virtual double Go::ParamObjectInt::startParam ( int paddir ) const` [pure virtual]

Return the start parameter value in the specified direction.

#### Parameters

<i>paddir</i>	the parameter direction in question. Indexing starts at 0.
---------------	------------------------------------------------------------

Implemented in [Go::ParamSurfaceInt](#), [Go::Param2FunctionInt](#), [Go::ParamCurveInt](#), [Go::Param1FunctionInt](#), [Go::ParamPointInt](#), [Go::Param0FunctionInt](#), and [Go::Spline2FunctionInt](#).

#### 29.331.4 Member Data Documentation

##### 29.331.4.1 ParamObjectInt\* Go::ParamObjectInt::parent\_ [protected]

Definition at line 239 of file ParamObjectInt.h.

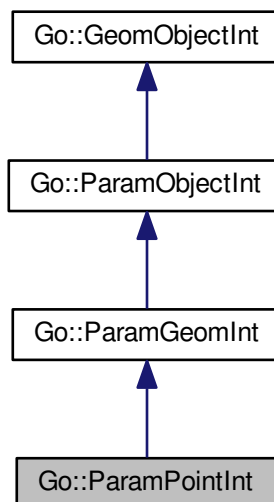
The documentation for this class was generated from the following file:

- [intersections/include/GoTools/intersections/ParamObjectInt.h](#)

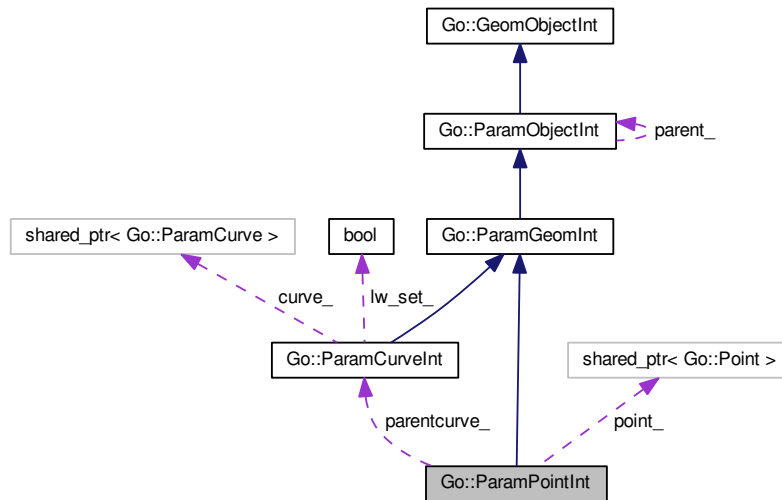
## 29.332 Go::ParamPointInt Class Reference

```
#include <ParamPointInt.h>
```

Inheritance diagram for Go::ParamPointInt:



Collaboration diagram for Go::ParamPointInt:



## Public Member Functions

- [ParamPointInt](#) (shared\_ptr< [Point](#) > point, [ParamGeomInt](#) \*parent=0)
- virtual [~ParamPointInt](#) ()
  - Destructor.*
- virtual void [point](#) ([Point](#) &pt, const double \*tpar) const
- virtual void [point](#) (std::vector< [Point](#) > &pt, const double \*tpar, int derivs, const bool \*from\_right=0, double resolution=1.0e-12) const
- virtual [ParamPointInt](#) \* [getParamPointInt](#) ()
- virtual int [numParams](#) () const
  - The number of parameters in the object.*
- virtual void [getLengthAndWiggle](#) (double \*length, double \*wiggle)
- virtual bool [hasInnerKnots](#) (int paddir) const
- virtual bool [hasCriticalVals](#) (int paddir) const
- virtual bool [hasCriticalValsOrKnots](#) (int paddir) const
- virtual bool [canDivide](#) (int paddir)
- virtual std::vector< double > [getCriticalVals](#) (int paddir) const
- virtual std::vector< double > [getInnerKnotVals](#) (int paddir, bool sort=false) const
- virtual std::vector< double > [getCriticalValsAndKnots](#) (int paddir) const
- virtual int [getMeshSize](#) (int dir)
- virtual double [paramFromMesh](#) (int dir, int idx)
- virtual std::vector< double >::iterator [getMesh](#) ()
  - Return the geometric sample mesh for the parametric function.*
- virtual double [startParam](#) (int paddir) const
- virtual double [endParam](#) (int paddir) const
- virtual bool [boundaryPoint](#) (const double \*par, double eps) const
- virtual void [subdivide](#) (int paddir, double par, std::vector< shared\_ptr< [ParamGeomInt](#) > > &subdiv\_objs, std::vector< shared\_ptr< [ParamGeomInt](#) > > &bd\_objs)
- virtual [CompositeBox](#) [compositeBox](#) () const
- virtual [DirectionCone](#) [directionCone](#) () const

- virtual void `getBoundaryObjects` (std::vector< shared\_ptr< [BoundaryGeomInt](#) > > &bd\_objs)
- virtual int `dimension` () const  
*The dimension of the geometric space.*
- virtual bool `isDegenerate` (double epsge, int dir, double \*par)
- virtual bool `isSpline` ()
- virtual double `getOptimizedConeAngle` (Point &axis1, Point &axis2)
- `RotatedBox` `getRotatedBox` (std::vector< Point > &axis) const

## Protected Attributes

- shared\_ptr< Point > `point_`
- int `dim_`
- ParamCurveInt \* `parentcurve_`
- std::vector< double > `coords_`

### 29.332.1 Detailed Description

Class representing the "intersection object" of a parametric curve.

Definition at line 57 of file ParamPointInt.h.

### 29.332.2 Constructor & Destructor Documentation

29.332.2.1 `Go::ParamPointInt::ParamPointInt ( shared_ptr< Point > point, ParamGeomInt * parent = 0 )`  
[explicit]

Constructor.

Parameters

<i>point</i>	the point defining the intersection object.
<i>parent</i>	the parent object to this object. Can be either a curve or a surface.

29.332.2.2 `virtual Go::ParamPointInt::~~ParamPointInt ( )` [inline],[virtual]

Destructor.

Definition at line 68 of file ParamPointInt.h.

### 29.332.3 Member Function Documentation

29.332.3.1 `virtual bool Go::ParamPointInt::boundaryPoint ( const double * par, double eps ) const` [inline],[virtual]

Return true if the specified point lies within eps from the boundary.

## Parameters

<i>par</i>	the parameter in which to evaluate. Size of array should be equal to <a href="#">numParams()</a> .
<i>eps</i>	the tolerance defining boundary neighbourhood.

Implements [Go::ParamObjectInt](#).

Definition at line 219 of file ParamPointInt.h.

**29.332.3.2** `virtual bool Go::ParamPointInt::canDivide ( int pardir ) [inline],[virtual]`

Return true if we are allowed to divide in the specified parameter direction.

## Parameters

<i>pardir</i>	the parameter direction in question.
---------------	--------------------------------------

Implements [Go::ParamObjectInt](#).

Definition at line 148 of file ParamPointInt.h.

**29.332.3.3** `virtual CompositeBox Go::ParamPointInt::compositeBox ( ) const [virtual]`

Create the [CompositeBox](#) for the parametric object.

## Returns

The [CompositeBox](#) for the parametric object.

Implements [Go::ParamGeomInt](#).

**29.332.3.4** `virtual int Go::ParamPointInt::dimension ( ) const [inline],[virtual]`

The dimension of the geometric space.

Implements [Go::ParamGeomInt](#).

Definition at line 251 of file ParamPointInt.h.

**29.332.3.5** `virtual DirectionCone Go::ParamPointInt::directionCone ( ) const [virtual]`

A cone which contains all normals of the object. As a normal is not defined for a point, the zero vector is returned.

## Returns

A cone which contains all normals of the object.

Implements [Go::ParamGeomInt](#).

**29.332.3.6** `virtual double Go::ParamPointInt::endParam ( int pardir ) const [inline],[virtual]`

Return the end parameter in the specified direction. Does not apply to a point.

## Parameters

<i>pardir</i>	the parameter direction in question. Indexing starts at 0.
---------------	------------------------------------------------------------

Implements [Go::ParamObjectInt](#).

Definition at line 211 of file ParamPointInt.h.

```
29.332.3.7 virtual void Go::ParamPointInt::getBoundaryObjects (std::vector< shared_ptr< BoundaryGeomInt > > &
 bd_objs) [virtual]
```

Return the boundary objects of this object.

## Parameters

<i>bd_objs</i>	the boundary objects of this object.
----------------	--------------------------------------

Implements [Go::ParamGeomInt](#).

```
29.332.3.8 virtual std::vector<double> Go::ParamPointInt::getCriticalVals (int pardir) const [inline],[virtual]
```

Return the critical parameter values in the specified direction.

## Parameters

<i>pardir</i>	the parameter direction in question. Indexing starts at 0.
---------------	------------------------------------------------------------

Implements [Go::ParamObjectInt](#).

Definition at line 154 of file ParamPointInt.h.

```
29.332.3.9 virtual std::vector<double> Go::ParamPointInt::getCriticalValsAndKnots (int pardir) const [inline],
 [virtual]
```

Return the critical parameter values and inner knots for object.

## Parameters

<i>pardir</i>	the parameter direction in question. Indexing starts at 0.
---------------	------------------------------------------------------------

Implements [Go::ParamObjectInt](#).

Definition at line 175 of file ParamPointInt.h.

29.332.3.10 `virtual std::vector<double> Go::ParamPointInt::getInnerKnotVals ( int pardir, bool sort = false ) const`  
`[inline],[virtual]`

Return the inner knot values in the specified direction.

#### Parameters

<i>pardir</i>	the parameter direction in question. Indexing starts at 0.
<i>sort</i>	the returned values may be sorted by the function.

Implements [Go::ParamObjectInt](#).

Definition at line 164 of file ParamPointInt.h.

29.332.3.11 `virtual void Go::ParamPointInt::getLengthAndWiggle ( double * length, double * wiggle )` `[inline],[virtual]`

Return an estimate on the size and wiggle of the object. Does not apply to a point.

#### Parameters

<i>length</i>	the approximative length of the object in the corresponding parameter directions. The size of the array should be equal to <a href="#">numParams()</a> .
<i>wiggle</i>	a scalar representing the wiggle of the object in the corresponding parameter directions. The size of the array should be equal to <a href="#">numParams()</a> .

Implements [Go::ParamObjectInt](#).

Definition at line 122 of file ParamPointInt.h.

29.332.3.12 `virtual std::vector<double>::iterator Go::ParamPointInt::getMesh ( )` `[inline],[virtual]`

Return the geometric sample mesh for the parametric function.

Implements [Go::ParamGeomInt](#).

Definition at line 198 of file ParamPointInt.h.

29.332.3.13 `virtual int Go::ParamPointInt::getMeshSize ( int dir )` `[inline],[virtual]`

Return the size of the geometric sample mesh in the specified direction.

#### Parameters

<i>dir</i>	the parameter direction in question. Indexing starts at 0.
------------	------------------------------------------------------------

Implements [Go::ParamGeomInt](#).



Definition at line 185 of file ParamPointInt.h.

**29.332.3.14** `virtual double Go::ParamPointInt::getOptimizedConeAngle ( Point & axis1, Point & axis2 ) [inline], [virtual]`

We try to treat problems which will never result in a simple case by shrinking the domain slightly, resulting in smaller cones. This is useful for scenarios where the normals are parallel in a boundary point.

#### Parameters

<i>axis1</i>	first vector defining a projection plane
<i>axis2</i>	second vector defining a projection plane

#### Returns

The optimized cone angle

Implements [Go::ParamGeomInt](#).

Definition at line 275 of file ParamPointInt.h.

**29.332.3.15** `virtual ParamPointInt* Go::ParamPointInt::getParamPointInt ( ) [inline], [virtual]`

Return a pointer to this object.

#### Returns

Pointer to this object.

Reimplemented from [Go::ParamObjectInt](#).

Definition at line 106 of file ParamPointInt.h.

**29.332.3.16** `RotatedBox Go::ParamPointInt::getRotatedBox ( std::vector< Point > & axis ) const`

Create a box containing the geometric sample mesh in the input coordinate system.

#### Parameters

<i>axis</i>	the axis defining the coordinate system. The size of vector axis is 2, the last point is given as the cross product axis[0]axis[1].
-------------	-------------------------------------------------------------------------------------------------------------------------------------

**29.332.3.17** `virtual bool Go::ParamPointInt::hasCriticalVals ( int pardir ) const [inline], [virtual]`

Return true if the object has any critical parameter values in the specified parameter direction.

## Parameters

<i>pardir</i>	the parameter direction in question. Indexing starts at 0.
---------------	------------------------------------------------------------

Implements [Go::ParamObjectInt](#).

Definition at line 135 of file ParamPointInt.h.

**29.332.3.18** `virtual bool Go::ParamPointInt::hasCriticalValsOrKnots ( int pardir ) const` `[inline],[virtual]`

Return true if the object has any critical parameter values or inner knots in the specified parameter direction.

## Parameters

<i>pardir</i>	the parameter direction in question. Indexing starts at 0.
---------------	------------------------------------------------------------

Implements [Go::ParamObjectInt](#).

Definition at line 142 of file ParamPointInt.h.

**29.332.3.19** `virtual bool Go::ParamPointInt::hasInnerKnots ( int pardir ) const` `[inline],[virtual]`

Return true if the object has any inner knots in the specified parameter direction.

## Parameters

<i>pardir</i>	the parameter direction in question. Indexing starts at 0.
---------------	------------------------------------------------------------

Implements [Go::ParamObjectInt](#).

Definition at line 128 of file ParamPointInt.h.

**29.332.3.20** `virtual bool Go::ParamPointInt::isDegenerate ( double epsge, int dir, double * par )` `[inline],[virtual]`

Verify whether or not the object is degenerate in the specified direction and parameter.

## Parameters

<i>epsge</i>	the geometric tolerance defining degeneracy.
<i>dir</i>	the parameter direction in question.
<i>par</i>	the parameter in which to evaluate. Size of array should be equal to <a href="#">numParams()</a> .

Implements [Go::ParamGeomInt](#).

Definition at line 260 of file ParamPointInt.h.

29.332.3.21 `virtual bool Go::ParamPointInt::isSpline ( ) [inline],[virtual]`

Verify whether the object is a spline.

#### Returns

True if the object is a spline.

Implements [Go::ParamGeomInt](#).

Definition at line 265 of file ParamPointInt.h.

29.332.3.22 `virtual int Go::ParamPointInt::numParams ( ) const [inline],[virtual]`

The number of parameters in the object.

Implements [Go::ParamObjectInt](#).

Definition at line 110 of file ParamPointInt.h.

29.332.3.23 `virtual double Go::ParamPointInt::paramFromMesh ( int dir, int idx ) [inline],[virtual]`

Return the corresponding mesh parameter. Does not apply to a point.

#### Parameters

<i>dir</i>	the parameter direction in question. Indexing starts at 0.
<i>idx</i>	the mesh idx in the specified direction. Indexing starts at 0.

Implements [Go::ParamGeomInt](#).

Definition at line 194 of file ParamPointInt.h.

29.332.3.24 `virtual void Go::ParamPointInt::point ( Point & pt, const double * tpar ) const [inline],[virtual]`

Evaluate the object in the input parameter.

#### Parameters

<i>pt</i>	the <a href="#">Point</a> to be returned.
<i>tpar</i>	the parameter in which to evaluate. The parameter is not used as the parameter domain of a point is of dimension 0.

Implements [Go::ParamObjectInt](#).

Definition at line 75 of file ParamPointInt.h.

29.332.3.25 `virtual void Go::ParamPointInt::point ( std::vector< Point > & pt, const double * tpar, int derivs, const bool * from_right = 0, double resolution = 1.0e-12 ) const [inline],[virtual]`

Evaluate the object in the input parameter, with the specified number of derivatives.

#### Parameters

<i>pt</i>	the <a href="#">Point</a> to be returned.
<i>tpar</i>	the parameter in which to evaluate. The parameter is not used as the parameter domain of a point is of dimension 0.
<i>derivs</i>	the number of derivatives to calculate. Should be 0.
<i>from_right</i>	if true the evaluation is to be performed from the right side of the parameter value. Not used.
<i>resolution</i>	tolerance used when determining whether parameters are located at special values of the parameter domain (in particular: knot values in case of spline objects. Not used.

Implements [Go::ParamObjectInt](#).

Definition at line 92 of file ParamPointInt.h.

29.332.3.26 `virtual double Go::ParamPointInt::startParam ( int paddir ) const [inline],[virtual]`

Return the start parameter value in the specified direction. Does not apply to a point.

#### Parameters

<i>paddir</i>	the parameter direction in question. Indexing starts at 0.
---------------	------------------------------------------------------------

Implements [Go::ParamObjectInt](#).

Definition at line 204 of file ParamPointInt.h.

29.332.3.27 `virtual void Go::ParamPointInt::subdivide ( int paddir, double par, std::vector< shared_ptr< ParamGeomInt > & subdiv_objs, std::vector< shared_ptr< ParamGeomInt > & bd_objs ) [inline],[virtual]`

Subdivide the object in the specified parameter direction and parameter value. Does not apply to a point.

#### Parameters

<i>paddir</i>	direction in which to subdivide. Indexing starts at 0.
<i>par</i>	parameter in which to subdivide.
<i>subdiv_objs</i>	The subparts of this object. Of the same geometric dimension as this object.
<i>bd_objs</i>	the boundaries between the returned <i>subdiv_objs</i> . Of geometric dimension 1 less than this object.

Implements [Go::ParamGeomInt](#).

Definition at line 232 of file ParamPointInt.h.

### 29.332.4 Member Data Documentation

29.332.4.1 `std::vector<double> Go::ParamPointInt::coords_` [mutable], [protected]

Definition at line 294 of file ParamPointInt.h.

29.332.4.2 `int Go::ParamPointInt::dim_` [protected]

Definition at line 290 of file ParamPointInt.h.

29.332.4.3 `ParamCurveInt* Go::ParamPointInt::parentcurve_` [protected]

Definition at line 292 of file ParamPointInt.h.

29.332.4.4 `shared_ptr<Point> Go::ParamPointInt::point_` [protected]

Definition at line 288 of file ParamPointInt.h.

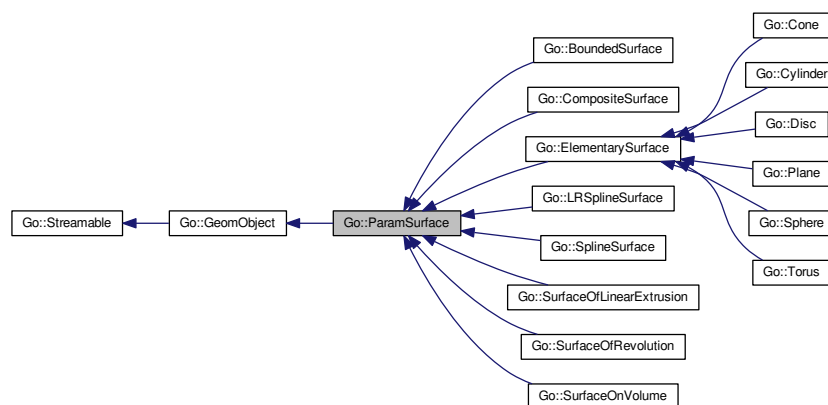
The documentation for this class was generated from the following file:

- [intersections/include/GoTools/intersections/ParamPointInt.h](#)

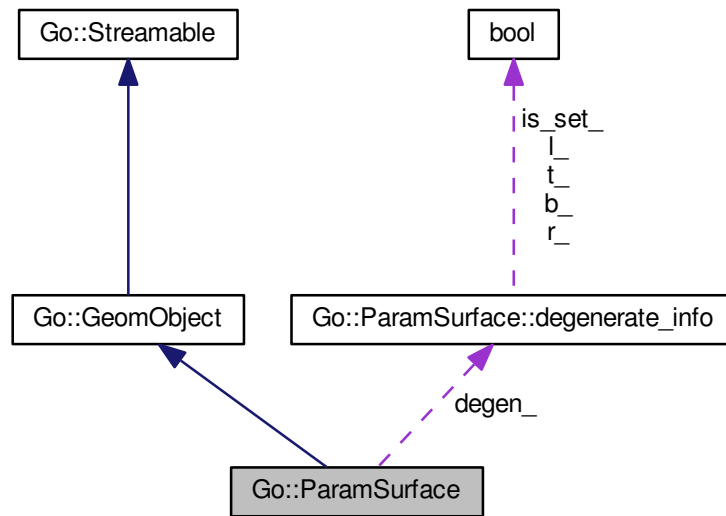
## 29.333 Go::ParamSurface Class Reference

```
#include <ParamSurface.h>
```

Inheritance diagram for Go::ParamSurface:



Collaboration diagram for Go::ParamSurface:



## Classes

- struct [degenerate\\_info](#)

*Degeneracy information regarding one boundary surface of the current surface.*

## Public Member Functions

- virtual [~ParamSurface](#) ()  
*Virtual destructor, enables safe inheritance.*
- virtual [ParamSurface](#) \* [clone](#) () const =0
- virtual [SplineSurface](#) \* [asSplineSurface](#) ()  
*Return the spline surface represented by this surface, if any.*
- virtual [const Domain](#) & [parameterDomain](#) () const =0
- virtual [RectDomain](#) [containingDomain](#) () const =0
- virtual [bool](#) [inDomain](#) ([double](#) u, [double](#) v, [double](#) eps=1.0e-4) const =0  
*Check if a parameter pair lies inside the domain of this surface.*
- virtual [int](#) [inDomain2](#) ([double](#) u, [double](#) v, [double](#) eps=1.0e-4) const =0
- virtual [bool](#) [onBoundary](#) ([double](#) u, [double](#) v, [double](#) eps=1.0e-4) const =0  
*Check if a parameter pair lies at the boundary of this surface.*
- virtual [Point](#) [closestInDomain](#) ([double](#) u, [double](#) v) const =0
- virtual void [setParameterDomain](#) ([double](#) u1, [double](#) u2, [double](#) v1, [double](#) v2)
- virtual [CurveLoop](#) [outerBoundaryLoop](#) ([double](#) degenerate\_epsilon=DEFAULT\_SPACE\_EPSILON) const =0
- virtual [std::vector](#)< [CurveLoop](#) > [allBoundaryLoops](#) ([double](#) degenerate\_epsilon=DEFAULT\_SPACE\_EPSILON) const =0
- virtual [ParamSurface](#) \* [mirrorSurface](#) ([const Point](#) &pos, [const Point](#) &norm) const  
*Mirror a surface around a specified plane.*
- virtual [DirectionCone](#) [normalCone](#) () const =0

- virtual [DirectionCone tangentCone](#) ([bool](#) paddir\_is\_u) [const](#) =0
- virtual [CompositeBox compositeBox](#) () [const](#)
- virtual void [point](#) ([Point](#) &pt, [double](#) upar, [double](#) vpar) [const](#) =0
- virtual void [point](#) ([std::vector](#)< [Point](#) > &pts, [double](#) upar, [double](#) vpar, [int](#) derivs, [bool](#) u\_from\_right=true, [bool](#) v\_from\_right=true, [double](#) resolution=1.0e-12) [const](#) =0
- [Point point](#) ([double](#) upar, [double](#) vpar) [const](#)
- [std::vector](#)< [Point](#) > [point](#) ([double](#) upar, [double](#) vpar, [int](#) derivs) [const](#)
- virtual void [normal](#) ([Point](#) &n, [double](#) upar, [double](#) vpar) [const](#) =0
- virtual void [evalGrid](#) ([int](#) num\_u, [int](#) num\_v, [double](#) umin, [double](#) umax, [double](#) vmin, [double](#) vmax, [std::vector](#)< [double](#) > &points, [double](#) nodata\_val=-9999) [const](#)
- virtual [Point getInternalPoint](#) ([double](#) &u, [double](#) &v) [const](#)
- virtual [std::vector](#)< [shared\\_ptr](#)< [ParamCurve](#) > > [constParamCurves](#) ([double](#) parameter, [bool](#) paddir\_is\_u) [const](#) =0
- virtual [std::vector](#)< [shared\\_ptr](#)< [ParamSurface](#) > > [subSurfaces](#) ([double](#) from\_upar, [double](#) from\_vpar, [double](#) to\_upar, [double](#) to\_vpar, [double](#) fuzzy=DEFAULT\_PARAMETER\_EPSILON) [const](#) =0
- virtual [double nextSegmentVal](#) ([int](#) dir, [double](#) par, [bool](#) forward, [double](#) tol) [const](#) =0
- virtual void [closestPoint](#) ([const Point](#) &pt, [double](#) &clo\_u, [double](#) &clo\_v, [Point](#) &clo\_pt, [double](#) &clo\_dist, [double](#) epsilon, [const RectDomain](#) \*domain\_of\_interest=NULL, [double](#) \*seed=0) [const](#)
- void [closestPoint](#) ([const Point](#) &pt, [double](#) &clo\_u, [double](#) &clo\_v, [Point](#) &clo\_pt, [double](#) &clo\_dist, [double](#) epsilon, [int](#) maxiter, [const RectDomain](#) \*domain\_of\_interest=NULL, [double](#) \*seed=0) [const](#)
- void [singularity](#) ([double](#) &sing\_u, [double](#) &sing\_v, [Point](#) &sing\_pt, [double](#) &sing\_dist, [double](#) epsilon, [const RectDomain](#) \*rd=NULL, [double](#) \*seed=0) [const](#)
- virtual void [closestBoundaryPoint](#) ([const Point](#) &pt, [double](#) &clo\_u, [double](#) &clo\_v, [Point](#) &clo\_pt, [double](#) &clo\_dist, [double](#) epsilon, [const RectDomain](#) \*rd=NULL, [double](#) \*seed=0) [const](#) =0
- virtual void [getBoundaryInfo](#) ([Point](#) &pt1, [Point](#) &pt2, [double](#) epsilon, [SplineCurve](#) \*&cv, [SplineCurve](#) \*&crosscv, [double](#) knot\_tol=1e-05) [const](#) =0
- virtual void [turnOrientation](#) ()=0  
*Turns the direction of the normal of the surface.*
- virtual void [reverseParameterDirection](#) ([bool](#) direction\_is\_u)=0
- virtual void [swapParameterDirection](#) ()=0  
*Swaps the two parameter directions.*
- virtual [double area](#) ([double](#) tol) [const](#) =0
- virtual [bool isDegenerate](#) ([bool](#) &b, [bool](#) &r, [bool](#) &t, [bool](#) &l, [double](#) tolerance) [const](#)
- virtual void [getDegenerateCorners](#) ([std::vector](#)< [Point](#) > &deg\_corners, [double](#) tol) [const](#) =0  
*Check for parallel and anti-parallel partial derivatives in surface corners.*
- virtual void [getCornerPoints](#) ([std::vector](#)< [std::pair](#)< [Point](#), [Point](#) > > &corners) [const](#) =0
- virtual void [setIterator](#) ([IteratorType](#) type)
- virtual [bool isAlsoTrimmed](#) ([double](#) tol) [const](#)  
*Check if the current surface is trimmed along constant parameter curves.*
- virtual [bool isSpline](#) () [const](#)  
*Check if the surface is of type spline.*
- virtual [bool isAxisRotational](#) ([Point](#) &centre, [Point](#) &axis, [Point](#) &vec, [double](#) &angle)
- virtual [bool isPlanar](#) ([Point](#) &normal, [double](#) tol)  
*Check if the surface is planar.*
- virtual [bool isLinear](#) ([Point](#) &dir1, [Point](#) &dir2, [double](#) tol)  
*Check if the surface is linear in one or both parameter directions.*
- void [estimateSfSize](#) ([double](#) &u\_size, [double](#) &v\_size, [int](#) u\_nmb=5, [int](#) v\_nmb=5) [const](#)  
*Estimate the size of the surface in the two parameter directions.*
- virtual [int ElementOnBoundary](#) ([int](#) elem\_ix, [double](#) eps)
- virtual [int ElementBoundaryStatus](#) ([int](#) elem\_ix, [double](#) eps)

## Protected Member Functions

- [ParamSurface](#) ()
- void [s1773](#) (const double ppoint[], double aepsge, double estart[], double eend[], double enext[], double gpos[], int maxiter, int \*jstat) const
- void [s1773\\_s9corr](#) (double gd[], double acoef1, double acoef2, double astart1, double aend1, double astart2, double aend2) const
- void [s1773\\_s9dir](#) (double \*cdist, double \*cdiff1, double \*cdiff2, double PS[], const double \*eval1, std::vector< [Point](#) > eval2, double aepsge, int idim, int \*jstat) const

## Protected Attributes

- [degenerate\\_info](#) `degen_`
- [IteratorType](#) `iterator_`

## Additional Inherited Members

### 29.333.1 Detailed Description

Base class for parametric surfaces in [Go](#)

Definition at line 66 of file `ParamSurface.h`.

### 29.333.2 Constructor & Destructor Documentation

29.333.2.1 virtual `Go::ParamSurface::~~ParamSurface ( )` `[virtual]`

Virtual destructor, enables safe inheritance.

29.333.2.2 `Go::ParamSurface::ParamSurface ( )` `[inline],[protected]`

Definition at line 519 of file `ParamSurface.h`.

### 29.333.3 Member Function Documentation

29.333.3.1 virtual `std::vector<CurveLoop> Go::ParamSurface::allBoundaryLoops ( double degenerate_epsilon = DEFAULT_SPACE_EPSILON ) const` `[pure virtual]`

Returns the anticlockwise outer boundary loop of the surface, together with clockwise loops of any interior boundaries, such that the surface always is 'to the left of' the loops.

#### Parameters

<code><i>degenerate_epsilon</i></code>	edges whose length is smaller than this value are ignored.
----------------------------------------	------------------------------------------------------------



**Returns**

a vector containing CurveLoops. The first of these describe the outer boundary of the surface (clockwise), whereas the others describe boundaries of interior holes (clockwise).

Implemented in [Go::SplineSurface](#), [Go::LRSplineSurface](#), [Go::BoundedSurface](#), [Go::SurfaceOnVolume](#), [Go::CompositeSurface](#), [Go::SurfaceOfRevolution](#), [Go::Disc](#), [Go::Torus](#), [Go::SurfaceOfLinearExtrusion](#), [Go::Plane](#), [Go::Cone](#), [Go::Cylinder](#), and [Go::Sphere](#).

**29.333.3.2** `virtual double Go::ParamSurface::area ( double tol ) const` [pure virtual]

Compute the total area of this surface up to some tolerance

**Parameters**

<i>tol</i>	the relative tolerance when approximating the area, i.e. stop iteration when error becomes smaller than tol/(surface area)
------------	----------------------------------------------------------------------------------------------------------------------------

**Returns**

the area calculated

Implemented in [Go::SplineSurface](#), [Go::BoundedSurface](#), [Go::LRSplineSurface](#), [Go::CompositeSurface](#), [Go::SurfaceOnVolume](#), [Go::SurfaceOfRevolution](#), [Go::SurfaceOfLinearExtrusion](#), and [Go::ElementarySurface](#).

**29.333.3.3** `virtual SplineSurface* Go::ParamSurface::asSplineSurface ( )` [inline],[virtual]

Return the spline surface represented by this surface, if any.

Reimplemented in [Go::SplineSurface](#), [Go::LRSplineSurface](#), [Go::BoundedSurface](#), [Go::SurfaceOnVolume](#), and [Go::ElementarySurface](#).

Definition at line 78 of file ParamSurface.h.

**29.333.3.4** `virtual ParamSurface* Go::ParamSurface::clone ( ) const` [pure virtual]

make a clone of this surface and return a pointer to it (user is responsible for clearing up memory afterwards).

**Returns**

pointer to cloned object

Implements [Go::GeomObject](#).

Implemented in [Go::SplineSurface](#), [Go::LRSplineSurface](#), [Go::BoundedSurface](#), [Go::SurfaceOnVolume](#), [Go::Disc](#), [Go::Torus](#), [Go::SurfaceOfRevolution](#), [Go::CompositeSurface](#), [Go::Plane](#), [Go::Cone](#), [Go::Cylinder](#), [Go::Sphere](#), [Go::SurfaceOfLinearExtrusion](#), and [Go::ElementarySurface](#).

```
29.333.3.5 virtual void Go::ParamSurface::closestBoundaryPoint (const Point & pt, double & clo_u, double & clo_v,
 Point & clo_pt, double & clo_dist, double epsilon, const RectDomain * rd = NULL, double * seed = 0)
 const [pure virtual]
```

Iterates to the closest point to pt on the boundary of the surface.

See also

[closestPoint\(\)](#)

Implemented in [Go::SplineSurface](#), [Go::BoundedSurface](#), [Go::LRSplineSurface](#), [Go::SurfaceOnVolume](#), [Go::CompositeSurface](#), [Go::SurfaceOfRevolution](#), [Go::Disc](#), [Go::Torus](#), [Go::SurfaceOfLinearExtrusion](#), [Go::Cone](#), [Go::Cylinder](#), [Go::Plane](#), and [Go::Sphere](#).

```
29.333.3.6 virtual Point Go::ParamSurface::closestInDomain (double u, double v) const [pure virtual]
```

Fetch the parameter value in the parameter domain of the surface closest to the parameter pair (u,v)

Implemented in [Go::CompositeSurface](#), [Go::SplineSurface](#), [Go::LRSplineSurface](#), [Go::BoundedSurface](#), [Go::SurfaceOnVolume](#), [Go::SurfaceOfRevolution](#), [Go::SurfaceOfLinearExtrusion](#), and [Go::ElementarySurface](#).

```
29.333.3.7 virtual void Go::ParamSurface::closestPoint (const Point & pt, double & clo_u, double & clo_v, Point &
 clo_pt, double & clo_dist, double epsilon, const RectDomain * domain_of_interest = NULL, double *
 seed = 0) const [virtual]
```

Iterates to the closest point to pt on the surface.

Parameters

<i>pt</i>	the point to find the closest point to
<i>clo_u</i>	u parameter of the closest point
<i>clo_v</i>	v parameter of the closest point
<i>clo_pt</i>	the geometric position of the closest point
<i>clo_dist</i>	the distance between pt and clo_pt
<i>epsilon</i>	parameter tolerance (will in any case not be higher than sqrt(machine_precision) x magnitude of solution)
<i>domain_of_interest</i>	pointer to parameter domain in which to search for closest point. If a NULL pointer is used, the entire surface is searched.
<i>seed</i>	pointer to parameter values where iteration starts.

Reimplemented in [Go::SplineSurface](#), [Go::BoundedSurface](#), [Go::SurfaceOnVolume](#), [Go::CompositeSurface](#), [Go::SurfaceOfRevolution](#), [Go::Disc](#), [Go::Torus](#), [Go::SurfaceOfLinearExtrusion](#), [Go::Cone](#), [Go::Cylinder](#), [Go::Plane](#), and [Go::Sphere](#).

```
29.333.3.8 void Go::ParamSurface::closestPoint (const Point & pt, double & clo_u, double & clo_v, Point & clo_pt,
 double & clo_dist, double epsilon, int maxiter, const RectDomain * domain_of_interest = NULL, double *
 seed = 0) const
```

29.333.3.9 virtual `CompositeBox` `Go::ParamSurface::compositeBox ( ) const` `[virtual]`

Creates a composite box enclosing the surface. The composite box consists of an inner and an edge box. The inner box is supposed to be made from the interior of the surface, while the edge box is made from the boundary curves. The default implementation simply makes both boxes identical to the regular bounding box.

#### Returns

the [CompositeBox](#) of the surface, as specified above

Reimplemented in [Go::SplineSurface](#), [Go::LRSplineSurface](#), [Go::SurfaceOnVolume](#), and [Go::CompositeSurface](#).

29.333.3.10 virtual `std::vector<shared_ptr<ParamCurve>>` `Go::ParamSurface::constParamCurves ( double parameter, bool pardir_is_u ) const` `[pure virtual]`

Get the curve(s) obtained by intersecting the surface with one of its constant parameter curves. For surfaces without holes, this will be the parameter curve itself; for surfaces with interior holes this may be a collection of several, disjoint curves.

#### Parameters

<i>parameter</i>	parameter value for the constant parameter (either u or v)
<i>pardir_is_u</i>	specify whether the <i>moving</i> parameter (as opposed to the <i>constant</i> parameter) is the first ('true') or the second ('false') one.

#### Returns

a vector containing shared pointers to the obtained, newly constructed constant-parameter curves.

Implemented in [Go::SplineSurface](#), [Go::LRSplineSurface](#), [Go::BoundedSurface](#), [Go::SurfaceOnVolume](#), [Go::CompositeSurface](#), [Go::SurfaceOfRevolution](#), [Go::Disc](#), [Go::Torus](#), [Go::SurfaceOfLinearExtrusion](#), [Go::Plane](#), [Go::Cone](#), [Go::Cylinder](#), and [Go::Sphere](#).

29.333.3.11 virtual `RectDomain` `Go::ParamSurface::containingDomain ( ) const` `[pure virtual]`

Get a rectangular parameter domain that is guaranteed to contain the surface's [parameterDomain\(\)](#). It may be the same. There is no guarantee that this is the smallest domain containing the actual domain.

#### Returns

a [RectDomain](#) that is guaranteed to include the surface's total parameter domain.

Implemented in [Go::SplineSurface](#), [Go::LRSplineSurface](#), [Go::BoundedSurface](#), [Go::SurfaceOnVolume](#), [Go::CompositeSurface](#), [Go::SurfaceOfRevolution](#), [Go::SurfaceOfLinearExtrusion](#), and [Go::ElementarySurface](#).

29.333.3.12 virtual `int` `Go::ParamSurface::ElementBoundaryStatus ( int elem_ix, double eps )` `[inline],[virtual]`

Check if a polynomial element (for spline surfaces) intersects the (trimming) boundaries of this [ftSurface](#), is inside or outside

## Parameters

<i>elem_ix</i>	Element index counted according to distinct knot values. Sequence of coordinates: x runs fastest, then y
<i>eps</i>	Intersection tolerance

## Returns

-1: Not a spline surface or element index out of range 0: Outside trimmed volume 1: On boundary (intersection with boundary found) 2: Internal to trimmed surfaces Note that a touch with the boundaries of the underlying surface is not considered a boundary intersection while touching a trimming curve is seen as an intersection

Reimplemented in [Go::SplineSurface](#), [Go::BoundedSurface](#), and [Go::SurfaceOnVolume](#).

Definition at line 493 of file ParamSurface.h.

```
29.333.3.13 virtual int Go::ParamSurface::ElementOnBoundary (int elem_ix, double eps) [inline],[virtual]
```

Check if a polynomial element (for spline surfaces) intersects the (trimming) boundaries of this surface

## Parameters

<i>elem_ix</i>	Element index counted according to distinct knot values. Sequence of coordinates: x runs fastest, then y
<i>eps</i>	Intersection tolerance

## Returns

-1: Not a spline surface or element index out of range 0: Not on boundary or touching a boundary curve 1: On boundary (intersection with boundary found) Note that a touch with the boundaries of the underlying surfaces is not considered a boundary intersection while touching a trimming curve is seen as an intersection

Reimplemented in [Go::SplineSurface](#), [Go::BoundedSurface](#), and [Go::SurfaceOnVolume](#).

Definition at line 476 of file ParamSurface.h.

```
29.333.3.14 void Go::ParamSurface::estimateSfSize (double & u_size, double & v_size, int u_nmb = 5, int v_nmb = 5) const
```

Estimate the size of the surface in the two parameter directions.

```
29.333.3.15 virtual void Go::ParamSurface::evalGrid (int num_u, int num_v, double umin, double umax, double vmin, double vmax, std::vector< double > & points, double nodata_val = -9999) const [virtual]
```

Evaluate points in a grid. The nodata value is applicable for bounded surfaces and grid points outside the trimming loop(s) will get this value

Reimplemented in [Go::SplineSurface](#), [Go::BoundedSurface](#), and [Go::LRSplineSurface](#).

29.333.3.16 `virtual void Go::ParamSurface::getBoundaryInfo ( Point & pt1, Point & pt2, double epsilon, SplineCurve *& cv, SplineCurve *& crosscv, double knot_tol = 1e-05 ) const` [pure virtual]

Get the boundary curve segment between two points on the boundary, as well as the cross-tangent curve. If the given points are not positioned on the same boundary (within a certain tolerance), no curves will be created.

#### Parameters

<i>pt1</i>	the first point on the boundary, given by the user
<i>pt2</i>	the second point on the boundary, given by the user
<i>epsilon</i>	the tolerance used when determining whether the given points are lying on a boundary, and if they do, whether they both lie on the <i>same</i> boundary.
<i>cv</i>	upon return, this will point to a newly created curve representing the boundary curve between 'pt1' and 'pt2'. The user assumes ownership of this object and is responsible for its deletion. No curve is created if the given points are not found to lie on the same boundary.
<i>crosscv</i>	upon return, this will point to a newly created curve representing the cross-boundary curve between 'pt1' and 'pt2'. The user assumes ownership of this object and is responsible for its deletion. The direction is outwards from the surface. No curve is created if the given points are not found to lie on the same boundary.
<i>knot_tol</i>	tolerance used when working with the knot-vector, to specify how close a parameter value must be to a knot in order to be considered 'on' the knot.

Implemented in [Go::SplineSurface](#), [Go::BoundedSurface](#), [Go::LRSplineSurface](#), [Go::CompositeSurface](#), [Go::SurfaceOnVolume](#), [Go::SurfaceOfRevolution](#), [Go::Disc](#), [Go::Torus](#), [Go::SurfaceOfLinearExtrusion](#), [Go::Cone](#), [Go::Cylinder](#), [Go::Plane](#), and [Go::Sphere](#).

29.333.3.17 `virtual void Go::ParamSurface::getCornerPoints ( std::vector< std::pair< Point, Point > > & corners ) const` [pure virtual]

Return surface corners, geometric and parametric points in that sequence

Implemented in [Go::SplineSurface](#), [Go::BoundedSurface](#), [Go::LRSplineSurface](#), [Go::SurfaceOnVolume](#), [Go::CompositeSurface](#), [Go::SurfaceOfRevolution](#), [Go::SurfaceOfLinearExtrusion](#), and [Go::ElementarySurface](#).

29.333.3.18 `virtual void Go::ParamSurface::getDegenerateCorners ( std::vector< Point > & deg_corners, double tol ) const` [pure virtual]

Check for parallel and anti-parallel partial derivatives in surface corners.

Implemented in [Go::SplineSurface](#), [Go::BoundedSurface](#), [Go::LRSplineSurface](#), [Go::CompositeSurface](#), [Go::SurfaceOnVolume](#), [Go::SurfaceOfRevolution](#), [Go::Torus](#), [Go::Disc](#), [Go::SurfaceOfLinearExtrusion](#), [Go::Sphere](#), [Go::Cone](#), [Go::Cylinder](#), and [Go::Plane](#).

29.333.3.19 `virtual Point Go::ParamSurface::getInternalPoint ( double & u, double & v ) const` [virtual]

Fetch an arbitrary internal point in the surface Used for localization purposes

Reimplemented in [Go::BoundedSurface](#).

29.333.3.20 `virtual bool Go::ParamSurface::inDomain ( double u, double v, double eps = 1.0e-4 ) const` [pure virtual]

Check if a parameter pair lies inside the domain of this surface.

Implemented in [Go::CompositeSurface](#), [Go::SplineSurface](#), [Go::LRSplineSurface](#), [Go::BoundedSurface](#), [Go::SurfaceOnVolume](#), [Go::SurfaceOfRevolution](#), [Go::SurfaceOfLinearExtrusion](#), and [Go::ElementarySurface](#).

29.333.3.21 `virtual int Go::ParamSurface::inDomain2 ( double u, double v, double eps = 1.0e-4 ) const` [pure virtual]

Check if a parameter pair lies inside the domain of this surface return value = 0: outside = 1: internal = 2: at the boundary

Implemented in [Go::CompositeSurface](#), [Go::SplineSurface](#), [Go::LRSplineSurface](#), [Go::BoundedSurface](#), [Go::SurfaceOnVolume](#), [Go::SurfaceOfRevolution](#), [Go::SurfaceOfLinearExtrusion](#), and [Go::ElementarySurface](#).

29.333.3.22 `virtual bool Go::ParamSurface::isAxisRotational ( Point & centre, Point & axis, Point & vec, double & angle )` [inline], [virtual]

Check if the surface is axis rotational. Only true if a connection to an axis rotational elementary surface exist The axis and rotational angle is only specified if the surface is actually rotational

Reimplemented in [Go::SplineSurface](#), [Go::BoundedSurface](#), [Go::Cone](#), [Go::Cylinder](#), and [Go::Sphere](#).

Definition at line 448 of file ParamSurface.h.

29.333.3.23 `virtual bool Go::ParamSurface::isDegenerate ( bool & b, bool & r, bool & t, bool & l, double tolerance ) const` [virtual]

The order of the edge indicators (bottom, right, top, left) matches the edge\_number of edgeCurve(). Query whether any of the four boundary curves are degenerate (zero length) within a certain tolerance. In the below, we refer to 'u' as the first parameter and 'v' as the second.

#### Parameters

<i>b</i>	'true' upon return of function if the boundary (v = v_min) is degenerate
<i>r</i>	'true' upon return of function if the boundary (v = v_max) is degenerate
<i>t</i>	'true' upon return of function if the boundary (u = u_min) is degenerate
<i>l</i>	'true' upon return of function if the boundary (u = u_max) is degenerate
<i>tolerance</i>	boundaries are considered degenerate if their length is shorter than this value, given by the user

#### Returns

'true' if at least one boundary curve was found to be degenerate, 'false' otherwise.

Reimplemented in [Go::SplineSurface](#), [Go::BoundedSurface](#), [Go::LRSplineSurface](#), [Go::SurfaceOnVolume](#), [Go::CompositeSurface](#), [Go::Disc](#), [Go::Torus](#), [Go::Sphere](#), [Go::Cone](#), [Go::Cylinder](#), and [Go::Plane](#).

29.333.3.24 `virtual bool Go::ParamSurface::isIsoTrimmed ( double tol ) const [inline],[virtual]`

Check if the current surface is trimmed along constant parameter curves.

Reimplemented in [Go::BoundedSurface](#), and [Go::SurfaceOnVolume](#).

Definition at line 433 of file ParamSurface.h.

29.333.3.25 `virtual bool Go::ParamSurface::isLinear ( Point & dir1, Point & dir2, double tol ) [virtual]`

Check if the surface is linear in one or both parameter directions.

Reimplemented in [Go::BoundedSurface](#), [Go::SurfaceOnVolume](#), [Go::Cylinder](#), and [Go::Plane](#).

29.333.3.26 `virtual bool Go::ParamSurface::isPlanar ( Point & normal, double tol ) [virtual]`

Check if the surface is planar.

Reimplemented in [Go::SplineSurface](#), [Go::BoundedSurface](#), and [Go::Plane](#).

29.333.3.27 `virtual bool Go::ParamSurface::isSpline ( ) const [inline],[virtual]`

Check if the surface is of type spline.

Reimplemented in [Go::SplineSurface](#).

Definition at line 439 of file ParamSurface.h.

29.333.3.28 `virtual ParamSurface* Go::ParamSurface::mirrorSurface ( const Point & pos, const Point & norm ) const [inline],[virtual]`

Mirror a surface around a specified plane.

Reimplemented in [Go::SplineSurface](#), and [Go::LRSplineSurface](#).

Definition at line 142 of file ParamSurface.h.

29.333.3.29 `virtual double Go::ParamSurface::nextSegmentVal ( int dir, double par, bool forward, double tol ) const [pure virtual]`

Determine the parameter value of the start of the 'next segment' from a parameter value, along a given parameter direction. A 'segment' is here defined as a parameter interval in which there will be no discontinuities in derivatives or other artifacts. For spline objects, a segment will typically be the interval between two consecutive, non-coincident knots.

#### Parameters

<i>dir</i>	the parameter direction in which we search for the next segment (0 or 1)
<i>par</i>	the parameter value starting from which we search for the start value of the next segment
<i>forward</i>	whether we shall move forward ('true') or backwards when searching along this parameter
<i>tol</i>	tolerance used for determining whether the 'par' is already located on the next segment value

**Returns**

the value of the start value of the next segment (or the end of the previous segment, if we are moving backwards...)

Implemented in [Go::SplineSurface](#), [Go::LRSplineSurface](#), [Go::BoundedSurface](#), [Go::SurfaceOnVolume](#), [Go::CompositeSurface](#), [Go::SurfaceOfRevolution](#), [Go::Disc](#), [Go::Torus](#), [Go::SurfaceOfLinearExtrusion](#), [Go::Cone](#), [Go::Cylinder](#), [Go::Plane](#), and [Go::Sphere](#).

**29.333.3.30** `virtual void Go::ParamSurface::normal ( Point & n, double upar, double vpar ) const` [pure virtual]

Evaluates the surface normal for a given parameter pair

**Parameters**

<i>n</i>	the computed normal will be written to this variable
<i>upar</i>	the first parameter
<i>vpar</i>	the second parameter

Implemented in [Go::SplineSurface](#), [Go::LRSplineSurface](#), [Go::BoundedSurface](#), [Go::SurfaceOnVolume](#), [Go::CompositeSurface](#), [Go::SurfaceOfRevolution](#), [Go::Disc](#), [Go::Torus](#), [Go::SurfaceOfLinearExtrusion](#), [Go::Plane](#), [Go::Cone](#), [Go::Cylinder](#), and [Go::Sphere](#).

**29.333.3.31** `virtual DirectionCone Go::ParamSurface::normalCone ( ) const` [pure virtual]

Creates a [DirectionCone](#) covering all normals to this surface.

**Returns**

a [DirectionCone](#) (not necessarily the smallest) containing all normals to this surface.

Implemented in [Go::SplineSurface](#), [Go::LRSplineSurface](#), [Go::SurfaceOnVolume](#), [Go::BoundedSurface](#), [Go::CompositeSurface](#), [Go::SurfaceOfRevolution](#), [Go::Disc](#), [Go::Torus](#), [Go::SurfaceOfLinearExtrusion](#), [Go::Plane](#), [Go::Cone](#), [Go::Cylinder](#), and [Go::Sphere](#).

**29.333.3.32** `virtual bool Go::ParamSurface::onBoundary ( double u, double v, double eps = 1.0e-4 ) const` [pure virtual]

Check if a parameter pair lies at the boundary of this surface.

Implemented in [Go::CompositeSurface](#), [Go::SplineSurface](#), [Go::LRSplineSurface](#), [Go::BoundedSurface](#), [Go::SurfaceOnVolume](#), [Go::SurfaceOfRevolution](#), [Go::SurfaceOfLinearExtrusion](#), and [Go::ElementarySurface](#).

**29.333.3.33** `virtual CurveLoop Go::ParamSurface::outerBoundaryLoop ( double degenerate_epsilon = DEFAULT_SPACE_EPSILON ) const` [pure virtual]

Returns the anticlockwise, outer boundary loop of the surface.



## Parameters

<i>degenerate_epsilon</i>	edges whose length is smaller than this value are ignored.
---------------------------	------------------------------------------------------------

## Returns

a [CurveLoop](#) describing the anticlockwise, outer boundary loop of the surface. A negative *degenerate\_epsilon* indicates that all curves, also the degenerate ones are wanted

Implemented in [Go::SplineSurface](#), [Go::LRSplineSurface](#), [Go::BoundedSurface](#), [Go::SurfaceOnVolume](#), [Go::CompositeSurface](#), [Go::SurfaceOfRevolution](#), [Go::SurfaceOfLinearExtrusion](#), and [Go::ElementarySurface](#).

**29.333.3.34** `virtual const Domain& Go::ParamSurface::parameterDomain ( ) const` [pure virtual]

Return the parameter domain of the surface. This may be a simple rectangular domain ([RectDomain](#)) or any other subclass of [Domain](#) (such as [GoCurveBoundedDomain](#), found in the `sisl_dependent` module).

## Returns

a [Domain](#) object describing the parametric domain of the surface

Implemented in [Go::SplineSurface](#), [Go::LRSplineSurface](#), [Go::BoundedSurface](#), [Go::SurfaceOnVolume](#), [Go::Disc](#), [Go::Torus](#), [Go::CompositeSurface](#), [Go::SurfaceOfRevolution](#), [Go::Plane](#), [Go::Cone](#), [Go::Cylinder](#), [Go::Sphere](#), and [Go::SurfaceOfLinearExtrusion](#).

**29.333.3.35** `virtual void Go::ParamSurface::point ( Point & pt, double upar, double vpar ) const` [pure virtual]

Evaluates the surface's position for a given parameter pair.

## Parameters

<i>pt</i>	the result of the evaluation is written here
<i>upar</i>	the first parameter
<i>vpar</i>	the second parameter

Implemented in [Go::SplineSurface](#), [Go::LRSplineSurface](#), [Go::BoundedSurface](#), [Go::SurfaceOnVolume](#), [Go::CompositeSurface](#), [Go::SurfaceOfRevolution](#), [Go::Disc](#), [Go::Torus](#), [Go::SurfaceOfLinearExtrusion](#), [Go::Plane](#), [Go::Cone](#), [Go::Cylinder](#), and [Go::Sphere](#).

**29.333.3.36** `virtual void Go::ParamSurface::point ( std::vector< Point > & pts, double upar, double vpar, int derivs, bool u_from_right = true, bool v_from_right = true, double resolution = 1.0e-12 ) const` [pure virtual]

Evaluates the surface's position and a certain number of derivatives for a given parameter pair.

## Parameters

<i>pts</i>	the vector containing the evaluated values. Its size must be set by the user prior to calling this function, and should be equal to $(\text{derivs}+1) * (\text{derivs}+2) / 2$ . Upon completion of the function, its first entry is the surface's position at the given parameter pair. Then, if 'derivs' > 0, the two next entries will be the surface tangents along the first and second parameter direction. The next three entries are the second- and cross derivatives, in the order (du2, dudv, dv2), and similar for even higher derivatives.
<i>upar</i>	the first parameter
<i>vpar</i>	the second parameter
<i>derivs</i>	number of requested derivatives
<i>u_from_right</i>	specify whether derivatives along the first parameter are to be calculated from the right ('true', default) or from the left ('false')
<i>v_from_right</i>	specify whether derivatives along the second parameter are to be calculated from the right ('true', default) or from the left ('false')
<i>resolution</i>	tolerance used when determining whether parameters are located at special values of the parameter domain (in particular; knot values in case of spline objects).

Implemented in [Go::SplineSurface](#), [Go::BoundedSurface](#), [Go::LRSplineSurface](#), [Go::SurfaceOnVolume](#), [Go::CompositeSurface](#), [Go::SurfaceOfRevolution](#), [Go::Disc](#), [Go::Torus](#), [Go::SurfaceOfLinearExtrusion](#), [Go::Plane](#), [Go::Cone](#), [Go::Cylinder](#), and [Go::Sphere](#).

### 29.333.3.37 Point `Go::ParamSurface::point ( double upar, double vpar ) const`

Evaluate the surface's position at a certain parameter pair

## Parameters

<i>upar</i>	the first parameter
<i>vpar</i>	the second parameter

## Returns

the surface's position for this parameter pair. NB: This function is implemented in terms of the [ParamSurface](#)'s virtual 'point(...)' function, but is itself not virtual. If you make a concrete subclass and wish to make this function visible to the user, you must put a 'using [ParamSurface::point](#)' statement in the class definition.

### 29.333.3.38 `std::vector<Point> Go::ParamSurface::point ( double upar, double vpar, int derivs ) const`

Evaluate the surface's position and a certain number of derivatives at a given parameter.

## Parameters

<i>upar</i>	the first parameter
<i>vpar</i>	the second parameter
<i>derivs</i>	number of requested derivatives

## Returns

the vector containing the evaluated values. Its size will be equal to  $(\text{derivs}+1) * (\text{derivs}+2) / 2$ . Its first entry is the surface's position at the given parameter pair. Then, if 'derivs' > 0, the two next entries will be the surface tangents along the first and second parameter direction. The next three entries are the second- and cross derivatives, in the order (du2, dudv, dv2), and similar for even higher derivatives. NB: This function is implemented in terms of the [ParamSurface](#)'s virtual 'point(...)' functions, but is itself not virtual. If you make a concrete subclass and wish to make this function visible to the user, you must put a 'using [ParamCurve::point](#)' in the class definition.

29.333.3.39 `virtual void Go::ParamSurface::reverseParameterDirection ( bool direction_is_u ) [pure virtual]`

Reverses the direction of the basis in input direction.

## Parameters

<i>direction_is_u</i>	if 'true', the first parameter direction will be reversed, otherwise, the second parameter direction will be reversed
-----------------------	-----------------------------------------------------------------------------------------------------------------------

Implemented in [Go::SplineSurface](#), [Go::BoundedSurface](#), [Go::LRSplineSurface](#), [Go::CompositeSurface](#), [Go::SurfaceOnVolume](#), [Go::SurfaceOfRevolution](#), [Go::SurfaceOfLinearExtrusion](#), and [Go::ElementarySurface](#).

29.333.3.40 `void Go::ParamSurface::s1773 ( const double ppoint[], double aepsge, double estart[], double eend[], double enext[], double gpos[], int maxiter, int * jstat ) const [protected]`

29.333.3.41 `void Go::ParamSurface::s1773_s9corr ( double gd[], double acoef1, double acoef2, double astart1, double aend1, double astart2, double aend2 ) const [protected]`

29.333.3.42 `void Go::ParamSurface::s1773_s9dir ( double * cdist, double * cdiff1, double * cdiff2, double PS[], const double * eval1, std::vector< Point > eval2, double aepsge, int idim, int * jstat ) const [protected]`

29.333.3.43 `virtual void Go::ParamSurface::setIterator ( IteratorType type ) [inline],[virtual]`

Set type of closest point iterator type == Iterator\_parametric - use conjugate gradient iteration type == Iterator\_geometric - sisl type geometric closest point iteration

Reimplemented in [Go::BoundedSurface](#).

Definition at line 427 of file ParamSurface.h.

29.333.3.44 `virtual void Go::ParamSurface::setParameterDomain ( double u1, double u2, double v1, double v2 ) [virtual]`

set the parameter domain to a given rectangle

## Parameters

<i>u1</i>	new min. value of first parameter span
<i>u2</i>	new max. value of first parameter span
<i>v1</i>	new min. value of second parameter span
<i>v2</i>	new max. value of second parameter span

Reimplemented in [Go::SplineSurface](#), [Go::BoundedSurface](#), [Go::LRSplineSurface](#), and [Go::CompositeSurface](#).

29.333.3.45 `void Go::ParamSurface::singularity ( double & sing_u, double & sing_v, Point & sing_pt, double & sing_dist, double epsilon, const RectDomain * rd = NULL, double * seed = 0 ) const`

29.333.3.46 `virtual std::vector<shared_ptr<ParamSurface> > Go::ParamSurface::subSurfaces ( double from_upar, double from_vpar, double to_upar, double to_vpar, double fuzzy = DEFAULT_PARAMETER_EPSILON ) const [pure virtual]`

Get the surface(s) obtained by cropping the parameter domain of this surface between given values for the first and second parameter. In general, for surfaces with no interior holes, the result will be *one* surface; however, for surfaces with interior holes, the result might be *several disjoint* surfaces.

#### Parameters

<i>from_upar</i>	lower value for the first parameter in the subdomain
<i>from_vpar</i>	lower value for the second parameter in the subdomain
<i>to_upar</i>	upper value for the first parameter in the subdomain
<i>to_vpar</i>	upper value for the second parameter in the subdomain
<i>fuzzy</i>	tolerance used when determining intersection with interior boundaries

#### Returns

a vector contained shared pointers to the obtained, newly constructed sub-surfaces.

Implemented in [Go::SplineSurface](#), [Go::LRSplineSurface](#), [Go::BoundedSurface](#), [Go::SurfaceOnVolume](#), [Go::CompositeSurface](#), [Go::SurfaceOfRevolution](#), [Go::Disc](#), [Go::Torus](#), [Go::SurfaceOfLinearExtrusion](#), [Go::Cone](#), [Go::Cylinder](#), [Go::Plane](#), and [Go::Sphere](#).

29.333.3.47 `virtual void Go::ParamSurface::swapParameterDirection ( ) [pure virtual]`

Swaps the two parameter directions.

Implemented in [Go::SplineSurface](#), [Go::BoundedSurface](#), [Go::LRSplineSurface](#), [Go::CompositeSurface](#), [Go::SurfaceOnVolume](#), [Go::SurfaceOfRevolution](#), [Go::SurfaceOfLinearExtrusion](#), and [Go::ElementarySurface](#).

29.333.3.48 `virtual DirectionCone Go::ParamSurface::tangentCone ( bool paddir_is_u ) const [pure virtual]`

Creates a [DirectionCone](#) covering all tangents to this surface along a given parameter direction.

#### Parameters

<i>paddir_is_u</i>	if 'true', then the <a href="#">DirectionCone</a> will be defined on basis of the surface's tangents along the first parameter direction. Otherwise the second parameter direction will be used.
--------------------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

**Returns**

a [DirectionCone](#) (not necessarily the smallest) containing all tangents to this surface along the specified parameter direction.

Implemented in [Go::SplineSurface](#), [Go::LRSplineSurface](#), [Go::SurfaceOnVolume](#), [Go::BoundedSurface](#), [Go::CompositeSurface](#), [Go::SurfaceOfRevolution](#), [Go::Disc](#), [Go::Torus](#), [Go::SurfaceOfLinearExtrusion](#), [Go::Plane](#), [Go::Cone](#), [Go::Cylinder](#), and [Go::Sphere](#).

**29.333.3.49** `virtual void Go::ParamSurface::turnOrientation ( ) [pure virtual]`

Turns the direction of the normal of the surface.

Implemented in [Go::SplineSurface](#), [Go::BoundedSurface](#), [Go::LRSplineSurface](#), [Go::CompositeSurface](#), [Go::SurfaceOnVolume](#), [Go::SurfaceOfRevolution](#), [Go::SurfaceOfLinearExtrusion](#), and [Go::ElementarySurface](#).

**29.333.4 Member Data Documentation**

**29.333.4.1** `degenerate_info Go::ParamSurface::degen_ [mutable], [protected]`

Definition at line 515 of file ParamSurface.h.

**29.333.4.2** `IteratorType Go::ParamSurface::iterator_ [protected]`

Definition at line 517 of file ParamSurface.h.

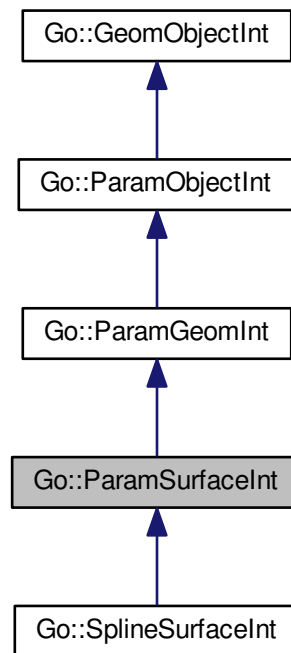
The documentation for this class was generated from the following file:

- [gotools-core/include/GoTools/geometry/ParamSurface.h](#)

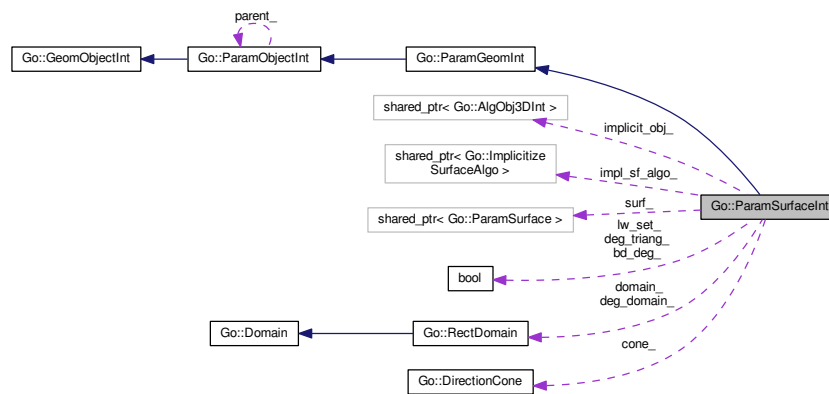
**29.334 Go::ParamSurfaceInt Class Reference**

```
#include <ParamSurfaceInt.h>
```

Inheritance diagram for Go::ParamSurfaceInt:



Collaboration diagram for Go::ParamSurfaceInt:



## Public Member Functions

- `ParamSurfaceInt` (`shared_ptr< ParamSurface > surf`, `ParamGeomInt *parent=0`)
- `virtual ~ParamSurfaceInt ()`

*Destructor.*

- virtual void [point](#) ([Point](#) &pt, [const double](#) \*tpar) [const](#)
- virtual void [point](#) ([std::vector](#)< [Point](#) > &pt, [const double](#) \*tpar, [int](#) derivs, [const bool](#) \*from\_right=0, [double](#) resolution=1.0e-12) [const](#)
- virtual [ParamSurfaceInt](#) \* [getParamSurfaceInt](#) ()
- [shared\\_ptr](#)< [ParamSurface](#) > [getParamSurface](#) ()
- [shared\\_ptr](#)< [const ParamSurface](#) > [getParamSurface](#) () [const](#)
- [shared\\_ptr](#)< [ParamSurface](#) > [getParentParamSurface](#) ([RectDomain](#) &domain)
- [shared\\_ptr](#)< [ParamSurface](#) > [getParentParamSurface](#) ()
- virtual [shared\\_ptr](#)< [ParamSurfaceInt](#) > [makeIntObject](#) ([shared\\_ptr](#)< [ParamSurface](#) > surf)
- virtual [shared\\_ptr](#)< [ParamCurveInt](#) > [makeIntCurve](#) ([shared\\_ptr](#)< [ParamCurve](#) > crv, [ParamGeomInt](#) \*parent)
- virtual [int](#) [numParams](#) () [const](#)

*The number of parameters in the object.*

- [shared\\_ptr](#)< [ParamCurve](#) > [getIsoCurve](#) ([double](#) param\_start, [double](#) param\_end, [double](#) isoval, [bool](#) paddir\_is\_u) [const](#)
- [shared\\_ptr](#)< [ParamCurve](#) > [getConstantParameterCurve](#) ([int](#) dir, [double](#) par)
- [shared\\_ptr](#)< [ParamCurve](#) > [getConstantParameterCurve](#) ([int](#) dir, [double](#) par, [double](#) tmin, [double](#) tmax)
- void [minimumAlongCurve](#) ([int](#) dir, [double](#) par, [double](#) tmin, [double](#) tmax, [Point](#) &pnt, [double](#) &minpar, [Point](#) &minval, [double](#) &mindist)
- virtual void [getLengthAndWiggle](#) ([double](#) \*length, [double](#) \*wiggle)
- virtual [bool](#) [hasInnerKnots](#) ([int](#) paddir) [const](#)
- virtual [bool](#) [hasCriticalVals](#) ([int](#) paddir) [const](#)
- virtual [bool](#) [hasCriticalValsOrKnots](#) ([int](#) paddir) [const](#)
- virtual [bool](#) [canDivide](#) ([int](#) paddir)
- virtual [bool](#) [canDivideTinyTriang](#) ([int](#) paddir)
- virtual [std::vector](#)< [double](#) > [getCriticalVals](#) ([int](#) paddir) [const](#)
- virtual [std::vector](#)< [double](#) > [getInnerKnotVals](#) ([int](#) paddir, [bool](#) sort=false) [const](#)
- virtual [std::vector](#)< [double](#) > [getCriticalValsAndKnots](#) ([int](#) paddir) [const](#)
- virtual [int](#) [getMeshSize](#) ([int](#) dir)
- virtual [double](#) [paramFromMesh](#) ([int](#) dir, [int](#) idx)
- virtual [std::vector](#)< [double](#) >::iterator [getMesh](#) ()

*Return the geometric sample mesh for the parametric function.*

- virtual [double](#) [startParam](#) ([int](#) paddir) [const](#)
- virtual [double](#) [endParam](#) ([int](#) paddir) [const](#)
- virtual [bool](#) [boundaryPoint](#) ([const double](#) \*par, [double](#) eps) [const](#)
- [std::vector](#)< [double](#) > [getMima](#) () [const](#)
- virtual [bool](#) [inCorner](#) ([const double](#) \*par, [double](#) epspar) [const](#)
- [bool](#) [atDegenerateBd](#) ([const double](#) \*par, [double](#) epsge, [double](#) epspar)
- virtual void [subdivide](#) ([int](#) paddir, [double](#) par, [std::vector](#)< [shared\\_ptr](#)< [ParamGeomInt](#) > > &subdiv\_objs, [std::vector](#)< [shared\\_ptr](#)< [ParamGeomInt](#) > > &bd\_objs)
- virtual [std::vector](#)< [shared\\_ptr](#)< [ParamSurfaceInt](#) > > [subSurfaces](#) ([double](#) from\_upar, [double](#) from\_vpar, [double](#) to\_upar, [double](#) to\_vpar, [double](#) fuzzy)
- virtual [CompositeBox](#) [compositeBox](#) () [const](#)
- virtual [DirectionCone](#) [directionCone](#) () [const](#)
- virtual void [getBoundaryObjects](#) ([std::vector](#)< [shared\\_ptr](#)< [BoundaryGeomInt](#) > > &bd\_objs)
- virtual [int](#) [checkPeriodicity](#) ([int](#) paddir) [const](#)
- virtual [int](#) [dimension](#) () [const](#)

*The dimension of the geometric space.*

- virtual [const RectDomain](#) & [getDomain](#) () [const](#)
- [const RectDomain](#) & [getDegDomain](#) ([double](#) epsge)
- virtual [bool](#) [isSimple](#) ()
- virtual [bool](#) [isSpline](#) ()
- virtual [bool](#) [isIsoParametric](#) ([ParamCurveInt](#) \*curve, [int](#) dir, [double](#) par, [double](#) ptol, [double](#) tol)
- virtual [double](#) [getOptimizedConeAngle](#) ([Point](#) &axis1, [Point](#) &axis2)

- virtual void [knotIntervalFuzzy](#) (double &u, double &v, double utol, double vtol) const
- virtual [double nextSegmentVal](#) (int dir, double par, bool forward, double tol) const
- [bool isDegenerate](#) (double epsge)
- virtual int [isDegenerate](#) (double epsge, int dir)
- virtual [bool isDegenerate](#) (double epsge, int dir, double \*par)
- [double isolateDegPar](#) (int dir, int deg\_edge, double threshold, double \*deg\_factor=NULL)
- void [setDegTriang](#) ()
- [bool getDegTriang](#) ()
- [double getParOffBd](#) (int dir, bool atstart, double tol) const
- void [first\\_fund\\_form](#) (double u, double v, bool u\_from\_right, bool v\_from\_right, double &E, double &F, double &G) const
- void [second\\_fund\\_form](#) (double u, double v, bool u\_from\_right, bool v\_from\_right, double &L, double &M, double &N) const
- void [derivs](#) (double u, double v, Point &deriv\_u, Point &deriv\_v, bool from\_right\_1=true, bool from\_right\_↔2=true) const
- void [normal](#) (double u, double v, Point &normal, bool from\_right\_1=true, bool from\_right\_2=true) const
- virtual [RotatedBox getRotatedBox](#) (std::vector< Point > &axis) const
- void [axisFromCorners](#) (Point &axis1, Point &axis2) const
- virtual void [splitAtG0](#) (double angtol, std::vector< shared\_ptr< ParamSurfaceInt > > &subG1)
- void [getSingularity](#) (double eps, double sing\_par[], Point &sing\_pt, double &sing\_val, double \*seed)
- [bool canSelfIntersect](#) (double epsge) const
- virtual shared\_ptr< ParamSurfaceInt > [getNormalSurface](#) () const
- virtual [bool canImplicitize](#) ()
- virtual [bool implicitize](#) (double tol)
- virtual [bool getImplicit](#) (double tol, double &tol2, AlgObj3DInt &alg\_obj\_3d\_int)
- void [maxCurvatures](#) (bool dir\_is\_u, int nmb\_params, double iso\_from, double iso\_to, double guess\_param, std::vector< double > &max\_curv\_params, std::vector< double > &max\_curvatures)

## Protected Attributes

- shared\_ptr< ParamSurface > [surf\\_](#)
- int [dim\\_](#)
- double [deg\\_tol\\_](#)
- bool [deg\\_triang\\_](#)
- std::vector< std::pair< double, int > > [segment\\_ \[2\]](#)
- bool [bd\\_deg\\_ \[4\]](#)
- RectDomain [domain\\_](#)
- RectDomain [deg\\_domain\\_](#)
- int [nmesh\\_ \[2\]](#)
- std::vector< double > [mesh\\_](#)
- DirectionCone [cone\\_](#)
- bool [lw\\_set\\_](#)
- double [length\\_ \[2\]](#)
- double [wiggle\\_ \[2\]](#)
- std::vector< Point > [temp\\_point\\_array\\_](#)
- shared\_ptr< ImplicitizeSurfaceAlgo > [impl\\_sf\\_algo\\_](#)
- double [implicit\\_tol\\_](#)
- int [impl\\_deg\\_](#)
- shared\_ptr< AlgObj3DInt > [implicit\\_obj\\_](#)
- double [implicit\\_err\\_](#)



### 29.334.1 Detailed Description

Class that represents the "intersection object" of a parametric surface.

Definition at line 59 of file ParamSurfacelnt.h.

### 29.334.2 Constructor & Destructor Documentation

29.334.2.1 `Go::ParamSurfacelnt::ParamSurfacelnt ( shared_ptr< ParamSurface > surf, ParamGeomInt * parent = 0 )`  
`[explicit]`

Constructor.

Parameters

<i>surf</i>	the parametric surface defining the intersection object.
<i>parent</i>	the parent object to this object.

29.334.2.2 `virtual Go::ParamSurfacelnt::~~ParamSurfacelnt ( )` `[virtual]`

Destructor.

### 29.334.3 Member Function Documentation

29.334.3.1 `bool Go::ParamSurfacelnt::atDegenerateBd ( const double * par, double epsge, double epspar )`

29.334.3.2 `void Go::ParamSurfacelnt::axisFromCorners ( Point & axis1, Point & axis2 ) const`

Use the corner points of the parameter domain to create axes.

Parameters

<i>axis1</i>	the first axis.
<i>axis2</i>	the second axis.

29.334.3.3 `virtual bool Go::ParamSurfacelnt::boundaryPoint ( const double * par, double eps ) const` `[virtual]`

Return true if the specified point lies within eps from the boundary.

Parameters

<i>par</i>	the parameter in which to evaluate. Size of array should be equal to <code>numParams()</code> .
<i>eps</i>	the tolerance defining the boundary neighbourhood.

Implements [Go::ParamObjectInt](#).

**29.334.3.4** `virtual bool Go::ParamSurfaceInt::canDivide ( int pardir ) [virtual]`

Return true if we are allowed to divide in the specified parameter direction.

#### Parameters

<i>pardir</i>	the parameter direction in question.
---------------	--------------------------------------

Implements [Go::ParamObjectInt](#).

**29.334.3.5** `virtual bool Go::ParamSurfaceInt::canDivideTinyTriang ( int pardir ) [virtual]`

Reimplemented from [Go::ParamObjectInt](#).

**29.334.3.6** `virtual bool Go::ParamSurfaceInt::canImplicitize ( ) [inline],[virtual]`

Check whether the objects fulfills the requirements to implicitize.

#### Returns

Return true if we may implicitize the object.

Reimplemented in [Go::SplineSurfaceInt](#).

Definition at line 621 of file ParamSurfaceInt.h.

**29.334.3.7** `bool Go::ParamSurfaceInt::canSelfIntersect ( double epsge ) const`

Check if the current surface can selfintersect. Uses normal cone and tangent cones.

#### Parameters

<i>epsge</i>	a geometric tolerance
--------------	-----------------------

**29.334.3.8** `virtual int Go::ParamSurfaceInt::checkPeriodicity ( int pardir ) const [virtual]`

Check if the object is periodic in the specified direction. Analyze periodicity of surface based on number of repeating knots and control points. The return value is -1 if the surface ends are disjoint, otherwise k if cv is  $C^k$  continuous. These are sufficient but not necessary conditions for periodicity, so it is possible that a higher degree of periodicity exists. Should not be called on this layer, should be overruled by inherited class.

## Parameters

<i>parDir</i>	the parameter direction in question.
---------------	--------------------------------------

## Returns

-1 if the curve ends are disjoint, or k if the surface is proven to be  $C^k$  continuous.

Reimplemented from [Go::ParamGeomInt](#).

Reimplemented in [Go::SplineSurfaceInt](#).

**29.334.3.9** virtual **CompositeBox** `Go::ParamSurfaceInt::compositeBox ( ) const` `[virtual]`

Create the [CompositeBox](#) for the parametric object.

## Returns

The [CompositeBox](#) for the parametric object.

Implements [Go::ParamGeomInt](#).

**29.334.3.10** void `Go::ParamSurfaceInt::derivs ( double u, double v, Point & deriv_u, Point & deriv_v, bool from_right_1 = true, bool from_right_2 = true ) const` `[inline]`

Return the partial derivatives in the input parameter point.

## Parameters

<i>u</i>	the first parameter value.
<i>v</i>	the second parameter value.
<i>deriv_u</i>	the partial derivative in the u-direction.
<i>deriv_v</i>	the partial derivative in the u-direction.
<i>from_right_1</i>	if true then calculate from the left int the first parameter parameter direction, otherwise from the right.
<i>from_right_2</i>	if true then calculate from the left int the second parameter parameter direction, otherwise from the right.

Definition at line 534 of file `ParamSurfaceInt.h`.

**29.334.3.11** virtual int `Go::ParamSurfaceInt::dimension ( ) const` `[inline],[virtual]`

The dimension of the geometric space.

Implements [Go::ParamGeomInt](#).

Definition at line 349 of file `ParamSurfaceInt.h`.

29.334.3.12 `virtual DirectionCone Go::ParamSurfaceInt::directionCone ( ) const [virtual]`

A cone which contains all normals of the object.

#### Returns

A cone which contains all normals of the object.

Implements [Go::ParamGeomInt](#).

Reimplemented in [Go::SplineSurfaceInt](#).

29.334.3.13 `virtual double Go::ParamSurfaceInt::endParam ( int pardir ) const [virtual]`

Return the end parameter in the specified direction.

#### Parameters

<i>pardir</i>	the parameter direction in question. Indexing starts at 0.
---------------	------------------------------------------------------------

Implements [Go::ParamObjectInt](#).

29.334.3.14 `void Go::ParamSurfaceInt::first_fund_form ( double u, double v, bool u_from_right, bool v_from_right, double & E, double & F, double & G ) const [inline]`

Return the coefficients necessary to calculate the first fundamental form of the surface in the parameter point.

#### Parameters

<i>u</i>	the first parameter value.
<i>v</i>	the second parameter value.
<i>u_from_right</i>	if true then evaluate from the right, otherwise from the left.
<i>v_from_right</i>	if true then evaluate from the right, otherwise from the left.
<i>E</i>	the first coefficient for the first fundamental form.
<i>F</i>	the second coefficient for the first fundamental form.
<i>G</i>	the third coefficient for the first fundamental form.

Definition at line 470 of file ParamSurfaceInt.h.

29.334.3.15 `virtual void Go::ParamSurfaceInt::getBoundaryObjects ( std::vector< shared_ptr< BoundaryGeomInt > > & bd_objs ) [virtual]`

Return the boundary objects of this object.

#### Parameters

<i>bd_objs</i>	the boundary objects of this object.
----------------	--------------------------------------

Implements [Go::ParamGeomInt](#).

Reimplemented in [Go::SplineSurfaceInt](#).

29.334.3.16 `shared_ptr<ParamCurve> Go::ParamSurfaceInt::getConstantParameterCurve ( int dir, double par )`

Return a curveOnSurface along the current surface in the given direction and parameter value

Parameters

<i>dir</i>	the direction of the constant parameter curve. Indexing starts at 0.
<i>par</i>	the isoparameter for the curve.

29.334.3.17 `shared_ptr<ParamCurve> Go::ParamSurfaceInt::getConstantParameterCurve ( int dir, double par, double tmin, double tmax )`

29.334.3.18 `virtual std::vector<double> Go::ParamSurfaceInt::getCriticalVals ( int pardir ) const [virtual]`

Return the critical parameter values in the specified direction.

Parameters

<i>pardir</i>	the parameter direction in question. Indexing starts at 0.
---------------	------------------------------------------------------------

Implements [Go::ParamObjectInt](#).

29.334.3.19 `virtual std::vector<double> Go::ParamSurfaceInt::getCriticalValsAndKnots ( int pardir ) const [virtual]`

Return the critical parameter values and inner knots for object.

Parameters

<i>pardir</i>	the parameter direction in question. Indexing starts at 0.
---------------	------------------------------------------------------------

Implements [Go::ParamObjectInt](#).

Reimplemented in [Go::SplineSurfaceInt](#).

29.334.3.20 `const RectDomain& Go::ParamSurfaceInt::getDegDomain ( double epsge )`

Return a rectangular domain of the surface which is slightly reduced in the degenerate case

Returns

Return the rectangular domain of the surface.

29.334.3.21 **bool** Go::ParamSurfaceInt::getDegTriang ( ) `[inline]`

Get deg\_triang info. Variable useful for surfaces with degenerated edge(s).

Definition at line 446 of file ParamSurfaceInt.h.

29.334.3.22 **virtual const RectDomain&** Go::ParamSurfaceInt::getDomain ( ) **const** `[inline],[virtual]`

Return the rectangular domain of the surface.

#### Returns

Return the rectangular domain of the surface.

Definition at line 354 of file ParamSurfaceInt.h.

29.334.3.23 **virtual bool** Go::ParamSurfaceInt::getImplicit ( **double** *tol*, **double &** *tol2*, **AlgObj3DInt &** *alg\_obj\_3d\_int* ) `[inline],[virtual]`

Get the implicit representation of the object. Garbage is returned if we are not able to implicitize.

#### Parameters

<i>tol</i>	geometric tolerance for the implicitization procedure.
<i>tol2</i>	geometric estimate for the accuracy of the implicitized object. Not yet in use!!!
<i>alg_obj_3d_int</i>	the algebraic object containing the implicitized surface.

#### Returns

True if the implicitization was a success.

Reimplemented in [Go::SplineSurfaceInt](#).

Definition at line 640 of file ParamSurfaceInt.h.

29.334.3.24 **virtual std::vector<double>** Go::ParamSurfaceInt::getInnerKnotVals ( **int** *parDir*, **bool** *sort = false* ) **const** `[virtual]`

Return the inner knot values in the specified direction.

#### Parameters

<i>parDir</i>	the parameter direction in question. Indexing starts at 0.
<i>sort</i>	the returned values may be sorted by the function.

Implements [Go::ParamObjectInt](#).

Reimplemented in [Go::SplineSurfaceInt](#).

29.334.3.25 `shared_ptr<ParamCurve> Go::ParamSurfaceInt::getIsoCurve ( double param_start, double param_end, double isoval, bool pardir_is_u ) const`

Returns the specified isocurve.

#### Parameters

<i>param_start</i>	start parameter for the isocurve.
<i>param_end</i>	end parameter for the isocurve.
<i>isoval</i>	the value for the isoparameter.
<i>pardir_is_u</i>	if 'pardir_is_u' is 'true', then the first parameter is the running direction and the second parameter is the isoparameter; vice versa for 'pardir_is_u' equal to 'false'.

29.334.3.26 `virtual void Go::ParamSurfaceInt::getLengthAndWiggle ( double * length, double * wiggle ) [virtual]`

Return an estimate on the size and wiggle of the object.

#### Parameters

<i>length</i>	the approximative length of the object in the corresponding parameter directions. The size of the array should be equal to <code>numParams()</code> .
<i>wiggle</i>	a scalar representing the wiggle of the object in the corresponding parameter directions. The size of the array should be equal to <code>numParams()</code> .

Implements [Go::ParamObjectInt](#).

29.334.3.27 `virtual std::vector<double>::iterator Go::ParamSurfaceInt::getMesh ( ) [virtual]`

Return the geometric sample mesh for the parametric function.

Implements [Go::ParamGeomInt](#).

Reimplemented in [Go::SplineSurfaceInt](#).

29.334.3.28 `virtual int Go::ParamSurfaceInt::getMeshSize ( int dir ) [virtual]`

Return the size of the geometric sample mesh in the specified direction.

#### Parameters

<i>dir</i>	the parameter direction in question. Indexing starts at 0.
------------	------------------------------------------------------------

Implements [Go::ParamGeomInt](#).

Reimplemented in [Go::SplineSurfaceInt](#).

29.334.3.29 `std::vector<double> Go::ParamSurfaceInt::getMima ( ) const`

Return the domain of the surface.

#### Returns

The returned vector consists of (umin, umax, vmin, vmax).

29.334.3.30 `virtual shared_ptr<ParamSurfaceInt> Go::ParamSurfaceInt::getNormalSurface ( ) const [inline], [virtual]`

Returns the normal surface corresponding to this surface.

#### Returns

Pointer to the [SplineSurface](#) defining the normal surface. Not implemented for a general parametric surface.

Reimplemented in [Go::SplineSurfaceInt](#).

Definition at line 612 of file ParamSurfaceInt.h.

29.334.3.31 `virtual double Go::ParamSurfaceInt::getOptimizedConeAngle ( Point & axis1, Point & axis2 ) [virtual]`

We try to treat problems which will never result in a simple case by shrinking the domain slightly, resulting in smaller cones. This is useful for scenarios where the normals are parallel in a boundary point.

#### Parameters

<i>axis1</i>	first vector defining a projection plane
<i>axis2</i>	second vector defining a projection plane

#### Returns

The optimized cone angle

Implements [Go::ParamGeomInt](#).

Reimplemented in [Go::SplineSurfaceInt](#).

29.334.3.32 `shared_ptr<ParamSurface> Go::ParamSurfaceInt::getParamSurface ( ) [inline]`

Return pointer to the parametric surface defining this object.

#### Returns

Pointer to the parametric surface defining this object.

Definition at line 116 of file ParamSurfaceInt.h.



29.334.3.33 `shared_ptr<const ParamSurface> Go::ParamSurfaceInt::getParamSurface ( ) const [inline]`

Return pointer to the parametric surface defining this object.

#### Returns

Pointer to the parametric surface defining this object.

Definition at line 122 of file ParamSurfaceInt.h.

29.334.3.34 `virtual ParamSurfaceInt* Go::ParamSurfaceInt::getParamSurfaceInt ( ) [virtual]`

Return a pointer to this object.

#### Returns

Pointer to this object.

Reimplemented from [Go::ParamObjectInt](#).

29.334.3.35 `shared_ptr<ParamSurface> Go::ParamSurfaceInt::getParentParamSurface ( RectDomain & domain )`

Return pointer to a subsurface of the parent surface for this object. If no such surface exist, we use the surface of this object instead.

#### Parameters

<i>domain</i>	the parametric domain of the subsurface.
---------------	------------------------------------------

#### Returns

Pointer to a subsurface of the parent surface for this object.

29.334.3.36 `shared_ptr<ParamSurface> Go::ParamSurfaceInt::getParentParamSurface ( )`

Return pointer to the parent surface for this object. If no such surface exist, we use the surface of this object instead.

#### Returns

Pointer to a subsurface of the parent surface for this object.

29.334.3.37 `double Go::ParamSurfaceInt::getParOffBd ( int dir, bool atstart, double tol ) const`

Estimate the parameter value of a surface a specified distance from a given edge.

## Parameters

<i>dir</i>	the parameter direction in question.
<i>atstart</i>	true if we are the start of the parameter interval, false otherwise.
<i>tol</i>	influencing how far from the edge we should move.

## Returns

the computed parameter value.

29.334.3.38 `virtual RotatedBox Go::ParamSurfaceInt::getRotatedBox ( std::vector< Point > & axis ) const` [virtual]

Create a box containing the geometric sample mesh in the input coordinate system.

## Parameters

<i>axis</i>	the axis defining the coordinate system. The size of vector axis is 2, the last point is given as the cross product axis[0]axis[1].
-------------	-------------------------------------------------------------------------------------------------------------------------------------

## Returns

the rotated box

Reimplemented in [Go::SplineSurfaceInt](#).

29.334.3.39 `void Go::ParamSurfaceInt::getSingularity ( double eps, double sing_par[], Point & sing_pt, double & sing_val, double * seed )`

Iterate for a surface singularity.

## Parameters

<i>eps</i>	numerical tolerance
<i>sing_par</i>	parameters of the singularity
<i>sing_pt</i>	the singular point
<i>sing_val</i>	a number that measures the quality of the singularity
<i>seed</i>	initial guess for singularity in parameter plane

29.334.3.40 `virtual bool Go::ParamSurfaceInt::hasCriticalVals ( int padir ) const` [virtual]

Return true if the object has any critical parameter values in the specified parameter direction.

## Parameters

<i>padir</i>	the parameter direction in question. Indexing starts at 0.
--------------	------------------------------------------------------------

Implements [Go::ParamObjectInt](#).

29.334.3.41 `virtual bool Go::ParamSurfaceInt::hasCriticalValsOrKnots ( int pardir ) const` [virtual]

Return true if the object has any critical parameter values or inner knots in the specified parameter direction.

#### Parameters

<i>pardir</i>	the parameter direction in question. Indexing starts at 0.
---------------	------------------------------------------------------------

Implements [Go::ParamObjectInt](#).

Reimplemented in [Go::SplineSurfaceInt](#).

29.334.3.42 `virtual bool Go::ParamSurfaceInt::hasInnerKnots ( int pardir ) const` [virtual]

Return true if the object has any inner knots in the specified parameter direction.

#### Parameters

<i>pardir</i>	the parameter direction in question. Indexing starts at 0.
---------------	------------------------------------------------------------

Implements [Go::ParamObjectInt](#).

Reimplemented in [Go::SplineSurfaceInt](#).

29.334.3.43 `virtual bool Go::ParamSurfaceInt::implicitize ( double tol )` [inline],[virtual]

Implicitize the object.

#### Parameters

<i>tol</i>	the geoemtric tolerance for the implicitization.
------------	--------------------------------------------------

#### Returns

True if the implicitization was a success.

Reimplemented in [Go::SplineSurfaceInt](#).

Definition at line 627 of file ParamSurfaceInt.h.

29.334.3.44 `virtual bool Go::ParamSurfaceInt::inCorner ( const double * par, double epspar ) const` [virtual]

Check if parameter point lies close to a corner of the parameter domain.

## Parameters

<i>par</i>	the input parameter point.
<i>epspar</i>	the parametric tolerance defining the neighbourhood.

Reimplemented from [Go::ParamObjectInt](#).

29.334.3.45 **bool** `Go::ParamSurfaceInt::isDegenerate ( double epsge )`

Verify whether the object is degenerate.

## Parameters

<i>epsge</i>	the geometric tolerance defining degeneracy.
--------------	----------------------------------------------

29.334.3.46 **virtual int** `Go::ParamSurfaceInt::isDegenerate ( double epsge, int dir )` [virtual]

Verify whether the object is degenerate in the specified direction.

## Parameters

<i>epsge</i>	the geometric tolerance defining degeneracy.
<i>dir</i>	the parameter direction in question.

Reimplemented from [Go::ParamGeomInt](#).

29.334.3.47 **virtual bool** `Go::ParamSurfaceInt::isDegenerate ( double epsge, int dir, double * par )` [virtual]

Verify whether the object is degenerate in the specified direction and parameter.

## Parameters

<i>epsge</i>	the geometric tolerance defining degeneracy.
<i>dir</i>	the parameter direction in question.
<i>par</i>	the parameter in which to evaluate. Size of array should be equal to <a href="#">numParams()</a> .

Implements [Go::ParamGeomInt](#).

29.334.3.48 **virtual bool** `Go::ParamSurfaceInt::isIsoParametric ( ParamCurveInt * curve, int dir, double par, double ptol, double tol )` [inline], [virtual]

Check if a curve is an iso parametric curve in the current surface. NB! Only valid for splines

Reimplemented in [Go::SplineSurfaceInt](#).

Definition at line 374 of file ParamSurfaceInt.h.

29.334.3.49 **double** Go::ParamSurfaceInt::isolateDegPar ( int *dir*, int *deg\_edge*, double *threshold*, double \* *deg\_factor* = NULL )

Return info on parameter domain which needs special treatment near a degenerated edge.

#### Parameters

<i>dir</i>	the parameter direction in question. Indexing starts at 0.
<i>deg_edge</i>	the degenerate edge
<i>threshold</i>	a tolerance

#### Returns

a safe parameter value that is isolated from the degenerate value

29.334.3.50 **virtual bool** Go::ParamSurfaceInt::isSimple ( ) [virtual]

Estimates if the current surface is simple enough for a singularity iteration. Checks the span of the normal cone and the size of the surface

#### Returns

True if the surface is characterized as simple.

Reimplemented in [Go::SplineSurfaceInt](#).

29.334.3.51 **virtual bool** Go::ParamSurfaceInt::isSpline ( ) [virtual]

Verify whether the object is a spline.

#### Returns

True if the object is a spline.

Implements [Go::ParamGeomInt](#).

Reimplemented in [Go::SplineSurfaceInt](#).

29.334.3.52 **virtual void** Go::ParamSurfaceInt::knotIntervalFuzzy ( double & *u*, double & *v*, double *utol*, double *vtoI* ) const [virtual]

Find the knot intervals for which *u* and *v* lie inside, moving the value *u* or *v* if they lie close to a knot.

#### Parameters

<i>u</i>	the <i>u</i> parameter value
<i>v</i>	the <i>v</i> parameter value
<i>utol</i>	the parametric tolerance deciding if the input parameter <i>u</i> should be moved.
<i>vtoI</i>	the parametric tolerance deciding if the input parameter <i>v</i> should be moved.

Reimplemented in [Go::SplineSurfaceInt](#).

29.334.3.53 `virtual shared_ptr<ParamCurveInt> Go::ParamSurfaceInt::makeIntCurve ( shared_ptr< ParamCurve > crv, ParamGeomInt * parent ) [virtual]`

Return an intersection object for the input curve, using parameter *parent* as parent.

#### Parameters

<i>crv</i>	the parametric curve defining the intersection object.
<i>parent</i>	the parent to the created intersection object.

Reimplemented in [Go::SplineSurfaceInt](#).

29.334.3.54 `virtual shared_ptr<ParamSurfaceInt> Go::ParamSurfaceInt::makeIntObject ( shared_ptr< ParamSurface > surf ) [virtual]`

Return an intersection object for the input surface, using this object as parent.

#### Parameters

<i>surf</i>	the parametric surface defining the intersection object.
-------------	----------------------------------------------------------

Reimplemented in [Go::SplineSurfaceInt](#).

29.334.3.55 `void Go::ParamSurfaceInt::maxCurvatures ( bool dir_is_u, int nmb_params, double iso_from, double iso_to, double guess_param, std::vector< double > & max_curv_params, std::vector< double > & max_curvatures )`

29.334.3.56 `void Go::ParamSurfaceInt::minimumAlongCurve ( int dir, double par, double tmin, double tmax, Point & pnt, double & minpar, Point & minval, double & mindist )`

29.334.3.57 `virtual double Go::ParamSurfaceInt::nextSegmentVal ( int dir, double par, bool forward, double tol ) const [virtual]`

Return the value of the knot next to the input parameter *par*.

#### Parameters

<i>dir</i>	the parameter direction in question. Indexing starts at 0.
<i>par</i>	the parameter value
<i>forward</i>	if true we return the closest knot to the right, otherwise the closest knot to the left.
<i>tol</i>	the tolerance to determine if <i>par</i> is already located on the start of the next segment.

## Returns

The knot closest to the input parameter.

Reimplemented in [Go::SplineSurfaceInt](#).

**29.334.3.58** `void Go::ParamSurfaceInt::normal ( double u, double v, Point & normal, bool from_right_1 = true, bool from_right_2 = true ) const` `[inline]`

Calculate the normal in the specified parameter point.

## Parameters

<i>u</i>	the first parameter value.
<i>v</i>	the second parameter value.
<i>normal</i>	the calculated normal in the parameter point.
<i>from_right_1</i>	if true then calculate from the left int the first parameter parameter direction, otherwise from the right.
<i>from_right_2</i>	if true then calculate from the left int the second parameter parameter direction, otherwise from the right.

Definition at line 557 of file ParamSurfaceInt.h.

**29.334.3.59** `virtual int Go::ParamSurfaceInt::numParams ( ) const` `[inline],[virtual]`

The number of parameters in the object.

Implements [Go::ParamObjectInt](#).

Definition at line 155 of file ParamSurfaceInt.h.

**29.334.3.60** `virtual double Go::ParamSurfaceInt::paramFromMesh ( int dir, int idx )` `[virtual]`

Return the corresponding mesh parameter.

## Parameters

<i>dir</i>	the parameter direction in question. Indexing starts at 0.
<i>idx</i>	the mesh idx in the specified direction. Indexing starts at 0.

Implements [Go::ParamGeomInt](#).

Reimplemented in [Go::SplineSurfaceInt](#).

**29.334.3.61** `virtual void Go::ParamSurfaceInt::point ( Point & pt, const double * tpar ) const` `[inline],[virtual]`

Evaluate the object in the input parameter.

## Parameters

<i>pt</i>	the <a href="#">Point</a> to be returned.
<i>tpar</i>	the parameter in which to evaluate. The size of the array should be equal to <a href="#">numParams()</a> .

Implements [Go::ParamObjectInt](#).

Definition at line 75 of file ParamSurfaceInt.h.

```
29.334.3.62 virtual void Go::ParamSurfaceInt::point (std::vector< Point > & pt, const double * tpar, int derivs, const
bool * from_right = 0, double resolution = 1.0e-12) const [inline],[virtual]
```

Evaluate the object in the input parameter, with the specified number of derivatives.

## Parameters

<i>pt</i>	the vecotr of points to be returned.
<i>tpar</i>	the parameter in which to evaluate. The size of the array should be equal to <a href="#">numParams()</a> .
<i>derivs</i>	the number of derivatives to calculate.
<i>from_right</i>	if true the evaluation is to be performed from the right side of the parameter value.
<i>resolution</i>	tolerance used when determining whether parameters are located at special values of the parameter domain (in particular: knot values in case of spline objects).

Implements [Go::ParamObjectInt](#).

Definition at line 89 of file ParamSurfaceInt.h.

```
29.334.3.63 void Go::ParamSurfaceInt::second_fund_form (double u, double v, bool u_from_right, bool v_from_right,
double & L, double & M, double & N) const [inline]
```

Return the coefficients necessary to calculate the second fundamental form of the surface in the parameter point.

## Parameters

<i>u</i>	the first parameter value.
<i>v</i>	the second parameter value.
<i>u_from_right</i>	if true then evaluate from the right, otherwise from the left.
<i>v_from_right</i>	if true then evaluate from the right, otherwise from the left.
<i>L</i>	the first coefficient for the second fundamental form.
<i>M</i>	the second coefficient for the second fundamental form.
<i>N</i>	the third coefficient for the second fundamental form.

Definition at line 501 of file ParamSurfaceInt.h.

```
29.334.3.64 void Go::ParamSurfaceInt::setDegTriang () [inline]
```

Set dege\_triang info. Variable useful for surfaces with degenerated edge(s).

Definition at line 441 of file ParamSurfaceInt.h.



29.334.3.65 `virtual void Go::ParamSurfaceInt::splitAtG0 ( double angtol, std::vector< shared_ptr< ParamSurfaceInt > > & subG1 ) [virtual]`

Requests from selfintersection computation. Split at G1 discontinuities

#### Parameters

<i>angtol</i>	angular tolerance defining G1 discontinuity
<i>subG1</i>	vector of subdivided patches that are G1

Reimplemented in [Go::SplineSurfaceInt](#).

29.334.3.66 `virtual double Go::ParamSurfaceInt::startParam ( int pardir ) const [virtual]`

Return the start parameter value in the specified direction.

#### Parameters

<i>pardir</i>	the parameter direction in question. Indexing starts at 0.
---------------	------------------------------------------------------------

Implements [Go::ParamObjectInt](#).

29.334.3.67 `virtual void Go::ParamSurfaceInt::subdivide ( int pardir, double par, std::vector< shared_ptr< ParamGeomInt > > & subdiv_objs, std::vector< shared_ptr< ParamGeomInt > > & bd_objs ) [virtual]`

Subdivide the object in the specified parameter direction and parameter value.

#### Parameters

<i>pardir</i>	direction in which to subdivide. Indexing starts at 0.
<i>par</i>	parameter in which to subdivide.
<i>subdiv_objs</i>	The subparts of this object. Of the same geometric dimension as this object.
<i>bd_objs</i>	the boundaries between the returned <i>subdiv_objs</i> . Of geometric dimension 1 less than this object.

Implements [Go::ParamGeomInt](#).

29.334.3.68 `virtual std::vector< shared_ptr< ParamSurfaceInt > > Go::ParamSurfaceInt::subSurfaces ( double from_upar, double from_vpar, double to_upar, double to_vpar, double fuzzy ) [virtual]`

Return the subsurface(s) on the input domain. For a trimmed surface there may be more than one surface, for a surface defined on a rectangular domain there will be only one.

#### Parameters

<i>from_upar</i>	start parameter in the first parameter direction.
<i>from_vpar</i>	start parameter in the second parameter direction.
<i>to_upar</i>	end parameter in the first parameter direction.

## Parameters

<i>to_vpar</i>	end parameter in the second parameter direction.
<i>fuzzy</i>	allowed alteration of an input parameter value. Typically this applies to a spline surface, where we do not want knots to lie too close whilst not being equal.

**29.334.4 Member Data Documentation**

**29.334.4.1** `bool Go::ParamSurfaceInt::bd_deg_`[4] [protected]

Definition at line 673 of file ParamSurfaceInt.h.

**29.334.4.2** `DirectionCone Go::ParamSurfaceInt::cone_` [mutable], [protected]

Definition at line 681 of file ParamSurfaceInt.h.

**29.334.4.3** `RectDomain Go::ParamSurfaceInt::deg_domain_` [protected]

Definition at line 675 of file ParamSurfaceInt.h.

**29.334.4.4** `double Go::ParamSurfaceInt::deg_tol_` [protected]

Definition at line 667 of file ParamSurfaceInt.h.

**29.334.4.5** `bool Go::ParamSurfaceInt::deg_triang_` [protected]

Definition at line 668 of file ParamSurfaceInt.h.

**29.334.4.6** `int Go::ParamSurfaceInt::dim_` [protected]

Definition at line 666 of file ParamSurfaceInt.h.

**29.334.4.7** `RectDomain Go::ParamSurfaceInt::domain_` [protected]

Definition at line 674 of file ParamSurfaceInt.h.

**29.334.4.8** `int Go::ParamSurfaceInt::impl_deg_` [protected]

Definition at line 699 of file ParamSurfaceInt.h.

29.334.4.9 `shared_ptr<ImplicitizeSurfaceAlgo> Go::ParamSurfaceInt::impl_sf_algo_` [protected]

Definition at line 696 of file ParamSurfaceInt.h.

29.334.4.10 `double Go::ParamSurfaceInt::implicit_err_` [protected]

Definition at line 701 of file ParamSurfaceInt.h.

29.334.4.11 `shared_ptr<AlgObj3DInt> Go::ParamSurfaceInt::implicit_obj_` [protected]

Definition at line 700 of file ParamSurfaceInt.h.

29.334.4.12 `double Go::ParamSurfaceInt::implicit_tol_` [protected]

Definition at line 697 of file ParamSurfaceInt.h.

29.334.4.13 `double Go::ParamSurfaceInt::length_[2]` [mutable], [protected]

Definition at line 684 of file ParamSurfaceInt.h.

29.334.4.14 `bool Go::ParamSurfaceInt::lw_set_` [mutable], [protected]

Definition at line 683 of file ParamSurfaceInt.h.

29.334.4.15 `std::vector<double> Go::ParamSurfaceInt::mesh_` [mutable], [protected]

Definition at line 679 of file ParamSurfaceInt.h.

29.334.4.16 `int Go::ParamSurfaceInt::nmesh_[2]` [mutable], [protected]

Definition at line 678 of file ParamSurfaceInt.h.

29.334.4.17 `std::vector<std::pair<double, int> > Go::ParamSurfaceInt::segment_[2]` [protected]

Definition at line 670 of file ParamSurfaceInt.h.

29.334.4.18 `shared_ptr<ParamSurface> Go::ParamSurfaceInt::surf_` [protected]

Definition at line 665 of file ParamSurfaceInt.h.

29.334.4.19 `std::vector<Point> Go::ParamSurfaceInt::temp_point_array_` [mutable],[protected]

Definition at line 693 of file ParamSurfaceInt.h.

29.334.4.20 `double Go::ParamSurfaceInt::wiggle_2` [mutable],[protected]

Definition at line 684 of file ParamSurfaceInt.h.

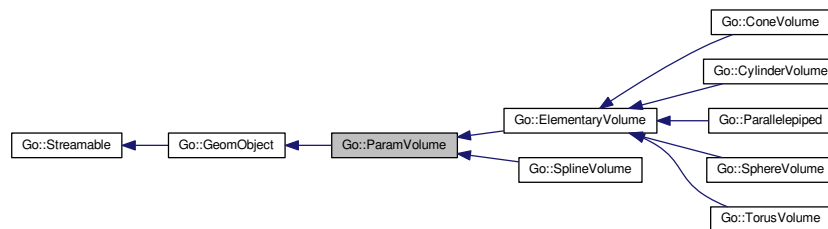
The documentation for this class was generated from the following file:

- intersections/include/GoTools/intersections/[ParamSurfaceInt.h](#)

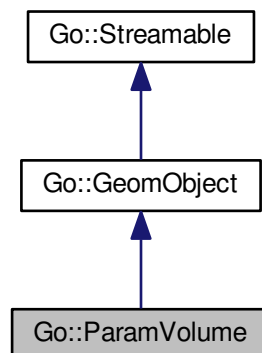
## 29.335 Go::ParamVolume Class Reference

```
#include <ParamVolume.h>
```

Inheritance diagram for Go::ParamVolume:



Collaboration diagram for Go::ParamVolume:



## Public Member Functions

- virtual [~ParamVolume](#) ()  
*Virtual destructor, enables safe inheritance.*
- virtual [ParamVolume](#) \* [clone](#) () const =0
- virtual [DirectionCone](#) [tangentCone](#) (int pdir) const =0
- virtual const [Array](#)< [double](#), 6 > [parameterSpan](#) () const =0
- virtual void [point](#) ([Point](#) &pt, [double](#) upar, [double](#) vpar, [double](#) wpar) const =0
- virtual void [point](#) (std::vector< [Point](#) > &pts, [double](#) upar, [double](#) vpar, [double](#) wpar, int derivs, [bool](#) u\_from←\_right=true, [bool](#) v\_from\_right=true, [bool](#) w\_from\_right=true, [double](#) resolution=1.0e-12) const =0
- virtual [double](#) [nextSegmentVal](#) (int dir, [double](#) par, [bool](#) forward, [double](#) tol) const =0
- virtual void [closestPoint](#) (const [Point](#) &pt, [double](#) &clo\_u, [double](#) &clo\_v, [double](#) &clo\_w, [Point](#) &clo\_pt, [double](#) &clo\_dist, [double](#) epsilon, [double](#) \*seed=0) const =0
- virtual void [reverseParameterDirection](#) (int pdir)=0
- virtual void [swapParameterDirection](#) (int pdir1, int pdir2)=0
- virtual std::vector< shared\_ptr< [ParamSurface](#) > > [getAllBoundarySurfaces](#) () const =0  
*Fetch all boundary surfaces corresponding to the volume.*
- virtual [ParamSurface](#) \* [constParamSurface](#) ([double](#) parameter, int pdir) const
- virtual [bool](#) [isSpline](#) () const  
*Check if the volume is of type spline.*
- virtual [SplineVolume](#) \* [asSplineVolume](#) ()  
*Return the spline volume represented by this volume, if any.*
- virtual void [translate](#) (const [Point](#) &vec)=0  
*Translate.*

## Additional Inherited Members

### 29.335.1 Detailed Description

Base class for parametric Volumes in [Go](#)

Definition at line 59 of file ParamVolume.h.

### 29.335.2 Constructor & Destructor Documentation

29.335.2.1 virtual [Go::ParamVolume::~ParamVolume](#) ( ) [virtual]

Virtual destructor, enables safe inheritance.

### 29.335.3 Member Function Documentation

29.335.3.1 virtual [SplineVolume](#)\* [Go::ParamVolume::asSplineVolume](#) ( ) [inline],[virtual]

Return the spline volume represented by this volume, if any.

Reimplemented in [Go::SplineVolume](#).

Definition at line 202 of file ParamVolume.h.

29.335.3.2 `virtual ParamVolume* Go::ParamVolume::clone ( ) const [pure virtual]`

make a clone of this volume and return a pointer to it (user is responsible for clearing up memory afterwards).

#### Returns

pointer to cloned object

Implements [Go::GeomObject](#).

Implemented in [Go::SplineVolume](#), [Go::TorusVolume](#), [Go::ConeVolume](#), [Go::CylinderVolume](#), [Go::SphereVolume](#), [Go::Parallelepiped](#), and [Go::ElementaryVolume](#).

29.335.3.3 `virtual void Go::ParamVolume::closestPoint ( const Point & pt, double & clo_u, double & clo_v, double & clo_w, Point & clo_pt, double & clo_dist, double epsilon, double * seed = 0 ) const [pure virtual]`

Iterates to the closest point to pt on the volume.

#### Parameters

<i>pt</i>	the point to find the closest point to
<i>clo_u</i>	u parameter of the closest point
<i>clo_v</i>	v parameter of the closest point
<i>clo_w</i>	w parameter of the closest point
<i>clo_pt</i>	the geometric position of the closest point
<i>clo_dist</i>	the distance between pt and clo_pt
<i>epsilon</i>	parameter tolerance (will in any case not be higher than sqrt(machine_precision) x magnitude of solution)
<i>seed</i>	pointer to parameter values where iteration starts.

Implemented in [Go::SplineVolume](#), [Go::TorusVolume](#), [Go::SphereVolume](#), [Go::ConeVolume](#), [Go::CylinderVolume](#), and [Go::Parallelepiped](#).

29.335.3.4 `virtual ParamSurface* Go::ParamVolume::constParamSurface ( double parameter, int paddir ) const [inline], [virtual]`

Generate and return a [ParamSurface](#) that represents a constant parameter surface on the volume

#### Parameters

<i>parameter</i>	value of the fixed parameter
<i>paddir</i>	0 if the surface is constant in the u-parameter, 1 if the surface is constant in the v-parameter, 2 if the surface is constant in the w-parameter.

#### Returns

pointer to a newly constructed [SplineSurface](#) representing the specified constant parameter surface. It is the user's responsibility to delete it when it is no longer needed.

Reimplemented in [Go::SplineVolume](#).

Definition at line 188 of file ParamVolume.h.

**29.335.3.5** `virtual std::vector<shared_ptr<ParamSurface> > Go::ParamVolume::getAllBoundarySurfaces ( ) const`  
[pure virtual]

Fetch all boundary surfaces corresponding to the volume.

Implemented in [Go::SplineVolume](#), [Go::TorusVolume](#), [Go::SphereVolume](#), [Go::ConeVolume](#), [Go::CylinderVolume](#), and [Go::Parallelepiped](#).

**29.335.3.6** `virtual bool Go::ParamVolume::isSpline ( ) const` [inline],[virtual]

Check if the volume is of type spline.

Reimplemented in [Go::SplineVolume](#).

Definition at line 196 of file ParamVolume.h.

**29.335.3.7** `virtual double Go::ParamVolume::nextSegmentVal ( int dir, double par, bool forward, double tol ) const`  
[pure virtual]

Determine the parameter value of the start of the 'next segment' from a parameter value, along a given parameter direction. A 'segment' is here defined as a parameter interval in which there will be no discontinuities in derivatives or other artifacts. For spline objects, a segment will typically be the interval between two consecutive, non-coincident knots.

#### Parameters

<i>dir</i>	the parameter direction in which we search for the next segment (0, 1 or 2)
<i>par</i>	the parameter value starting from which we search for the start value of the next segment
<i>forward</i>	define whether we shall move forward ('true') or backwards when searching along this parameter
<i>tol</i>	tolerance used for determining whether the 'par' is already located <i>on</i> the next segment value

#### Returns

the value of the start value of the next segment (or the end of the previous segment, if we are moving backwards...)

Implemented in [Go::SplineVolume](#), [Go::TorusVolume](#), [Go::SphereVolume](#), [Go::ConeVolume](#), [Go::CylinderVolume](#), and [Go::Parallelepiped](#).

**29.335.3.8** `virtual const Array<double,6> Go::ParamVolume::parameterSpan ( ) const` [pure virtual]

Return the parameter domain of the volume. This is an array containing the start and end parameter values. The spline's parameter *i* has its start value at the array position (2*i*) and its end parameter value at the array position (2*i*+1)

**Returns**

An array describing the parametric domain of the volume

Implemented in [Go::SplineVolume](#), [Go::TorusVolume](#), [Go::SphereVolume](#), [Go::ConeVolume](#), [Go::CylinderVolume](#), and [Go::Parallelepiped](#).

```
29.335.3.9 virtual void Go::ParamVolume::point (Point & pt, double upar, double vpar, double wpar) const [pure virtual]
```

Evaluates the volume's position for a given parameter triple.

**Parameters**

<i>pt</i>	the result of the evaluation is written here
<i>upar</i>	the first parameter
<i>vpar</i>	the second parameter
<i>wpar</i>	the third parameter

Implemented in [Go::SplineVolume](#), [Go::TorusVolume](#), [Go::SphereVolume](#), [Go::ConeVolume](#), [Go::CylinderVolume](#), and [Go::Parallelepiped](#).

```
29.335.3.10 virtual void Go::ParamVolume::point (std::vector< Point > & pts, double upar, double vpar, double wpar, int derivs, bool u_from_right = true, bool v_from_right = true, bool w_from_right = true, double resolution = 1.0e-12) const [pure virtual]
```

Evaluates the volume's position and a certain number of derivatives for a given parameter triple.

**Parameters**

<i>pts</i>	the vector containing the evaluated values. Its size must be set by the user prior to calling this function. Upon completion of the function, its first entry is the volume's position at the given parameter pair. Then, if 'derivs' > 0, the two next entries will be the volume tangents along the first, second and third parameter direction. The next three entries are the second- and cross derivatives, in the order (du2, dudv, dudw, dv2, dvdw, dw2), and similar for even higher derivatives.
<i>upar</i>	the first parameter
<i>vpar</i>	the second parameter
<i>wpar</i>	the third parameter
<i>derivs</i>	number of requested derivatives
<i>u_from_right</i>	specify whether derivatives along the first parameter are to be calculated from the right ('true', default) or from the left ('false')
<i>v_from_right</i>	specify whether derivatives along the second parameter are to be calculated from the right ('true', default) or from the left ('false')
<i>w_from_right</i>	specify whether derivatives along the third parameter are to be calculated from the right ('true', default) or from the left ('false')
<i>resolution</i>	tolerance used when determining whether parameters are located at special values of the parameter domain (in particular; knot values in case of spline objects).

Implemented in [Go::SplineVolume](#), [Go::TorusVolume](#), [Go::SphereVolume](#), [Go::ConeVolume](#), [Go::CylinderVolume](#),



and [Go::Parallelepiped](#).

29.335.3.11 `virtual void Go::ParamVolume::reverseParameterDirection ( int pardir ) [pure virtual]`

Reverses the direction of the basis in input direction.

#### Parameters

<i>pardir</i>	which parameter direction to reverse
---------------	--------------------------------------

Implemented in [Go::SplineVolume](#), [Go::TorusVolume](#), [Go::SphereVolume](#), [Go::ConeVolume](#), [Go::CylinderVolume](#), and [Go::Parallelepiped](#).

29.335.3.12 `virtual void Go::ParamVolume::swapParameterDirection ( int pardir1, int pardir2 ) [pure virtual]`

Swaps two parameter directions

#### Parameters

<i>pardir1</i>	One of the parameter directions (0, 1 or 2) to be swapped
<i>pardir2</i>	The other parameter direction (0, 1 or 2) to be swapped

Implemented in [Go::SplineVolume](#), [Go::TorusVolume](#), [Go::SphereVolume](#), [Go::ConeVolume](#), [Go::CylinderVolume](#), and [Go::Parallelepiped](#).

29.335.3.13 `virtual DirectionCone Go::ParamVolume::tangentCone ( int pardir ) const [pure virtual]`

Creates a [DirectionCone](#) covering all tangents to this volume along a given parameter direction.

#### Parameters

<i>pardir</i>	if 1, the <a href="#">DirectionCone</a> will be defined on basis of the volume's tangents along the first parameter direction. If 2, the second parameter direction will be used. If 3, the third parameter direction will be used.
---------------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

#### Returns

a [DirectionCone](#) (not necessarily the smallest) containing all tangents to this volume along the specified parameter direction.

Implemented in [Go::SplineVolume](#), [Go::TorusVolume](#), [Go::SphereVolume](#), [Go::ConeVolume](#), [Go::CylinderVolume](#), and [Go::Parallelepiped](#).

29.335.3.14 `virtual void Go::ParamVolume::translate ( const Point & vec ) [pure virtual]`

Translate.

Implemented in [Go::SplineVolume](#), [Go::TorusVolume](#), [Go::SphereVolume](#), [Go::ConeVolume](#), [Go::CylinderVolume](#), and [Go::Parallelepiped](#).

The documentation for this class was generated from the following file:

- [trivariate/include/GoTools/trivariate/ParamVolume.h](#)

## 29.336 PathType Class Reference

```
#include <PrPathTriangleSeq.h>
```

### Public Member Functions

- void [initPathType](#) ()
- void [funnelOfPath](#) ([PathType](#) prec)
  - Computes the funnel for the funnel algorithm.*
- void [modification\\_l\\_path\\_](#) (int new\_pt, const vector< [UnfNodeType](#) > &nodes\_unf)
  - Modification of the left path in the funnel algorithm.*
- void [modification\\_r\\_path\\_](#) (int new\_pt, const vector< [UnfNodeType](#) > &nodes\_unf)
  - Modification of the right path in the funnel algorithm.*
- void [print](#) ()

### Public Attributes

- std::vector< int > [l\\_path\\_](#)
- std::vector< int > [r\\_path\\_](#)
- std::vector< int > [funnel\\_](#)

#### 29.336.1 Detailed Description

[PathType](#) - used in [PrPathTriangleSeq](#)

Definition at line 83 of file [PrPathTriangleSeq.h](#).

#### 29.336.2 Member Function Documentation

##### 29.336.2.1 void PathType::funnelOfPath ( PathType prec )

Computes the funnel for the funnel algorithm.

##### 29.336.2.2 void PathType::initPathType ( )

##### 29.336.2.3 void PathType::modification\_l\_path\_ ( int new\_pt, const vector< UnfNodeType > & nodes\_unf )

Modification of the left path in the funnel algorithm.

29.336.2.4 void PathType::modification\_r\_path\_ ( int *new\_pt*, const vector< UnfNodeType > & *nodes\_unf* )

Modification of the right path in the funnel algorithm.

29.336.2.5 void PathType::print ( )

### 29.336.3 Member Data Documentation

29.336.3.1 std::vector<int> PathType::funnel\_

Definition at line 88 of file PrPathTriangleSeq.h.

29.336.3.2 std::vector<int> PathType::l\_path\_

Definition at line 86 of file PrPathTriangleSeq.h.

29.336.3.3 std::vector<int> PathType::r\_path\_

Definition at line 87 of file PrPathTriangleSeq.h.

The documentation for this class was generated from the following file:

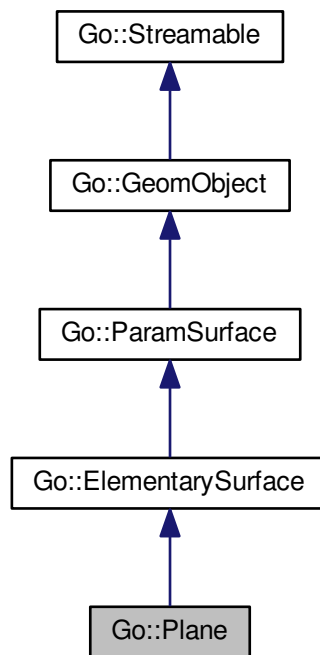
- parametrization/include/GoTools/parametrization/[PrPathTriangleSeq.h](#)

## 29.337 Go::Plane Class Reference

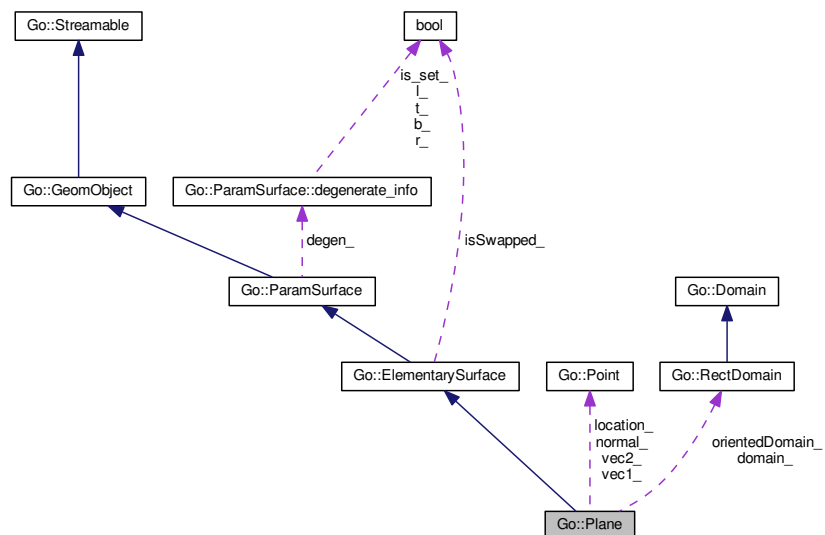
Class that represents a plane. It is a subclass of [ElementarySurface](#), and thus has a parametrization and is non-selfintersecting.

```
#include <Plane.h>
```

Inheritance diagram for Go::Plane:



Collaboration diagram for Go::Plane:



## Public Member Functions

- [Plane \(\)](#)

- [Plane](#) ([Point](#) location, [Point](#) normal, [bool](#) isSwapped=false)
  - Constructor. Input is location and normal.*
- [Plane](#) ([Point](#) location, [Point](#) normal, [Point](#) x\_axis, [bool](#) isSwapped=false)
- [Plane](#) ([double](#) a, [double](#) b, [double](#) c, [double](#) d, [bool](#) isSwapped=false)
- virtual [~Plane](#) ()
  - virtual destructor - ensures safe inheritance*
- virtual void [read](#) ([std::istream](#) &is)
- virtual void [write](#) ([std::ostream](#) &os) [const](#)
- virtual int [dimension](#) () [const](#)
  - Return the dimension of the space in which the object lies (usually 2 or 3)*
- virtual [ClassType](#) [instanceType](#) () [const](#)
  - Return the class type identifier of a given, derived instance of [GeomObject](#).*
- virtual [BoundingBox](#) [boundingBox](#) () [const](#)
  - Return empty box if infinite plane.*
- virtual [Plane](#) \* [clone](#) () [const](#)
- [const](#) [RectDomain](#) & [parameterDomain](#) () [const](#)
- [std::vector](#)< [CurveLoop](#) > [allBoundaryLoops](#) ([double](#) degenerate\_epsilon=DEFAULT\_SPACE\_EPSILON) [const](#)
- [DirectionCone](#) [normalCone](#) () [const](#)
- [DirectionCone](#) [tangentCone](#) ([bool](#) pdir\_is\_u) [const](#)
- void [point](#) ([Point](#) &pt, [double](#) upar, [double](#) vpar) [const](#)
- void [point](#) ([std::vector](#)< [Point](#) > &pts, [double](#) upar, [double](#) vpar, int derivs, [bool](#) u\_from\_right=true, [bool](#) v\_from\_right=true, [double](#) resolution=1.0e-12) [const](#)
- void [normal](#) ([Point](#) &n, [double](#) upar, [double](#) vpar) [const](#)
- [std::vector](#)< [shared\\_ptr](#)< [ParamCurve](#) > > [constParamCurves](#) ([double](#) parameter, [bool](#) pdir\_is\_u) [const](#)
- [std::vector](#)< [shared\\_ptr](#)< [ParamSurface](#) > > [subSurfaces](#) ([double](#) from\_upar, [double](#) from\_vpar, [double](#) to\_upar, [double](#) to\_vpar, [double](#) fuzzy=DEFAULT\_PARAMETER\_EPSILON) [const](#)
- [double](#) [nextSegmentVal](#) (int dir, [double](#) par, [bool](#) forward, [double](#) tol) [const](#)
- void [closestPoint](#) ([const](#) [Point](#) &pt, [double](#) &clo\_u, [double](#) &clo\_v, [Point](#) &clo\_pt, [double](#) &clo\_dist, [double](#) epsilon, [const](#) [RectDomain](#) \*domain\_of\_interest=NULL, [double](#) \*seed=0) [const](#)
- void [closestBoundaryPoint](#) ([const](#) [Point](#) &pt, [double](#) &clo\_u, [double](#) &clo\_v, [Point](#) &clo\_pt, [double](#) &clo\_dist, [double](#) epsilon, [const](#) [RectDomain](#) \*rd=NULL, [double](#) \*seed=0) [const](#)
- void [getBoundaryInfo](#) ([Point](#) &pt1, [Point](#) &pt2, [double](#) epsilon, [SplineCurve](#) \*&cv, [SplineCurve](#) \*&crosscv, [double](#) knot\_tol=1e-05) [const](#)
- [bool](#) [isDegenerate](#) ([bool](#) &b, [bool](#) &r, [bool](#) &t, [bool](#) &l, [double](#) tolerance) [const](#)
- virtual void [getDegenerateCorners](#) ([std::vector](#)< [Point](#) > &deg\_corners, [double](#) tol) [const](#)
  - Check for parallel and anti parallel partial derivatives in surface corners.*
- [Point](#) [getPoint](#) ()
  - Point in plane.*
- [Point](#) [getNormal](#) ()
- void [getSpanningVectors](#) ([Point](#) &axis1, [Point](#) &axis2)
  - Vectors in plane.*
- [Point](#) [projectPoint](#) ([const](#) [Point](#) &pnt) [const](#)
  - Projection of the point pnt in the plane.*
- [double](#) [distance](#) ([const](#) [Point](#) &pnt) [const](#)
  - Distance between the point pnt and the plane.*
- virtual void [setParameterBounds](#) ([double](#) from\_upar, [double](#) from\_vpar, [double](#) to\_upar, [double](#) to\_vpar)
- [Plane](#) \* [subSurface](#) ([double](#) from\_upar, [double](#) from\_vpar, [double](#) to\_upar, [double](#) to\_vpar, [double](#) fuzzy=DEFAULT\_PARAMETER\_EPSILON) [const](#)
  - Fetch a part of the plane.*
- virtual [SplineSurface](#) \* [geometrySurface](#) () [const](#)
  - Create a [SplineSurface](#) representation of the [Plane](#).*
- virtual [SplineSurface](#) \* [createSplineSurface](#) () [const](#)

Create a [SplineSurface](#) representation of the [Plane](#).

- [bool isBounded](#) ( ) const
- [bool isClosed](#) (bool &closed\_dir\_u, bool &closed\_dir\_v) const

Check if the plane is closed. Virtual function - always false.

- [Plane \\* intersect](#) (const [RotatedBox](#) &bd\_box) const
- virtual [bool isPlanar](#) ([Point](#) &normal, double tol)

Confirm that the surface is a plane and return the plane normal.

- virtual [bool isLinear](#) ([Point](#) &dir1, [Point](#) &dir2, double tol)

### Static Public Member Functions

- static [ClassType classType](#) ( )

### Protected Member Functions

- void [setSpanningVectors](#) ( )
- void [setSpanningVectorsSafe](#) ( )

### Protected Attributes

- [Point location\\_](#)
- [Point normal\\_](#)
- [Point vec1\\_](#)
- [Point vec2\\_](#)
- [RectDomain domain\\_](#)
- [RectDomain orientedDomain\\_](#)

## 29.337.1 Detailed Description

Class that represents a plane. It is a subclass of [ElementarySurface](#), and thus has a parametrization and is non-selfintersecting.

A [Plane](#) has a natural parametrization in terms of its location  $\mathbf{C}$  and spanning vectors  $\mathbf{x}$  and  $\mathbf{y}$ :  $p(u, v) = \mathbf{C} + u\mathbf{x} + v\mathbf{y}$ . This parametrization might be unbounded:  $-\infty < u, v < \infty$ .

Definition at line 63 of file [Plane.h](#).

## 29.337.2 Constructor & Destructor Documentation

### 29.337.2.1 [Go::Plane::Plane](#) ( ) [inline]

Default constructor. Constructs an uninitialized [Plane](#) which can only be assigned to or read into.

Definition at line 68 of file [Plane.h](#).

### 29.337.2.2 [Go::Plane::Plane](#) ( [Point location](#), [Point normal](#), bool *isSwapped* = false )

Constructor. Input is location and normal.

29.337.2.3 `Go::Plane::Plane ( Point location, Point normal, Point x_axis, bool isSwapped = false )`

Constructor. Input is location, normal and (local, approximate) x-axis

29.337.2.4 `Go::Plane::Plane ( double a, double b, double c, double d, bool isSwapped = false )`

Constructor. Input is coefficients of implicit equation + point suggestion

29.337.2.5 `virtual Go::Plane::~~Plane ( ) [virtual]`

virtual destructor - ensures safe inheritance

### 29.337.3 Member Function Documentation

29.337.3.1 `std::vector<CurveLoop> Go::Plane::allBoundaryLoops ( double degenerate_epsilon = DEFAULT_SPACE_EPSILON ) const [virtual]`

Returns the anticlockwise outer boundary loop of the surface, together with clockwise loops of any interior boundaries, such that the surface always is 'to the left of' the loops.

#### Parameters

<code>degenerate_epsilon</code>	edges whose length is smaller than this value are ignored.
---------------------------------	------------------------------------------------------------

#### Returns

a vector containing CurveLoops. The first of these describe the outer boundary of the surface (clockwise), whereas the others describe boundaries of interior holes (clockwise).

Implements [Go::ParamSurface](#).

29.337.3.2 `virtual BoundingBox Go::Plane::boundingBox ( ) const [virtual]`

Return empty box if infinite plane.

Implements [Go::GeomObject](#).

29.337.3.3 `static ClassType Go::Plane::classType ( ) [inline],[static]`

Definition at line 102 of file Plane.h.

29.337.3.4 `virtual Plane* Go::Plane::clone ( ) const [virtual]`

make a clone of this surface and return a pointer to it (user is responsible for clearing up memory afterwards).

#### Returns

pointer to cloned object

Implements [Go::ElementarySurface](#).

29.337.3.5 `void Go::Plane::closestBoundaryPoint ( const Point & pt, double & clo_u, double & clo_v, Point & clo_pt, double & clo_dist, double epsilon, const RectDomain * rd=NULL, double * seed=0 ) const [virtual]`

Iterates to the closest point to pt on the boundary of the surface.

#### See also

[closestPoint\(\)](#)

Implements [Go::ParamSurface](#).

29.337.3.6 `void Go::Plane::closestPoint ( const Point & pt, double & clo_u, double & clo_v, Point & clo_pt, double & clo_dist, double epsilon, const RectDomain * domain_of_interest=NULL, double * seed=0 ) const [virtual]`

Iterates to the closest point to pt on the surface.

#### Parameters

<i>pt</i>	the point to find the closest point to
<i>clo_u</i>	u parameter of the closest point
<i>clo_v</i>	v parameter of the closest point
<i>clo_pt</i>	the geometric position of the closest point
<i>clo_dist</i>	the distance between pt and clo_pt
<i>epsilon</i>	parameter tolerance (will in any case not be higher than sqrt(machine_precision) x magnitude of solution)
<i>domain_of_interest</i>	pointer to parameter domain in which to search for closest point. If a NULL pointer is used, the entire surface is searched.
<i>seed</i>	pointer to parameter values where iteration starts.

Reimplemented from [Go::ParamSurface](#).

29.337.3.7 `std::vector<shared_ptr<ParamCurve>> Go::Plane::constParamCurves ( double parameter, bool pardir_is_u ) const [virtual]`

Get the curve(s) obtained by intersecting the surface with one of its constant parameter curves. For surfaces without holes, this will be the parameter curve itself; for surfaces with interior holes this may be a collection of several, disjoint curves.



## Parameters

<i>parameter</i>	parameter value for the constant parameter (either u or v)
<i>pardir_is_u</i>	specify whether the <i>moving</i> parameter (as opposed to the <i>constant</i> parameter) is the first ('true') or the second ('false') one.

## Returns

a vector containing shared pointers to the obtained, newly constructed constant-parameter curves.

Implements [Go::ParamSurface](#).

**29.337.3.8** `virtual SplineSurface* Go::Plane::createSplineSurface ( ) const` [virtual]

Create a [SplineSurface](#) representation of the [Plane](#).

Implements [Go::ElementarySurface](#).

**29.337.3.9** `virtual int Go::Plane::dimension ( ) const` [virtual]

Return the dimension of the space in which the object lies (usually 2 or 3)

Implements [Go::GeomObject](#).

**29.337.3.10** `double Go::Plane::distance ( const Point & pnt ) const`

Distance between the point pnt and the plane.

**29.337.3.11** `virtual SplineSurface* Go::Plane::geometrySurface ( ) const` [virtual]

Create a [SplineSurface](#) representation of the [Plane](#).

Implements [Go::ElementarySurface](#).

**29.337.3.12** `void Go::Plane::getBoundaryInfo ( Point & pt1, Point & pt2, double epsilon, SplineCurve *& cv, SplineCurve *& crosscv, double knot_tol = 1e-05 ) const` [virtual]

Get the boundary curve segment between two points on the boundary, as well as the cross-tangent curve. If the given points are not positioned on the same boundary (within a certain tolerance), no curves will be created.

## Parameters

<i>pt1</i>	the first point on the boundary, given by the user
<i>pt2</i>	the second point on the boundary, given by the user
<i>epsilon</i>	the tolerance used when determining whether the given points are lying on a boundary, and if they do, whether they both lie on the <i>same</i> boundary.
<i>cv</i>	upon return, this will point to a newly created curve representing the boundary curve between 'pt1' and 'pt2'. The user assumes ownership of this object and is responsible for its deletion. No curve is created if the given points are not found to lie on the same boundary.
<i>crosscv</i>	upon return, this will point to a newly created curve representing the cross-boundary curve between 'pt1' and 'pt2'. The user assumes ownership of this object and is responsible for its deletion. The

Implements [Go::ParamSurface](#).

**29.337.3.13** `virtual void Go::Plane::getDegenerateCorners ( std::vector< Point > & deg_corners, double tol ) const`  
`[virtual]`

Check for paralell and anti paralell partial derivatives in surface corners.

Implements [Go::ParamSurface](#).

**29.337.3.14** `Point Go::Plane::getNormal ( ) [inline]`

[Plane](#) normal. NB: This function returns the "defining normal" and does not take swapped parameter directions into account! To get the "oriented normal", use `Point::normal()`

Definition at line 180 of file `Plane.h`.

**29.337.3.15** `Point Go::Plane::getPoint ( ) [inline]`

[Point](#) in plane.

Definition at line 174 of file `Plane.h`.

**29.337.3.16** `void Go::Plane::getSpanningVectors ( Point & axis1, Point & axis2 ) [inline]`

Vectors in plane.

Definition at line 184 of file `Plane.h`.

**29.337.3.17** `virtual ClassType Go::Plane::instanceType ( ) const [virtual]`

Return the class type identifier of a given, derived instance of [GeomObject](#).

Implements [Go::GeomObject](#).

**29.337.3.18** `Plane* Go::Plane::intersect ( const RotatedBox & bd_box ) const`

Return the result from intersecting the unbounded plane with a rotated bounding box (having `axis[0]=vec1_↔`, `axis[1]=vec2_`, `axis[2]=normal_`). Useful for visualizing the (unbounded) plane. If intersection is empty, the returned plane is the NULL pointer. The rotated box may be created from a boundingbox by defining the coordinate system and the 8 corner points of the `bd_box`.

**29.337.3.19** `bool Go::Plane::isBounded ( ) const [virtual]`

Query if parametrization is bounded. All four parameter bounds must be finite for this to be true.

Returns

*true* if bounded, *false* otherwise

Reimplemented from [Go::ElementarySurface](#).

29.337.3.20 `bool Go::Plane::isClosed ( bool & closed_dir_u, bool & closed_dir_v ) const` [virtual]

Check if the plane is closed. Virtual function - always false.

Reimplemented from [Go::ElementarySurface](#).

29.337.3.21 `bool Go::Plane::isDegenerate ( bool & b, bool & r, bool & t, bool & l, double tolerance ) const`  
[virtual]

The order of the edge indicators (bottom, right, top, left) matches the `edge_number` of `edgeCurve()`. Query whether any of the four boundary curves are degenerate (zero length) within a certain tolerance. In the below, we refer to 'u' as the first parameter and 'v' as the second.

#### Parameters

<i>b</i>	'true' upon return of function if the boundary ( $v = v_{\min}$ ) is degenerate
<i>r</i>	'true' upon return of function if the boundary ( $v = v_{\max}$ ) is degenerate
<i>t</i>	'true' upon return of function if the boundary ( $u = u_{\min}$ ) is degenerate
<i>l</i>	'true' upon return of function if the boundary ( $u = u_{\max}$ ) is degenerate
<i>tolerance</i>	boundaries are considered degenerate if their length is shorter than this value, given by the user

#### Returns

'true' if at least one boundary curve was found to be degenerate, 'false' otherwise.

Reimplemented from [Go::ParamSurface](#).

29.337.3.22 `virtual bool Go::Plane::isLinear ( Point & dir1, Point & dir2, double tol )` [virtual]

The surface is linear in all directions. Fetch the parameter directions

Reimplemented from [Go::ParamSurface](#).

29.337.3.23 `virtual bool Go::Plane::isPlanar ( Point & normal, double tol )` [virtual]

Confirm that the surface is a plane and return the plane normal.

Reimplemented from [Go::ParamSurface](#).

29.337.3.24 `double Go::Plane::nextSegmentVal ( int dir, double par, bool forward, double tol ) const` [virtual]

Determine the parameter value of the start of the 'next segment' from a parameter value, along a given parameter direction. A 'segment' is here defined as a parameter interval in which there will be no discontinuities in derivatives or other artifacts. For spline objects, a segment will typically be the interval between two consecutive, non-coincident knots.

## Parameters

<i>dir</i>	the parameter direction in which we search for the next segment (0 or 1)
<i>par</i>	the parameter value starting from which we search for the start value of the next segment
<i>forward</i>	define whether we shall move forward ('true') or backwards when searching along this parameter
<i>tol</i>	tolerance used for determining whether the 'par' is already located <i>on</i> the next segment value

## Returns

the value of the start value of the next segment (or the end of the previous segment, if we are moving backwards...)

Implements [Go::ParamSurface](#).

29.337.3.25 `void Go::Plane::normal ( Point & n, double upar, double vpar ) const [virtual]`

Evaluates the surface normal for a given parameter pair

## Parameters

<i>n</i>	the computed normal will be written to this variable
<i>upar</i>	the first parameter
<i>vpar</i>	the second parameter

Implements [Go::ParamSurface](#).

29.337.3.26 `DirectionCone Go::Plane::normalCone ( ) const [virtual]`

Creates a [DirectionCone](#) covering all normals to this surface.

## Returns

a [DirectionCone](#) (not necessarily the smallest) containing all normals to this surface.

Implements [Go::ParamSurface](#).

29.337.3.27 `const RectDomain& Go::Plane::parameterDomain ( ) const [virtual]`

Return the parameter domain of the surface. This may be a simple rectangular domain ([RectDomain](#)) or any other subclass of [Domain](#) (such as [GoCurveBoundedDomain](#), found in the `sisl_dependent` module).

## Returns

a [Domain](#) object describing the parametric domain of the surface

Implements [Go::ParamSurface](#).

29.337.3.28 `void Go::Plane::point ( Point & pt, double upar, double vpar ) const [virtual]`

Evaluates the surface's position for a given parameter pair.

## Parameters

<i>pt</i>	the result of the evaluation is written here
<i>upar</i>	the first parameter
<i>vpar</i>	the second parameter

Implements [Go::ParamSurface](#).

**29.337.3.29** `void Go::Plane::point ( std::vector< Point > & pts, double upar, double vpar, int derivs, bool u_from_right = true, bool v_from_right = true, double resolution = 1.0e-12 ) const [virtual]`

Evaluates the surface's position and a certain number of derivatives for a given parameter pair.

## Parameters

<i>pts</i>	the vector containing the evaluated values. Its size must be set by the user prior to calling this function, and should be equal to $(derivs+1) * (derivs+2) / 2$ . Upon completion of the function, its first entry is the surface's position at the given parameter pair. Then, if 'derivs' > 0, the two next entries will be the surface tangents along the first and second parameter direction. The next three entries are the second- and cross derivatives, in the order (du2, dudv, dv2), and similar for even higher derivatives.
<i>upar</i>	the first parameter
<i>vpar</i>	the second parameter
<i>derivs</i>	number of requested derivatives
<i>u_from_right</i>	specify whether derivatives along the first parameter are to be calculated from the right ('true', default) or from the left ('false')
<i>v_from_right</i>	specify whether derivatives along the second parameter are to be calculated from the right ('true', default) or from the left ('false')
<i>resolution</i>	tolerance used when determining whether parameters are located at special values of the parameter domain (in particular; knot values in case of spline objects).

Implements [Go::ParamSurface](#).

**29.337.3.30** `Point Go::Plane::projectPoint ( const Point & pnt ) const`

Projection of the point pnt in the plane.

**29.337.3.31** `virtual void Go::Plane::read ( std::istream & is ) [virtual]`

read object from stream

## Parameters

<i>is</i>	stream from which object is read
-----------	----------------------------------

Implements [Go::Streamable](#).

29.337.3.32 `virtual void Go::Plane::setParameterBounds ( double from_upar, double from_vpar, double to_upar, double to_vpar ) [virtual]`

Restrict the plane by restricting the parameter domain. It is initially infinite

Implements [Go::ElementarySurface](#).

29.337.3.33 `void Go::Plane::setSpanningVectors ( ) [protected]`

29.337.3.34 `void Go::Plane::setSpanningVectorsSafe ( ) [protected]`

29.337.3.35 `Plane* Go::Plane::subSurface ( double from_upar, double from_vpar, double to_upar, double to_vpar, double fuzzy = DEFAULT_PARAMETER_EPSILON ) const`

Fetch a part of the plane.

29.337.3.36 `std::vector<shared_ptr<ParamSurface>> Go::Plane::subSurfaces ( double from_upar, double from_vpar, double to_upar, double to_vpar, double fuzzy = DEFAULT_PARAMETER_EPSILON ) const [virtual]`

Get the surface(s) obtained by cropping the parameter domain of this surface between given values for the first and second parameter. In general, for surfaces with no interior holes, the result will be *one* surface; however, for surfaces with interior holes, the result might be *several disjoint* surfaces.

#### Parameters

<i>from_upar</i>	lower value for the first parameter in the subdomain
<i>from_vpar</i>	lower value for the second parameter in the subdomain
<i>to_upar</i>	upper value for the first parameter in the subdomain
<i>to_vpar</i>	upper value for the second parameter in the subdomain
<i>fuzzy</i>	tolerance used when determining intersection with interior boundaries

#### Returns

a vector contained shared pointers to the obtained, newly constructed sub-surfaces.

Implements [Go::ParamSurface](#).

29.337.3.37 `DirectionCone Go::Plane::tangentCone ( bool pardir_is_u ) const [virtual]`

Creates a [DirectionCone](#) covering all tangents to this surface along a given parameter direction.

#### Parameters

<i>pardir_is_u</i>	if 'true', then the <a href="#">DirectionCone</a> will be defined on basis of the surface's tangents along the first parameter direction. Otherwise the second parameter direction will be used.
--------------------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

**Returns**

a [DirectionCone](#) (not necessarily the smallest) containing all tangents to this surface along the specified parameter direction.

Implements [Go::ParamSurface](#).

**29.337.3.38** `virtual void Go::Plane::write ( std::ostream & os ) const` `[virtual]`

write object to stream

**Parameters**

<code>os</code>	stream to which object is written
-----------------	-----------------------------------

Implements [Go::Streamable](#).

**29.337.4 Member Data Documentation**

**29.337.4.1** `RectDomain Go::Plane::domain_` `[protected]`

Definition at line 246 of file Plane.h.

**29.337.4.2** `Point Go::Plane::location_` `[protected]`

Definition at line 238 of file Plane.h.

**29.337.4.3** `Point Go::Plane::normal_` `[protected]`

Definition at line 242 of file Plane.h.

**29.337.4.4** `RectDomain Go::Plane::orientedDomain_` `[mutable], [protected]`

Definition at line 247 of file Plane.h.

**29.337.4.5** `Point Go::Plane::vec1_` `[protected]`

Definition at line 243 of file Plane.h.

**29.337.4.6** `Point Go::Plane::vec2_` `[protected]`

Definition at line 244 of file Plane.h.

The documentation for this class was generated from the following file:

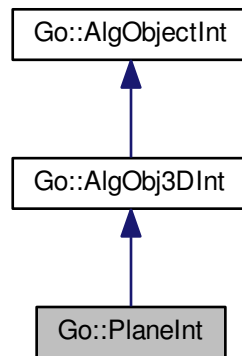
- `gotools-core/include/GoTools/geometry/Plane.h`

## 29.338 Go::PlaneInt Class Reference

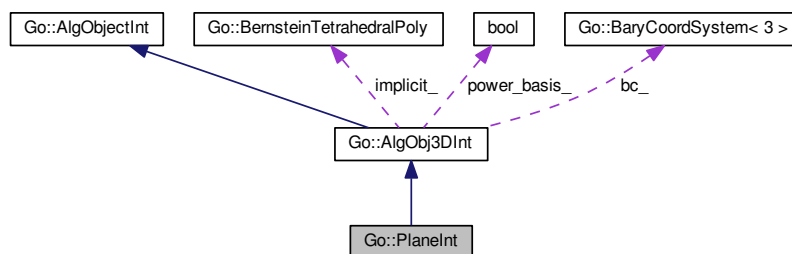
Class representing planar algebraic intersection objects.

```
#include <PlaneInt.h>
```

Inheritance diagram for Go::PlaneInt:



Collaboration diagram for Go::PlaneInt:



### Public Member Functions

- [PlaneInt](#) ()
  - [PlaneInt](#) ([Point](#) point, [Point](#) normal)
  - [PlaneInt](#) ([double](#) a, [double](#) b, [double](#) c, [double](#) d)
  - [virtual](#) [~PlaneInt](#) ()
- Constructor.*
- [void](#) [read](#) ([std::istream](#) &is)
  - [double](#) [a](#) () [const](#)
  - [double](#) [b](#) () [const](#)
  - [double](#) [c](#) () [const](#)
  - [double](#) [d](#) () [const](#)
  - [shared\\_ptr](#)< [SplineSurface](#) > [surface](#) ([Point](#) mid\_pt, [double](#) length\_x, [double](#) length\_y) [const](#)



## Additional Inherited Members

### 29.338.1 Detailed Description

Class representing planar algebraic intersection objects.

Definition at line 57 of file PlaneInt.h.

### 29.338.2 Constructor & Destructor Documentation

#### 29.338.2.1 Go::PlaneInt::PlaneInt ( )

Constructor. Used when reading from file.

#### 29.338.2.2 Go::PlaneInt::PlaneInt ( Point *point*, Point *normal* )

Constructor.

##### Parameters

<i>point</i>	reference point in the plane.
<i>normal</i>	normal to the plane.

#### 29.338.2.3 Go::PlaneInt::PlaneInt ( double *a*, double *b*, double *c*, double *d* )

Constructor. The plane is described by the expression  $ax + by + cz + d = 0$ .

##### Parameters

<i>a</i>	the x multiplier.
<i>b</i>	the y multiplier.
<i>c</i>	the z multiplier.
<i>d</i>	the constant.

#### 29.338.2.4 virtual Go::PlaneInt::~~PlaneInt ( ) [virtual]

Constructor.

### 29.338.3 Member Function Documentation

#### 29.338.3.1 double Go::PlaneInt::a ( ) const

Get the x multiplier.

**Returns**

a The x multiplier.

**29.338.3.2 double Go::PlaneInt::b ( ) const**

Get the y multiplier.

**Returns**

b The y multiplier.

**29.338.3.3 double Go::PlaneInt::c ( ) const**

Get the z multiplier.

**Returns**

c The z multiplier.

**29.338.3.4 double Go::PlaneInt::d ( ) const**

Get the constant.

**Returns**

d The constant.

**29.338.3.5 void Go::PlaneInt::read ( std::istream & *is* )**

Read a plane description from file.

**Parameters**

<i>is</i>	the stream containing the plane description.
-----------	----------------------------------------------

**29.338.3.6 shared\_ptr<SplineSurface> Go::PlaneInt::surface ( Point *mid\_pt*, double *length\_x*, double *length\_y* ) const**

Get a SplineSurface representing the plane. This can be used to visualize the object. Since the object is to be tessellated we make sure it is bounded.

**Parameters**

<i>mid_pt</i>	the intersection of the diagonals in a rectangle in the plane.
---------------	----------------------------------------------------------------

## Parameters

<i>length</i> ↔ _x	the length of the rectangle in the x direction.
<i>length</i> ↔ _y	the length of the rectangle in the y direction.

## Returns

The spline surface of the plane.

The documentation for this class was generated from the following file:

- intersections/include/GoTools/intersections/[PlaneInt.h](#)

## 29.339 Go::Point Class Reference

```
#include <Point.h>
```

### Public Member Functions

- [Point](#) ()
- [Point](#) (int dim)
- [Point](#) (double x, double y)
- [Point](#) (double x, double y, double z)
- [template<typename T, int Dim>](#)  
[Point](#) (const [Array](#)< T, Dim > &v)
- [template<typename RandomAccessIterator >](#)  
[Point](#) (RandomAccessIterator first, RandomAccessIterator last)  
*Constructor making a [Point](#) from an iterator range.*
- [Point](#) (double \*begin, double \*end, bool own)
- [Point](#) (const [Point](#) &v)  
*Copy constructor.*
- [Point](#) & [operator=](#) (const [Point](#) &v)  
*Assignment operator.*
- [~Point](#) ()  
*Destructor.*
- void [swap](#) ([Point](#) &other)  
*Swaps two [Point](#) instances. Never throws.*
- void [read](#) (std::istream &is)
- void [write](#) (std::ostream &os) const
- const double & [operator\[\]](#) (int i) const  
*Read-only index access.*
- double & [operator\[\]](#) (int i)  
*Index access.*
- const double \* [begin](#) () const  
*Get a read-only start iterator.*
- double \* [begin](#) ()

- Get a start iterator.*

  - `const double * end () const`

*Get a read-only end iterator.*
- `double * end ()`

*Get a end iterator.*
- `int size () const`

*Get the dimension of the `Point`.*
- `int dimension () const`

*Get the dimension of the `Point`. Same as `size()`.*
- `void resize (int d)`

*Changing dimension. This loses all info in the point.*
- `void setValue (double x, double y)`

*Set function for 2D.*
- `void setValue (double x, double y, double z)`

*Set function for 3D.*
- `void setValue (const double *array)`
- `void setValue (double val)`
- `void resetValue (int idx, double val)`
- `double length2 () const`
- `double length () const`
- `double lengthInf () const`
- `double dist2 (const Point &v) const`
- `double dist (const Point &v) const`
- `double distInf (const Point &v) const`
- `void normalize ()`
- `double normalize_checked ()`
- `Point operator+ (const Point &v) const`

*The sum of two vectors.*
- `void operator+= (const Point &v)`

*Add a vector to this vector.*
- `Point operator- (const Point &v) const`

*The difference between two vectors.*
- `void operator-= (const Point &v)`

*Subtract a vector from this vector.*
- `Point operator* (double d) const`

*The product of a vector and a scalar.*
- `void operator*= (double d)`

*Multiply this vector by a scalar.*
- `Point operator/ (double d) const`

*A vector divided by a scalar.*
- `void operator/= (double d)`

*Divide this vector with a scalar.*
- `Point operator- () const`

*The negation of a vector.*
- `double operator* (const Point &v) const`
- `Point operator% (const Point &v) const`
- `Point cross (const Point &v) const`
- `void setToCrossProd (const Point &u, const Point &v)`

*Set this `Point` to the cross product of two other `Points`.*
- `double cosAngle (const Point &v) const`
- `double angle (const Point &v) const`

*The angle between this and another vector. Range is  $[0.0, M\_PI]$ .*

- `double angle2 (const Point &v) const`  
*The angle between this and another vector of dimension 2. Range is [0.0, 2\*M\_PI].*
- `double angle_smallest (const Point &v) const`  
*of orientation. Range is [0.0, 0.5\*M\_PI].*

### 29.339.1 Detailed Description

Run-time sized point class. Encapsulates a point or vector with full value semantics, i.e. copying and assignment works as expected. The class has some vector algebra functionality, such as scalar product, multiplication by scalars etc, and objects will sometimes be called 'vectors' in the following. Based on double precision floating point numbers.

Definition at line 63 of file Point.h.

### 29.339.2 Constructor & Destructor Documentation

#### 29.339.2.1 `Go::Point::Point ( ) [inline]`

Default constructor, does not initialize elements. The only functions you are allowed to use with a default constructed (0-dim) `Point` are the assignment operator, `resize` and `setValue(...)`. This is not enforced.

Definition at line 75 of file Point.h.

#### 29.339.2.2 `Go::Point::Point ( int dim ) [inline],[explicit]`

Constructor taking a dimension argument. Resulting point is of the specified dimension, but not initialized.

Definition at line 81 of file Point.h.

#### 29.339.2.3 `Go::Point::Point ( double x, double y ) [inline]`

Constructor taking 2 arguments, makes the 2D-point (x,y).

Definition at line 86 of file Point.h.

#### 29.339.2.4 `Go::Point::Point ( double x, double y, double z ) [inline]`

Constructor taking 3 arguments, makes the 3D-point (x,y,z).

Definition at line 94 of file Point.h.

#### 29.339.2.5 `template<typename T , int Dim> Go::Point::Point ( const Array< T, Dim > & v ) [inline],[explicit]`

Constructor making a `Point` as a copy of an `Array`. The `Array` must have a value type convertible to double.

Definition at line 108 of file Point.h.

29.339.2.6 `template<typename RandomAccessIterator > Go::Point::Point ( RandomAccessIterator first, RandomAccessIterator last ) [inline]`

Constructor making a [Point](#) from an iterator range.

Definition at line 122 of file Point.h.

29.339.2.7 `Go::Point::Point ( double * begin, double * end, bool own ) [inline]`

Make a point from an existing range of doubles. If the own parameter is false, the [Point](#) will refer to the input data, and not own them. If it is true, it will act as a regular point, owning its data.

Definition at line 162 of file Point.h.

29.339.2.8 `Go::Point::Point ( const Point & v ) [inline]`

Copy constructor.

Definition at line 178 of file Point.h.

29.339.2.9 `Go::Point::~~Point ( ) [inline]`

Destructor.

Definition at line 198 of file Point.h.

### 29.339.3 Member Function Documentation

29.339.3.1 `double Go::Point::angle ( const Point & v ) const [inline]`

The angle between this and another vector. Range is [0.0, M\_PI].

Definition at line 505 of file Point.h.

29.339.3.2 `double Go::Point::angle2 ( const Point & v ) const [inline]`

The angle between this and another vector of dimension 2. Range is [0.0, 2\*M\_PI].

Definition at line 511 of file Point.h.

29.339.3.3 `double Go::Point::angle_smallest ( const Point & v ) const [inline]`

of orientation. Range is [0.0, 0.5\*M\_PI].

The smallest angle between this and another vector regardless

Definition at line 526 of file Point.h.

**29.339.3.4** `const double* Go::Point::begin ( ) const` `[inline]`

Get a read-only start iterator.

Definition at line 228 of file Point.h.

**29.339.3.5** `double* Go::Point::begin ( )` `[inline]`

Get a start iterator.

Definition at line 230 of file Point.h.

**29.339.3.6** `double Go::Point::cosAngle ( const Point & v ) const` `[inline]`

The cosine of the angle between this and another vector.

Definition at line 489 of file Point.h.

**29.339.3.7** `Point Go::Point::cross ( const Point & v ) const` `[inline]`

The cross product of two vectors. Throws if dimensions are not 3.

Definition at line 462 of file Point.h.

**29.339.3.8** `int Go::Point::dimension ( ) const` `[inline]`

Get the dimension of the [Point](#). Same as [size\(\)](#).

Definition at line 238 of file Point.h.

**29.339.3.9** `double Go::Point::dist ( const Point & v ) const` `[inline]`

Get the euclidian length of the difference between this vector and another vector.

Definition at line 333 of file Point.h.

**29.339.3.10** `double Go::Point::dist2 ( const Point & v ) const` `[inline]`

Get the square of the euclidian length of the difference between this vector and another vector.

Definition at line 320 of file Point.h.

**29.339.3.11** `double Go::Point::distInf ( const Point & v ) const` `[inline]`

Get the infinity-norm (or max-norm) length of the difference between this vector and another vector.

Definition at line 340 of file Point.h.

**29.339.3.12** `const double* Go::Point::end ( ) const` [inline]

Get a read-only end iterator.

Definition at line 232 of file Point.h.

**29.339.3.13** `double* Go::Point::end ( )` [inline]

Get a end iterator.

Definition at line 234 of file Point.h.

**29.339.3.14** `double Go::Point::length ( ) const` [inline]

Get the euclidian length of the vector.

Definition at line 303 of file Point.h.

**29.339.3.15** `double Go::Point::length2 ( ) const` [inline]

Get the square of the euclidian length of the vector.

Definition at line 294 of file Point.h.

**29.339.3.16** `double Go::Point::lengthInf ( ) const` [inline]

Get the infinity-norm (or max-norm) length of the vector.

Definition at line 309 of file Point.h.

**29.339.3.17** `void Go::Point::normalize ( )` [inline]

Normalize this vector, i.e. divide every element by [length\(\)](#).

Definition at line 352 of file Point.h.

**29.339.3.18** `double Go::Point::normalize_checked ( )` [inline]

Definition at line 359 of file Point.h.

**29.339.3.19** `Point Go::Point::operator%( const Point & v ) const` [inline]

The cross product of two vectors. Dimensions should be 3.

Definition at line 451 of file Point.h.



**29.339.3.20** `Point Go::Point::operator*( double d ) const` [inline]

The product of a vector and a scalar.

Definition at line 401 of file Point.h.

**29.339.3.21** `double Go::Point::operator*( const Point & v ) const` [inline]

The scalar product (or inner product, or dot product) of two vectors.

Definition at line 439 of file Point.h.

**29.339.3.22** `void Go::Point::operator*=( double d )` [inline]

Multiply this vector by a scalar.

Definition at line 408 of file Point.h.

**29.339.3.23** `Point Go::Point::operator+( const Point & v ) const` [inline]

The sum of two vectors.

Definition at line 370 of file Point.h.

**29.339.3.24** `void Go::Point::operator+=( const Point & v )` [inline]

Add a vector to this vector.

Definition at line 377 of file Point.h.

**29.339.3.25** `Point Go::Point::operator-( const Point & v ) const` [inline]

The difference between two vectors.

Definition at line 386 of file Point.h.

**29.339.3.26** `Point Go::Point::operator-( ) const` [inline]

The negation of a vector.

Definition at line 429 of file Point.h.

**29.339.3.27** `void Go::Point::operator-=( const Point & v )` [inline]

Subtract a vector from this vector.

Definition at line 393 of file Point.h.

**29.339.3.28** `Point Go::Point::operator/( double d ) const` `[inline]`

A vector divided by a scalar.

Definition at line 415 of file Point.h.

**29.339.3.29** `void Go::Point::operator/=( double d )` `[inline]`

Divide this vector with a scalar.

Definition at line 422 of file Point.h.

**29.339.3.30** `Point& Go::Point::operator=( const Point & v )` `[inline]`

Assignment operator.

Definition at line 190 of file Point.h.

**29.339.3.31** `const double& Go::Point::operator[]( int i ) const` `[inline]`

Read-only index access.

Definition at line 224 of file Point.h.

**29.339.3.32** `double& Go::Point::operator[]( int i )` `[inline]`

Index access.

Definition at line 226 of file Point.h.

**29.339.3.33** `void Go::Point::read ( std::istream & is )`

Reads a [Point](#) elementwise from a standard istream. The [Point](#) must already be initialized with the correct dimension.

**29.339.3.34** `void Go::Point::resetValue ( int idx, double val )` `[inline]`

Definition at line 285 of file Point.h.

**29.339.3.35** `void Go::Point::resize ( int d )` `[inline]`

Changing dimension. This loses all info in the point.

Definition at line 241 of file Point.h.

29.339.3.36 `void Go::Point::setToCrossProd ( const Point & u, const Point & v ) [inline]`

Set this [Point](#) to the cross product of two other Points.

Definition at line 468 of file Point.h.

29.339.3.37 `void Go::Point::setValue ( double x, double y ) [inline]`

Set function for 2D.

Definition at line 252 of file Point.h.

29.339.3.38 `void Go::Point::setValue ( double x, double y, double z ) [inline]`

Set function for 3D.

Definition at line 259 of file Point.h.

29.339.3.39 `void Go::Point::setValue ( const double * array ) [inline]`

Set function that copies values from an input array. Dimension must already be initialized!

Definition at line 269 of file Point.h.

29.339.3.40 `void Go::Point::setValue ( double val ) [inline]`

Set function that sets all elements equal to the input. Dimension must already be initialized!

Definition at line 278 of file Point.h.

29.339.3.41 `int Go::Point::size ( ) const [inline]`

Get the dimension of the [Point](#).

Definition at line 236 of file Point.h.

29.339.3.42 `void Go::Point::swap ( Point & other ) [inline]`

Swaps two [Point](#) instances. Never throws.

Definition at line 204 of file Point.h.

29.339.3.43 void Go::Point::write ( std::ostream & os ) const

Writes a [Point](#) elementwise to a standard ostream. Precision is set to 16 by this function. Dimension is not stored.

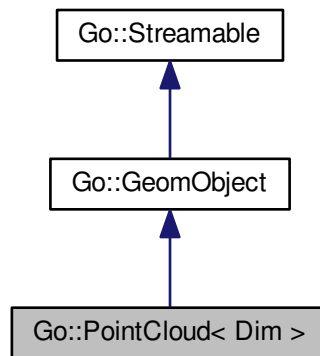
The documentation for this class was generated from the following file:

- [gotools-core/include/GoTools/utils/Point.h](#)

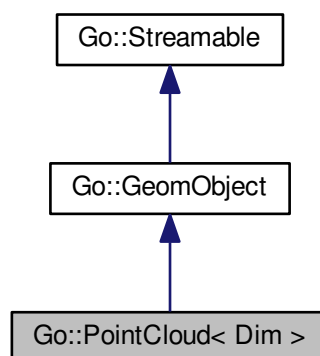
## 29.340 Go::PointCloud< Dim > Class Template Reference

```
#include <PointCloud.h>
```

Inheritance diagram for Go::PointCloud< Dim >:



Collaboration diagram for Go::PointCloud< Dim >:



## Public Member Functions

- [PointCloud](#) ()
- `template<typename ForwardIterator >`  
[PointCloud](#) (ForwardIterator start, int numpoints)
- [PointCloud](#) (std::vector< [Array](#)< double, Dim > > &points)  
*Constructor specifying a [PointCloud](#) from an [Array](#) of points.*
- virtual `~PointCloud` ()  
*Virtual destructor, enables safe inheritance.*
- virtual [BoundingBox](#) boundingBox () const  
*Return the object's bounding box.*
- virtual int [dimension](#) () const  
*Query the dimension of the space where the [PointCloud](#) lies.*
- virtual [ClassType](#) instanceType () const  
*Return the class type identifier of a given, derived instance of [GeomObject](#).*
- virtual [PointCloud](#) \* clone () const  
*Clone the [GeomObject](#) and return a pointer to the clone.*
- int [numPoints](#) () const
- [Array](#)< double, Dim > & [point](#) (int i)
- const [Array](#)< double, Dim > & [point](#) (int i) const
- double \* [rawData](#) ()
- const double \* [rawData](#) () const
- const std::vector< [Array](#)< double, Dim > > & [pointVector](#) ()
- void [translate](#) ([Array](#)< double, Dim > vec)
- void [rotate](#) (const [Vector3D](#) &p, const [Vector3D](#) &q)
- virtual void [read](#) (std::istream &is)
- virtual void [write](#) (std::ostream &os) const

## Static Public Member Functions

- static [ClassType](#) classType ()

### 29.340.1 Detailed Description

```
template<int Dim>
class Go::PointCloud< Dim >
```

Represent a cloud of points in 'Dim'-dimensional space.

Definition at line 61 of file PointCloud.h.

### 29.340.2 Constructor & Destructor Documentation

29.340.2.1 `template<int Dim> Go::PointCloud< Dim >::PointCloud ( ) [inline]`

Constructor for an empty [PointCloud](#) that can only be assigned to or read(...) into.

Definition at line 65 of file PointCloud.h.

29.340.2.2 `template<int Dim> template<typename ForwardIterator > Go::PointCloud< Dim >::PointCloud ( ForwardIterator start, int numpoints ) [inline]`

start is supposed to point to the start of the array to be copied, its valuetype should be convertible to double  
 Constructor specifying a [PointCloud](#) from a number of predefined points.

## Parameters

<i>start</i>	iterator to the beginning of an array where the defining points are stored (these will be copied to the objects internal datastructure).
<i>numpoints</i>	number of points to be read into the <a href="#">PointCloud</a>

Definition at line 77 of file PointCloud.h.

```
29.340.2.3 template<int Dim> Go::PointCloud< Dim >::PointCloud (std::vector< Array< double, Dim > &
 points) [inline]
```

Constructor specifying a [PointCloud](#) from an [Array](#) of points.

Definition at line 88 of file PointCloud.h.

```
29.340.2.4 template<int Dim> virtual Go::PointCloud< Dim >::~~PointCloud () [inline],[virtual]
```

Virtual destructor, enables safe inheritance.

Definition at line 93 of file PointCloud.h.

### 29.340.3 Member Function Documentation

```
29.340.3.1 template<int Dim> virtual BoundingBox Go::PointCloud< Dim >::boundingBox () const [inline],
 [virtual]
```

Return the object's bounding box.

Implements [Go::GeomObject](#).

Definition at line 97 of file PointCloud.h.

```
29.340.3.2 template<int Dim> static ClassType Go::PointCloud< Dim >::classType () [inline],[static]
```

Definition at line 113 of file PointCloud.h.

```
29.340.3.3 template<int Dim> virtual PointCloud* Go::PointCloud< Dim >::clone () const [inline],
 [virtual]
```

Clone the [GeomObject](#) and return a pointer to the clone.

Implements [Go::GeomObject](#).

Definition at line 126 of file PointCloud.h.

```
29.340.3.4 template<int Dim> virtual int Go::PointCloud< Dim >::dimension () const [inline],[virtual]
```

Query the dimension of the space where the [PointCloud](#) lies.

Implements [Go::GeomObject](#).

Definition at line 105 of file PointCloud.h.

```
29.340.3.5 template<int Dim> virtual ClassType Go::PointCloud< Dim >::instanceType () const [inline],
 [virtual]
```

Return the class type identifier of a given, derived instance of [GeomObject](#).

Implements [Go::GeomObject](#).

Definition at line 109 of file PointCloud.h.

```
29.340.3.6 template<int Dim> int Go::PointCloud< Dim >::numPoints () const [inline]
```

Query the number of points in the [PointCloud](#)

#### Returns

the number of points in the [PointCloud](#)

Definition at line 132 of file PointCloud.h.

```
29.340.3.7 template<int Dim> Array<double, Dim>& Go::PointCloud< Dim >::point (int i) [inline]
```

Get the coordinates of a point in the point cloud

#### Parameters

<i>i</i>	the index of the requested point
----------	----------------------------------

#### Returns

a reference to the [Array](#) containing the coordinates of the requested point

Definition at line 139 of file PointCloud.h.

```
29.340.3.8 template<int Dim> const Array<double, Dim>& Go::PointCloud< Dim >::point (int i) const
 [inline]
```

Get the coordinates of a point in the point cloud (const-version)

## Parameters

<i>i</i>	the index of the requested point
----------	----------------------------------

## Returns

a const reference to the [Array](#) containing the coordinates of the requested point

Definition at line 146 of file PointCloud.h.

```
29.340.3.9 template<int Dim> const std::vector<Array<double, Dim> >& Go::PointCloud< Dim >::pointVector ()
 [inline]
```

Get a reference to the vector where the points are stored

## Returns

a reference to the point coordinate vector

Definition at line 163 of file PointCloud.h.

```
29.340.3.10 template<int Dim> double* Go::PointCloud< Dim >::rawData () [inline]
```

Get a pointer to the memory area where the point coordinates are (consecutively) stored.

## Returns

a pointer to the coordinate storage area.

Definition at line 152 of file PointCloud.h.

```
29.340.3.11 template<int Dim> const double* Go::PointCloud< Dim >::rawData () const [inline]
```

Get a const pointer to the memory area where the point coordinates are (consecutively) stored.

## Returns

a constant pointer to the coordinate storage area

Definition at line 158 of file PointCloud.h.

```
29.340.3.12 template<int Dim> virtual void Go::PointCloud< Dim >::read (std::istream & is) [inline],
 [virtual]
```

read object from stream



## Parameters

<i>is</i>	stream from which object is read
-----------	----------------------------------

Implements [Go::Streamable](#).

Definition at line 192 of file PointCloud.h.

**29.340.3.13** `template<int Dim> void Go::PointCloud< Dim >::rotate ( const Vector3D & p, const Vector3D & q )`  
[inline]

Rotate by multiplying all points with a matrix that takes vector p to vector q Only for Dim == 3

Definition at line 177 of file PointCloud.h.

**29.340.3.14** `template<int Dim> void Go::PointCloud< Dim >::translate ( Array< double, Dim > vec )` [inline]

Definition at line 166 of file PointCloud.h.

**29.340.3.15** `template<int Dim> virtual void Go::PointCloud< Dim >::write ( std::ostream & os ) const` [inline],  
[virtual]

write object to stream

## Parameters

<i>os</i>	stream to which object is written
-----------	-----------------------------------

Implements [Go::Streamable](#).

Definition at line 228 of file PointCloud.h.

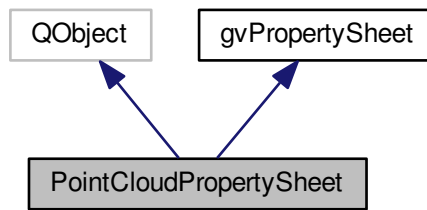
The documentation for this class was generated from the following file:

- [gotools-core/include/GoTools/geometry/PointCloud.h](#)

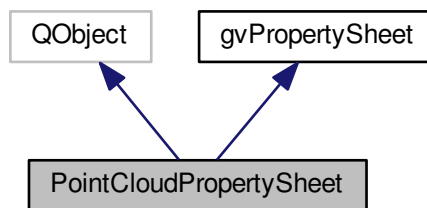
## 29.341 PointCloudPropertySheet Class Reference

```
#include <PointCloudPropertySheet.h>
```

Inheritance diagram for PointCloudPropertySheet:



Collaboration diagram for PointCloudPropertySheet:



## Public Slots

- void [apply](#) ()

## Public Member Functions

- [PointCloudPropertySheet](#) ([gvPointCloudPaintable](#) \*pable)
- virtual void [createSheet](#) ([QWidget](#) \*parent, [gvObserver](#) \*obs)

### 29.341.1 Detailed Description

Documentation ... The property sheet for PointCloud.

Definition at line 57 of file `PointCloudPropertySheet.h`.

## 29.341.2 Constructor & Destructor Documentation

29.341.2.1 `PointCloudPropertySheet::PointCloudPropertySheet ( gvPointCloudPaintable * pable )`

## 29.341.3 Member Function Documentation

29.341.3.1 `void PointCloudPropertySheet::apply ( ) [slot]`

29.341.3.2 `virtual void PointCloudPropertySheet::createSheet ( QWidget * parent, gvObserver * obs ) [virtual]`

Implements [gvPropertySheet](#).

The documentation for this class was generated from the following file:

- [viewlib/include/GoTools/viewlib/PointCloudPropertySheet.h](#)

## 29.342 Go::PointOnCurve Class Reference

```
#include <PointOnCurve.h>
```

### Public Member Functions

- [PointOnCurve \(\)](#)  
*Default constructor.*
- [PointOnCurve \(shared\\_ptr< ParamCurve > curve, double par\)](#)  
*Constructor taking a curve and a parameter value.*
- [PointOnCurve \(shared\\_ptr< ParamCurve > curve, Point pnt\)](#)  
*Constructor taking a curve and a point.*
- [~PointOnCurve \(\)](#)  
*Destructor.*
- [Point getPos \(\) const](#)
- [shared\\_ptr< ParamCurve > getCurve \(\) const](#)  
*Fetch the curve associated to the point.*
- [double getPar \(\) const](#)  
*Fetch the curve parameter associated to the point.*
- [void evaluate \(int der, std::vector< Point > &deriv\) const](#)  
*The curve is evaluated including der derivatives.*
- [void setParInterval \(double start, double end\)](#)  
*Add limiting information to the current curve.*

### 29.342.1 Detailed Description

Represents a point on a curve. Lean functionality for the time being, but should be extended to cover for instance curvature computations

Definition at line 53 of file `PointOnCurve.h`.

## 29.342.2 Constructor & Destructor Documentation

### 29.342.2.1 `Go::PointOnCurve::PointOnCurve ( )`

Default constructor.

### 29.342.2.2 `Go::PointOnCurve::PointOnCurve ( shared_ptr< ParamCurve > curve, double par )`

Constructor taking a curve and a parameter value.

### 29.342.2.3 `Go::PointOnCurve::PointOnCurve ( shared_ptr< ParamCurve > curve, Point pnt )`

Constructor taking a curve and a point.

### 29.342.2.4 `Go::PointOnCurve::~~PointOnCurve ( )`

Destructor.

## 29.342.3 Member Function Documentation

### 29.342.3.1 `void Go::PointOnCurve::evaluate ( int der, std::vector< Point > & deriv ) const`

The curve is evaluated including der derivatives.

### 29.342.3.2 `shared_ptr<ParamCurve> Go::PointOnCurve::getCurve ( ) const [inline]`

Fetch the curve associated to the point.

Definition at line 74 of file `PointOnCurve.h`.

### 29.342.3.3 `double Go::PointOnCurve::getPar ( ) const [inline]`

Fetch the curve parameter associated to the point.

Definition at line 80 of file `PointOnCurve.h`.

### 29.342.3.4 `Point Go::PointOnCurve::getPos ( ) const`

Evaluate If an input point is given, this is the point that will be returned even if it does not lie exactly on the curve

29.342.3.5 void Go::PointOnCurve::setParInterval ( double start, double end )

Add limiting information to the current curve.

The documentation for this class was generated from the following file:

- [gotools-core/include/GoTools/geometry/PointOnCurve.h](#)

## 29.343 Go::PointOnEdge Class Reference

```
#include <PointOnEdge.h>
```

### Public Member Functions

- [PointOnEdge](#) (ftEdge \*edge, double par)
- [~PointOnEdge](#) ()  
*Destructor.*
- [const Point & position](#) () const  
*Point position.*
- [ftEdge \\* edge](#) () const  
*Associated edge.*
- [double par](#) () const  
*Associated edge parameter.*

### 29.343.1 Detailed Description

[PointOnEdge](#) - represents a point lying on an edge. Storage.

Definition at line 56 of file PointOnEdge.h.

### 29.343.2 Constructor & Destructor Documentation

29.343.2.1 Go::PointOnEdge::PointOnEdge ( ftEdge \* edge, double par )

Constructor

Parameters

<i>the</i>	edge on which the point lies
<i>par</i>	associated edge parameter

29.343.2.2 Go::PointOnEdge::~~PointOnEdge ( )

Destructor.

### 29.343.3 Member Function Documentation

29.343.3.1 `ftEdge* Go::PointOnEdge::edge ( ) const` `[inline]`

Associated edge.

Definition at line 71 of file PointOnEdge.h.

29.343.3.2 `double Go::PointOnEdge::par ( ) const` `[inline]`

Associated edge parameter.

Definition at line 73 of file PointOnEdge.h.

29.343.3.3 `const Point& Go::PointOnEdge::position ( ) const` `[inline]`

[Point](#) position.

Definition at line 69 of file PointOnEdge.h.

The documentation for this class was generated from the following file:

- [compositemodel/include/GoTools/compositemodel/PointOnEdge.h](#)

## 29.344 Go::PointSequence Class Reference

```
#include <PointSequence.h>
```

### Public Member Functions

- [PointSequence](#) ()  
*Default constructor.*
- `template<typename RandomIterator >`  
[PointSequence](#) (int `dim`, int `nmb_pts`, RandomIterator `coefs_start`, [PointSequenceType](#) `pst=PSTPoint`)  
*Constructor given a sequence of points of a specified type.*
- `template<typename RandomIterator >`  
[PointSequence](#) (int `dim`, int `nmb_pts_1`, int `nmb_pts_2`, RandomIterator `coefs_start`, [PointSequenceType](#) `pst=PSTPoint`)
- `template<typename RandomIterator >`  
[PointSequence](#) (int `dim`, int `nmb_pts_1`, int `nmb_pts_2`, int `nmb_pts_3`, RandomIterator `coefs_start`, [PointSequenceType](#) `pst=PSTPoint`)  
*Constructor given a 3 dimensional grid of points of a specified type.*
- `virtual ~PointSequence` ()
- int `dimension` () `const`  
*Point dimension.*
- int `grid_dimension` () `const`  
*Dimension of grid.*
- int `grid_length` (int `i`) `const`

- Length of grid in the given direction.*
- [PointSequenceType type \(\) const](#)  
*Point type.*
- `std::vector< double >::iterator` [coefs\\_begin \(\)](#)  
*Iterator to grid start.*
- `std::vector< double >::iterator` [coefs\\_end \(\)](#)  
*Iterator to grid end.*
- `std::vector< double >::const_iterator` [coefs\\_begin \(\) const](#)  
*Constant iterator to grid start.*
- `std::vector< double >::const_iterator` [coefs\\_end \(\) const](#)  
*Constant iterator to grid end.*

### 29.344.1 Detailed Description

Definition at line 64 of file PointSequence.h.

### 29.344.2 Constructor & Destructor Documentation

#### 29.344.2.1 Go::PointSequence::PointSequence ( ) [inline]

Default constructor.

Definition at line 70 of file PointSequence.h.

#### 29.344.2.2 `template<typename RandomIterator > Go::PointSequence::PointSequence ( int dim, int nmb_pts, RandomIterator coefs_start, PointSequenceType pst = PSTPoint ) [inline]`

Constructor given a sequence of points of a specified type.

Definition at line 78 of file PointSequence.h.

#### 29.344.2.3 `template<typename RandomIterator > Go::PointSequence::PointSequence ( int dim, int nmb_pts_1, int nmb_pts_2, RandomIterator coefs_start, PointSequenceType pst = PSTPoint ) [inline]`

Constructor given a number of sequences of points of a specified type, Each point sequence must contain the same number of points, i.e. two dimensional grid

Definition at line 105 of file PointSequence.h.

#### 29.344.2.4 `template<typename RandomIterator > Go::PointSequence::PointSequence ( int dim, int nmb_pts_1, int nmb_pts_2, int nmb_pts_3, RandomIterator coefs_start, PointSequenceType pst = PSTPoint ) [inline]`

Constructor given a 3 dimensional grid of points of a specified type.

Definition at line 132 of file PointSequence.h.

29.344.2.5 `virtual Go::PointSequence::~~PointSequence ( ) [inline],[virtual]`

Definition at line 160 of file PointSequence.h.

### 29.344.3 Member Function Documentation

29.344.3.1 `std::vector<double>::iterator Go::PointSequence::coefs_begin ( ) [inline]`

Iterator to grid start.

Definition at line 176 of file PointSequence.h.

29.344.3.2 `std::vector<double>::const_iterator Go::PointSequence::coefs_begin ( ) const [inline]`

Constant iterator to grid start.

Definition at line 184 of file PointSequence.h.

29.344.3.3 `std::vector<double>::iterator Go::PointSequence::coefs_end ( ) [inline]`

Iterator to grid end.

Definition at line 180 of file PointSequence.h.

29.344.3.4 `std::vector<double>::const_iterator Go::PointSequence::coefs_end ( ) const [inline]`

Constant iterator to grid end.

Definition at line 188 of file PointSequence.h.

29.344.3.5 `int Go::PointSequence::dimension ( ) const`

[Point](#) dimension.

29.344.3.6 `int Go::PointSequence::grid_dimension ( ) const`

Dimension of grid.

29.344.3.7 `int Go::PointSequence::grid_length ( int i ) const`

Length of grid in the given direction.



## 29.344.3.8 PointSequenceType Go::PointSequence::type ( ) const

[Point](#) type.

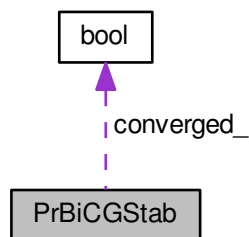
The documentation for this class was generated from the following file:

- [gtools-core/include/GoTools/geometry/PointSequence.h](#)

## 29.345 PrBiCGStab Class Reference

```
#include <PrBiCGStab.h>
```

Collaboration diagram for PrBiCGStab:



## Public Member Functions

- [PrBiCGStab](#) ()  
*Constructor.*
- [~PrBiCGStab](#) ()  
*Destructor.*
- void [setTolerance](#) (double tolerance=1.0e-6)  
*Set the tolerance for the residual.*
- void [setMaxIterations](#) (int max\_ iterations)  
*Set the maximum number of iterations.*
- void [solve](#) (const PrMatrix &A, PrVec &x, const PrVec &b)  
*Solve the linear system, replacing the start vector with the solution.*
- int [getItCount](#) ()  
*Get the number of iterations spent for the last call of 'solve()'.*
- double [getCPUTime](#) ()  
*Get the CPU time spent for the last call of 'solve()'.*
- bool [converged](#) ()  
*Check if the last call of 'solve()' managed to converge to a solution.*

## Protected Attributes

- [double tolerance\\_](#)
- [int max\\_iterations\\_](#)
- [int it\\_count\\_](#)
- [double cpu\\_time\\_](#)
- [bool converged\\_](#)

### 29.345.1 Detailed Description

[PrBiCGStab](#) - Implements the BiCGStab method for solving sparse linear systems.

Definition at line 51 of file PrBiCGStab.h.

### 29.345.2 Constructor & Destructor Documentation

#### 29.345.2.1 PrBiCGStab::PrBiCGStab ( )

Constructor.

#### 29.345.2.2 PrBiCGStab::~PrBiCGStab ( ) [inline]

Destructor.

Definition at line 66 of file PrBiCGStab.h.

### 29.345.3 Member Function Documentation

#### 29.345.3.1 bool PrBiCGStab::converged ( ) [inline]

Check if the last call of ['solve\(\)'](#) managed to converge to a solution.

Definition at line 84 of file PrBiCGStab.h.

#### 29.345.3.2 double PrBiCGStab::getCPUTime ( ) [inline]

Get the CPU time spent for the last call of ['solve\(\)'](#).

Definition at line 81 of file PrBiCGStab.h.

#### 29.345.3.3 int PrBiCGStab::getItCount ( ) [inline]

Get the number of iterations spent for the last call of ['solve\(\)'](#).

Definition at line 78 of file PrBiCGStab.h.

29.345.3.4 void PrBiCGStab::setMaxIterations ( int *max\_iterations* ) [inline]

Set the maximum number of iterations.

Definition at line 72 of file PrBiCGStab.h.

29.345.3.5 void PrBiCGStab::setTolerance ( double *tolerance* = 1.0e-6 ) [inline]

Set the tolerance for the residual.

Definition at line 69 of file PrBiCGStab.h.

29.345.3.6 void PrBiCGStab::solve ( const PrMatrix & A, PrVec & x, const PrVec & b )

Solve the linear system, replacing the start vector with the solution.

## 29.345.4 Member Data Documentation

29.345.4.1 bool PrBiCGStab::converged\_ [protected]

Definition at line 60 of file PrBiCGStab.h.

29.345.4.2 double PrBiCGStab::cpu\_time\_ [protected]

Definition at line 59 of file PrBiCGStab.h.

29.345.4.3 int PrBiCGStab::it\_count\_ [protected]

Definition at line 58 of file PrBiCGStab.h.

29.345.4.4 int PrBiCGStab::max\_iterations\_ [protected]

Definition at line 56 of file PrBiCGStab.h.

29.345.4.5 double PrBiCGStab::tolerance\_ [protected]

Definition at line 55 of file PrBiCGStab.h.

The documentation for this class was generated from the following file:

- parametrization/include/GoTools/parametrization/[PrBiCGStab.h](#)

## 29.346 PrCellStructure Class Reference

```
#include <PrCellStructure.h>
```

### Public Member Functions

- [PrCellStructure](#) ()  
*Default constructor.*
- [PrCellStructure](#) (int *n*, double \*xyz\_points, int max\_no\_cells=10)
- virtual [~PrCellStructure](#) ()  
*Destructor.*
- void [attach](#) (int *n*, const double \*xyz\_points)
- void [setNumCells](#) (int max\_no\_cells)
- int [getNumCells](#) () const
- int [getNumNodes](#) () const  
*Get the number of points (nodes) stored in the cell structure.*
- Vector3D [get3dNode](#) (int *i*) const  
*Get a specific point (node) in the PrCellStructure by its index.*
- void [set3dNode](#) (int *i*, const Vector3D &p)
- void [getBall](#) (const Vector3D &p, double radius, vector< int > &neighbours, int notP=0) const
- void [getKNearest](#) (const Vector3D &p, int *k*, vector< int > &neighbours, int notP=0) const
- int [getI](#) (int *i*, int *j*, int *k*) const  
*Get the index of the cell positioned at 'i', 'j', 'k'.*
- void [getIJK](#) (int *ii*, int &*i*, int &*j*, int &*k*) const  
*Get the 'i', 'j', 'k'-position of the cell indexed 'ii'.*
- void [whichCell](#) (const Vector3D &xyz, int &*i*, int &*j*, int &*k*) const
- void [print](#) (ostream &os)  
*write points to stream*
- void [scan](#) (istream &is)  
*read points from stream and regenerate cell structure.*

### 29.346.1 Detailed Description

[PrCellStructure](#) - Represents a set of points in three dimensions and a cell structure: a 3D grid of cubical cells each with a local list of those points which lie inside it. This allows one to perform various queries, such as finding all points in a ball, very efficiently. Creating the cell structure requires only  $O(N)$  operations, where  $N$  is the number of points.

Definition at line 60 of file PrCellStructure.h.

### 29.346.2 Constructor & Destructor Documentation

#### 29.346.2.1 PrCellStructure::PrCellStructure ( ) [inline]

Default constructor.

Definition at line 77 of file PrCellStructure.h.

#### 29.346.2.2 PrCellStructure::PrCellStructure ( int *n*, double \* xyz\_points, int max\_no\_cells = 10 )

Constructor

## Parameters

<i>n</i>	total number of points
<i>xyz_points</i>	pointer to an array of points stored xyz-wise (The points will be copied to internal data structure.
<i>max_no_cells</i>	indicate maximum number of cells along the X, Y and Z axis.

29.346.2.3 virtual PrCellStructure::~~PrCellStructure ( ) [inline],[virtual]

Destructor.

Definition at line 87 of file PrCellStructure.h.

### 29.346.3 Member Function Documentation

29.346.3.1 void PrCellStructure::attach ( int *n*, const double \* *xyz\_points* )

Reset the [PrCellStructure](#) to a set of new points, deleting old content.

## Parameters

<i>n</i>	number of points
<i>xyz_points</i>	pointer to the array of stored points. (The points will be copied to internal data structure).

29.346.3.2 Vector3D PrCellStructure::get3dNode ( int *i* ) const [inline]

Get a specific point (node) in the [PrCellStructure](#) by its index.

Definition at line 108 of file PrCellStructure.h.

29.346.3.3 void PrCellStructure::getBall ( const Vector3D & *p*, double *radius*, vector< int > & *neighbours*, int *notP* = 0 ) const

Return all points within the ball of radius *radius* around the point *p*. Don't include *p* itself if *notP* = 1.

## Parameters

<i>p</i>	the center of the ball
<i>radius</i>	the radius of the ball

## Return values

<i>neighbours</i>	will be filled with the indexes of the neighbour points.
-------------------	----------------------------------------------------------

## Parameters

<i>notP</i>	if != 0, 'p' itself will not be included in the list of returned points.
-------------	--------------------------------------------------------------------------

**29.346.3.4** `int PrCellStructure::getI ( int i, int j, int k ) const` `[inline]`

Get the index of the cell positioned at 'i', 'j', 'k'.

Definition at line 139 of file PrCellStructure.h.

**29.346.3.5** `void PrCellStructure::getJK ( int ii, int & i, int & j, int & k ) const`

Get the 'i', 'j', 'k'-position of the cell indexed 'ii'.

**29.346.3.6** `void PrCellStructure::getKNearest ( const Vector3D & p, int k, vector< int > & neighbours, int notP = 0 ) const`

Return k nearest points to the point p. Don't include p itself if notP = 1.

## Parameters

<i>p</i>	the point for which we seek the k nearest neighbours
<i>k</i>	the number of neighbours we seek

## Return values

<i>neighbours</i>	will be filled with the indexes of the neighbour points.
-------------------	----------------------------------------------------------

## Parameters

<i>notP</i>	if != 0, 'p' itself will not be included in the list of returned points.
-------------	--------------------------------------------------------------------------

**29.346.3.7** `int PrCellStructure::getNumCells ( ) const` `[inline]`

Get the previously set limit for the maximum number of cells in the X, Y and Z direction.

Definition at line 102 of file PrCellStructure.h.

**29.346.3.8** `int PrCellStructure::getNumNodes ( ) const` `[inline]`

Get the number of points (nodes) stored in the cell structure.

Definition at line 105 of file PrCellStructure.h.

29.346.3.9 void PrCellStructure::print ( ostream & os )

write points to stream

29.346.3.10 void PrCellStructure::scan ( istream & is )

read points from stream and regenerate cell structure.

29.346.3.11 void PrCellStructure::set3dNode ( int *i*, const Vector3D & *p* ) [inline]

Change the coordinates of a specific point in the [PrCellStructure](#).

#### Parameters

<i>i</i>	index of node to change
<i>p</i>	the new coordinates

Definition at line 113 of file PrCellStructure.h.

29.346.3.12 void PrCellStructure::setNumCells ( int *max\_no\_cells* ) [inline]

Set the maximum number of cells in X, Y and Z direction to 'max\_no\_cells'. This must be set before the cells are generated (ie. before the ['attach\(\)'](#) command).

Definition at line 98 of file PrCellStructure.h.

29.346.3.13 void PrCellStructure::whichCell ( const Vector3D & xyz, int & *i*, int & *j*, int & *k* ) const

Locate the cell which contains the coordinate 'xyz'

#### Parameters

<i>xyz</i>	we want to find the cell containing these coordinates
------------	-------------------------------------------------------

#### Return values

<i>i</i>	i-position of the located cell
<i>j</i>	j-position of the located cell
<i>k</i>	k-position of the located cell

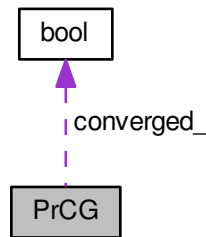
The documentation for this class was generated from the following file:

- parametrization/include/GoTools/parametrization/[PrCellStructure.h](#)

## 29.347 PrCG Class Reference

```
#include <PrCG.h>
```

Collaboration diagram for PrCG:



### Public Member Functions

- [PrCG \(\)](#)  
*Constructor.*
- [~PrCG \(\)](#)  
*Destructor.*
- void [setTolerance](#) (double tolerance=1.0e-6)  
*Set the tolerance for the residual.*
- void [setMaxIterations](#) (int max\_iterations)  
*Set the maximum number of iterations.*
- void [solve](#) (const PrMatrix &A, PrVec &x, const PrVec &b)  
*Solve the linear system, replacing the start vector with the solution.*
- int [getItCount](#) ()  
*Get the number of iterations spent for the last call of 'solve()'.*
- double [getCPUTime](#) ()  
*Get the CPU time spent for the last call of 'solve()'.*
- bool [converged](#) ()  
*Check if the last call of 'solve()' managed to converge to a solution.*

### Protected Attributes

- double [tolerance\\_](#)
- int [max\\_iterations\\_](#)
- int [it\\_count\\_](#)
- double [cpu\\_time\\_](#)
- bool [converged\\_](#)



### 29.347.1 Detailed Description

[PrCG](#) - This class implements the (unpreconditioned) CG method for solving sparse symmetric positive definite linear systems.

Definition at line 51 of file PrCG.h.

### 29.347.2 Constructor & Destructor Documentation

#### 29.347.2.1 PrCG::PrCG ( )

Constructor.

#### 29.347.2.2 PrCG::~PrCG ( ) [inline]

Destructor.

Definition at line 66 of file PrCG.h.

### 29.347.3 Member Function Documentation

#### 29.347.3.1 bool PrCG::converged ( ) [inline]

Check if the last call of '[solve\(\)](#)' managed to converge to a solution.

Definition at line 83 of file PrCG.h.

#### 29.347.3.2 double PrCG::getCPUTime ( ) [inline]

Get the CPU time spent for the last call of '[solve\(\)](#)'.

Definition at line 80 of file PrCG.h.

#### 29.347.3.3 int PrCG::getItCount ( ) [inline]

Get the number of iterations spent for the last call of '[solve\(\)](#)'.

Definition at line 77 of file PrCG.h.

#### 29.347.3.4 void PrCG::setMaxIterations ( int *max\_iterations* ) [inline]

Set the maximum number of iterations.

Definition at line 71 of file PrCG.h.

29.347.3.5 void PrCG::setTolerance ( double *tolerance* = 1.0e-6 ) [inline]

Set the tolerance for the residual.

Definition at line 69 of file PrCG.h.

29.347.3.6 void PrCG::solve ( const PrMatrix & A, PrVec & x, const PrVec & b )

Solve the linear system, replacing the start vector with the solution.

#### 29.347.4 Member Data Documentation

29.347.4.1 bool PrCG::converged\_ [protected]

Definition at line 60 of file PrCG.h.

29.347.4.2 double PrCG::cpu\_time\_ [protected]

Definition at line 59 of file PrCG.h.

29.347.4.3 int PrCG::it\_count\_ [protected]

Definition at line 58 of file PrCG.h.

29.347.4.4 int PrCG::max\_iterations\_ [protected]

Definition at line 56 of file PrCG.h.

29.347.4.5 double PrCG::tolerance\_ [protected]

Definition at line 55 of file PrCG.h.

The documentation for this class was generated from the following file:

- parametrization/include/GoTools/parametrization/[PrCG.h](#)

#### 29.348 Go::preEvaluationSf Struct Reference

```
#include <SfSolution.h>
```

## Public Attributes

- `std::vector< double > gauss_par1_`
- `std::vector< double > gauss_par2_`
- `std::vector< double > basisvals_u_`
- `std::vector< double > basisvals_v_`
- `std::vector< int > left_u_`
- `std::vector< int > left_v_`
- `std::vector< double > points_`
- `std::vector< double > deriv_u_`
- `std::vector< double > deriv_v_`

### 29.348.1 Detailed Description

Definition at line 58 of file SfSolution.h.

### 29.348.2 Member Data Documentation

#### 29.348.2.1 `std::vector<double> Go::preEvaluationSf::basisvals_u_`

Definition at line 63 of file SfSolution.h.

#### 29.348.2.2 `std::vector<double> Go::preEvaluationSf::basisvals_v_`

Definition at line 64 of file SfSolution.h.

#### 29.348.2.3 `std::vector<double> Go::preEvaluationSf::deriv_u_`

Definition at line 72 of file SfSolution.h.

#### 29.348.2.4 `std::vector<double> Go::preEvaluationSf::deriv_v_`

Definition at line 73 of file SfSolution.h.

#### 29.348.2.5 `std::vector<double> Go::preEvaluationSf::gauss_par1_`

Definition at line 61 of file SfSolution.h.

#### 29.348.2.6 `std::vector<double> Go::preEvaluationSf::gauss_par2_`

Definition at line 62 of file SfSolution.h.

29.348.2.7 `std::vector<int> Go::preEvaluationSf::left_u_`

Definition at line 67 of file SfSolution.h.

29.348.2.8 `std::vector<int> Go::preEvaluationSf::left_v_`

Definition at line 68 of file SfSolution.h.

29.348.2.9 `std::vector<double> Go::preEvaluationSf::points_`

Definition at line 71 of file SfSolution.h.

The documentation for this struct was generated from the following file:

- [isogeometric\\_model/include/GoTools/isogeometric\\_model/SfSolution.h](#)

## 29.349 Go::preEvaluationVol Struct Reference

```
#include <VolSolution.h>
```

### Public Attributes

- `std::vector< double > gauss_par1_`
- `std::vector< double > gauss_par2_`
- `std::vector< double > gauss_par3_`
- `std::vector< double > basisvals_u_`
- `std::vector< double > basisvals_v_`
- `std::vector< double > basisvals_w_`
- `std::vector< int > left_u_`
- `std::vector< int > left_v_`
- `std::vector< int > left_w_`
- `std::vector< double > points_`
- `std::vector< double > deriv_u_`
- `std::vector< double > deriv_v_`
- `std::vector< double > deriv_w_`

### 29.349.1 Detailed Description

Definition at line 60 of file VolSolution.h.

### 29.349.2 Member Data Documentation

29.349.2.1 `std::vector<double> Go::preEvaluationVol::basisvals_u_`

Definition at line 66 of file VolSolution.h.

29.349.2.2 `std::vector<double> Go::preEvaluationVol::basisvals_v_`

Definition at line 67 of file VolSolution.h.

29.349.2.3 `std::vector<double> Go::preEvaluationVol::basisvals_w_`

Definition at line 68 of file VolSolution.h.

29.349.2.4 `std::vector<double> Go::preEvaluationVol::deriv_u_`

Definition at line 76 of file VolSolution.h.

29.349.2.5 `std::vector<double> Go::preEvaluationVol::deriv_v_`

Definition at line 77 of file VolSolution.h.

29.349.2.6 `std::vector<double> Go::preEvaluationVol::deriv_w_`

Definition at line 78 of file VolSolution.h.

29.349.2.7 `std::vector<double> Go::preEvaluationVol::gauss_par1_`

Definition at line 63 of file VolSolution.h.

29.349.2.8 `std::vector<double> Go::preEvaluationVol::gauss_par2_`

Definition at line 64 of file VolSolution.h.

29.349.2.9 `std::vector<double> Go::preEvaluationVol::gauss_par3_`

Definition at line 65 of file VolSolution.h.

29.349.2.10 `std::vector<int> Go::preEvaluationVol::left_u_`

Definition at line 69 of file VolSolution.h.

29.349.2.11 `std::vector<int> Go::preEvaluationVol::left_v_`

Definition at line 70 of file VolSolution.h.

29.349.2.12 `std::vector<int> Go::preEvaluationVol::left_w_`

Definition at line 71 of file VolSolution.h.

29.349.2.13 `std::vector<double> Go::preEvaluationVol::points_`

Definition at line 75 of file VolSolution.h.

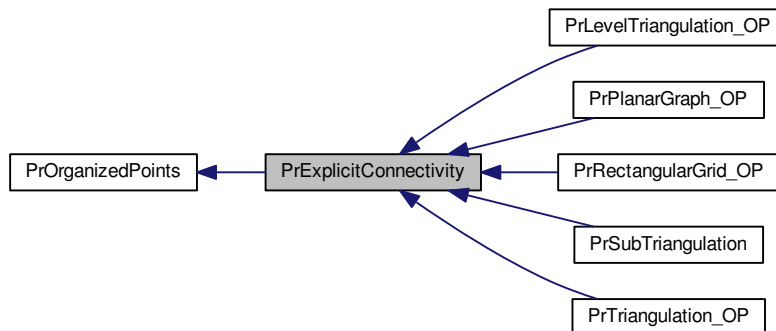
The documentation for this struct was generated from the following file:

- `isogeometric_model/include/GoTools/isogeometric_model/VolSolution.h`

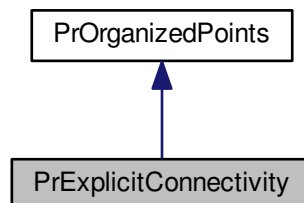
## 29.350 PrExplicitConnectivity Class Reference

```
#include <PrExplicitConnectivity.h>
```

Inheritance diagram for PrExplicitConnectivity:



Collaboration diagram for PrExplicitConnectivity:



## Public Member Functions

- virtual int [findNumFaces](#) () const  
*count the number of faces in the graph*
- void [findFace](#) (int i, int j, std::vector< int > &neighbours, std::vector< int > &face) const
- int [findGenus](#) () const  
*Compute the genus of the planar graph.*
- bool [isTriangulation](#) () const  
*Check if the planar graph is a triangulation.*
- virtual void [printXYZFaces](#) (std::ostream &os) const  
*Print out the faces of the graph.*
- virtual void [printXYZFacesML](#) (std::ostream &os) const  
*Print out the faces of the graph, ML format.*
- virtual void [printXYZFacesVRML](#) (std::ostream &os) const  
*Print out the faces of the graph, VRML format.*
- virtual void [printUVFaces](#) (std::ostream &os) const  
*Print out the faces of the parameterization.*
- void [findNextEdgeInFace](#) (int &i, int &j, std::vector< int > &neighbours) const
- virtual void [printInfo](#) (std::ostream &os) const  
*Print general information about this [PrOrganizedPoints](#) object (number of nodes, etc.)*
- virtual void [printTexture](#) (std::ostream &os) const

## Additional Inherited Members

### 29.350.1 Detailed Description

[PrExplicitConnectivity](#) - This class implements an interface to a planar graph embedded in  $R^2$  or  $R^3$ , where connectivity is explicitly defined. The two common examples are a triangulation and a topologically rectangular grid. What differentiates this class from [PrOrganizedPoints](#) is that for [PrExplicitConnectivity](#), it is possible to directly identify faces. The methods in this class are used by [PrParametrize](#).

Definition at line 55 of file [PrExplicitConnectivity.h](#).

### 29.350.2 Member Function Documentation

**29.350.2.1** void [PrExplicitConnectivity::findFace](#) ( int i, int j, std::vector< int > & neighbours, std::vector< int > & face ) const

Find the face of the graph which lies to the left of the directed edge whose end points are the node i and its j-th neighbour. The nodes of the face, starting with i, will be filled out in the list vector<int> face. The vector<int> neighbours is used for temporarily storing neighbours (for efficiency when calling this routine many times).

#### Parameters

<i>i</i>	the index of the node specifying the directed edge
<i>j</i>	the 'j'th neighbour of 'i' will be the second node specifying the edge

## Return values

<i>neighbours</i>	a temporary vector holding the neighbours of the last node examined by the implementation (might not always be needed; its presence in the parameter list is mainly an efficiency issue).
<i>face</i>	upon function return, will contain the indexes of the nodes in the requested face (anticlockwise order).

29.350.2.2 `int PrExplicitConnectivity::findGenus ( ) const`

Compute the genus of the planar graph.

29.350.2.3 `void PrExplicitConnectivity::findNextEdgeInFace ( int & i, int & j, std::vector< int > & neighbours ) const`

Find the next edge in the face which lies to the left of the directed edge whose end points are the node 'i' and its j-th neighbour.

## Parameters

<i>i</i>	the index of the node specifying the directed edge. Upon function return, it will contain the corresponding value for the next edge in the face.
<i>j</i>	the 'j'th neighbour of 'i' will be the second node specifying the edge. Upon function return, it will contain the corresponding value for the next edge in the face.
<i>neighbours</i>	when calling the function, this vector should contain the indexes of the neighbour nodes of node 'i'. Upon function return, it will contain the corresponding value for the next node in the face.

29.350.2.4 `virtual int PrExplicitConnectivity::findNumFaces ( ) const [virtual]`

count the number of faces in the graph

Reimplemented in [PrTriangulation\\_OP](#), and [PrLevelTriangulation\\_OP](#).

29.350.2.5 `bool PrExplicitConnectivity::isTriangulation ( ) const`

Check if the planar graph is a triangulation.

29.350.2.6 `virtual void PrExplicitConnectivity::printInfo ( std::ostream & os ) const [virtual]`

Print general information about this [PrOrganizedPoints](#) object (number of nodes, etc.)

Reimplemented from [PrOrganizedPoints](#).

29.350.2.7 `virtual void PrExplicitConnectivity::printTexture ( std::ostream & os ) const [virtual]`

29.350.2.8 `virtual void PrExplicitConnectivity::printUVFaces ( std::ostream & os ) const [virtual]`

Print out the faces of the parameterization.



29.350.2.9 `virtual void PrExplicitConnectivity::printXYZFaces ( std::ostream & os ) const` [virtual]

Print out the faces of the graph.

29.350.2.10 `virtual void PrExplicitConnectivity::printXYZFacesML ( std::ostream & os ) const` [virtual]

Print out the faces of the graph, ML format.

29.350.2.11 `virtual void PrExplicitConnectivity::printXYZFacesVRML ( std::ostream & os ) const` [virtual]

Print out the faces of the graph, VRML format.

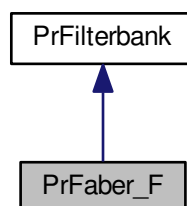
The documentation for this class was generated from the following file:

- [parametrization/include/GoTools/parametrization/PrExplicitConnectivity.h](#)

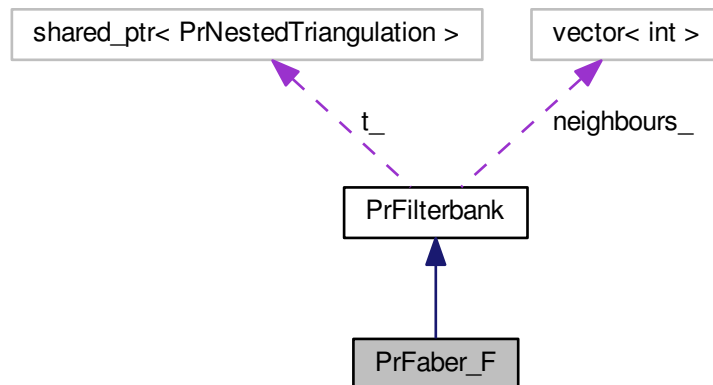
## 29.351 PrFaber\_F Class Reference

```
#include <PrFaber_F.h>
```

Inheritance diagram for PrFaber\_F:



Collaboration diagram for PrFaber\_F:



## Public Member Functions

- [PrFaber\\_F \(\)](#)  
*Constructor.*
- [virtual ~PrFaber\\_F \(\)](#)  
*Destructor.*
- virtual void [decompose](#) (int jlev, int dim=1)
- virtual void [compose](#) (int jlev, int dim=1)

## Additional Inherited Members

### 29.351.1 Detailed Description

- Implements algorithms for composing and decomposing piecewise linear functions over nested triangulations, using Faber decomposition. For a comparison with prewavelets, see

M. S. Floater, E. G. Quak and M. Reimers,  
Filter Bank Algorithms for Piecewise Linear  
Prewavelets on Arbitrary Triangulations,  
J. Comp. Appl. Math. \e 119 (2000), 185--207.

Definition at line 57 of file PrFaber\_F.h.

### 29.351.2 Constructor & Destructor Documentation

#### 29.351.2.1 PrFaber\_F::PrFaber\_F ( ) [inline]

Constructor.

Definition at line 63 of file PrFaber\_F.h.

29.351.2.2 virtual PrFaber\_F::~~PrFaber\_F ( ) [inline],[virtual]

Destructor.

Definition at line 65 of file PrFaber\_F.h.

### 29.351.3 Member Function Documentation

29.351.3.1 virtual void PrFaber\_F::compose ( int *jlev*, int *dim* = 1 ) [virtual]

The nodes in the triangulation belonging to  $V^j$  form a piecewise linear function  $f^j$  in the space  $S^j$ . This routines composes  $f^j$  from  $f^{j-1} + g^{j-1}$  where  $f^{j-1} \in S^{j-1}$  and  $g^{j-1} \in W^{j-1}$ . Use  $\text{dim} = 1$  for explicit triangulations and  $\text{dim} = 3$  for non-explicit. Default is  $\text{dim} = 1$ .

Implements [PrFilterbank](#).

29.351.3.2 virtual void PrFaber\_F::decompose ( int *jlev*, int *dim* = 1 ) [virtual]

The nodes in the triangulation belonging to  $V^j$  form a piecewise linear function  $f^j$  in the space  $S^j$  (where  $j$  = level). This routines decomposes  $f^j$  into  $f^{j-1} + g^{j-1}$  where  $f^{j-1} \in S^{j-1}$  and  $g^{j-1} \in W^{j-1}$ . Use  $\text{dim} = 1$  for explicit triangulations and  $\text{dim} = 3$  for non-explicit. Default is  $\text{dim} = 1$ .

Implements [PrFilterbank](#).

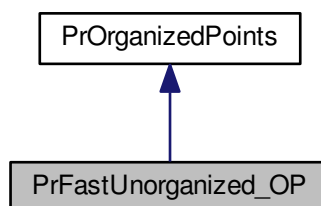
The documentation for this class was generated from the following file:

- parametrization/include/GoTools/parametrization/[PrFaber\\_F.h](#)

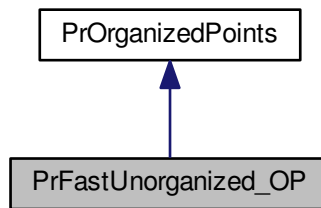
## 29.352 PrFastUnorganized\_OP Class Reference

```
#include <PrFastUnorganized_OP.h>
```

Inheritance diagram for PrFastUnorganized\_OP:



Collaboration diagram for PrFastUnorganized\_OP:



## Public Member Functions

- [PrFastUnorganized\\_OP](#) (int num\_cells=10)  
*Constructor.*
- [PrFastUnorganized\\_OP](#) (int n, int n\_int, double \*xyz\_points, int num\_cells=10)  
*Constructor.*
- [~PrFastUnorganized\\_OP](#) ()  
*Destructor.*
- void [initNeighbours](#) ()
- virtual int [getNumNodes](#) () const  
*Return the number of nodes in the graph.*
- virtual Vector3D [get3dNode](#) (int i) const  
*Return the i-th node in the graph if the nodes are three-dimensional.*
- virtual void [set3dNode](#) (int i, const Vector3D &p)
- virtual void [getNeighbours](#) (int i, vector< int > &neighbours) const
- virtual bool [isBoundary](#) (int i) const  
*Is 'i' a boundary node or interior node?*
- virtual double [getU](#) (int i) const  
*return the u parameter value of the i-th node.*
- virtual double [getV](#) (int i) const  
*return the v parameter value of the i-th node.*
- virtual void [setU](#) (int i, double u)  
*reset the u parameter value of the i-th node.*
- virtual void [setV](#) (int i, double v)  
*reset the v parameter value of the i-th node.*
- double [getRadius](#) ()  
*get the specified radius for defining neighbours by distance*
- void [setRadius](#) (double radius)  
*specify the radius for defining neighbourhood by distance*
- int [getKNearest](#) ()
- void [setKNearest](#) (int knearest)
- void [useK](#) ()
- void [useRadius](#) ()  
*Use an euclidean distance to define the neighbourhood of a point.*
- void [print](#) (ostream &os)  
*Write contents to stream.*
- void [scanRawData](#) (istream &is, int num\_cells=10, double noise=0.0)

## Additional Inherited Members

### 29.352.1 Detailed Description

- [PrFastUnorganized\\_OP](#) represents a set of points in three dimensions. All we know about them is their boundary in sequence. No other topology is given. This kind of data can be parametrized by [PrParametrizeBdy](#) and [PrParametrizeInt](#). The major difference to the class [PrUnorganized\\_OP](#) is that this class stores the neighbours for each vertex. Note that this requires the call of "initNeighbours" first!

Definition at line 59 of file PrFastUnorganized\_OP.h.

### 29.352.2 Constructor & Destructor Documentation

#### 29.352.2.1 PrFastUnorganized\_OP::PrFastUnorganized\_OP ( int num\_cells = 10 )

Constructor.

#### 29.352.2.2 PrFastUnorganized\_OP::PrFastUnorganized\_OP ( int n, int n\_int, double \* xyz\_points, int num\_cells = 10 )

Constructor.

#### 29.352.2.3 PrFastUnorganized\_OP::~PrFastUnorganized\_OP ( ) [inline]

Destructor.

Definition at line 82 of file PrFastUnorganized\_OP.h.

### 29.352.3 Member Function Documentation

#### 29.352.3.1 virtual Vector3D PrFastUnorganized\_OP::get3dNode ( int i ) const [inline],[virtual]

Return the i-th node in the graph if the nodes are three-dimensional.

Implements [PrOrganizedPoints](#).

Definition at line 94 of file PrFastUnorganized\_OP.h.

#### 29.352.3.2 int PrFastUnorganized\_OP::getKNearest ( ) [inline]

get the specified number for defining neighbourhoods by a fixed number of neighbours

Definition at line 127 of file PrFastUnorganized\_OP.h.

#### 29.352.3.3 virtual void PrFastUnorganized\_OP::getNeighbours ( int i, vector< int > & neighbours ) const [virtual]

Return the indices of the neighbours of the i-th node. (here there is no ordering: it's not a planar graph.)

**29.352.3.4** `virtual int PrFastUnorganized_OP::getNumNodes ( ) const [inline],[virtual]`

Return the number of nodes in the graph.

Implements [PrOrganizedPoints](#).

Definition at line 92 of file PrFastUnorganized\_OP.h.

**29.352.3.5** `double PrFastUnorganized_OP::getRadius ( ) [inline]`

get the specified radius for defining neighbours by distance

Definition at line 115 of file PrFastUnorganized\_OP.h.

**29.352.3.6** `virtual double PrFastUnorganized_OP::getU ( int i ) const [inline],[virtual]`

return the u parameter value of the i-th node.

Implements [PrOrganizedPoints](#).

Definition at line 109 of file PrFastUnorganized\_OP.h.

**29.352.3.7** `virtual double PrFastUnorganized_OP::getV ( int i ) const [inline],[virtual]`

return the v parameter value of the i-th node.

Implements [PrOrganizedPoints](#).

Definition at line 110 of file PrFastUnorganized\_OP.h.

**29.352.3.8** `void PrFastUnorganized_OP::initNeighbours ( )`

Compute (and internally store) information about the neighbours of each point. Neighbours are defined either by the  $k$  nearest points or by all points within a specified radius. Use the member functions [useK\(\)](#) and [useRadius\(\)](#) to specify this.

**29.352.3.9** `virtual bool PrFastUnorganized_OP::isBoundary ( int i ) const [inline],[virtual]`

Is 'i' a boundary node or interior node?

Implements [PrOrganizedPoints](#).

Definition at line 107 of file PrFastUnorganized\_OP.h.

**29.352.3.10** `void PrFastUnorganized_OP::print ( ostream & os )`

Write contents to stream.

**29.352.3.11** `void PrFastUnorganized_OP::scanRawData ( istream & is, int num_cells = 10, double noise = 0.0 )`

Read in points from a stream and regenerate internal structures (no parameterization of the points).

## Parameters

<i>is</i>	The stream containing the points. The format of the stream must be: <ul style="list-style-type: none"> <li>• first: the total number of points</li> <li>• then: the number of interior points (not boundary points)</li> <li>• then: all internal point coordinates listed as xyz xyz, etc.</li> <li>• finally: the boundary point coordinates</li> </ul>
<i>num_cells</i>	number of cells in each spatial direction when dividing up the space containing the points.
<i>noise</i>	magnitude of noise added to the internal points. Zero by default.

**29.352.3.12** `virtual void PrFastUnorganized_OP::set3dNode ( int i, const Vector3D & p ) [inline],[virtual]`

This inherited member function does not apply to this derived class. As of now, it will throw if you try to run it.

Implements [PrOrganizedPoints](#).

Definition at line 97 of file PrFastUnorganized\_OP.h.

**29.352.3.13** `void PrFastUnorganized_OP::setKNearest ( int knearest ) [inline]`

specify a number for defining neighbourhoods by a fixed number of neighbours.

Definition at line 130 of file PrFastUnorganized\_OP.h.

**29.352.3.14** `void PrFastUnorganized_OP::setRadius ( double radius ) [inline]`

specify the radius for defining neighbourhood by distance

Definition at line 124 of file PrFastUnorganized\_OP.h.

**29.352.3.15** `virtual void PrFastUnorganized_OP::setU ( int i, double u ) [inline],[virtual]`

reset the u parameter value of the i-th node.

Implements [PrOrganizedPoints](#).

Definition at line 111 of file PrFastUnorganized\_OP.h.

**29.352.3.16** `virtual void PrFastUnorganized_OP::setV ( int i, double v ) [inline],[virtual]`

reset the v parameter value of the i-th node.

Implements [PrOrganizedPoints](#).

Definition at line 112 of file PrFastUnorganized\_OP.h.

29.352.3.17 `void PrFastUnorganized_OP::useK ( ) [inline]`

Use a fixed number of neighbours to define the neighbourhood of a point.

Definition at line 133 of file PrFastUnorganized\_OP.h.

29.352.3.18 `void PrFastUnorganized_OP::useRadius ( ) [inline]`

Use an euclidean distance to define the neighbourhood of a point.

Definition at line 135 of file PrFastUnorganized\_OP.h.

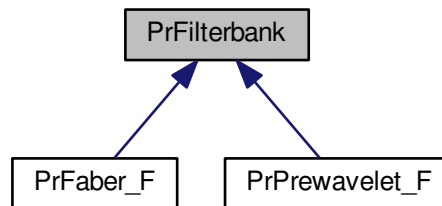
The documentation for this class was generated from the following file:

- parametrization/include/GoTools/parametrization/[PrFastUnorganized\\_OP.h](#)

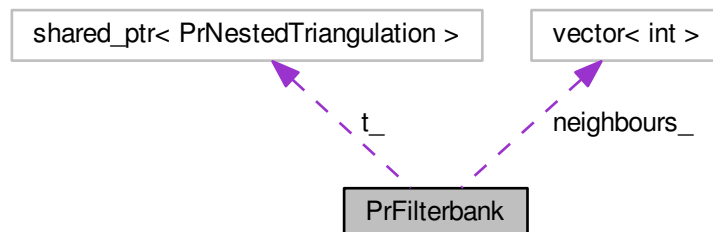
## 29.353 PrFilterbank Class Reference

```
#include <PrFilterbank.h>
```

Inheritance diagram for PrFilterbank:



Collaboration diagram for PrFilterbank:





## Public Member Functions

- virtual void [decompose](#) (int jlev, int dim=1)=0
- virtual void [compose](#) (int jlev, int dim=1)=0
- virtual [~PrFilterbank](#) ()
  - Destructor.*
- void [attach](#) (shared\_ptr< [PrNestedTriangulation](#) > t)
  - Set the graph.*
- void [decomposeAll](#) (int dim=1)
- void [decomposeFrom](#) (int jlev, int dim=1)
- void [composeAll](#) (int dim=1)
- void [composeUpTo](#) (int jlev, int dim=1)

## Protected Attributes

- shared\_ptr< [PrNestedTriangulation](#) > t\_
- vector< int > neighbours\_

### 29.353.1 Detailed Description

[PrFilterbank](#) - implements an algorithm for decomposing a piecewise linear function  $f^k$  over a nested triangulation  $T^k$  into the sum  $f^0 + g^0 + g^1 + \dots + g^{k-1}$  where  $f^0 \in S^0$  and  $g^0 \in W^0, \dots, g^{k-1} \in W^{k-1}$ . Here the set of nodes  $V^k$  of  $T^k$  satisfy  $V^0 \subset V^1 \subset \dots \subset V^k$  and the corresponding spaces of piecewise linear  $S^0 \subset S^1 \subset \dots \subset S^k$ . The space  $W^{j-1}$  is a complement space of  $S^{j-1} \in S^j$ . Examples are prewavelet and Faber decomposition.

Definition at line 66 of file PrFilterbank.h.

### 29.353.2 Constructor & Destructor Documentation

29.353.2.1 virtual [PrFilterbank::~PrFilterbank](#) ( ) [virtual]

Destructor.

### 29.353.3 Member Function Documentation

29.353.3.1 void [PrFilterbank::attach](#) ( shared\_ptr< [PrNestedTriangulation](#) > t )

Set the graph.

29.353.3.2 virtual void [PrFilterbank::compose](#) ( int jlev, int dim = 1 ) [pure virtual]

The nodes in the triangulation belonging to  $V^j$  form a piecewise linear function  $f^j$  in the space  $S^j$ . This routines composes  $f^j$  from  $f^{j-1} + g^{j-1}$  where  $f^{j-1} \in S^{j-1}$  and  $g^{j-1} \in W^{j-1}$ . Use dim = 1 for explicit triangulations dim = 3 for non-explicit. Default is dim = 1.

Implemented in [PrPrewavelet\\_F](#), and [PrFaber\\_F](#).

**29.353.3.3** void PrFilterbank::composeAll ( int *dim* = 1 )

The coarsest triangulation is a piecewise linear function  $f^0$  in the space  $S^0$ . This routine composes  $f^k$  from  $f^0 + g^0 + g^1 + \dots + g^{k-1}$  where  $f^k \in S^k, g^0 \in W^0, \dots, g^{k-1} \in W^{k-1}$ . Use *dim* = 1 for explicit triangulations and *dim* = 3 for non-explicit. Default is *dim* = 1.

**29.353.3.4** void PrFilterbank::composeUpTo ( int *jlev*, int *dim* = 1 )

**29.353.3.5** virtual void PrFilterbank::decompose ( int *jlev*, int *dim* = 1 ) [pure virtual]

The nodes in the triangulation belonging to  $V^j$  form a piecewise linear function  $f^j$  in the space  $S^j$  (where *j* = level). This routine decomposes  $f^j$  into  $f^{j-1} + g^{j-1}$  where  $f^{j-1} \in S^{j-1}$  and  $g^{j-1} \in W^{j-1}$ . Use *dim* = 1 for explicit triangulations and *dim* = 3 for non-explicit. Default is *dim* = 1.

Implemented in [PrPrewavelet\\_F](#), and [PrFaber\\_F](#).

**29.353.3.6** void PrFilterbank::decomposeAll ( int *dim* = 1 )

The triangulation is a piecewise linear function  $f^k$  in the space  $S^k$ . This routine decomposes  $f^k$  into  $f^0 + g^0 + g^1 + \dots + g^{k-1}$  where  $f^0 \in S^0, g^0 \in W^0, \dots, g^{k-1} \in W^{k-1}$ . Use *dim* = 1 for explicit triangulations and *dim* = 3 for non-explicit. Default is *dim* = 1.

**29.353.3.7** void PrFilterbank::decomposeFrom ( int *jlev*, int *dim* = 1 )

## 29.353.4 Member Data Documentation

**29.353.4.1** vector<int> PrFilterbank::neighbours\_ [protected]

Definition at line 71 of file PrFilterbank.h.

**29.353.4.2** shared\_ptr<PrNestedTriangulation> PrFilterbank::t\_ [protected]

Definition at line 70 of file PrFilterbank.h.

The documentation for this class was generated from the following file:

- parametrization/include/GoTools/parametrization/[PrFilterbank.h](#)

## 29.354 PrHeap Class Reference

```
#include <PrHeap.h>
```

## Public Member Functions

- [PrHeap](#) ()  
*Default constructor.*
- [PrHeap](#) (int maxsize)
- [PrHeap](#) (int \*elms, int n)  
*Constructor. Not implemented.*
- [PrHeap](#) (const [PrHeap](#) &heap)  
*Copy constructor. Not implemented, generates a warning.*
- [~PrHeap](#) ()  
*Destructor. Deletes arrays.*
- void [redim](#) (int size)
- int [getMaxSize](#) () const  
*return max. size for the heap.*
- int [getSize](#) () const  
*return size for the heap.*
- void [push](#) (double key, int element)  
*Inserts an element with a key in the heap.*
- void [modify](#) (double new\_key, int element)  
*Modifies an elements key and restores the heap property.*
- int [pop](#) ()  
*Extracts the best element from the heap.*
- void [pop](#) (double &key, int &element)  
*Extracts the best element from the heap.*
- void [emptyHeap](#) ()  
*Restarts the heap.*
- void [print](#) (std::ostream &os) const  
*print to stream*

### 29.354.1 Detailed Description

[PrHeap](#) - This is a class for minimum heaps. The [PrHeap](#)'s functionality is to store prioritized elements. It has a function for adding elements, a function for returning (and removing from the heap) the element with the "best" priority (least key) and a function for modifying elements. These functions are implemented in a way making sure that they are very fast in run time. All functions run in  $O(\log n)$  time.

Definition at line 55 of file [PrHeap.h](#).

### 29.354.2 Constructor & Destructor Documentation

#### 29.354.2.1 [PrHeap::PrHeap](#) ( )

Default constructor.

#### 29.354.2.2 [PrHeap::PrHeap](#) ( int maxsize )

Constructor. Creates a heap wich is able to store maxsize elements. maxsize will typically be set to the total number of elements (here total number of vertices).

**29.354.2.3 PrHeap::PrHeap ( int \* *elms*, int *n* )**

Constructor. Not implemented.

**29.354.2.4 PrHeap::PrHeap ( const PrHeap & *heap* )**

Copy constructor. Not implemented, generates a warning.

**29.354.2.5 PrHeap::~PrHeap ( )**

Destructor. Deletes arrays.

**29.354.3 Member Function Documentation****29.354.3.1 void PrHeap::emptyHeap ( )**

Restarts the heap.

**29.354.3.2 int PrHeap::getMaxSize ( ) const [inline]**

return max. size for the heap.

Definition at line 86 of file PrHeap.h.

**29.354.3.3 int PrHeap::getSize ( ) const [inline]**

return size for the heap.

Definition at line 89 of file PrHeap.h.

**29.354.3.4 void PrHeap::modify ( double *new\_key*, int *element* )**

Modifies an elements key and restores the heap property.

**29.354.3.5 int PrHeap::pop ( )**

Extracts the best element from the heap.

**29.354.3.6 void PrHeap::pop ( double & *key*, int & *element* )**

Extracts the best element from the heap.

29.354.3.7 void PrHeap::print ( std::ostream & os ) const

print to stream

29.354.3.8 void PrHeap::push ( double key, int element )

Inserts an element with a key in the heap.

29.354.3.9 void PrHeap::redim ( int size )

The documentation for this class was generated from the following file:

- parametrization/include/GoTools/parametrization/[PrHeap.h](#)

## 29.355 PrintCounter Class Reference

### Public Member Functions

- [~PrintCounter](#) ()
- [PrintCounter](#) (const char \*sx)
- void [operator++](#) ()

### 29.355.1 Detailed Description

Definition at line 18 of file tmt.cpp.

### 29.355.2 Constructor & Destructor Documentation

29.355.2.1 [PrintCounter::~PrintCounter](#) ( )

Definition at line 31 of file tmt.cpp.

29.355.2.2 [PrintCounter::PrintCounter](#) ( const char \* sx ) `[inline]`

Definition at line 24 of file tmt.cpp.

### 29.355.3 Member Function Documentation

29.355.3.1 void [PrintCounter::operator++](#) ( ) `[inline]`

Definition at line 25 of file tmt.cpp.

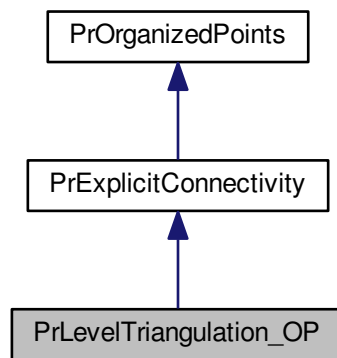
The documentation for this class was generated from the following file:

- newmat/app/[tmt.cpp](#)

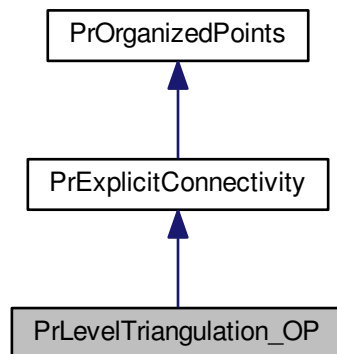
## 29.356 PrLevelTriangulation\_OP Class Reference

```
#include <PrLevelTriangulation_OP.h>
```

Inheritance diagram for PrLevelTriangulation\_OP:



Collaboration diagram for PrLevelTriangulation\_OP:



### Public Member Functions

- [PrLevelTriangulation\\_OP](#) ()  
*Empty default constructor.*
- [PrLevelTriangulation\\_OP](#) (vector< [PrNestedNode](#) > \*node, vector< [PrTriangle](#) > &nt, int level=0)
- [~PrLevelTriangulation\\_OP](#) ()  
*Empty default destructor.*

**Derived from base class**

- virtual int [getNumNodes](#) () const  
*Return the number of nodes in the graph.*
- virtual Vector3D [get3dNode](#) (int i) const  
*Return the i-th node in the graph.*
- virtual void [set3dNode](#) (int i, const Vector3D &p)
- virtual void [getNeighbours](#) (int i, vector< int > &neighbours) const
- virtual bool [isBoundary](#) (int i) const  
*Is i a boundary node or interior node?*
- virtual double [getU](#) (int i) const  
*return the u parameter value of the i-th node.*
- virtual double [getV](#) (int i) const  
*return the v parameter value of the i-th node.*
- virtual void [setU](#) (int i, double u)  
*reset the u parameter value of the i-th node.*
- virtual void [setV](#) (int i, double v)  
*reset the v parameter value of the i-th node.*
- virtual int [findNumFaces](#) () const  
*count the number of faces in the graph*

**Other functions**

- [PrNestedNode](#) & [getPrNestedNode](#) (int i)  
*Grab internal structure for doing thinning.*
- [PrTriangle](#) & [getPrTriangle](#) (int i)  
*Grab internal structure for doing thinning.*
- void [printXYZTriangles](#) (ostream &os, bool num=false)
- void [printUVTriangles](#) (ostream &os, bool num=false)
- void [print](#) (ostream &os)  
*print whole class*
- void [printRawData](#) (ostream &os)  
*print without neighbourhood information*
- void [printUV](#) (ostream &os)  
*for converting uv to xyz with z=0*

**Additional Inherited Members****29.356.1 Detailed Description**

[PrLevelTriangulation\\_OP](#) - This class represents a triangulation of points in  $R^3$  with or without parameter points in  $R^2$ . It implements the virtual functions in [PrOrganizedPoints](#). This kind of triangulation can be parametrized by [PrParametrize](#).

Definition at line 55 of file [PrLevelTriangulation\\_OP.h](#).

**29.356.2 Constructor & Destructor Documentation****29.356.2.1 PrLevelTriangulation\_OP::PrLevelTriangulation\_OP ( ) [inline]**

Empty default constructor.

Definition at line 68 of file [PrLevelTriangulation\\_OP.h](#).

**29.356.2.2 PrLevelTriangulation\_OP::PrLevelTriangulation\_OP ( vector< PrNestedNode > \* node, vector< PrTriangle > & nt, int level = 0 )**

Constructor.

## Parameters

<i>node</i>	pointer to a vector of nodes. The vector will not be copied internally.
<i>nt</i>	vector of triangles. Will be copied internally
<i>level</i>	specify the level represented by this triangulation

**29.356.2.3** `PrLevelTriangulation_OP::~PrLevelTriangulation_OP ( ) [inline]`

Empty default destructor.

Definition at line 81 of file `PrLevelTriangulation_OP.h`.

### 29.356.3 Member Function Documentation

**29.356.3.1** `virtual int PrLevelTriangulation_OP::findNumFaces ( ) const [inline],[virtual]`

count the number of faces in the graph

Reimplemented from [PrExplicitConnectivity](#).

Definition at line 115 of file `PrLevelTriangulation_OP.h`.

**29.356.3.2** `virtual Vector3D PrLevelTriangulation_OP::get3dNode ( int i ) const [inline],[virtual]`

Return the i-th node in the graph.

Implements [PrOrganizedPoints](#).

Definition at line 90 of file `PrLevelTriangulation_OP.h`.

**29.356.3.3** `virtual void PrLevelTriangulation_OP::getNeighbours ( int i, vector< int > & neighbours ) const [virtual]`

Return the indices of the neighbours of the i-th node in:

1. any anticlockwise order if i is an interior node
2. the unique anticlockwise order if i is a boundary node.

**29.356.3.4** `virtual int PrLevelTriangulation_OP::getNumNodes ( ) const [inline],[virtual]`

Return the number of nodes in the graph.

Implements [PrOrganizedPoints](#).

Definition at line 87 of file `PrLevelTriangulation_OP.h`.



29.356.3.5 `PrNestedNode& PrLevelTriangulation_OP::getPrNestedNode ( int i ) [inline]`

Grab internal structure for doing thinning.

Definition at line 123 of file PrLevelTriangulation\_OP.h.

29.356.3.6 `PrTriangle& PrLevelTriangulation_OP::getPrTriangle ( int i ) [inline]`

Grab internal structure for doing thinning.

Definition at line 125 of file PrLevelTriangulation\_OP.h.

29.356.3.7 `virtual double PrLevelTriangulation_OP::getU ( int i ) const [inline],[virtual]`

return the u parameter value of the i-th node.

Implements [PrOrganizedPoints](#).

Definition at line 104 of file PrLevelTriangulation\_OP.h.

29.356.3.8 `virtual double PrLevelTriangulation_OP::getV ( int i ) const [inline],[virtual]`

return the v parameter value of the i-th node.

Implements [PrOrganizedPoints](#).

Definition at line 107 of file PrLevelTriangulation\_OP.h.

29.356.3.9 `virtual bool PrLevelTriangulation_OP::isBoundary ( int i ) const [virtual]`

Is i a boundary node or interior node?

Implements [PrOrganizedPoints](#).

29.356.3.10 `void PrLevelTriangulation_OP::print ( ostream & os )`

print whole class

29.356.3.11 `void PrLevelTriangulation_OP::printRawData ( ostream & os )`

print without neighbourhood information

29.356.3.12 `void PrLevelTriangulation_OP::printUV ( ostream & os )`

for converting uv to xyz with z=0

29.356.3.13 `void PrLevelTriangulation_OP::printUVTriangles ( ostream & os, bool num = false )`

Print out the uv triangles of the graph, useful for plotting. If num = 1, print first the number of triangles.

29.356.3.14 `void PrLevelTriangulation_OP::printXYZTriangles ( ostream & os, bool num = false )`

Print out the triangles of the graph, useful for plotting. If num = 1, print first the number of triangles.

29.356.3.15 `virtual void PrLevelTriangulation_OP::set3dNode ( int i, const Vector3D & p ) [inline],[virtual]`

set the coordinates of the i-th node in the graph if the nodes are three-dimensional.

Implements [PrOrganizedPoints](#).

Definition at line 91 of file PrLevelTriangulation\_OP.h.

29.356.3.16 `virtual void PrLevelTriangulation_OP::setU ( int i, double u ) [inline],[virtual]`

reset the u parameter value of the i-th node.

Implements [PrOrganizedPoints](#).

Definition at line 110 of file PrLevelTriangulation\_OP.h.

29.356.3.17 `virtual void PrLevelTriangulation_OP::setV ( int i, double v ) [inline],[virtual]`

reset the v parameter value of the i-th node.

Implements [PrOrganizedPoints](#).

Definition at line 113 of file PrLevelTriangulation\_OP.h.

The documentation for this class was generated from the following file:

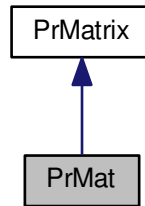
- parametrization/include/GoTools/parametrization/[PrLevelTriangulation\\_OP.h](#)

## 29.357 PrMat Class Reference

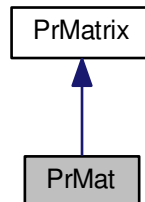
This class implements a matrix.

```
#include <PrMat.h>
```

Inheritance diagram for PrMat:



Collaboration diagram for PrMat:



### Public Member Functions

#### Derived from base class

- virtual int [rows](#) () [const](#)  
*Number of rows.*
- virtual int [colms](#) () [const](#)  
*Number of columns.*
- virtual [double operator\(\)](#) (int i, int j) [const](#)  
*value of matrix element*
- virtual void [prod](#) (const [PrVec](#) &x, [PrVec](#) &y) [const](#)  
*Find  $y = Ax$ .*
- virtual void [read](#) (std::istream &is)  
*read contents from stream*
- virtual [~PrMat](#) ()

### Other functions

- [PrMat](#) (int m, int n, double val=0.0)
- [PrMat](#) ()  
*Empty default constructor.*
- [double & operator\(\)](#) (int i, int j)  
*access element (i, j)*
- void [redim](#) (int m, int n)  
*change size of matrix*

### Protected Attributes

- std::vector< double > a\_
- int m\_
- int n\_

### 29.357.1 Detailed Description

This class implements a matrix.

Definition at line 53 of file PrMat.h.

### 29.357.2 Constructor & Destructor Documentation

29.357.2.1 virtual PrMat::~PrMat ( ) [virtual]

29.357.2.2 PrMat::PrMat ( int m, int n, double val = 0.0 )

Constructor

Parameters

<i>m</i>	number of columns
<i>n</i>	number of rows

29.357.2.3 PrMat::PrMat ( ) [inline]

Empty default constructor.

Definition at line 80 of file PrMat.h.

### 29.357.3 Member Function Documentation

29.357.3.1 virtual int PrMat::columns ( ) const [inline],[virtual]

Number of columns.

Implements [PrMatrix](#).

Definition at line 64 of file PrMat.h.

29.357.3.2 `virtual double PrMat::operator()( int i, int j ) const` [virtual]

value of matrix element

Implements [PrMatrix](#).

29.357.3.3 `double & PrMat::operator()( int i, int j )` [inline]

access element (i, j)

Definition at line 91 of file PrMat.h.

29.357.3.4 `virtual void PrMat::prod ( const PrVec & x, PrVec & y ) const` [virtual]

Find  $y = Ax$ .

Implements [PrMatrix](#).

29.357.3.5 `virtual void PrMat::read ( std::istream & is )` [virtual]

read contents from stream

Implements [PrMatrix](#).

29.357.3.6 `void PrMat::redim ( int m, int n )`

change size of matrix

29.357.3.7 `virtual int PrMat::rows ( ) const` [inline],[virtual]

Number of rows.

Implements [PrMatrix](#).

Definition at line 63 of file PrMat.h.

## 29.357.4 Member Data Documentation

29.357.4.1 `std::vector<double> PrMat::a_` [protected]

Definition at line 57 of file PrMat.h.

29.357.4.2 `int PrMat::m_` [protected]

Definition at line 58 of file PrMat.h.

29.357.4.3 `int PrMat::n_ [protected]`

Definition at line 58 of file PrMat.h.

The documentation for this class was generated from the following file:

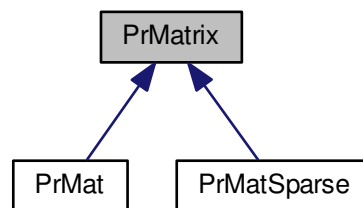
- parametrization/include/GoTools/parametrization/PrMat.h

## 29.358 PrMatrix Class Reference

This class implements a matrix.

```
#include <PrMatrix.h>
```

Inheritance diagram for PrMatrix:



### Public Member Functions

- virtual `int rows () const =0`  
*Number of rows.*
- virtual `int colms () const =0`  
*Number of columns.*
- virtual `double operator() (int i, int j) const =0`  
*value of matrix element*
- virtual `void prod (const PrVec &x, PrVec &y) const =0`  
*Multiply matrix with 'x' and return result in 'y'. ( $y = Ax$ )*
- virtual `~PrMatrix ()`  
*Virtual destructor.*
- virtual `void print (std::ostream &os)`  
*print contents to stream*
- virtual `void read (std::istream &is)=0`  
*read contents from stream*

### 29.358.1 Detailed Description

This class implements a matrix.

Definition at line 49 of file PrMatrix.h.

### 29.358.2 Constructor & Destructor Documentation

29.358.2.1 virtual PrMatrix::~PrMatrix ( ) [virtual]

Virtual destructor.

### 29.358.3 Member Function Documentation

29.358.3.1 virtual int PrMatrix::columns ( ) const [pure virtual]

Number of columns.

Implemented in [PrMatSparse](#), and [PrMat](#).

29.358.3.2 virtual double PrMatrix::operator()( int *i*, int *j* ) const [pure virtual]

value of matrix element

Implemented in [PrMatSparse](#), and [PrMat](#).

29.358.3.3 virtual void PrMatrix::print ( std::ostream & *os* ) [virtual]

print contents to stream

Reimplemented in [PrMatSparse](#).

29.358.3.4 virtual void PrMatrix::prod ( const PrVec & *x*, PrVec & *y* ) const [pure virtual]

Multiply matrix with 'x' and return result in 'y'. ( $y = Ax$ )

Implemented in [PrMatSparse](#), and [PrMat](#).

29.358.3.5 virtual void PrMatrix::read ( std::istream & *is* ) [pure virtual]

read contents from stream

Implemented in [PrMatSparse](#), and [PrMat](#).

29.358.3.6 `virtual int PrMatrix::rows ( ) const` [pure virtual]

Number of rows.

Implemented in [PrMatSparse](#), and [PrMat](#).

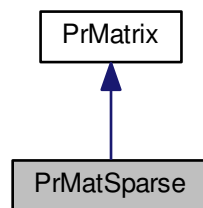
The documentation for this class was generated from the following file:

- [parametrization/include/GoTools/parametrization/PrMatrix.h](#)

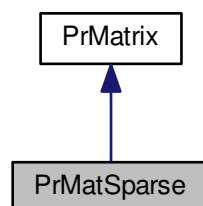
## 29.359 PrMatSparse Class Reference

```
#include <PrMatSparse.h>
```

Inheritance diagram for PrMatSparse:



Collaboration diagram for PrMatSparse:





## Public Member Functions

### Derived from base class

- virtual int `rows () const`  
*Number of rows.*
- virtual int `columns () const`  
*Number of columns.*
- virtual `double operator() (int i, int j) const`  
*value of matrix element*
- virtual void `prod (const PrVec &x, PrVec &y) const`  
*Find  $y = Ax$ .*
- virtual void `print (std::ostream &os)`  
*print contents to stream*
- virtual void `read (std::istream &is)`  
*read contents from stream*
- virtual `~PrMatSparse ()`  
*virtual destructor*

### Other functions

- void `matProd (PrMatSparse &B, PrMatSparse &C) const`
- void `scalProd (double d)`  
*multiply all elements with 'd'*
- void `printFull (std::ostream &os)`  
*print the matrix to stream in 'full' (not sparse) format*
- `PrMatSparse ()`  
*constructor for a sparse matrix of zero size.*
- `PrMatSparse (int m, int n, int num_nonzero)`  
*constructor for an  $(m \times n)$  matrix with 'num\_nonzero' nonzero elements*
- `PrMatSparse (int m, int n, int num_nonzero, const int *irow, const int *jcol, const double *data)`  
*Set to sparse matrix given by irow, jcol and data.*
- void `redim (int m, int n, int num_nonzero)`  
*resize matrix to  $(m \times n)$ , with 'num\_nonzero' nonzero elements*
- void `setToMatrix (const PrMatrix &m, double tol=0.0)`  
*Set equal to another matrix, sparsify if entries  $\leq$  tol.*
- int & `irow (int k)`  
 $< 0 \leq k \leq m_-$  (last element is past-the-end index of a\_)
- const int & `irow (int k) const`  
 $< 0 \leq k \leq m_-$  (last element is past-the-end index of a\_)
- int & `jcol (int k)`  
 $< 0 \leq k \leq p_- - 1$
- const int & `jcol (int k) const`  
 $< 0 \leq k \leq p_- - 1$
- double & `operator() (int k)`  
 $< 0 \leq k \leq p_- - 1$
- const double & `operator() (int k) const`  
 $< 0 \leq k \leq p_- - 1$

### 29.359.1 Detailed Description

Definition at line 49 of file PrMatSparse.h.

## 29.359.2 Constructor & Destructor Documentation

29.359.2.1 `virtual PrMatSparse::~~PrMatSparse ( ) [virtual]`

virtual destructor

29.359.2.2 `PrMatSparse::PrMatSparse ( ) [inline]`

constructor for a sparse matrix of zero size.

Definition at line 87 of file PrMatSparse.h.

29.359.2.3 `PrMatSparse::PrMatSparse ( int m, int n, int num_nonzero )`

constructor for an (m x n) matrix with 'num\_nonzero' nonzero elements

29.359.2.4 `PrMatSparse::PrMatSparse ( int m, int n, int num_nonzero, const int * irow, const int * jcol, const double * data )`

Set to sparse matrix given by irow, jcol and data.

## 29.359.3 Member Function Documentation

29.359.3.1 `virtual int PrMatSparse::columns ( ) const [virtual]`

Number of columns.

Implements [PrMatrix](#).

29.359.3.2 `int & PrMatSparse::irow ( int k ) [inline]`

$0 < k \leq m$  (last element is past-the-end index of `a_`)

Definition at line 134 of file PrMatSparse.h.

29.359.3.3 `const int & PrMatSparse::irow ( int k ) const [inline]`

$0 < k \leq m$  (last element is past-the-end index of `a_`)

Definition at line 127 of file PrMatSparse.h.

29.359.3.4 `int & PrMatSparse::jcol ( int k ) [inline]`

$0 < k \leq p-1$

Definition at line 148 of file PrMatSparse.h.

29.359.3.5 `const int & PrMatSparse::jcol ( int k ) const` `[inline]`

$0 < k \leq p_-1$

Definition at line 141 of file PrMatSparse.h.

29.359.3.6 `void PrMatSparse::matProd ( PrMatSparse & B, PrMatSparse & C ) const`

find  $C = A * B$ . Multiplies two sparse matrices, "A=(this)" times "B" and stores the result in "C"

29.359.3.7 `virtual double PrMatSparse::operator() ( int i, int j ) const` `[virtual]`

value of matrix element

Implements [PrMatrix](#).

29.359.3.8 `double & PrMatSparse::operator() ( int k )` `[inline]`

$0 < k \leq p_-1$

Definition at line 162 of file PrMatSparse.h.

29.359.3.9 `const double & PrMatSparse::operator() ( int k ) const` `[inline]`

$0 < k \leq p_-1$

Definition at line 155 of file PrMatSparse.h.

29.359.3.10 `virtual void PrMatSparse::print ( std::ostream & os )` `[virtual]`

print contents to stream

Reimplemented from [PrMatrix](#).

29.359.3.11 `void PrMatSparse::printFull ( std::ostream & os )`

print the matrix to stream in 'full' (not sparse) format

29.359.3.12 `virtual void PrMatSparse::prod ( const PrVec & x, PrVec & y ) const` `[virtual]`

Find  $y = Ax$ .

Implements [PrMatrix](#).

29.359.3.13 `virtual void PrMatSparse::read ( std::istream & is ) [virtual]`

read contents from stream

Implements [PrMatrix](#).

29.359.3.14 `void PrMatSparse::redim ( int m, int n, int num_nonzero )`

resize matrix to (m x n), with 'num\_nonzero' nonzero elements

29.359.3.15 `virtual int PrMatSparse::rows ( ) const [virtual]`

Number of rows.

Implements [PrMatrix](#).

29.359.3.16 `void PrMatSparse::scalProd ( double d ) [inline]`

multiply all elements with 'd'

Definition at line 117 of file `PrMatSparse.h`.

29.359.3.17 `void PrMatSparse::setToMatrix ( const PrMatrix & m, double tol = 0.0 )`

Set equal to another matrix, sparsify if entries  $\leq$  tol.

The documentation for this class was generated from the following file:

- parametrization/include/GoTools/parametrization/[PrMatSparse.h](#)

## 29.360 PrNestedNode Class Reference

```
#include <PrNestedNode.h>
```

## Public Member Functions

- [PrNestedNode](#) ()  
*Empty default constructor.*
- [PrNestedNode](#) (double x, double y, double z, double u, double v, int level)  
*Constructor.*
- [~PrNestedNode](#) ()  
*Empty default destructor.*
- void [init](#) (double x, double y, double z, double u, double v, int level)
- const [Vector3D](#) & [pnt](#) () const
- const double & [x](#) () const
- const double & [y](#) () const
- const double & [z](#) () const
- const double & [u](#) () const
- const double & [v](#) () const
- const int & [level](#) () const
- const int & [tr](#) (int i) const
- [Vector3D](#) & [pnt](#) ()
- double & [x](#) ()
- double & [y](#) ()
- double & [z](#) ()
- double & [u](#) ()
- double & [v](#) ()
- int & [level](#) ()
- int & [tr](#) (int i)
- void [addTrianglePtr](#) (int t)
- void [print](#) (ostream &os)
- void [printXYZ](#) (ostream &os)
- void [printUV](#) (ostream &os)

### 29.360.1 Detailed Description

This class represents a node for use in [PrLevelTriangulation\\_OP](#) and [PrNestedTriangulation](#). It has double x,y,z values and double u and v values (parameter values). It also has the index of its "first" incident triangle at each triangulation level. If the node is interior then tr[j] is any incident triangle. Otherwise the node is a boundary node, in which case tr[j] is the incident triangle which is the first one in the anticlockwise direction around the node (equivalently the only triangle containing the node whose right neighbour from the point of view of the node is 0).

Definition at line 64 of file PrNestedNode.h.

### 29.360.2 Constructor & Destructor Documentation

#### 29.360.2.1 PrNestedNode::PrNestedNode( ) [inline]

Empty default constructor.

Definition at line 76 of file PrNestedNode.h.

29.360.2.2 `PrNestedNode::PrNestedNode ( double x, double y, double z, double u, double v, int level )` `[inline]`

Constructor.

Definition at line 78 of file PrNestedNode.h.

29.360.2.3 `PrNestedNode::~PrNestedNode ( )` `[inline]`

Empty default destructor.

Definition at line 84 of file PrNestedNode.h.

### 29.360.3 Member Function Documentation

29.360.3.1 `void PrNestedNode::addTrianglePtr ( int t )` `[inline]`

Definition at line 110 of file PrNestedNode.h.

29.360.3.2 `void PrNestedNode::init ( double x, double y, double z, double u, double v, int level )` `[inline]`

Definition at line 121 of file PrNestedNode.h.

29.360.3.3 `const int & PrNestedNode::level ( ) const` `[inline]`

Definition at line 178 of file PrNestedNode.h.

29.360.3.4 `int & PrNestedNode::level ( )` `[inline]`

Definition at line 237 of file PrNestedNode.h.

29.360.3.5 `const Vector3D & PrNestedNode::pnt ( ) const` `[inline]`

Definition at line 136 of file PrNestedNode.h.

29.360.3.6 `Vector3D & PrNestedNode::pnt ( )` `[inline]`

Definition at line 195 of file PrNestedNode.h.

29.360.3.7 void PrNestedNode::print ( ostream & os )

29.360.3.8 void PrNestedNode::printUV ( ostream & os )

29.360.3.9 void PrNestedNode::printXYZ ( ostream & os )

29.360.3.10 const int & PrNestedNode::tr ( int i ) const [inline]

Definition at line 185 of file PrNestedNode.h.

29.360.3.11 int & PrNestedNode::tr ( int i ) [inline]

Definition at line 244 of file PrNestedNode.h.

29.360.3.12 const double & PrNestedNode::u ( ) const [inline]

Definition at line 164 of file PrNestedNode.h.

29.360.3.13 double & PrNestedNode::u ( ) [inline]

Definition at line 223 of file PrNestedNode.h.

29.360.3.14 const double & PrNestedNode::v ( ) const [inline]

Definition at line 171 of file PrNestedNode.h.

29.360.3.15 double & PrNestedNode::v ( ) [inline]

Definition at line 230 of file PrNestedNode.h.

29.360.3.16 const double & PrNestedNode::x ( ) const [inline]

Definition at line 143 of file PrNestedNode.h.

29.360.3.17 double & PrNestedNode::x ( ) [inline]

Definition at line 202 of file PrNestedNode.h.

29.360.3.18 const double & PrNestedNode::y ( ) const [inline]

Definition at line 150 of file PrNestedNode.h.

29.360.3.19 `double & PrNestedNode::y( )` [inline]

Definition at line 209 of file PrNestedNode.h.

29.360.3.20 `const double & PrNestedNode::z( ) const` [inline]

Definition at line 157 of file PrNestedNode.h.

29.360.3.21 `double & PrNestedNode::z( )` [inline]

Definition at line 216 of file PrNestedNode.h.

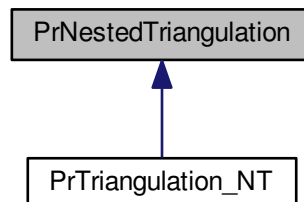
The documentation for this class was generated from the following file:

- parametrization/include/GoTools/parametrization/PrNestedNode.h

## 29.361 PrNestedTriangulation Class Reference

```
#include <PrNestedTriangulation.h>
```

Inheritance diagram for PrNestedTriangulation:



### Public Member Functions

#### Pure virtual functions...

- virtual int `getFinestLevel` ()=0
- virtual int `getNumNodes` (int jlev)=0  
*return the number of nodes in  $V^j$ ,  $j = 0, 1, \dots, k$ .*
- virtual double `getX` (int i)=0  
*return the x coeff. of the i-th node in the triangulation.*
- virtual double `getY` (int i)=0  
*return the y coeff. of the i-th node in the triangulation.*
- virtual double `getZ` (int i)=0  
*return the z coeff. of the i-th node in the triangulation.*



- virtual void `setX` (int *i*, const double &x)=0  
*set the x coeff. of the i-th node in the triangulation.*
- virtual void `setY` (int *i*, const double &y)=0  
*set the y coeff. of the i-th node in the triangulation.*
- virtual void `setZ` (int *i*, const double &z)=0  
*set the z coeff. of the i-th node in the triangulation.*
- virtual void `getNeighbours` (int *i*, int *jlev*, vector< int > &neighbours)=0
- virtual bool `isBoundary` (int *i*)=0  
*is the i-th node a boundary node or interior node?*

#### Other functions

- virtual `~PrNestedTriangulation` ()  
*Empty virtual default destructor.*
- void `getParents` (int *i*, int *jlev*, int &p1, int &p2)

### 29.361.1 Detailed Description

This class implements a set of nested triangulations. The methods in this class are used by PrFilterBank.

Definition at line 51 of file PrNestedTriangulation.h.

### 29.361.2 Constructor & Destructor Documentation

29.361.2.1 virtual PrNestedTriangulation::~PrNestedTriangulation ( ) [inline],[virtual]

Empty virtual default destructor.

Definition at line 93 of file PrNestedTriangulation.h.

### 29.361.3 Member Function Documentation

29.361.3.1 virtual int PrNestedTriangulation::getFinestLevel ( ) [pure virtual]

Return the finest level in the hierarchy, i.e.  $k$  if the vertices are arranged  $V^0 \subset V^1 \subset \dots \subset V^k$ .

Implemented in [PrTriangulation\\_NT](#).

29.361.3.2 virtual void PrNestedTriangulation::getNeighbours ( int *i*, int *jlev*, vector< int > &neighbours ) [pure virtual]

Return the indices of the neighbours of the *i*-th node with respect to  $V^j$ . This routine assumes that node *i* also belongs to  $V^j$ . The order of the neighbours returned will be

1. any anticlockwise order if *i* is an interior node
2. the unique anticlockwise order if *i* is a boundary node.

Implemented in [PrTriangulation\\_NT](#).

29.361.3.3 `virtual int PrNestedTriangulation::getNumNodes ( int jlev ) [pure virtual]`

return the number of nodes in  $V^j, j = 0, 1, \dots, k$ .

Implemented in [PrTriangulation\\_NT](#).

29.361.3.4 `void PrNestedTriangulation::getParents ( int i, int jlev, int & p1, int & p2 )`

Assuming that node  $i$  belongs to  $V^j \setminus V^{j-1}$  and that  $jlev \geq 1$ , find  $i$ 's two parent nodes in  $V^{j-1}$   $p1$  and  $p2$ .

29.361.3.5 `virtual double PrNestedTriangulation::getX ( int i ) [pure virtual]`

return the x coeff. of the  $i$ -th node in the triangulation.

Implemented in [PrTriangulation\\_NT](#).

29.361.3.6 `virtual double PrNestedTriangulation::getY ( int i ) [pure virtual]`

return the y coeff. of the  $i$ -th node in the triangulation.

Implemented in [PrTriangulation\\_NT](#).

29.361.3.7 `virtual double PrNestedTriangulation::getZ ( int i ) [pure virtual]`

return the z coeff. of the  $i$ -th node in the triangulation.

Implemented in [PrTriangulation\\_NT](#).

29.361.3.8 `virtual bool PrNestedTriangulation::isBoundary ( int i ) [pure virtual]`

is the  $i$ -th node a boundary node or interior node?

Implemented in [PrTriangulation\\_NT](#).

29.361.3.9 `virtual void PrNestedTriangulation::setX ( int i, const double & x ) [pure virtual]`

set the x coeff. of the  $i$ -th node in the triangulation.

Implemented in [PrTriangulation\\_NT](#).

29.361.3.10 `virtual void PrNestedTriangulation::setY ( int i, const double & y ) [pure virtual]`

set the y coeff. of the  $i$ -th node in the triangulation.

Implemented in [PrTriangulation\\_NT](#).

29.361.3.11 `virtual void PrNestedTriangulation::setZ ( int i, const double & z ) [pure virtual]`

set the z coeff. of the i-th node in the triangulation.

Implemented in [PrTriangulation\\_NT](#).

The documentation for this class was generated from the following file:

- parametrization/include/GoTools/parametrization/[PrNestedTriangulation.h](#)

## 29.362 PrNode Class Reference

```
#include <PrNode.h>
```

### Public Member Functions

- [PrNode](#) ()
- [PrNode](#) (double x, double y, double z, double u, double v, int tr)
- void [init](#) (double x, double y, double z, double u, double v, int tr)
- const [Vector3D](#) & [point](#) () const
- const double & [u](#) () const
- const double & [v](#) () const
- const double & [x](#) () const
- const double & [y](#) () const
- const double & [z](#) () const
- const int & [tr](#) () const
- double & [u](#) ()
- double & [v](#) ()
- double & [x](#) ()
- double & [y](#) ()
- double & [z](#) ()
- int & [tr](#) ()
- void [print](#) (std::ostream &os) const
- void [printXYZ](#) (std::ostream &os) const
- void [printUV](#) (std::ostream &os) const
- void [scan](#) (std::istream &is)

### 29.362.1 Detailed Description

This class represents a node for use in [PrTriangulation\\_OP](#). It has double x,y,z values contained in its parent class [Vector3D](#). It also has double u and v values (parameter values) and the index of its "first" incident triangle tr. If the node is interior then tr is any incident triangle. Otherwise the node is a boundary node, in which case tr is the incident triangle which is the first one in the anticlockwise direction around the node (equivalently the only triangle containing the node whose right neighbour from the point of view of the node is 0).

Definition at line 60 of file [PrNode.h](#).

## 29.362.2 Constructor & Destructor Documentation

29.362.2.1 `PrNode::PrNode ( )` `[inline]`

Definition at line 68 of file PrNode.h.

29.362.2.2 `PrNode::PrNode ( double x, double y, double z, double u, double v, int tr )` `[inline]`

Definition at line 71 of file PrNode.h.

## 29.362.3 Member Function Documentation

29.362.3.1 `void PrNode::init ( double x, double y, double z, double u, double v, int tr )` `[inline]`

Definition at line 104 of file PrNode.h.

29.362.3.2 `const Vector3D& PrNode::point ( ) const` `[inline]`

Definition at line 78 of file PrNode.h.

29.362.3.3 `void PrNode::print ( std::ostream & os ) const`

29.362.3.4 `void PrNode::printUV ( std::ostream & os ) const`

29.362.3.5 `void PrNode::printXYZ ( std::ostream & os ) const`

29.362.3.6 `void PrNode::scan ( std::istream & is )`

29.362.3.7 `const int & PrNode::tr ( ) const` `[inline]`

Definition at line 130 of file PrNode.h.

29.362.3.8 `int & PrNode::tr ( )` `[inline]`

Definition at line 151 of file PrNode.h.

29.362.3.9 `const double & PrNode::u ( ) const` `[inline]`

Definition at line 116 of file PrNode.h.

29.362.3.10 `double & PrNode::u ( ) [inline]`

Definition at line 137 of file PrNode.h.

29.362.3.11 `const double & PrNode::v ( ) const [inline]`

Definition at line 123 of file PrNode.h.

29.362.3.12 `double & PrNode::v ( ) [inline]`

Definition at line 144 of file PrNode.h.

29.362.3.13 `const double & PrNode::x ( ) const [inline]`

Definition at line 158 of file PrNode.h.

29.362.3.14 `double & PrNode::x ( ) [inline]`

Definition at line 165 of file PrNode.h.

29.362.3.15 `const double & PrNode::y ( ) const [inline]`

Definition at line 172 of file PrNode.h.

29.362.3.16 `double & PrNode::y ( ) [inline]`

Definition at line 179 of file PrNode.h.

29.362.3.17 `const double & PrNode::z ( ) const [inline]`

Definition at line 186 of file PrNode.h.

29.362.3.18 `double & PrNode::z ( ) [inline]`

Definition at line 193 of file PrNode.h.

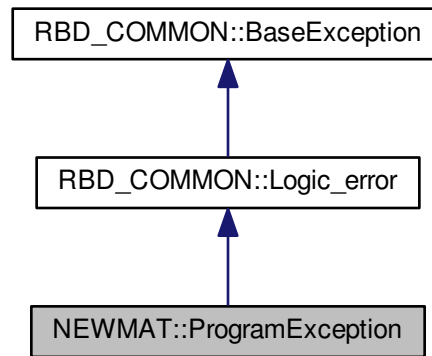
The documentation for this class was generated from the following file:

- parametrization/include/GoTools/parametrization/[PrNode.h](#)

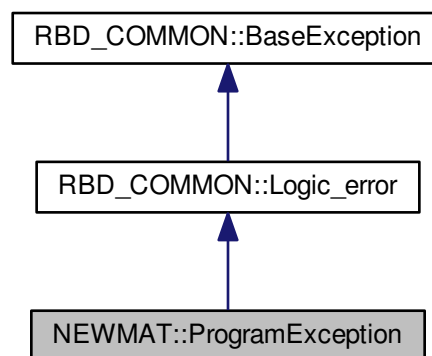
## 29.363 NEWMAT::ProgramException Class Reference

```
#include <newmat.h>
```

Inheritance diagram for NEWMAT::ProgramException:



Collaboration diagram for NEWMAT::ProgramException:



### Public Member Functions

- [ProgramException](#) (const char \*c)
- [ProgramException](#) (const char \*c, const GeneralMatrix &)
- [ProgramException](#) (const char \*c, const GeneralMatrix &, const GeneralMatrix &)
- [ProgramException](#) (const char \*c, MatrixType, MatrixType)

## Static Public Attributes

- static unsigned long [Select](#)

## Protected Member Functions

- [ProgramException](#) ()

## Additional Inherited Members

### 29.363.1 Detailed Description

Definition at line 1624 of file newmat.h.

### 29.363.2 Constructor & Destructor Documentation

29.363.2.1 NEWMAT::ProgramException::ProgramException ( ) `[protected]`

29.363.2.2 ProgramException::ProgramException ( const char \* c )

Definition at line 88 of file newmatex.cpp.

29.363.2.3 NEWMAT::ProgramException::ProgramException ( const char \* c, const GeneralMatrix & )

29.363.2.4 NEWMAT::ProgramException::ProgramException ( const char \* c, const GeneralMatrix &, const GeneralMatrix & )

29.363.2.5 NEWMAT::ProgramException::ProgramException ( const char \* c, MatrixType, MatrixType )

### 29.363.3 Member Data Documentation

29.363.3.1 unsigned long ProgramException::Select `[static]`

Definition at line 1629 of file newmat.h.

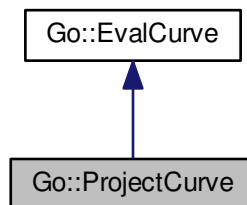
The documentation for this class was generated from the following files:

- newmat/include/newmat.h
- newmat/src/newmatex.cpp

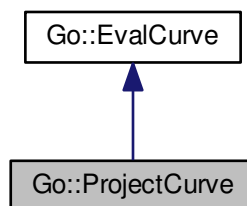
## 29.364 Go::ProjectCurve Class Reference

```
#include <ProjectCurve.h>
```

Inheritance diagram for Go::ProjectCurve:



Collaboration diagram for Go::ProjectCurve:



### Public Member Functions

- [ProjectCurve](#) (shared\_ptr< [Go::ParamCurve](#) > &space\_crv, shared\_ptr< [Go::ParamSurface](#) > &surf, shared\_ptr< [Go::Point](#) > &start\_par\_pt, shared\_ptr< [Go::Point](#) > &end\_par\_pt, double epsgeo1, const [RectDomain](#) \*domain\_of\_interest=NULL)
- virtual [~ProjectCurve](#) ()
  - virtual destructor ensures safe inheritance*
- virtual [Go::Point](#) [eval](#) (double t) const
- [Go::Point](#) [eval](#) (double t, [Go::Point](#) seed) const
  - Evaluate point, given seed for the closest point iteration involved.*
- virtual void [eval](#) (double t, int n, [Go::Point](#) der[]) const
- virtual double [start](#) () const
- virtual double [end](#) () const
- virtual int [dim](#) () const
- virtual bool [approximationOK](#) (double par, [Go::Point](#) approxpos, double tol1, double tol2) const



### 29.364.1 Detailed Description

This class represents the curve obtained by projecting a given 3D curve onto a given part of a given 3D surface.

Definition at line 58 of file ProjectCurve.h.

### 29.364.2 Constructor & Destructor Documentation

29.364.2.1 `Go::ProjectCurve::ProjectCurve ( shared_ptr< Go::ParamCurve > & space_crv, shared_ptr< Go::ParamSurface > & surf, shared_ptr< Go::Point > & start_par_pt, shared_ptr< Go::Point > & end_par_pt, double epsgeo1, const RectDomain * domain_of_interest = NULL )`

Constructor, taking one 3D curve and one 3D surface. The user may additionally provide explicit values for the start and end points of the curve (supposedly on the surface) as well as specifying the parameter domain of interest of the surface.

#### Parameters

<i>space_crv</i>	the 3D space curve that will be projected onto a surface
<i>surf</i>	the 3D surface that we will project the space curve onto. Assuming that the surface is k-regular.
<i>start_par_pt</i>	explicit position of start point (can be a zero pointer, in which case the start point will be evaluated by projection, just like any other point).
<i>end_par_pt</i>	explicit position of end point (can be a zero pointer, in which case the start point will be evaluated by projection, just like any other point).
<i>epsgeo1</i>	geometric tolerance to use when projecting curve onto surface, and when using the <a href="#">approximationOK()</a> function, max dist from exact proj.
<i>epsgeo2</i>	geometric tolerance when using the <a href="#">approximationOK()</a> function, max dist from space_crv. Negative = ignored.
<i>domain_of_interest</i>	if the user wants to limit the surface to a certain parametric domain, it can be specified here.

29.364.2.2 `virtual Go::ProjectCurve::~~ProjectCurve ( ) [virtual]`

virtual destructor ensures safe inheritance

### 29.364.3 Member Function Documentation

29.364.3.1 `virtual bool Go::ProjectCurve::approximationOK ( double par, Go::Point approxpos, double tol1, double tol2 ) const [virtual]`

Inherited from [EvalCurve::approximationOK\(\)](#). For this class, the specified tolerances are not used; the internally stored 'epsgeo' value is used as tolerance (this value was specified in the constructor).

#### Parameters

<i>par</i>	the parameter at which to check the curve
<i>approxpos</i>	the position we want to check whether or not the curve approximates for parameter 'par'.
<i>tol1</i>	unused
<i>tol2</i>	unused

**Returns**

'true' if the curve approximates the point at the parameter, 'false' otherwise.

Implements [Go::EvalCurve](#).

**29.364.3.2** `virtual int Go::ProjectCurve::dim ( ) const [virtual]`

Inherited from [EvalCurve::dim\(\)](#). For this class, the returned dimension will be that of the surface parameter domain, ie. 2, NOT that of the space curve.

Implements [Go::EvalCurve](#).

**29.364.3.3** `virtual double Go::ProjectCurve::end ( ) const [virtual]`

Get the end parameter of the curve.

**Returns**

the end parameter of the curve.

Implements [Go::EvalCurve](#).

**29.364.3.4** `virtual Go::Point Go::ProjectCurve::eval ( double t ) const [virtual]`

Evaluate a point on the curve for a given parameter

**Parameters**

<i>t</i>	the parameter for which to evaluate the curve.
----------	------------------------------------------------

**Returns**

the evaluated point

Implements [Go::EvalCurve](#).

**29.364.3.5** `Go::Point Go::ProjectCurve::eval ( double t, Go::Point seed ) const`

Evaluate point, given seed for the closest point iteration involved.

**29.364.3.6** `virtual void Go::ProjectCurve::eval ( double t, int n, Go::Point der[] ) const [virtual]`

Evaluate a point and a certain number of derivatives on the curve for a given parameter.

## Parameters

<i>t</i>	the parameter for which to evaluate the curve.
<i>n</i>	the number of derivatives (0 or more)

## Return values

<i>der</i>	pointer to an array of Points where the result will be written. The position will be stored first, then the first derivative (tangent), then the second, etc.. <b>NB:</b> For most (all) derived classes of 'EvalCurve', the implementation actually only supports the computation of one derivative, i.e. if $n > 1$ , only one derivative will be computed anyway.
------------	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Implements [Go::EvalCurve](#).

29.364.3.7 virtual double Go::ProjectCurve::start ( ) const [virtual]

Get the start parameter of the curve.

## Returns

the start parameter of the curve.

Implements [Go::EvalCurve](#).

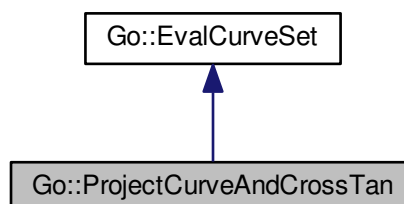
The documentation for this class was generated from the following file:

- [gotools-core/include/GoTools/creators/ProjectCurve.h](#)

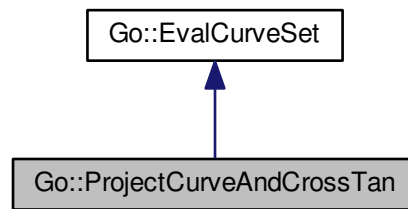
## 29.365 Go::ProjectCurveAndCrossTan Class Reference

```
#include <ProjectCurveAndCrossTan.h>
```

Inheritance diagram for Go::ProjectCurveAndCrossTan:



Collaboration diagram for Go::ProjectCurveAndCrossTan:



## Public Member Functions

- [ProjectCurveAndCrossTan](#) ([const SplineCurve](#) &space\_crv, [const SplineCurve](#) &crosstan\_crv, [const SplineSurface](#) &surf, [const Point](#) \*start\_par\_pt, [const Point](#) \*end\_par\_pt, [double](#) epsgeo, [const RectDomain](#) \*domain\_of\_interest=NULL)
- virtual [~ProjectCurveAndCrossTan](#) ()
  - Destructor.*
- virtual [std::vector< Go::Point > eval](#) ([double](#) t)
- virtual [void eval](#) ([double](#) t, [int](#) n, [std::vector< std::vector< Go::Point > >](#) &ders)
- virtual [double start](#) ()
- virtual [double end](#) ()
- virtual [int dim](#) ()
- virtual [bool approximationOK](#) ([double](#) par, [const std::vector< Go::Point >](#) &approxpos, [double](#) tol1, [double](#) tol2)
- virtual [int nmbCvs](#) ()

### 29.365.1 Detailed Description

Evaluator class representing the projection and and tangent curve given a set of input curves. In total 3 points are computed: the parameter point of the projection on an input surface, the corresponding space pt and the projected cross tangent point.

Definition at line 60 of file ProjectCurveAndCrossTan.h.

### 29.365.2 Constructor & Destructor Documentation

**29.365.2.1** [Go::ProjectCurveAndCrossTan::ProjectCurveAndCrossTan](#) ( [const SplineCurve](#) & space\_crv, [const SplineCurve](#) & crosstan\_crv, [const SplineSurface](#) & surf, [const Point](#) \* start\_par\_pt, [const Point](#) \* end\_par\_pt, [double](#) epsgeo, [const RectDomain](#) \* domain\_of\_interest = NULL )

Constructor.

Parameters

<i>space_crv</i>	the input space curve to project.
------------------	-----------------------------------

## Parameters

<i>crosstan_crv</i>	the cross tangent curve associated with <i>space_crv</i> .
<i>surf</i>	the parametric surface onto which to project the two curves.
<i>start_par_pt</i>	We may require that the projected curve shall start in a specific point in the parameter domain (input pointer may be NULL).
<i>end_par_pt</i>	We may require that the projected curve shall end in a specific point in the parameter domain (input pointer may be NULL).
<i>epsgeo</i>	geometric tolerance for the projection.
<i>domain_of_interest</i>	if not null, specifies the part of the <a href="#">SplineSurface</a> s parametric domain that will be considered.

29.365.2.2 `virtual Go::ProjectCurveAndCrossTan::~~ProjectCurveAndCrossTan ( ) [virtual]`

Destructor.

## 29.365.3 Member Function Documentation

29.365.3.1 `virtual bool Go::ProjectCurveAndCrossTan::approximationOK ( double par, const std::vector< Go::Point > & approxpos, double tol1, double tol2 ) [virtual]`

Whether the approximation is within tolerances in input parameter.

## Parameters

<i>par</i>	parameter in which to evaluate.
<i>approxpos</i>	whether the input points are within tolerance from the evaluated points (as given by <a href="#">eval()</a> ).
<i>tol1</i>	tolerance used to decide approximation accuracy.
<i>tol2</i>	tolerance used to decide approximation accuracy.

## Returns

whether the approximation is within tolerances in input parameter.

29.365.3.2 `virtual int Go::ProjectCurveAndCrossTan::dim ( ) [virtual]`

Dimension of space (parameter domain, i.e. 2).

## Returns

geometric dimension of the space.

Implements [Go::EvalCurveSet](#).

29.365.3.3 `virtual double Go::ProjectCurveAndCrossTan::end ( ) [virtual]`

End parameter of domain.

#### Returns

end parameter of domain.

Implements [Go::EvalCurveSet](#).

29.365.3.4 `virtual std::vector<Go::Point> Go::ProjectCurveAndCrossTan::eval ( double t ) [virtual]`

Evaluate the curve set.

#### Parameters

<i>t</i>	parameter in which to evaluate.
----------	---------------------------------

#### Returns

the evaluated points in the input parameter, size of vector is 3.

Implements [Go::EvalCurveSet](#).

29.365.3.5 `virtual void Go::ProjectCurveAndCrossTan::eval ( double t, int n, std::vector< std::vector< Go::Point > > & ders ) [virtual]`

Evaluate derivatives in the curve set.

#### Parameters

<i>t</i>	parameter in which to evaluate.
<i>n</i>	the number of derivatives to evaluate.
<i>ders</i>	the evaluated points in the input parameter. Size of vector is 3, each holding 'n + 1' points.

29.365.3.6 `virtual int Go::ProjectCurveAndCrossTan::nmbCvs ( ) [virtual]`

The number of curves in the curve set.

#### Returns

the number of curves in the curve set, i.e. 3.

Implements [Go::EvalCurveSet](#).

29.365.3.7 virtual double Go::ProjectCurveAndCrossTan::start ( ) [virtual]

Start parameter of domain.

#### Returns

start parameter of domain.

Implements [Go::EvalCurveSet](#).

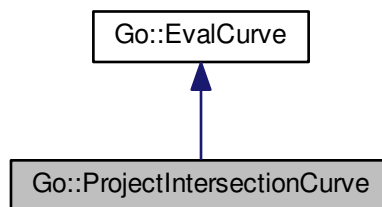
The documentation for this class was generated from the following file:

- [gotools-core/include/GoTools/creators/ProjectCurveAndCrossTan.h](#)

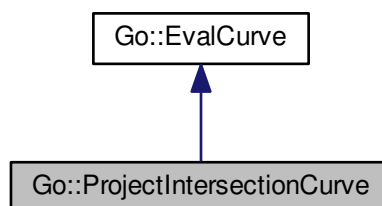
## 29.366 Go::ProjectIntersectionCurve Class Reference

```
#include <ProjectIntersectionCurve.h>
```

Inheritance diagram for Go::ProjectIntersectionCurve:



Collaboration diagram for Go::ProjectIntersectionCurve:



## Public Member Functions

- [ProjectIntersectionCurve](#) (shared\_ptr< [SplineCurve](#) > &inters\_crv, shared\_ptr< [SplineCurve](#) > &p\_crv, shared\_ptr< [SplineCurve](#) > &other\_p\_crv, shared\_ptr< [ParamSurface](#) > &surf, shared\_ptr< [ParamSurface](#) > &other\_surf, double offset\_dist, double other\_offset\_dist, double epsgeo)
- virtual [~ProjectIntersectionCurve](#) ()
  - Destructor.*
- virtual [Point](#) eval (double t) const
- virtual void eval (double t, int n, [Point](#) der[]) const
- virtual double start () const
- virtual double end () const
- virtual int dim () const
- virtual bool approximationOK (double par, [Point](#) approxpos, double tol1, double tol2) const

### 29.366.1 Detailed Description

This class provides an interface to a curve that can be evaluated. This evaluator based class computes the projected point in the first surface of the offset point defined by input.

Definition at line 58 of file [ProjectIntersectionCurve.h](#).

### 29.366.2 Constructor & Destructor Documentation

- 29.366.2.1 `Go::ProjectIntersectionCurve::ProjectIntersectionCurve ( shared_ptr< SplineCurve > & inters_crv, shared_ptr< SplineCurve > & p_crv, shared_ptr< SplineCurve > & other_p_crv, shared_ptr< ParamSurface > & surf, shared_ptr< ParamSurface > & other_surf, double offset_dist, double other_offset_dist, double epsgeo )`

Constructor.

#### Parameters

<i>inters_crv</i>	the space intersection curve between surf & other_surf.
<i>p_crv</i>	the corresponding parameter curve in surf.
<i>other_p_crv</i>	the corresponding parameter curve in other_surf.
<i>surf</i>	the first input surface.
<i>other_surf</i>	the second input surface.
<i>offset_dist</i>	the offset distance in surf.
<i>other_offset_dist</i>	the offset distance in other_surf.
<i>epsgeo</i>	the geometrical tolerance (for closes point evaluations).

- 29.366.2.2 `virtual Go::ProjectIntersectionCurve::~~ProjectIntersectionCurve ( ) [virtual]`

Destructor.

### 29.366.3 Member Function Documentation



29.366.3.1 `virtual bool Go::ProjectIntersectionCurve::approximationOK ( double par, Point approxpos, double tol1, double tol2 ) const` `[virtual]`

Whether the evaluated point in *par* is close enough to *approxpos*.

#### Parameters

<i>par</i>	the parameter in which to evaluate.
<i>approxpos</i>	position to check for accuracy.
<i>tol1</i>	currently not used.
<i>tol2</i>	currently not used.

#### Returns

whether *approxpos* is within satisfactory accuracy (i.e. `epsgeo_`).

Implements [Go::EvalCurve](#).

29.366.3.2 `virtual int Go::ProjectIntersectionCurve::dim ( ) const` `[virtual]`

Dimension of `inters_crv_`.

#### Returns

the dimension of the evaluator point.

Implements [Go::EvalCurve](#).

29.366.3.3 `virtual double Go::ProjectIntersectionCurve::end ( ) const` `[virtual]`

End parameter of curve.

#### Returns

the end parameter.

Implements [Go::EvalCurve](#).

29.366.3.4 `virtual Point Go::ProjectIntersectionCurve::eval ( double t ) const` `[virtual]`

The evaluator part of the class, returns the projected offset point in the first surface.

#### Parameters

<i>t</i>	the parameter in which to evaluate.
----------	-------------------------------------

Implements [Go::EvalCurve](#).

29.366.3.5 `virtual void Go::ProjectIntersectionCurve::eval ( double t, int n, Point der[] ) const [virtual]`

The evaluator part of the class, returns the projected offset point in the first surface. For  $n == 1$  the tangent in the projected offset curve is also computed.

#### Parameters

<i>t</i>	the parameter in which to evaluate.
<i>n</i>	the number of derivatives to compute (at most 1).
<i>der</i>	the evaluated point. Size of array is 'n + 1'.

Implements [Go::EvalCurve](#).

29.366.3.6 `virtual double Go::ProjectIntersectionCurve::start ( ) const [virtual]`

Start parameter of curve.

#### Returns

the start parameter.

Implements [Go::EvalCurve](#).

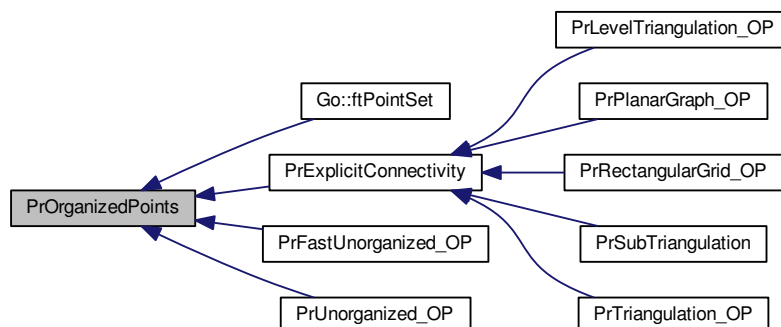
The documentation for this class was generated from the following file:

- `gotools-core/include/GoTools/creators/ProjectIntersectionCurve.h`

## 29.367 PrOrganizedPoints Class Reference

```
#include <PrOrganizedPoints.h>
```

Inheritance diagram for PrOrganizedPoints:



## Public Member Functions

### Pure virtual functions...

- virtual int [getNumNodes](#) () const =0  
*return the number of nodes in the graph.*
- virtual Vector3D [get3dNode](#) (int i) const =0  
*return the i-th node in the graph if the nodes are three-dimensional*
- virtual void [set3dNode](#) (int i, const Vector3D &p)=0
- virtual void [getNeighbours](#) (int i, std::vector< int > &neighbours) const =0
- virtual bool [isBoundary](#) (int i) const =0  
*is the i-th node a boundary node or interior node?*
- virtual double [getU](#) (int i) const =0  
*return the u parameter value of the i-th node.*
- virtual double [getV](#) (int i) const =0  
*return the v parameter value of the i-th node.*
- virtual void [setU](#) (int i, double u)=0  
*reset the u parameter value of the i-th node.*
- virtual void [setV](#) (int i, double v)=0  
*reset the v parameter value of the i-th node.*
- virtual [~PrOrganizedPoints](#) ()  
*Empty default destructor.*

### Print routines

- virtual void [printXYZNodes](#) (std::ostream &os, bool num=0) const  
*Print out the XYZ nodes of the graph. If num = true, print first the number of nodes.*
- virtual void [printUVNodes](#) (std::ostream &os, bool num=0) const  
*Print out the UV nodes of the graph. If num = 1, print first the number of nodes.*
- virtual void [printUVXYZNodes](#) (std::ostream &os, bool num=0) const  
*Print out the nodes of the graph. If num = 1, print first the number of nodes.*
- virtual void [printXYZEdges](#) (std::ostream &os) const  
*Print out the edges of the graph.*
- virtual void [printUVEEdges](#) (std::ostream &os) const  
*Print out the edges of the parametrization.*
- virtual void [printInfo](#) (std::ostream &os) const  
*Print general information about this PrOrganizedPoints object (number of nodes, etc.)*

### Other functions

- virtual int [findNumBdyNodes](#) () const  
*count the number of boundary nodes in the graph.*
- virtual int [findNumEdges](#) () const  
*count the number of edges in the graph*
- int [findNumComponents](#) () const  
*Compute the number of connected components of the graph.*
- void [labelNode](#) (int i, int ic, std::vector< int > &component) const
- int [findNumBdyComponents](#) () const  
*Compute the number of separate boundaries of the graph.*
- void [labelBdyNode](#) (int i, int ic, std::vector< int > &component) const
- int [indexComponents](#) (std::vector< int > &component, std::vector< int > &newIndex) const
- void [labelNode](#) (int i, int ic, int &index, std::vector< int > &component, std::vector< int > &newIndex) const
- int [findIndex](#) (Vector3D &point) const
- void [topologicalDistToBdy](#) (std::vector< int > &label) const
- void [getCommonNeighbours](#) (int j, int k, std::vector< int > &neighbours) const  
*Return the common neighbours of the j-th and k-th node in some arbitrary order.*
- void [get2Neighbours](#) (int i, std::vector< int > &neighbours) const  
*Get the 2-neighbours of the node 'i'.*
- static bool [isMinimum](#) (int i, std::vector< int > &face)

### 29.367.1 Detailed Description

[PrOrganizedPoints](#) - This class implements an interface to a planar graph embedded in  $R^2$  or  $R^3$ . The two common examples are a triangulation and a topologically rectangular grid, but it can also represent point clouds with no explicit connectivity, as long as the boundary is identified. The methods in this class are used by [PrParametrize](#).

Definition at line 60 of file [PrOrganizedPoints.h](#).

### 29.367.2 Constructor & Destructor Documentation

29.367.2.1 `virtual PrOrganizedPoints::~PrOrganizedPoints ( ) [virtual]`

Empty default destructor.

### 29.367.3 Member Function Documentation

29.367.3.1 `int PrOrganizedPoints::findIndex ( Vector3D & point ) const`

find the index of the node whose (x,y,z) point is exactly equal to the given point. This can be useful for relocating "corner" points after triangulating (if the graph is a triangulation).

29.367.3.2 `int PrOrganizedPoints::findNumBdyComponents ( ) const`

Compute the number of separate boundaries of the graph.

29.367.3.3 `virtual int PrOrganizedPoints::findNumBdyNodes ( ) const [virtual]`

count the number of boundary nodes in the graph.

Reimplemented in [PrSubTriangulation](#).

29.367.3.4 `int PrOrganizedPoints::findNumComponents ( ) const`

Compute the number of connected components of the graph.

29.367.3.5 `virtual int PrOrganizedPoints::findNumEdges ( ) const [virtual]`

count the number of edges in the graph

29.367.3.6 `void PrOrganizedPoints::get2Neighbours ( int i, std::vector< int > & neighbours ) const`

Get the 2-neighbours of the node 'i'.

29.367.3.7 `virtual Vector3D PrOrganizedPoints::get3dNode ( int i ) const [pure virtual]`

return the i-th node in the graph if the nodes are three-dimensional

Implemented in [Go::ftPointSet](#), [PrTriangulation\\_OP](#), [PrSubTriangulation](#), [PrPlanarGraph\\_OP](#), [PrFastUnorganized\\_OP](#), [PrRectangularGrid\\_OP](#), and [PrLevelTriangulation\\_OP](#).

29.367.3.8 `void PrOrganizedPoints::getCommonNeighbours ( int j, int k, std::vector< int > & neighbours ) const`

Return the common neighbours of the j-th and k-th node in some arbitrary order.

29.367.3.9 `virtual void PrOrganizedPoints::getNeighbours ( int i, std::vector< int > & neighbours ) const [pure virtual]`

Return the indices of the neighbours of the i-th node in:

1. Any anticlockwise order if i is an interior node.
2. The unique anticlockwise order if i is a boundary node. This routine should be implemented in any derived class by first calling `clear()` followed successively by `push_back()` on the neighbours vector.

Implemented in [Go::ftPointSet](#), and [PrTriangulation\\_OP](#).

29.367.3.10 `virtual int PrOrganizedPoints::getNumNodes ( ) const [pure virtual]`

return the number of nodes in the graph.

Implemented in [Go::ftPointSet](#), [PrTriangulation\\_OP](#), [PrSubTriangulation](#), [PrPlanarGraph\\_OP](#), [PrFastUnorganized\\_OP](#), [PrRectangularGrid\\_OP](#), and [PrLevelTriangulation\\_OP](#).

29.367.3.11 `virtual double PrOrganizedPoints::getU ( int i ) const [pure virtual]`

return the u parameter value of the i-th node.

Implemented in [Go::ftPointSet](#), [PrTriangulation\\_OP](#), [PrPlanarGraph\\_OP](#), [PrSubTriangulation](#), [PrFastUnorganized\\_OP](#), [PrLevelTriangulation\\_OP](#), and [PrRectangularGrid\\_OP](#).

29.367.3.12 `virtual double PrOrganizedPoints::getV ( int i ) const [pure virtual]`

return the v parameter value of the i-th node.

Implemented in [Go::ftPointSet](#), [PrPlanarGraph\\_OP](#), [PrTriangulation\\_OP](#), [PrSubTriangulation](#), [PrFastUnorganized\\_OP](#), [PrLevelTriangulation\\_OP](#), and [PrRectangularGrid\\_OP](#).

29.367.3.13 `int PrOrganizedPoints::indexComponents ( std::vector< int > & component, std::vector< int > & newIndex ) const`

Locate all connected components of the graph, give each of them an unique index (starting from 0), and giving all nodes of each component "local" indexes pertaining to that component.

## Return values

<i>component</i>	will be resized to number of nodes. Each entry will contain the index of the component in which the corresponding node is located
<i>newIndex</i>	will be resized to number of nodes. Each entry will contain the LOCAL index of the corresponding node in the component it belongs to.

**29.367.3.14** `virtual bool PrOrganizedPoints::isBoundary ( int i ) const [pure virtual]`

is the i-th node a boundary node or interior node?

Implemented in [Go::ftPointSet](#), [PrTriangulation\\_OP](#), [PrPlanarGraph\\_OP](#), [PrSubTriangulation](#), [PrFastUnorganizedOP](#), [PrLevelTriangulation\\_OP](#), and [PrRectangularGrid\\_OP](#).

**29.367.3.15** `static bool PrOrganizedPoints::isMinimum ( int i, std::vector< int > & face ) [static]`

Check if there are any elements of the vector 'face' that are inferior to the value of 'i'.

**29.367.3.16** `void PrOrganizedPoints::labelBdyNode ( int i, int ic, std::vector< int > & component ) const`

Locate all nodes on the same boundary as the node 'i' (supposedly a boundary node) and mark their corresponding entry in the vector 'component' with the value 'ic'.

## Parameters

<i>i</i>	index specifying a boundary node
<i>ic</i>	the 'label value' to write to the concerned entries in the 'component' vector

## Return values

<i>component</i>	the vector where the labels will be written. Its size should be equal to the number of nodes in the <a href="#">PrOrganizedPoints</a> object before calling this function.
------------------	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------

**29.367.3.17** `void PrOrganizedPoints::labelNode ( int i, int ic, std::vector< int > & component ) const`

Locate all nodes connected to node 'i', and mark their corresponding entry in the vector 'component' with the value 'ic'.

## Parameters

<i>i</i>	index specifying the node
<i>ic</i>	the 'label value' to write to the concerned entries in the 'component' vector

## Return values

<i>component</i>	the vector where the labels will be written. Its size should be equal to the number of nodes in the <a href="#">PrOrganizedPoints</a> object before calling this function.
------------------	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------

**29.367.3.18** void [PrOrganizedPoints::labelNode](#) ( int *i*, int *ic*, int & *index*, std::vector< int > & *component*, std::vector< int > & *newIndex* ) const

Locate all nodes in the same connected component as the node 'i', and label their corresponding entries in the vector 'component' with the value 'ic'. These nodes are also given "local" indexes for that component. These indexes will be written to the corresponding entries of the 'newIndex' vector.

## Parameters

<i>i</i>	the index of the node specifying the connected component to examine.
<i>ic</i>	the label to give to the entries in the 'component'-vector that correspond with nodes in the specified connected component.
<i>index</i>	start value for local indexing of nodes within the component.

## Return values

<i>component</i>	this vector is where the labeling is written. Before calling the function, the user should ensure that its size is equal to the total number of nodes in the graph.
<i>newIndex</i>	this vector is where the new local coordinates are written. Before calling the function, the user should ensure that its size is equal to the total number of nodes in the graph.

**29.367.3.19** virtual void [PrOrganizedPoints::printInfo](#) ( std::ostream & *os* ) const [virtual]

Print general information about this [PrOrganizedPoints](#) object (number of nodes, etc.)

Reimplemented in [PrExplicitConnectivity](#).

**29.367.3.20** virtual void [PrOrganizedPoints::printUVEEdges](#) ( std::ostream & *os* ) const [virtual]

Print out the edges of the parametrization.

**29.367.3.21** virtual void [PrOrganizedPoints::printUVNodes](#) ( std::ostream & *os*, bool *num* = 0 ) const [virtual]

Print out the UV nodes of the graph. If num = 1, print first the number of nodes.

**29.367.3.22** virtual void [PrOrganizedPoints::printUVXYZNodes](#) ( std::ostream & *os*, bool *num* = 0 ) const [virtual]

Print out the nodes of the graph. If num = 1, print first the number of nodes.

29.367.3.23 `virtual void PrOrganizedPoints::printXYZEdges ( std::ostream & os ) const` [virtual]

Print out the edges of the graph.

29.367.3.24 `virtual void PrOrganizedPoints::printXYZNodes ( std::ostream & os, bool num = 0 ) const` [virtual]

Print out the XYZ nodes of the graph. If num = true, print first the number of nodes.

29.367.3.25 `virtual void PrOrganizedPoints::set3dNode ( int i, const Vector3D & p )` [pure virtual]

set the coordinates of the i-th node in the graph if the nodes are three-dimensional.

Implemented in [PrTriangulation\\_OP](#), [PrPlanarGraph\\_OP](#), [PrSubTriangulation](#), [PrFastUnorganized\\_OP](#), [PrRectangularGrid\\_OP](#), [PrLevelTriangulation\\_OP](#), and [PrUnorganized\\_OP](#).

29.367.3.26 `virtual void PrOrganizedPoints::setU ( int i, double u )` [pure virtual]

reset the u parameter value of the i-th node.

Implemented in [Go::ftPointSet](#), [PrPlanarGraph\\_OP](#), [PrTriangulation\\_OP](#), [PrSubTriangulation](#), [PrFastUnorganized\\_OP](#), [PrLevelTriangulation\\_OP](#), [PrUnorganized\\_OP](#), and [PrRectangularGrid\\_OP](#).

29.367.3.27 `virtual void PrOrganizedPoints::setV ( int i, double v )` [pure virtual]

reset the v parameter value of the i-th node.

Implemented in [Go::ftPointSet](#), [PrPlanarGraph\\_OP](#), [PrTriangulation\\_OP](#), [PrSubTriangulation](#), [PrLevelTriangulation\\_OP](#), [PrFastUnorganized\\_OP](#), [PrUnorganized\\_OP](#), and [PrRectangularGrid\\_OP](#).

29.367.3.28 `void PrOrganizedPoints::topologicalDistToBdy ( std::vector< int > & label ) const`

Compute for each node the number of edges that must be traversed in order to reach the boundary (For boundary edges, this number is 0).

Return values

<i>label</i>	this vector will contain the computed number for each node. Its size will therefore be equal to the number of nodes.
--------------	----------------------------------------------------------------------------------------------------------------------

The documentation for this class was generated from the following file:

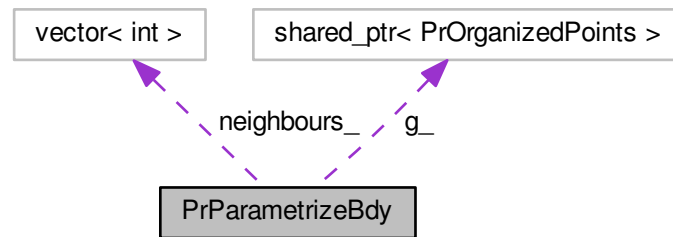
- parametrization/include/GoTools/parametrization/[PrOrganizedPoints.h](#)

## 29.368 PrParametrizeBdy Class Reference

```
#include <PrParametrizeBdy.h>
```



Collaboration diagram for PrParametrizeBdy:



## Public Member Functions

- [PrParametrizeBdy \(\)](#)  
*Default constructor.*
- virtual [~PrParametrizeBdy \(\)](#)  
*Empty default constructor.*
- virtual void [attach](#) (shared\_ptr< [PrOrganizedPoints](#) > graph)  
*Set the graph.*
- void [setParamKind](#) ([PrBdyParamKind](#) bdyparamtype=[PrCHORDLENGTHBDY](#))
- [double](#) [boundaryLength](#) (int i1, int i2)
- [bool](#) [parametrize](#) ()
- [bool](#) [parametrizeSide](#) (int i1, int i2)
- [bool](#) [parametrize](#) (int c1, int c2, int c3, int c4, [double](#) umin=0.0, [double](#) umax=1.0, [double](#) vmin=0.0, [double](#) vmax=1.0)
- void [findCornersFromXYZ](#) (int \*c)
- void [findCornersFromUV](#) (int \*c)
- int [findBdyNode](#) ()  
*Find the bounday node with the least index.*
- int [getNextBdyNode](#) (int i)

## Protected Member Functions

- [double](#) [chord](#) (const [Vector3D](#) &a, const [Vector3D](#) &b)  
*Return "length" of chord between two points in  $R^3$ .*

## Protected Attributes

- [PrBdyParamKind](#) bdyparamtype\_
- shared\_ptr< [PrOrganizedPoints](#) > g\_
- vector< int > neighbours\_

### 29.368.1 Detailed Description

This class implements an algorithm for parametrizing the boundary of a planar graph in  $R^3$ . Typical choices are uniform and chord length, either round the whole boundary or along each "side".

Definition at line 60 of file PrParametrizeBdy.h.

### 29.368.2 Constructor & Destructor Documentation

#### 29.368.2.1 PrParametrizeBdy::PrParametrizeBdy ( )

Default constructor.

#### 29.368.2.2 virtual PrParametrizeBdy::~PrParametrizeBdy ( ) [virtual]

Empty default constructor.

### 29.368.3 Member Function Documentation

#### 29.368.3.1 virtual void PrParametrizeBdy::attach ( shared\_ptr< PrOrganizedPoints > graph ) [virtual]

Set the graph.

#### 29.368.3.2 double PrParametrizeBdy::boundaryLength ( int *i1*, int *i2* )

Calculate the "length" of the section of the xyz boundary between nodes boundary *i* and *j* according to the kind of parametrization specified by [setParamKind\(\)](#). Choices are PrCHORDLENGTHBDY, PrCENTRIPETAL and PrUN↔IFBDY.

#### 29.368.3.3 double PrParametrizeBdy::chord ( const Vector3D & *a*, const Vector3D & *b* ) [protected]

Return "length" of chord between two points in  $R^3$ .

#### 29.368.3.4 int PrParametrizeBdy::findBdyNode ( )

Find the boundary node with the least index.

#### 29.368.3.5 void PrParametrizeBdy::findCornersFromUV ( int \* *c* )

Find the indices *c*[0],*c*[1],*c*[2],*c*[3] of the four vertices of the graph whose (*u*,*v*) points are the furthest SW, SE, NE, and NW in that order. The array *c* should be allocated outside with length 4.

29.368.3.6 `void PrParametrizeBdy::findCornersFromXYZ ( int * c )`

Find four corner nodes `c[0],c[1],c[2],c[3]` by taking `c[0]` to be an arbitrary boundary node and choosing `c[1],c[2],c[3]` so that the lengths of four boundary curves delimited by `c[0],c[1],c[2],c[3]` are as equal in "chord" length (defined by [setParamKind\(\)](#)) as possible. The array `c` should be allocated outside with length 4.

29.368.3.7 `int PrParametrizeBdy::getNextBdyNode ( int i )`

Given a boundary node `i`, return the next boundary node in an anticlockwise direction around the boundary.

29.368.3.8 `bool PrParametrizeBdy::parametrize ( )`

Parametrize the boundary of the planar graph. The parameter domain will be the unit circle.

29.368.3.9 `bool PrParametrizeBdy::parametrize ( int c1, int c2, int c3, int c4, double umin = 0.0, double umax = 1.0, double vmin = 0.0, double vmax = 1.0 )`

Parametrize the boundary of the given planar graph. The parameter domain will be the rectangle `[umin,umax]*[vmin,vmax]`. Node `c1` will be mapped to `(umin,vmin)`, `c2` to `(umax,vmin)`, `c3` to `(umax,vmax)` and `c4` to `(umin,vmax)`. The nodes `c1,c2,c3,c4` should be boundary nodes and should be in anticlockwise sequence around the boundary. The kind of parametrization along the four sides of the square will be specified by [setParamKind\(\)](#).

29.368.3.10 `bool PrParametrizeBdy::parametrizeSide ( int i1, int i2 )`

Parametrize the boundary nodes between two boundary nodes `i1,i2` of a given planar graph by chord length. The two nodes `i1` and `i2` are assumed to be have `u` and `v` values. The boundary parameter points between `i1` and `i2`, starting from `i1` in an anticlockwise direction around the boundary, will be placed along the straight line between the two parameter points.

29.368.3.11 `void PrParametrizeBdy::setParamKind ( PrBdyParamKind bdyparamtype = PrCHORDLENGTHBDY )`  
`[inline]`

Set type of parametrization to be used along the boundary. Unless you have a strong reason to do so you should set the parametrization to "PrCHORDLENGTHBDY". (Other choices are PrCENTRIPETAL and PrUNIFBDY).

Definition at line 88 of file PrParametrizeBdy.h.

## 29.368.4 Member Data Documentation

29.368.4.1 `PrBdyParamKind PrParametrizeBdy::bdyparamtype_` `[protected]`

Definition at line 64 of file PrParametrizeBdy.h.

29.368.4.2 `shared_ptr<PrOrganizedPoints> PrParametrizeBdy::g_` `[protected]`

Definition at line 65 of file PrParametrizeBdy.h.

29.368.4.3 `vector<int> PrParametrizeBdy::neighbours_` [protected]

Definition at line 66 of file `PrParametrizeBdy.h`.

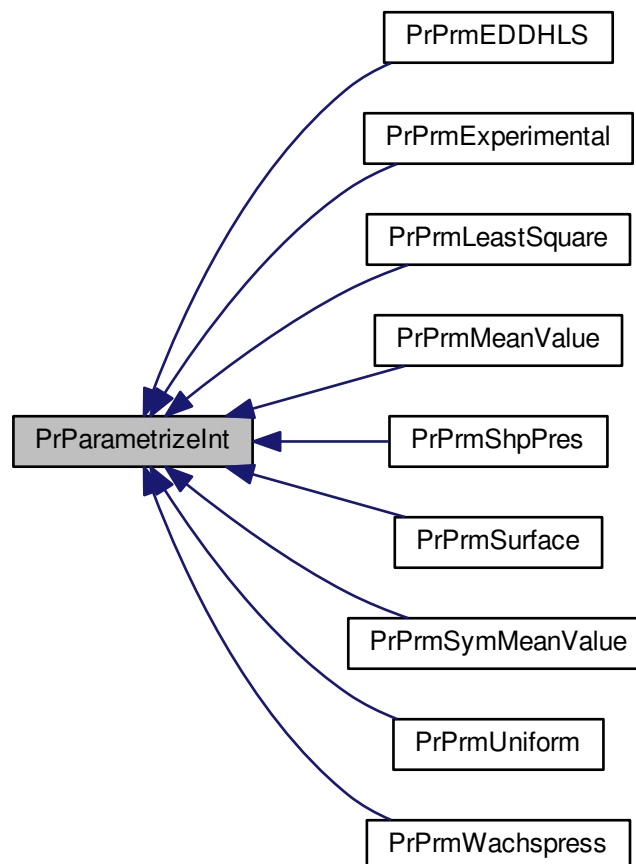
The documentation for this class was generated from the following file:

- [parametrization/include/GoTools/parametrization/PrParametrizeBdy.h](#)

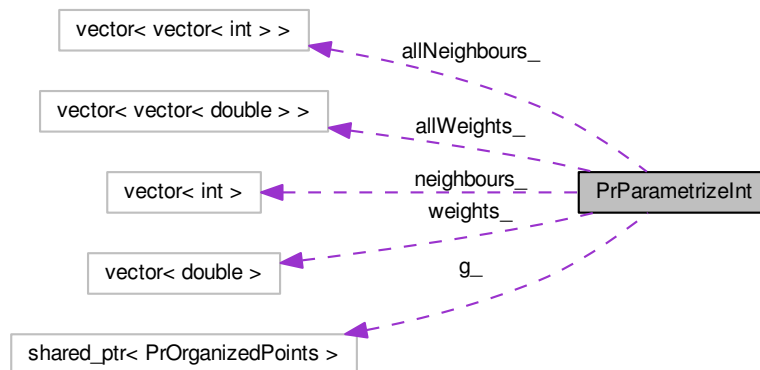
## 29.369 PrParametrizeInt Class Reference

```
#include <PrParametrizeInt.h>
```

Inheritance diagram for `PrParametrizeInt`:



Collaboration diagram for PrParametrizeInt:



## Public Member Functions

- [PrParametrizeInt](#) ()  
*Default constructor.*
- virtual [~PrParametrizeInt](#) ()  
*Empty default destructor.*
- void [attach](#) (shared\_ptr< [PrOrganizedPoints](#) > graph)  
*Set the graph.*
- void [setStartVectorKind](#) ([PrParamStartVector](#) svtype=[PrBARYCENTRE](#))
- void [setBiCGTolerance](#) (double tolerance=1.0e-6)  
*Set tolerance for Bi-CGSTAB.*
- bool [parametrize](#) ()  
*Parametrize the given planar graph.*
- bool [parametrize3d](#) (vector< int > &, vector< double > &)
- bool [new\\_parametrize3d](#) (vector< int > &, vector< double > &)
- void [findFixedPntsFromXYZ](#) (vector< int > &fixedPnts)
- void [smooth](#) (int nmb, vector< int > &fixedPnts)  
*performs "nmb" Gauss-Seidel smoothing steps on the sphere*
- void [computeWeights](#) ()  
*computes and stores the weights for each node*

## Protected Member Functions

- int [getNumIntNghrs](#) (int i)
- int [getNumInt2Nghrs](#) (int i, vector< int > &)
- void [findBarycentre](#) (double &ucentre, double &vcentre)
- bool [isFixed](#) (int, vector< int > &)
- virtual bool [makeWeights](#) (int i)=0
- const vector< vector< double > > & [getAllWeights](#) () const
- const vector< vector< int > > & [getAllNeighbours](#) () const

## Protected Attributes

- [double tolerance\\_](#)
- [PrParamStartVector startvectortype\\_](#)
- [shared\\_ptr< PrOrganizedPoints > g\\_](#)
- [vector< int > neighbours\\_](#)
- [vector< double > weights\\_](#)
- [vector< vector< double > > allWeights\\_](#)
- [vector< vector< int > > allNeighbours\\_](#)

### 29.369.1 Detailed Description

This class implements an algorithm for creating a parametrization in  $R^2$  of the interior of a given embedding of a planar graph in  $R^3$ . The method is described in the paper: M. S. Floater, "Parametrization and smooth approximation of surface triangulations", CAGD 14 (1997), 231-250. The method sets u and v values to each interior node of the graph. The class [PrParametrizeInt](#) is an abstract base class. One must call one of its derived classes in order to choose a particular method of parametrization. We recommend the shape preserving parametrization implemented in [PrPrmShpPres](#).

Definition at line 67 of file [PrParametrizeInt.h](#).

### 29.369.2 Constructor & Destructor Documentation

#### 29.369.2.1 PrParametrizeInt::PrParametrizeInt ( )

Default constructor.

#### 29.369.2.2 virtual PrParametrizeInt::~~PrParametrizeInt ( ) [virtual]

Empty default destructor.

### 29.369.3 Member Function Documentation

#### 29.369.3.1 void PrParametrizeInt::attach ( shared\_ptr< PrOrganizedPoints > graph )

Set the graph.

#### 29.369.3.2 void PrParametrizeInt::computeWeights ( )

computes and stores the weights for each node

#### 29.369.3.3 void PrParametrizeInt::findBarycentre ( double & ucentre, double & vcentre ) [protected]

#### 29.369.3.4 void PrParametrizeInt::findFixedPntsFromXYZ ( vector< int > & fixedPnts )

This is a simple routine which finds the indices `fixedPnts [0...3]` of the four vertices of the graph whose (x,y,z) points are the furthest in the directions (-1/-1/1), (1/1/1), (-1/1/-1), and (1/-1/-1) in that order. The indices can be used as fixed points for parametrising in 3D.

29.369.3.5 `const vector< vector<int> >& PrParametrizeInt::getAllNeighbours ( ) const` [inline],[protected]

Definition at line 96 of file PrParametrizeInt.h.

29.369.3.6 `const vector< vector<double> >& PrParametrizeInt::getAllWeights ( ) const` [inline],[protected]

Definition at line 92 of file PrParametrizeInt.h.

29.369.3.7 `int PrParametrizeInt::getNumInt2Nghrs ( int i, vector< int > & )` [protected]

29.369.3.8 `int PrParametrizeInt::getNumIntNghrs ( int i )` [protected]

29.369.3.9 `bool PrParametrizeInt::isFixed ( int , vector< int > & )` [protected]

29.369.3.10 `virtual bool PrParametrizeInt::makeWeights ( int i )` [protected],[pure virtual]

Implemented in [PrPrmMeanValue](#), [PrPrmShpPres](#), [PrPrmLeastSquare](#), [PrPrmSymMeanValue](#), [PrPrmEDDHLS](#), [PrPrmExperimental](#), and [PrPrmWachspres](#).

29.369.3.11 `bool PrParametrizeInt::new_parametrize3d ( vector< int > & , vector< double > & )`

Parametrize the nodes of the 3D graph `g_` except those with indices in "fixedPnts". The parameterization is done in 3D and the result is returned as vector "uvw"

29.369.3.12 `bool PrParametrizeInt::parametrize ( )`

Parametrize the given planar graph.

29.369.3.13 `bool PrParametrizeInt::parametrize3d ( vector< int > & , vector< double > & )`

Parametrize the nodes of the 3D graph `g_` except those with indices in "fixedPnts". The parameterization is done in 3D and the result is returned as vector "uvw"

29.369.3.14 `void PrParametrizeInt::setBiCGTolerance ( double tolerance = 1.0e-6 )` [inline]

Set tolerance for Bi-CGSTAB.

Definition at line 116 of file PrParametrizeInt.h.

29.369.3.15 `void PrParametrizeInt::setStartVectorKind ( PrParamStartVector svtype = PrBARYCENTRE )`  
[inline]

Choose how to initialize the solution vector when running the internal solver. Choices are PrBARYCENTRE or PrFROMUV.

Definition at line 112 of file PrParametrizeInt.h.

29.369.3.16 `void PrParametrizeInt::smooth ( int nmb, vector< int > & fixedPnts )`

performs "nmb" Gauss-Seidel smoothing steps on the sphere

#### 29.369.4 Member Data Documentation

29.369.4.1 `vector< vector<int> > PrParametrizeInt::allNeighbours_` [protected]

Definition at line 80 of file PrParametrizeInt.h.

29.369.4.2 `vector< vector<double> > PrParametrizeInt::allWeights_` [protected]

Definition at line 79 of file PrParametrizeInt.h.

29.369.4.3 `shared_ptr<PrOrganizedPoints> PrParametrizeInt::g_` [protected]

Definition at line 74 of file PrParametrizeInt.h.

29.369.4.4 `vector<int> PrParametrizeInt::neighbours_` [protected]

Definition at line 76 of file PrParametrizeInt.h.

29.369.4.5 `PrParamStartVector PrParametrizeInt::startvectortype_` [protected]

Definition at line 72 of file PrParametrizeInt.h.

29.369.4.6 `double PrParametrizeInt::tolerance_` [protected]

Definition at line 71 of file PrParametrizeInt.h.

29.369.4.7 `vector<double> PrParametrizeInt::weights_` [protected]

Definition at line 77 of file PrParametrizeInt.h.

The documentation for this class was generated from the following file:

- parametrization/include/GoTools/parametrization/[PrParametrizeInt.h](#)



## 29.370 PrParametrizeMesh Class Reference

```
#include <PrParametrizeMesh.h>
```

### Public Member Functions

- [PrParametrizeMesh](#) ()  
*Default constructor.*
- virtual [~PrParametrizeMesh](#) ()  
*Default destructor.*
- void [attach](#) ([PrTriangulation\\_OP](#) \*mesh, [PrTriangulation\\_OP](#) \*basemesh)
- [PrParamTriangulation](#) \* [parametrize](#) ()
- void [makeSubTriangulationFromTriangle](#) (int i, [PrSubTriangulation](#) &subtri, [std::vector](#)< int > &corners)
- [bool](#) [makePolygon](#) ([PrTriangulation\\_OP](#) &triangulation, [const](#) [std::vector](#)< int > &nodes, [std::vector](#)< int > &polygon)
- [bool](#) [makePath](#) ([PrTriangulation\\_OP](#) &triangulation, int n1, int n2, [std::vector](#)< int > &path)
- [bool](#) [makeSubTriangulation](#) ([PrTriangulation\\_OP](#) &triangulation, [const](#) [vector](#)< int > &nodes, [PrSubTriangulation](#) &subtriangulation)
- [bool](#) [makeSubTriangulationInsidePolygon](#) ([PrTriangulation\\_OP](#) &triangulation, [const](#) [std::vector](#)< int > &polygon, [PrSubTriangulation](#) &subtriangulation)
- [bool](#) [getConnectedNodes](#) ([PrTriangulation\\_OP](#) &triangulation, [const](#) [std::vector](#)< int > &polygon, [std::set](#)< int > &nodes)
- int [getLeftNode](#) ([PrTriangulation\\_OP](#) &triangulation, int n1, int n2)
- void [parametrizeSubTriangulation](#) ([shared\\_ptr](#)< [PrSubTriangulation](#) > sub\_tri, [std::vector](#)< int > &corners)
- void [getInteriorNeighbours](#) (int v, [const](#) [std::vector](#)< int > &neighbours, [const](#) [std::vector](#)< int > &boundary, [std::vector](#)< int > &new\_nbrs)
- [bool](#) [edgeInPolygon](#) (int n1, int n2, [const](#) [std::vector](#)< int > &polygon)
- int [castRay](#) ([PrTriangulation\\_OP](#) &triangulation, int vs, [double](#) angle, [double](#) dMax, [const](#) [std::set](#)< int > &T, [std::vector](#)< int > &t\_path, [std::vector](#)< [Vector3D](#) > &v\_path)
- [bool](#) [getNextV](#) ([const](#) [Vector3D](#) &p1, [const](#) [Vector3D](#) &p2, [const](#) [Vector3D](#) &p3, [const](#) [Vector3D](#) &p4, [double](#) a, [double](#) b, [bool](#) f, [double](#) &c)

### Static Public Member Functions

- static [double](#) [getIsoline](#) ([const](#) [PrTriangulation\\_OP](#) &triangulation, int vs, int vd, [std::vector](#)< int > &t\_path, [std::vector](#)< [Vector3D](#) > &v\_path)
- static void [convertAngle](#) ([const](#) [PrTriangulation\\_OP](#) &triangulation, [double](#) theta, int v, [double](#) &bc, int &t)

#### 29.370.1 Detailed Description

[PrParametrizeMesh](#) - Short description. Detailed description.

Definition at line 58 of file PrParametrizeMesh.h.

#### 29.370.2 Constructor & Destructor Documentation

##### 29.370.2.1 PrParametrizeMesh::PrParametrizeMesh ( )

Default constructor.

29.370.2.2 `virtual PrParametrizeMesh::~~PrParametrizeMesh ( ) [virtual]`

Default destructor.

### 29.370.3 Member Function Documentation

29.370.3.1 `void PrParametrizeMesh::attach ( PrTriangulation_OP * mesh, PrTriangulation_OP * basemesh ) [inline]`

Definition at line 76 of file PrParametrizeMesh.h.

29.370.3.2 `int PrParametrizeMesh::castRay ( PrTriangulation_OP & triangulation, int vs, double angle, double dMax, const std::set< int > & T, std::vector< int > & t_path, std::vector< Vector3D > & v_path )`

Casts a ray from the vertex "vs" into the direction of "angle". angle = 0 is the direction to the ccw node in vs's leading triangle (remark: that is also the first node in the result of the "getNeighbours" routine). The ray is cast as long as its length does not exceed "dMax" and no triangle in "T" is visited. The triangles visited are stored in "t\_path". If the path does not hit a triangle in "T", the return value is -1, otherwise it is the triangle index "v\_path" contains the vertices of the path.

29.370.3.3 `static void PrParametrizeMesh::convertAngle ( const PrTriangulation_OP & triangulation, double theta, int v, double & bc, int & t ) [static]`

Converts the direction indicated by "theta" around the node "v" into a (double,int) pair. The "int" value is the index of the triangle around "v" in direction "theta", the "double" value is the barycentric coordinate of the intersection point of a ray in direction "theta" with the edge opposite to "v" in the triangle w.r.t to the vertices of that edge.

29.370.3.4 `bool PrParametrizeMesh::edgeInPolygon ( int n1, int n2, const std::vector< int > & polygon )`

29.370.3.5 `bool PrParametrizeMesh::getConnectedNodes ( PrTriangulation_OP & triangulation, const std::vector< int > & polygon, std::set< int > & nodes )`

Assume the vertices in polygon is a closed oriented polygon on triangulation. Returns the indices of the vertices interior to polygon. If the orientation is reversed it returns the exterior. If the polygon is open and does not split triangulation all nodes will be found

29.370.3.6 `void PrParametrizeMesh::getInteriorNeighbours ( int v, const std::vector< int > & neighbours, const std::vector< int > & boundary, std::vector< int > & new_nbrs )`

29.370.3.7 `static double PrParametrizeMesh::getisoline ( const PrTriangulation_OP & triangulation, int vs, int vd, std::vector< int > & t_path, std::vector< Vector3D > & v_path ) [static]`

runs the [Dijkstra](#) algorithm from source point "vs" to destination point "vd" and determines the iso-distance-line of vd. The vertices of that isoline are stored in "v\_path", whereas the indices of the triangles visited by that line are stored in "t\_path". Return value is the distance from "vs" to "vd".

29.370.3.8 `int PrParametrizeMesh::getLeftNode ( PrTriangulation_OP & triangulation, int n1, int n2 )`

Find the third node of the triangle with nodes n1, n2 in anti-clockwise order (i.e. the triangle to the "left")

29.370.3.9 `bool PrParametrizeMesh::getNextV ( const Vector3D & p1, const Vector3D & p2, const Vector3D & p3, const Vector3D & p4, double a, double b, bool f, double & c )`

Determine barycentric coordinates (r,s,t) of p4 w.r.t (p1,p2,p3) after flattening

29.370.3.10 `bool PrParametrizeMesh::makePath ( PrTriangulation_OP & triangulation, int n1, int n2, std::vector< int > & path )`

29.370.3.11 `bool PrParametrizeMesh::makePolygon ( PrTriangulation_OP & triangulation, const std::vector< int > & nodes, std::vector< int > & polygon )`

Given a "triangulation" and a (ordered) set of "nodes", this routine returns a "polygon", consisting of those nodes (ordered) who form the shortest paths between the given nodes in the triangulation

i.e.: given a triangle (or arbitrary polygon) in the coarse mesh, this algorithm finds the corresponding triangle (polygon) in the original (fine) mesh

29.370.3.12 `bool PrParametrizeMesh::makeSubTriangulation ( PrTriangulation_OP & triangulation, const vector< int > & nodes, PrSubTriangulation & subtriangulation )`

Make a subtriangulation of triang along the polygon induced by the shortest paths between the nodes in nodes. Use the same nodeset ?

29.370.3.13 `void PrParametrizeMesh::makeSubTriangulationFromTriangle ( int i, PrSubTriangulation & subtri, std::vector< int > & corners )`

29.370.3.14 `bool PrParametrizeMesh::makeSubTriangulationInsidePolygon ( PrTriangulation_OP & triangulation, const std::vector< int > & polygon, PrSubTriangulation & subtriangulation )`

Make a subtriangulation of triang inside polygon. Assume polygon to be a closed ring of neighbouring nodes

29.370.3.15 `PrParamTriangulation* PrParametrizeMesh::parametrize ( )`

29.370.3.16 `void PrParametrizeMesh::parametrizeSubTriangulation ( shared_ptr< PrSubTriangulation > sub_tri, std::vector< int > & corners )`

The documentation for this class was generated from the following file:

- parametrization/include/GoTools/parametrization/PrParametrizeMesh.h

## 29.371 PrParamTriangulation Class Reference

```
#include <PrParamTriangulation.h>
```

### Public Member Functions

- [PrParamTriangulation](#) ()  
*Empty default constructor.*
- virtual [~PrParamTriangulation](#) ()  
*Empty default destructor.*
- void [open](#) (istream &is)  
*Read from stream.*
- void [save](#) (ostream &os)  
*Write to stream.*
- [PrTriangulation\\_OP](#) \* [getFineMesh](#) ()  
*Get a pointer to the fine mesh.*
- [PrTriangulation\\_OP](#) \* [getCoarseMesh](#) ()  
*Get a pointer to the coarse mesh.*
- void [attach](#) ([PrTriangulation\\_OP](#) \*mesh, [PrTriangulation\\_OP](#) \*basemesh)
- void [makeCorrespondences](#) (int idx, [PrSubTriangulation](#) &sub\_tri)
- void [printBaseTriangles](#) ()  
*(Print out debug information)*
- Vector3D [getParamPoint](#) (int i)
- void [findTriangleContainingPoint](#) (int coarseT, const Vector3D &coarseBC, int &fineT, Vector3D &fineBC)
- void [getUV](#) (int node, int tri, double &u, double &v)
- bool [triangleContainsPoint](#) (int fineTri, int coarseTri, const Vector3D &coarseBC, Vector3D &fineBC)
- Vector3D [getSurfPoint](#) (int coarseTri, const Vector3D &coarseBC)

### Static Public Member Functions

- static Vector3D [evaluator](#) ([PrTriangulation\\_OP](#) &mesh, int idx, Vector3D &bc)

#### 29.371.1 Detailed Description

This class represents a fine triangulation embedded in a coarser one. The points on the faces on the coarser triangulation can be seen as parametrizing the points on the finer triangulation.

Definition at line 54 of file PrParamTriangulation.h.

#### 29.371.2 Constructor & Destructor Documentation

##### 29.371.2.1 PrParamTriangulation::PrParamTriangulation ( ) [inline]

Empty default constructor.

Definition at line 63 of file PrParamTriangulation.h.

29.371.2.2 `virtual PrParamTriangulation::~PrParamTriangulation ( ) [inline],[virtual]`

Empty default destructor.

Definition at line 66 of file PrParamTriangulation.h.

### 29.371.3 Member Function Documentation

29.371.3.1 `void PrParamTriangulation::attach ( PrTriangulation_OP * mesh, PrTriangulation_OP * basemesh )`

Reset this objects to handle the fine/coarse triangulation pair represented by 'mesh' and 'basemesh'.

29.371.3.2 `static Vector3D PrParamTriangulation::evaluator ( PrTriangulation_OP & mesh, int idx, Vector3D & bc ) [static]`

Evaluate a point on a given mesh by specifying a triangle on the mesh and the barycentric coordinates within this triangle. The point is computed by linear interpolation of the triangle's corner nodes.

29.371.3.3 `void PrParamTriangulation::findTriangleContainingPoint ( int coarseT, const Vector3D & coarseBC, int & fineT, Vector3D & fineBC )`

Given a triangle on the coarse mesh and the barycentric coordinates of a point in this triangle, find the triangle on the fine mesh that contains this point.

#### Parameters

<i>coarseT</i>	index of the triangle on the coarse mesh
<i>coarseBC</i>	barycentric coordinates of the point with respect to the triangle on the coarse mesh

#### Return values

<i>fineT</i>	gives the index to the triangle on the fine mesh containing the specified point
<i>fineBC</i>	gives the barycentric coordinates of the point with respect to the triangle on the fine mesh.

29.371.3.4 `PrTriangulation_OP* PrParamTriangulation::getCoarseMesh ( ) [inline]`

Get a pointer to the coarse mesh.

Definition at line 78 of file PrParamTriangulation.h.

29.371.3.5 `PrTriangulation_OP* PrParamTriangulation::getFineMesh ( ) [inline]`

Get a pointer to the fine mesh.

Definition at line 75 of file PrParamTriangulation.h.

29.371.3.6 Vector3D PrParamTriangulation::getParamPoint ( int *i* )

Get the 'parameter point' of a certain node on the fine mesh. By this we mean to find its "corresponding" point on the coarse mesh.

## Parameters

<i>i</i>	index to the node on the fine mesh
----------	------------------------------------

## Returns

a 3D point on the coarse mesh that corresponds to this node

29.371.3.7 Vector3D PrParamTriangulation::getSurfPoint ( int *coarseTri*, const Vector3D & *coarseBC* )

Given a base triangle (triangle on the coarse mesh) and barycentric coordinates with respect to this triangle, compute the corresponding point on the fine mesh.

## Parameters

<i>coarseTri</i>	index of the base triangle in question
<i>coarseBC</i>	the barycentric coordinates

## Returns

a 3D point on the fine mesh corresponding to (parameterized by) the point on the coarse mesh specified by 'coarseTri' and 'coarseBC'.

29.371.3.8 void PrParamTriangulation::getUV ( int *node*, int *tri*, double & *u*, double & *v* )

Returns the barycentric coordinates of node with respect to a triangle on the fine mesh. Usually, these are the uv-values obtained by the parameterization, but the handling becomes more complicated on edges.

## Parameters

<i>node</i>	index to a node on the fine mesh, for which we want to determine the barycentric coordinates
<i>tri</i>	index to a base triangle (ie. triangle on the coarse mesh), for with we want to find the barycentric coordinates of the node given by 'node'.

## Return values

<i>u</i>	the u-coordinate of the node
<i>v</i>	the v-coordinate of the node (the w-coordinate is given by 1-u-v).

29.371.3.9 void PrParamTriangulation::makeCorrespondences ( int *idx*, PrSubTriangulation & *sub\_tri* )

Set all triangles contained in a sub-triangulation of the fine mesh to "correspond" to a certain triangle in the coarse mesh.

#### Parameters

<i>idx</i>	index to the triangle in the coarse mesh
<i>sub_↔_tri</i>	a sub-triangulation that represents those triangles in the fine mesh that are to be associated with the 'idx' triangle in the coarse mesh.

29.371.3.10 void PrParamTriangulation::open ( istream & *is* )

Read from stream.

29.371.3.11 void PrParamTriangulation::printBaseTriangles ( )

(Print out debug information)

29.371.3.12 void PrParamTriangulation::save ( ostream & *os* )

Write to stream.

29.371.3.13 bool PrParamTriangulation::triangleContainsPoint ( int *fineTri*, int *coarseTri*, const Vector3D & *coarseBC*, Vector3D & *fineBC* )

Determine whether a given triangle on the fine mesh contains a point specified by its barycentric coordinates on the underlying coarse triangle.

#### Parameters

<i>fineTri</i>	index of the triangle on the fine mesh
<i>coarseTri</i>	index of the triangle on the coarse mesh
<i>coarseBC</i>	barycentric coordinates of the point on the triangle in the coarse mesh

#### Return values

<i>fineBC</i>	if the triangle on the fine mesh was found to contain the point in question, its barycentric coordinates with respect to this triangle will be returned here
---------------	--------------------------------------------------------------------------------------------------------------------------------------------------------------

#### Returns

'true' if the specified point is contained n the triangle on the fine mesh, 'false' otherwise.

The documentation for this class was generated from the following file:

- parametrization/include/GoTools/parametrization/PrParamTriangulation.h

## 29.372 PrPGNode Class Reference

```
#include <PrPGNode.h>
```

### Public Member Functions

- [PrPGNode \(\)](#)  
*Empty default constructor.*
- [PrPGNode \(double x, double y, double z, double u, double v, int end\)](#)  
*Constructor.*
- [PrPGNode \(const PrPGNode &p\)](#)  
*Copy constructor.*
- [~PrPGNode \(\)](#)  
*Empty destructor.*
- [void init \(double x, double y, double z, double u, double v, int end\)](#)
- [const Vector3D & pnt \(\) const](#)
- [const double & x \(\) const](#)
- [const double & y \(\) const](#)
- [const double & z \(\) const](#)
- [const double & u \(\) const](#)
- [const double & v \(\) const](#)
- [const int & end \(\) const](#)
- [Vector3D & pnt \(\)](#)
- [double & x \(\)](#)
- [double & y \(\)](#)
- [double & z \(\)](#)
- [double & u \(\)](#)
- [double & v \(\)](#)
- [int & end \(\)](#)
- [void print \(std::ostream &os\)](#)
- [void printXYZ \(std::ostream &os\)](#)
- [void printUV \(std::ostream &os\)](#)
- [void scan \(std::istream &is\)](#)

### 29.372.1 Detailed Description

[PrPGNode](#) - This class represents a node for use in [PrPlanarGraph\\_OP](#). It has double x,y,z coordinate values. It also has double u and v values (parameter values) and the index of its last neighbour in the `adj_` array of [PrPlanarGraph\\_OP](#).

Definition at line 52 of file `PrPGNode.h`.

### 29.372.2 Constructor & Destructor Documentation

#### 29.372.2.1 [PrPGNode::PrPGNode \( \)](#) [`inline`]

Empty default constructor.

Definition at line 61 of file `PrPGNode.h`.



29.372.2.2 PrPGNode::PrPGNode ( double *x*, double *y*, double *z*, double *u*, double *v*, int *end* ) [inline]

Constructor.

Definition at line 63 of file PrPGNode.h.

29.372.2.3 PrPGNode::PrPGNode ( const PrPGNode & *p* ) [inline]

Copy constructor.

Definition at line 66 of file PrPGNode.h.

29.372.2.4 PrPGNode::~PrPGNode ( ) [inline]

Empty destructor.

Definition at line 69 of file PrPGNode.h.

### 29.372.3 Member Function Documentation

29.372.3.1 const int & PrPGNode::end ( ) const [inline]

Definition at line 162 of file PrPGNode.h.

29.372.3.2 int & PrPGNode::end ( ) [inline]

Definition at line 214 of file PrPGNode.h.

29.372.3.3 void PrPGNode::init ( double *x*, double *y*, double *z*, double *u*, double *v*, int *end* ) [inline]

Definition at line 103 of file PrPGNode.h.

29.372.3.4 const Vector3D & PrPGNode::pnt ( ) const [inline]

Definition at line 117 of file PrPGNode.h.

29.372.3.5 Vector3D & PrPGNode::pnt ( ) [inline]

Definition at line 169 of file PrPGNode.h.

29.372.3.6 void PrPGNode::print ( std::ostream & os )

29.372.3.7 void PrPGNode::printUV ( std::ostream & os )

29.372.3.8 void PrPGNode::printXYZ ( std::ostream & os )

29.372.3.9 void PrPGNode::scan ( std::istream & is )

29.372.3.10 const double & PrPGNode::u ( ) const [inline]

Definition at line 148 of file PrPGNode.h.

29.372.3.11 double & PrPGNode::u ( ) [inline]

Definition at line 200 of file PrPGNode.h.

29.372.3.12 const double & PrPGNode::v ( ) const [inline]

Definition at line 155 of file PrPGNode.h.

29.372.3.13 double & PrPGNode::v ( ) [inline]

Definition at line 207 of file PrPGNode.h.

29.372.3.14 const double & PrPGNode::x ( ) const [inline]

Definition at line 124 of file PrPGNode.h.

29.372.3.15 double & PrPGNode::x ( ) [inline]

Definition at line 176 of file PrPGNode.h.

29.372.3.16 const double & PrPGNode::y ( ) const [inline]

Definition at line 132 of file PrPGNode.h.

29.372.3.17 double & PrPGNode::y ( ) [inline]

Definition at line 184 of file PrPGNode.h.

29.372.3.18 `const double & PrPGNode::z( ) const` [inline]

Definition at line 140 of file PrPGNode.h.

29.372.3.19 `double & PrPGNode::z( )` [inline]

Definition at line 192 of file PrPGNode.h.

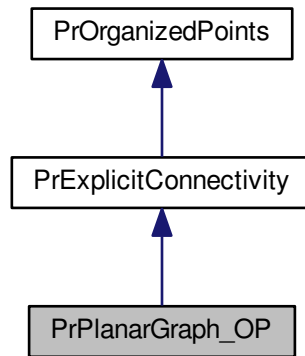
The documentation for this class was generated from the following file:

- parametrization/include/GoTools/parametrization/[PrPGNode.h](#)

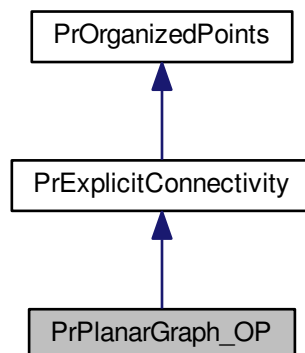
## 29.373 PrPlanarGraph\_OP Class Reference

```
#include <PrPlanarGraph_OP.h>
```

Inheritance diagram for PrPlanarGraph\_OP:



Collaboration diagram for PrPlanarGraph\_OP:



## Public Member Functions

- [PrPlanarGraph\\_OP](#) ()  
*Empty default constructor.*
- [PrPlanarGraph\\_OP](#) (int npts, [double](#) \*xyz\_nodes, [double](#) \*uv\_nodes, int \*end, int \*adj)
- [PrPlanarGraph\\_OP](#) (int npts, [double](#) \*xyz\_nodes, int \*end, int \*adj)
- [PrPlanarGraph\\_OP](#) ([PrOrganizedPoints](#) &op)  
*Constructor. Construct the [PrPlanarGraph\\_OP](#) from a [PrOrganizedPoints](#) class.*
- [~PrPlanarGraph\\_OP](#) ()  
*Empty destructor.*
- virtual int [getNumNodes](#) () [const](#)  
*Return the number of nodes in the graph.*
- virtual [Vector3D](#) [get3dNode](#) (int i) [const](#)  
*Return the i-th node in the graph.*
- virtual void [set3dNode](#) (int i, [const](#) [Vector3D](#) &p)  
*Doesn't really do anything. But needed anyway.*
- virtual void [getNeighbours](#) (int i, [vector](#)< int > &neighbours) [const](#)
- virtual [bool](#) [isBoundary](#) (int i) [const](#)  
*Is i a boundary node or interior node?*
- virtual [double](#) [getU](#) (int i) [const](#)  
*Return the u parameter value of the i-th node.*
- virtual [double](#) [getV](#) (int i) [const](#)  
*Return the v parameter value of the i-th node.*
- virtual void [setU](#) (int i, [double](#) u)  
*Reset the u parameter value of the i-th node.*
- virtual void [setV](#) (int i, [double](#) v)  
*Reset the v parameter value of the i-th node.*
- void [print](#) (std::ostream &os)
- void [scan](#) (std::istream &is)
- void [scan2](#) (std::istream &is)  
*alternative scan, reading the end array after points*

## Additional Inherited Members

### 29.373.1 Detailed Description

[PrPlanarGraph\\_OP](#) - This class represents a planar graph of points in  $R^3$  with or without parameter points in  $R^2$ . It implements the virtual functions in [PrOrganizedPoints](#). This kind of planar graph can be parametrized by [Pr↔Parametrize](#).

Definition at line 53 of file [PrPlanarGraph\\_OP.h](#).

### 29.373.2 Constructor & Destructor Documentation

#### 29.373.2.1 [PrPlanarGraph\\_OP::PrPlanarGraph\\_OP](#) ( ) [[inline](#)]

Empty default constructor.

Definition at line 64 of file [PrPlanarGraph\\_OP.h](#).

29.373.2.2 `PrPlanarGraph_OP::PrPlanarGraph_OP ( int npts, double * xyz_nodes, double * uv_nodes, int * end, int * adj )`

Constructor. Construct the [PrPlanarGraph\\_OP](#) from an array of nodes and an array of adjacency lists. The length of the arrays `xyz_nodes` and `uv_nodes` should be  $3*npts$  and  $2*npts$  respectively. The length of the array `end` should be `npts`. The length of the array `adj` should be `end[npts]`. If node 1 is an interior node, its neighbours are `adj[1],...,adj[end[1]]` in any anticlockwise sequence. If node 1 is a boundary node, its neighbours are `adj[1],...,adj[end[1]-1]` in the unique anticlockwise sequence. and `adj[end[1]] = 0` to indicate that node 1 is on the boundary. For  $i=2, \dots, n$ : If node  $i$  is an interior node, its neighbours are `adj[end[i-1]],...,adj[end[i]]` in any anticlockwise sequence. If node  $i$  is a boundary node, its neighbours are `adj[end[i-1]],...,adj[end[i]-1]` in the unique anticlockwise sequence. and `adj[end[i]] = 0` to indicate that node  $i$  is on the boundary. This is the data structure proposed by Cline and Renka.

29.373.2.3 `PrPlanarGraph_OP::PrPlanarGraph_OP ( int npts, double * xyz_nodes, int * end, int * adj )`

Constructor. Construct the [PrPlanarGraph\\_OP](#) from an array of nodes and an array of adjacency lists. Like the previous constructor, only we set the `uv`'s to zero.

29.373.2.4 `PrPlanarGraph_OP::PrPlanarGraph_OP ( PrOrganizedPoints & op )`

Constructor. Construct the [PrPlanarGraph\\_OP](#) from a [PrOrganizedPoints](#) class.

29.373.2.5 `PrPlanarGraph_OP::~~PrPlanarGraph_OP ( ) [inline]`

Empty destructor.

Definition at line 100 of file `PrPlanarGraph_OP.h`.

### 29.373.3 Member Function Documentation

29.373.3.1 `virtual Vector3D PrPlanarGraph_OP::get3dNode ( int i ) const [inline],[virtual]`

Return the  $i$ -th node in the graph.

Implements [PrOrganizedPoints](#).

Definition at line 108 of file `PrPlanarGraph_OP.h`.

29.373.3.2 `virtual void PrPlanarGraph_OP::getNeighbours ( int i, vector< int > & neighbours ) const [virtual]`

Return the indices of the neighbours of the  $i$ -th node in:

1. any anticlockwise order if  $i$  is an interior node
2. the unique anticlockwise order if  $i$  is a boundary node.

29.373.3.3 `virtual int PrPlanarGraph_OP::getNumNodes ( ) const [inline],[virtual]`

Return the number of nodes in the graph.

Implements [PrOrganizedPoints](#).

Definition at line 105 of file PrPlanarGraph\_OP.h.

29.373.3.4 `virtual double PrPlanarGraph_OP::getU ( int i ) const [inline],[virtual]`

Return the u parameter value of the i-th node.

Implements [PrOrganizedPoints](#).

Definition at line 123 of file PrPlanarGraph\_OP.h.

29.373.3.5 `virtual double PrPlanarGraph_OP::getV ( int i ) const [inline],[virtual]`

Return the v parameter value of the i-th node.

Implements [PrOrganizedPoints](#).

Definition at line 126 of file PrPlanarGraph\_OP.h.

29.373.3.6 `virtual bool PrPlanarGraph_OP::isBoundary ( int i ) const [virtual]`

Is i a boundary node or interior node?

Implements [PrOrganizedPoints](#).

29.373.3.7 `void PrPlanarGraph_OP::print ( std::ostream & os )`

29.373.3.8 `void PrPlanarGraph_OP::scan ( std::istream & is )`

29.373.3.9 `void PrPlanarGraph_OP::scan2 ( std::istream & is )`

alternative scan, reading the end array after points

29.373.3.10 `virtual void PrPlanarGraph_OP::set3dNode ( int i, const Vector3D & p ) [inline],[virtual]`

Doesn't really do anything. But needed anyway.

Implements [PrOrganizedPoints](#).

Definition at line 111 of file PrPlanarGraph\_OP.h.

29.373.3.11 `virtual void PrPlanarGraph_OP::setU ( int i, double u ) [inline],[virtual]`

Reset the *u* parameter value of the *i*-th node.

Implements [PrOrganizedPoints](#).

Definition at line 129 of file PrPlanarGraph\_OP.h.

29.373.3.12 `virtual void PrPlanarGraph_OP::setV ( int i, double v ) [inline],[virtual]`

Reset the *v* parameter value of the *i*-th node.

Implements [PrOrganizedPoints](#).

Definition at line 132 of file PrPlanarGraph\_OP.h.

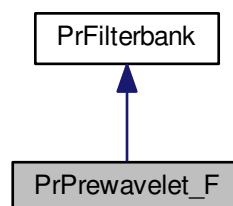
The documentation for this class was generated from the following file:

- parametrization/include/GoTools/parametrization/[PrPlanarGraph\\_OP.h](#)

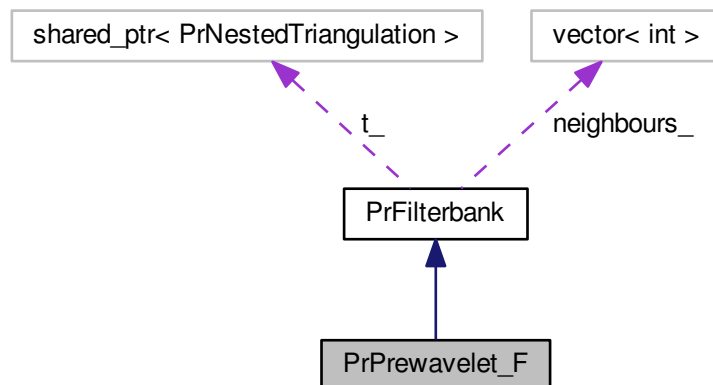
## 29.374 PrPrewavelet\_F Class Reference

```
#include <PrPrewavelet_F.h>
```

Inheritance diagram for PrPrewavelet\_F:



Collaboration diagram for PrPrewavelet\_F:



## Public Member Functions

- [PrPrewavelet\\_F](#) ()  
*Default constructor.*
- virtual [~PrPrewavelet\\_F](#) ()  
*Empty destructor.*
- void [setCGTolerance](#) (double tolerance=1.0e-6)
- virtual void [decompose](#) (int jlev, int dim=1)
- virtual void [compose](#) (int jlev, int dim=1)

## Additional Inherited Members

### 29.374.1 Detailed Description

PrPrewavelet \_ This class implements algorithms for decomposing and decomposing piecewise linear functions over nested triangulations. The complemenet spaces are taken here to be wavelet spaces with the prewavelet basis constructed in

M. S. Floater and E. G. Quak, Piecewise Linear Prewavelets on Arbitrary Triangulations, Numer. Math. **82** (1999), 221–252.

Definition at line 60 of file PrPrewavelet\_F.h.

### 29.374.2 Constructor & Destructor Documentation

#### 29.374.2.1 PrPrewavelet\_F::PrPrewavelet\_F ( ) [inline]

Default constructor.

Definition at line 77 of file PrPrewavelet\_F.h.



29.374.2.2 virtual PrPrewavelet\_F::~~PrPrewavelet\_F ( ) [virtual]

Empty destructor.

### 29.374.3 Member Function Documentation

29.374.3.1 virtual void PrPrewavelet\_F::compose ( int *jlev*, int *dim* = 1 ) [virtual]

The nodes in the triangulation belonging to  $V^j$  form a piecewise linear function  $f^j$  in the space  $S^j$ . This routine composes  $f^j$  from  $f^{j-1} + g^{j-1}$  where  $f^{j-1} \in S^{j-1}$  and  $g^{j-1} \in W^{j-1}$ . Use  $dim = 1$  for explicit triangulations and  $dim = 3$  for non-explicit. Default is  $dim = 1$ .

Implements [PrFilterbank](#).

29.374.3.2 virtual void PrPrewavelet\_F::decompose ( int *jlev*, int *dim* = 1 ) [virtual]

The nodes in the triangulation belonging to  $V^j$  form a piecewise linear function  $f^j$  in the space  $S^j$  (where  $j = \text{level}$ ). This routine decomposes  $f^j$  into  $f^{j-1} + g^{j-1}$  where  $f^{j-1} \in S^{j-1}$  and  $g^{j-1} \in W^{j-1}$ . Use  $dim = 1$  for explicit triangulations and  $dim = 3$  for non-explicit. Default is  $dim = 1$ .

Implements [PrFilterbank](#).

29.374.3.3 void PrPrewavelet\_F::setCGTolerance ( double *tolerance* = 1.0e-6 ) [inline]

Set the numerical tolerance used for internal computations with the conjugated gradient algorithm.

Definition at line 83 of file PrPrewavelet\_F.h.

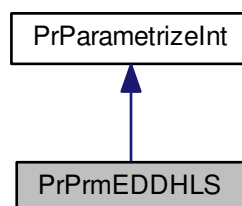
The documentation for this class was generated from the following file:

- parametrization/include/GoTools/parametrization/PrPrewavelet\_F.h

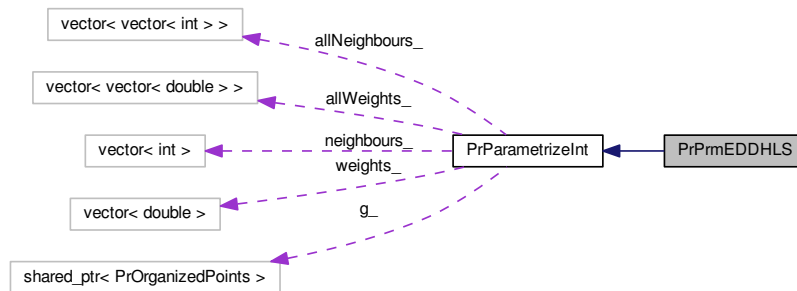
## 29.375 PrPrmEDDHLS Class Reference

```
#include <PrPrmEDDHLS.h>
```

Inheritance diagram for PrPrmEDDHLS:



Collaboration diagram for PrPrmEDDHLS:



## Public Member Functions

- [PrPrmEDDHLS \(\)](#)  
*Default constructor.*
- [virtual ~PrPrmEDDHLS \(\)](#)  
*Empty destructor.*

## Protected Member Functions

- [virtual bool makeWeights \(int i\)](#)

## Additional Inherited Members

### 29.375.1 Detailed Description

[PrPrmEDDHLS](#) - Implement the EDDHLS parametrization by implementing the virtual function `makeWeights`

Definition at line 50 of file `PrPrmEDDHLS.h`.

### 29.375.2 Constructor & Destructor Documentation

#### 29.375.2.1 `PrPrmEDDHLS::PrPrmEDDHLS ( )`

Default constructor.

#### 29.375.2.2 `virtual PrPrmEDDHLS::~~PrPrmEDDHLS ( ) [virtual]`

Empty destructor.

### 29.375.3 Member Function Documentation

29.375.3.1 `virtual bool PrPrmEDDHLS::makeWeights ( int i )` [protected],[virtual]

Implements [PrParametrizeInt](#).

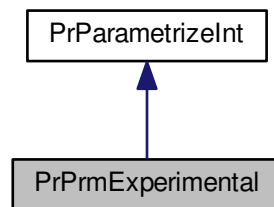
The documentation for this class was generated from the following file:

- parametrization/include/GoTools/parametrization/[PrPrmEDDHLS.h](#)

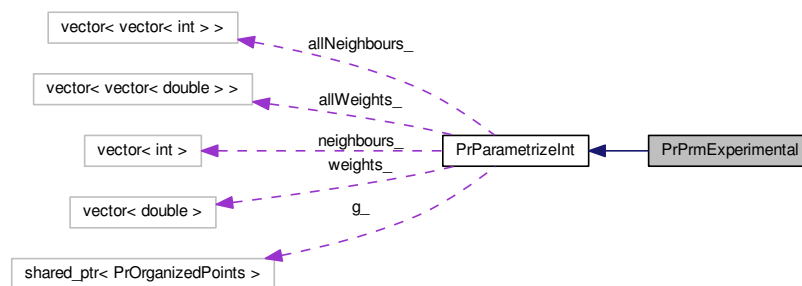
## 29.376 PrPrmExperimental Class Reference

```
#include <PrPrmExperimental.h>
```

Inheritance diagram for PrPrmExperimental:



Collaboration diagram for PrPrmExperimental:



### Public Member Functions

- [PrPrmExperimental](#) ()  
*Default constructor.*
- `virtual ~PrPrmExperimental` ()  
*Empty destructor.*

## Protected Member Functions

- virtual [bool makeWeights](#) (int i)

## Additional Inherited Members

### 29.376.1 Detailed Description

[PrPmExperimental](#) - Implement the experimental parametrization by implementing the virtual function [makeWeights](#)

Definition at line 48 of file [PrPmExperimental.h](#).

### 29.376.2 Constructor & Destructor Documentation

#### 29.376.2.1 [PrPmExperimental::PrPmExperimental](#) ( )

Default constructor.

#### 29.376.2.2 [virtual PrPmExperimental::~~PrPmExperimental](#) ( ) [virtual]

Empty destructor.

### 29.376.3 Member Function Documentation

#### 29.376.3.1 [virtual bool PrPmExperimental::makeWeights](#) ( int *i* ) [protected],[virtual]

Implements [PrParametrizeInt](#).

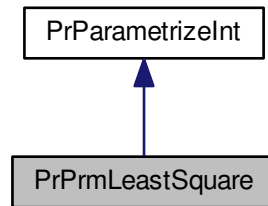
The documentation for this class was generated from the following file:

- parametrization/include/GoTools/parametrization/[PrPmExperimental.h](#)

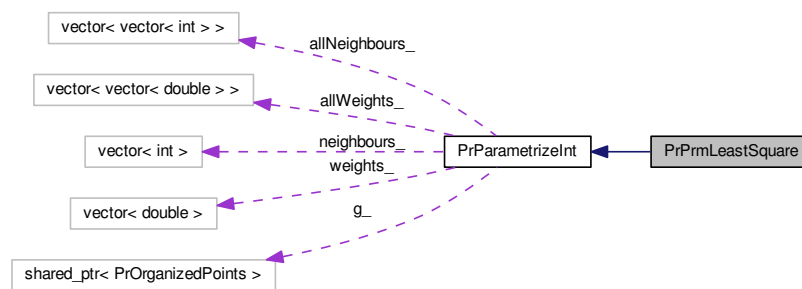
## 29.377 PrPrmLeastSquare Class Reference

```
#include <PrPrmLeastSquare.h>
```

Inheritance diagram for PrPrmLeastSquare:



Collaboration diagram for PrPrmLeastSquare:



### Public Member Functions

- [PrPrmLeastSquare](#) ()  
*Default constructor.*
- virtual [~PrPrmLeastSquare](#) ()  
*Empty destructor.*

### Protected Member Functions

- virtual [bool makeWeights](#) (int i)

## Additional Inherited Members

### 29.377.1 Detailed Description

[PrPrmLeastSquare](#) - Implement the reciprocal least square parametrization by implementing the virtual function `makeWeights`

Definition at line 50 of file `PrPrmLeastSquare.h`.

### 29.377.2 Constructor & Destructor Documentation

#### 29.377.2.1 `PrPrmLeastSquare::PrPrmLeastSquare ( )`

Default constructor.

#### 29.377.2.2 `virtual PrPrmLeastSquare::~~PrPrmLeastSquare ( )` [virtual]

Empty destructor.

### 29.377.3 Member Function Documentation

#### 29.377.3.1 `virtual bool PrPrmLeastSquare::makeWeights ( int i )` [protected],[virtual]

Implements [PrParametrizeInt](#).

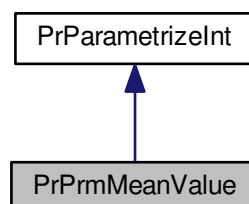
The documentation for this class was generated from the following file:

- `parametrization/include/GoTools/parametrization/PrPrmLeastSquare.h`

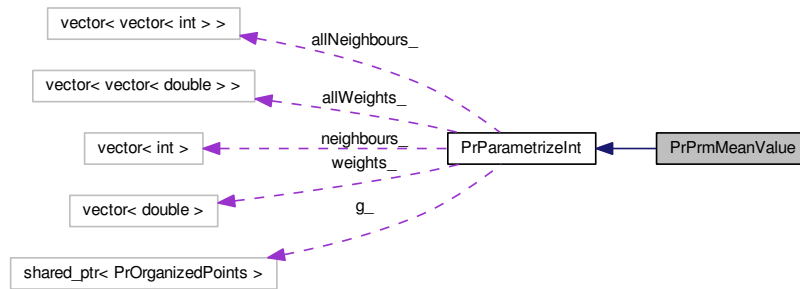
## 29.378 PrPrmMeanValue Class Reference

```
#include <PrPrmMeanValue.h>
```

Inheritance diagram for `PrPrmMeanValue`:



Collaboration diagram for PrPrmMeanValue:



## Public Member Functions

- [PrPrmMeanValue](#) ()  
*Default constructor.*
- virtual [~PrPrmMeanValue](#) ()  
*Empty destructor.*

## Protected Member Functions

- virtual [bool makeWeights](#) (int i)

## Additional Inherited Members

### 29.378.1 Detailed Description

[PrPrmMeanValue](#) - Implement the mean value parametrization by implementing the virtual function `makeWeights`. See preprint "Mean Value Coordinates", M.S. Floater. This method is now THE recommended method. The mapped points of the mean value parameterization depend infinitely smoothly on the original data points, unlike the shape-preserving parameterization. However, in many examples that have been tested, the two methods give similar results. (MF. August 2002).

Definition at line 58 of file `PrPrmMeanValue.h`.

### 29.378.2 Constructor & Destructor Documentation

#### 29.378.2.1 `PrPrmMeanValue::PrPrmMeanValue ( )`

Default constructor.

#### 29.378.2.2 `virtual PrPrmMeanValue::~~PrPrmMeanValue ( )` [virtual]

Empty destructor.

### 29.378.3 Member Function Documentation

29.378.3.1 `virtual bool PrPmMeanValue::makeWeights ( int i )` [protected],[virtual]

Implements [PrParametrizeInt](#).

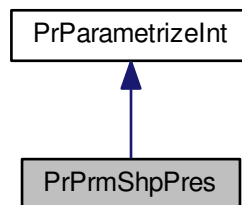
The documentation for this class was generated from the following file:

- parametrization/include/GoTools/parametrization/[PrPmMeanValue.h](#)

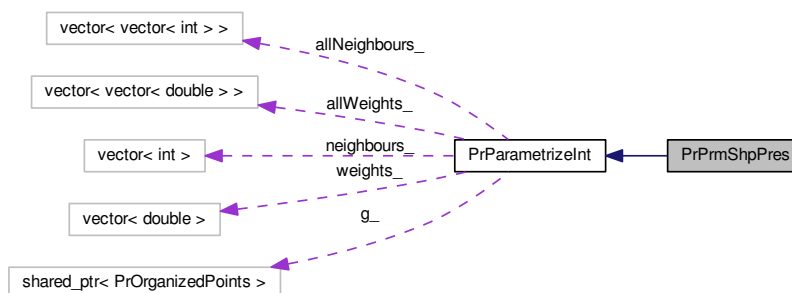
### 29.379 PrPmShpPres Class Reference

```
#include <PrPmShpPres.h>
```

Inheritance diagram for PrPmShpPres:



Collaboration diagram for PrPmShpPres:



#### Public Member Functions

- [PrPmShpPres \(\)](#)  
*Default constructor.*
- [virtual ~PrPmShpPres \(\)](#)  
*Empty destructor.*



## Protected Member Functions

- virtual [bool makeWeights](#) (int i)
- [bool localParam](#) (int i)

## Protected Attributes

- `std::vector< double > u_`
- `std::vector< double > v_`
- `std::vector< double > alpha_`
- `std::vector< double > len_`

### 29.379.1 Detailed Description

[PrPmShpPres](#) - Implement the shape-preserving parametrization by implementing the virtual function `makeWeights`

Definition at line 50 of file `PrPmShpPres.h`.

### 29.379.2 Constructor & Destructor Documentation

#### 29.379.2.1 `PrPmShpPres::PrPmShpPres ( )`

Default constructor.

#### 29.379.2.2 `virtual PrPmShpPres::~PrPmShpPres ( ) [virtual]`

Empty destructor.

### 29.379.3 Member Function Documentation

#### 29.379.3.1 `bool PrPmShpPres::localParam ( int i ) [protected]`

#### 29.379.3.2 `virtual bool PrPmShpPres::makeWeights ( int i ) [protected], [virtual]`

Implements [PrParametrizeInt](#).

### 29.379.4 Member Data Documentation

#### 29.379.4.1 `std::vector<double> PrPmShpPres::alpha_ [protected]`

Definition at line 55 of file `PrPmShpPres.h`.

29.379.4.2 `std::vector<double> PrPmShpPres::len_` [protected]

Definition at line 56 of file PrPmShpPres.h.

29.379.4.3 `std::vector<double> PrPmShpPres::u_` [protected]

Definition at line 53 of file PrPmShpPres.h.

29.379.4.4 `std::vector<double> PrPmShpPres::v_` [protected]

Definition at line 54 of file PrPmShpPres.h.

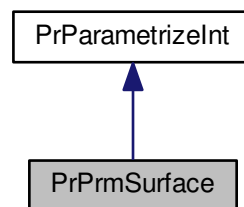
The documentation for this class was generated from the following file:

- parametrization/include/GoTools/parametrization/PrPmShpPres.h

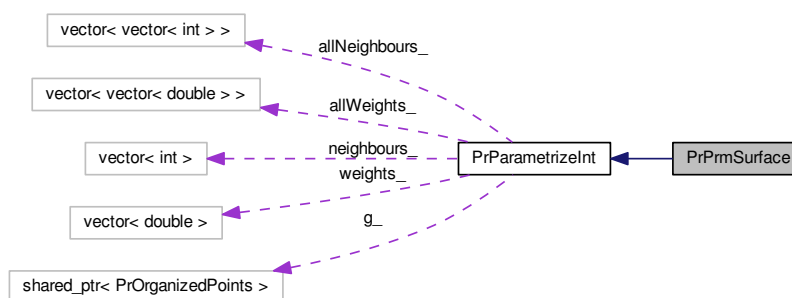
## 29.380 PrPmSurface Class Reference

```
#include <PrPmSurface.h>
```

Inheritance diagram for PrPmSurface:



Collaboration diagram for PrPmSurface:



## Public Member Functions

- [PrPrmSurface](#) ()  
*Default constructor.*
- virtual [~PrPrmSurface](#) ()  
*Empty destructor.*

## Additional Inherited Members

### 29.380.1 Detailed Description

[PrPrmSurface](#) - Implement the shape-preserving parametrization by implementing the virtual function `makeWeights`

Definition at line 50 of file `PrPrmSurface.h`.

### 29.380.2 Constructor & Destructor Documentation

#### 29.380.2.1 `PrPrmSurface::PrPrmSurface ( )`

Default constructor.

#### 29.380.2.2 `virtual PrPrmSurface::~~PrPrmSurface ( ) [virtual]`

Empty destructor.

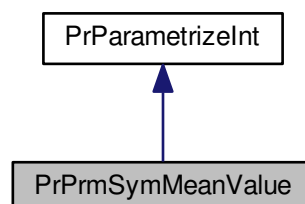
The documentation for this class was generated from the following file:

- `parametrization/include/GoTools/parametrization/PrPrmSurface.h`

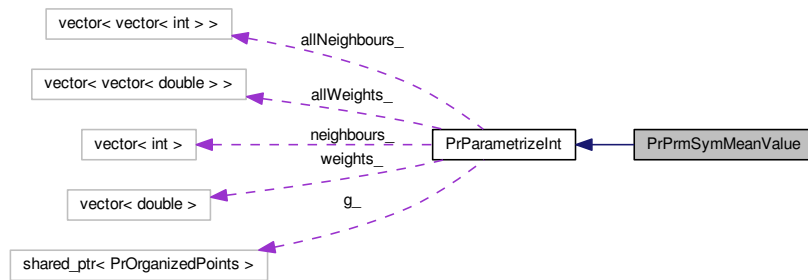
## 29.381 PrPrmSymMeanValue Class Reference

```
#include <PrPrmSymMeanValue.h>
```

Inheritance diagram for `PrPrmSymMeanValue`:



Collaboration diagram for PrPrmSymMeanValue:



## Public Member Functions

- [PrPrmSymMeanValue \(\)](#)  
*Default constructor.*
- [~PrPrmSymMeanValue \(\)](#)  
*Empty destructor.*

## Protected Member Functions

- virtual [bool makeWeights](#) (int i)
- [double tanThetaOverTwo](#) (Vector3D &a, Vector3D &b, Vector3D &c)

## Additional Inherited Members

### 29.381.1 Detailed Description

[PrPrmSymMeanValue](#) - Implements the virtual function `makeWeights`. This is an attempt to improve the mean value method of [PrPrmMeanValue](#) by making it symmetric.

Definition at line 51 of file `PrPrmSymMeanValue.h`.

### 29.381.2 Constructor & Destructor Documentation

#### 29.381.2.1 PrPrmSymMeanValue::PrPrmSymMeanValue ( )

Default constructor.

#### 29.381.2.2 PrPrmSymMeanValue::~PrPrmSymMeanValue ( )

Empty destructor.

### 29.381.3 Member Function Documentation

29.381.3.1 `virtual bool PrPrmSymMeanValue::makeWeights ( int i )` [protected],[virtual]

Implements [PrParametrizeInt](#).

29.381.3.2 `double PrPrmSymMeanValue::tanThetaOverTwo ( Vector3D & a, Vector3D & b, Vector3D & c )` [protected]

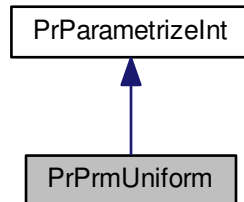
The documentation for this class was generated from the following file:

- parametrization/include/GoTools/parametrization/[PrPrmSymMeanValue.h](#)

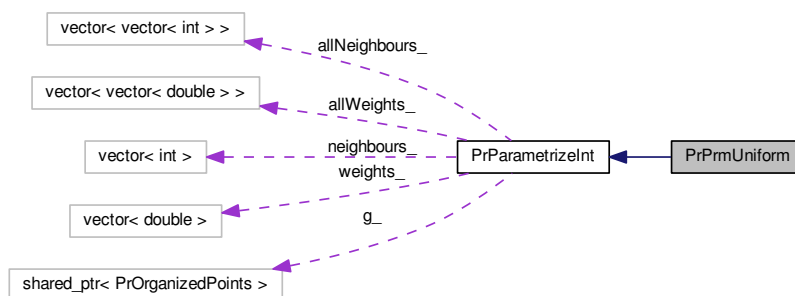
## 29.382 PrPrmUniform Class Reference

```
#include <PrPrmUniform.h>
```

Inheritance diagram for PrPrmUniform:



Collaboration diagram for PrPrmUniform:



## Public Member Functions

- [PrPrmUniform](#) ()  
*Default constructor.*
- virtual [~PrPrmUniform](#) ()  
*Empty destructor.*

## Additional Inherited Members

### 29.382.1 Detailed Description

[PrPrmUniform](#) - Implement the uniform parametrization by implementing the virtual function `makeWeights`

Definition at line 50 of file `PrPrmUniform.h`.

### 29.382.2 Constructor & Destructor Documentation

#### 29.382.2.1 `PrPrmUniform::PrPrmUniform ( )`

Default constructor.

#### 29.382.2.2 virtual `PrPrmUniform::~~PrPrmUniform ( )` [virtual]

Empty destructor.

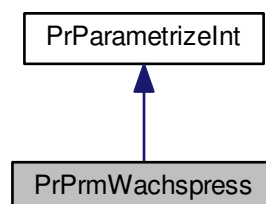
The documentation for this class was generated from the following file:

- `parametrization/include/GoTools/parametrization/PrPrmUniform.h`

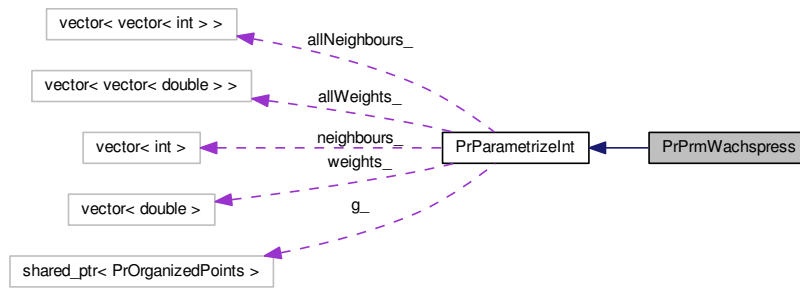
## 29.383 PrPrmWachspress Class Reference

```
#include <PrPrmWachspress.h>
```

Inheritance diagram for `PrPrmWachspress`:



Collaboration diagram for PrPrmWachspress:



## Public Member Functions

- [PrPrmWachspress \(\)](#)  
*Default constructor.*
- [virtual ~PrPrmWachspress \(\)](#)  
*Empty destructor.*

## Protected Member Functions

- [virtual bool makeWeights \(int i\)](#)
- [double tanThetaOverTwo \(Vector3D &a, Vector3D &b, Vector3D &c\)](#)

## Additional Inherited Members

### 29.383.1 Detailed Description

[PrPrmWachspress](#) - Implement the Wachspress parametrization by implementing the virtual function `makeWeights`

Definition at line 49 of file `PrPrmWachspress.h`.

### 29.383.2 Constructor & Destructor Documentation

#### 29.383.2.1 PrPrmWachspress::PrPrmWachspress ( )

Default constructor.

#### 29.383.2.2 virtual PrPrmWachspress::~~PrPrmWachspress ( ) [virtual]

Empty destructor.

### 29.383.3 Member Function Documentation

29.383.3.1 `virtual bool PrPrmWachspress::makeWeights ( int i )` [protected],[virtual]

Implements [PrParametrizeInt](#).

29.383.3.2 `double PrPrmWachspress::tanThetaOverTwo ( Vector3D & a, Vector3D & b, Vector3D & c )` [protected]

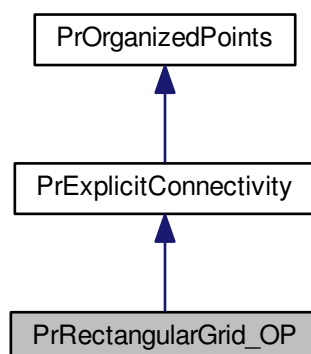
The documentation for this class was generated from the following file:

- parametrization/include/GoTools/parametrization/[PrPrmWachspress.h](#)

### 29.384 PrRectangularGrid\_OP Class Reference

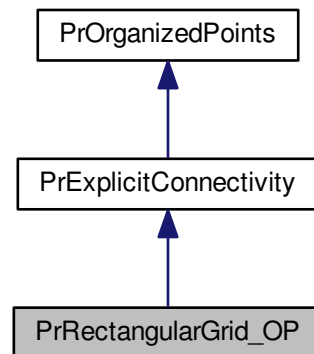
```
#include <PrRectangularGrid_OP.h>
```

Inheritance diagram for PrRectangularGrid\_OP:





Collaboration diagram for PrRectangularGrid\_OP:



## Public Member Functions

- [PrRectangularGrid\\_OP](#) ()  
*Default constructor.*
- [PrRectangularGrid\\_OP](#) (int numcols, int numRows, [const double](#) \*grid)  
*Constructor. Construct the [PrRectangularGrid\\_OP](#) from an array of nodes.*
- [PrRectangularGrid\\_OP](#) (int numcols, int numRows, [const double](#) \*grid, [const double](#) \*uvgrid)  
*Constructor. Construct the [PrRectangularGrid\\_OP](#) from an array of nodes.*
- virtual [~PrRectangularGrid\\_OP](#) ()  
*Default destructor.*
- int [gridToGraph](#) (int col, int row) [const](#)  
*compute the index of the point located at column 'col' and row 'row'.*
- void [graphToGrid](#) (int index, int &col, int &row) [const](#)  
*Get the column and row position of the point indexed 'index'.*
- void [print](#) (std::ostream &os)  
*print object to stream*
- void [scan](#) (std::istream &is)  
*read object from stream*
- void [printRawData](#) (std::ostream &os)  
*print raw data to stream (no parameterization involved)*
- void [scanRawData](#) (std::istream &is)  
*scan raw data from stream (no parameterization involved)*
- [double](#) \* [uvData](#) ()
- [double](#) \* [xyzData](#) ()
- [const double](#) \* [uvData](#) () [const](#)
- [const double](#) \* [xyzData](#) () [const](#)

## Derived from base class

- virtual int [getNumNodes](#) () [const](#)  
*return the number of nodes in the graph.*

- virtual `Vector3D get3dNode (int i) const`  
*return the i-th node in the graph if the nodes are three-dimensional*
- virtual void `set3dNode (int i, const Vector3D &p)`
- virtual void `getNeighbours (int i, vector< int > &neighbours) const`
- virtual `bool isBoundary (int i) const`  
*is the i-th node a boundary node or interior node?*
- virtual `double getU (int i) const`  
*return the u parameter value of the i-th node.*
- virtual `double getV (int i) const`  
*return the v parameter value of the i-th node.*
- virtual void `setU (int i, double u)`  
*reset the u parameter value of the i-th node.*
- virtual void `setV (int i, double v)`  
*reset the v parameter value of the i-th node.*

### Other routines

- void `setDim (int numcols, int numRows)`  
*Set the dimensions of the grid.*
- void `setXYZVertices (const double *grid)`  
*Reset all the xyz points from an array.*
- void `setUVVertices (const double *grid)`  
*Reset all the uv points from an array.*
- void `getDim (int &numcols, int &numrows) const`  
*Get the dimensions of the grid.*
- `double getU (int i, int j) const`  
*Get U value for gridpoint (i,j)*
- `double getV (int i, int j) const`  
*Get V value for gridpoint (i,j)*
- void `getCorners (int &c1, int &c2, int &c3, int &c4) const`
- `bool getUVTriangle (double u, double v, int &i, int &j, bool &right, double &tau0, double &tau1, double &tau2) const`
- `bool getUVTriangle (double &u, double &v, int ilast, int jlast, int &i, int &j, bool &right, double &tau0, double &tau1, double &tau2) const`
- `bool getTriangle (double u, double v, int ii, int jj, bool &right, double &tau0, double &tau1, double &tau2) const`

## Additional Inherited Members

### 29.384.1 Detailed Description

`PrRectangularGrid_OP` - This class represents a (topologically) rectangular grid of points in  $R^3$  with or without parameter points in  $R^2$ . It implements the virtual functions in `PrOrganizedPoints` specific to a grid. This kind of grid can be parametrized by `PrParametrize`.

Definition at line 55 of file `PrRectangularGrid_OP.h`.

### 29.384.2 Constructor & Destructor Documentation

#### 29.384.2.1 `PrRectangularGrid_OP::PrRectangularGrid_OP ( ) [inline]`

Default constructor.

Definition at line 70 of file `PrRectangularGrid_OP.h`.

29.384.2.2 `PrRectangularGrid_OP::PrRectangularGrid_OP ( int numcols, int numRows, const double * grid )`

Constructor. Construct the [PrRectangularGrid\\_OP](#) from an array of nodes.

29.384.2.3 `PrRectangularGrid_OP::PrRectangularGrid_OP ( int numcols, int numRows, const double * grid, const double * uvgrid )`

Constructor. Construct the [PrRectangularGrid\\_OP](#) from an array of nodes.

29.384.2.4 `virtual PrRectangularGrid_OP::~~PrRectangularGrid_OP ( ) [virtual]`

Default destructor.

### 29.384.3 Member Function Documentation

29.384.3.1 `virtual Vector3D PrRectangularGrid_OP::get3dNode ( int i ) const [inline],[virtual]`

return the i-th node in the graph if the nodes are three-dimensional

Implements [PrOrganizedPoints](#).

Definition at line 92 of file `PrRectangularGrid_OP.h`.

29.384.3.2 `void PrRectangularGrid_OP::getCorners ( int & c1, int & c2, int & c3, int & c4 ) const`

Get the indices of the four corner nodes in an anticlockwise direction, starting with the bottom left hand corner.

29.384.3.3 `void PrRectangularGrid_OP::getDim ( int & numcols, int & numRows ) const [inline]`

Get the dimensions of the grid.

Definition at line 112 of file `PrRectangularGrid_OP.h`.

29.384.3.4 `virtual void PrRectangularGrid_OP::getNeighbours ( int i, vector< int > & neighbours ) const [virtual]`

29.384.3.5 `virtual int PrRectangularGrid_OP::getNumNodes ( ) const [inline],[virtual]`

return the number of nodes in the graph.

Implements [PrOrganizedPoints](#).

Definition at line 91 of file `PrRectangularGrid_OP.h`.

29.384.3.6 `bool PrRectangularGrid_OP::getTriangle ( double u, double v, int ii, int jj, bool & right, double & tau0, double & tau1, double & tau2 ) const`

Check if the parameter point (u,v) can be found in one of the two triangles specified by the nodes at indexes (ii, jj), (ii+1, jj), (ii, jj+1) and (ii+1, jj+1). In that case, return its barycentric coordinates with respect to that triangle

## Parameters

<i>u</i>	u-coordinate of the specified point
<i>v</i>	v-coordinate of the specified point
<i>ii</i>	row-index of the "lower left" node of the rectangle specifying the two triangles where we want to look for the point
<i>jj</i>	column-index of the "lower left" node of the rectangle specifying the two triangles where we want to look for the point

## Return values

<i>right</i>	if the point was found to be in the triangle specified by the nodes (ii,jj), (ii+1, jj) and (ii+1, jj+1), then the value of 'right' will be set to 'true'. If the point was found to be in the triangle specified by the nodes (ii, jj), (ii, jj+1) and (ii+1, jj+1), then the value of 'right' will be set to 'false'. Otherwise, the value of this parameter will remain unchanged
<i>tau0</i>	first barycentric coordinate of the point in the located triangle
<i>tau1</i>	second barycentric coordinate of the point in the located triangle
<i>tau2</i>	third barycentric coordinate of the point in the located triangle

## Returns

'true' if the parameter point was found to be within one of the two triangles.

**29.384.3.7** `virtual double PrRectangularGrid_OP::getU ( int i ) const [inline],[virtual]`

return the u parameter value of the i-th node.

Implements [PrOrganizedPoints](#).

Definition at line 97 of file PrRectangularGrid\_OP.h.

**29.384.3.8** `double PrRectangularGrid_OP::getU ( int i, int j ) const [inline]`

Get U value for gridpoint (i,j)

Definition at line 116 of file PrRectangularGrid\_OP.h.

**29.384.3.9** `bool PrRectangularGrid_OP::getUVTriangle ( double u, double v, int & i, int & j, bool & right, double & tau0, double & tau1, double & tau2 ) const`

Locate the triangle in the parameter plane that contains the point (u,v). Report back the indexes of the "lower left" node of this triangle, as well as the point's barycentric coordinates in this triangle

## Parameters

<i>u</i>	u-coordinate of the specified point
<i>v</i>	v-coordinate of the specified point

## Return values

<i>i</i>	if triangle found, this will be the row index of the node in the "lower left" corner of the triangle.
<i>j</i>	if triangle found, this will be the column index of the node in the "lower left" corner of the triangle.
<i>right</i>	if the triangle was found, this variable reports whether the triangle consists of the "right" or "left" part of a quad in the grid. (Where the right part is defined as (i, j), (i+1, j) (i+1, j+1), whereas the left part is defined as (i, j), (i, j+1) (i+1, j+1).
<i>tau0</i>	if triangle found, this parameter is set to the first barycentric coordinate of the point (u,v) in the triangle.
<i>tau1</i>	if triangle found, this parameter is set to the second barycentric coordinate of the point (u,v) in the triangle.
<i>tau2</i>	if triangle found, this parameter is set to the third barycentric coordinate of the point (u,v) in the triangle.

## Returns

'true' if the triangle was located, 'false' otherwise

**29.384.3.10** `bool PrRectangularGrid_OP::getUVTriangle ( double & u, double & v, int llast, int jlast, int & i, int & j, bool & right, double & tau0, double & tau1, double & tau2 ) const`

This function behaves as the other [getUVTriangle\(\)](#) function, with the difference that the user can specify two additional parameters, '*l*last' and '*j*last' as a suggestion for where to start the search for a triangle.

**29.384.3.11** `virtual double PrRectangularGrid_OP::getV ( int i ) const [inline], [virtual]`

return the v parameter value of the i-th node.

Implements [PrOrganizedPoints](#).

Definition at line 98 of file PrRectangularGrid\_OP.h.

**29.384.3.12** `double PrRectangularGrid_OP::getV ( int i, int j ) const [inline]`

Get V value for gridpoint (i,j)

Definition at line 118 of file PrRectangularGrid\_OP.h.

**29.384.3.13** `void PrRectangularGrid_OP::graphToGrid ( int index, int & col, int & row ) const [inline]`

Get the column and row position of the point indexed '*index*'.

Definition at line 86 of file PrRectangularGrid\_OP.h.

**29.384.3.14** `int PrRectangularGrid_OP::gridToGraph ( int col, int row ) const [inline]`

compute the index of the point located at column '*col*' and row '*row*'.

Definition at line 83 of file PrRectangularGrid\_OP.h.

29.384.3.15 `virtual bool PrRectangularGrid_OP::isBoundary ( int i ) const` [virtual]

is the *i*-th node a boundary node or interior node?

Implements [PrOrganizedPoints](#).

29.384.3.16 `void PrRectangularGrid_OP::print ( std::ostream & os )`

print object to stream

29.384.3.17 `void PrRectangularGrid_OP::printRawData ( std::ostream & os )`

print raw data to stream (no parameterization involved)

29.384.3.18 `void PrRectangularGrid_OP::scan ( std::istream & is )`

read object from stream

29.384.3.19 `void PrRectangularGrid_OP::scanRawData ( std::istream & is )`

scan raw data from stream (no parameterization involved)

29.384.3.20 `virtual void PrRectangularGrid_OP::set3dNode ( int i, const Vector3D & p )` [inline],[virtual]

set the coordinates of the *i*-th node in the graph if the nodes are three-dimensional.

Implements [PrOrganizedPoints](#).

Definition at line 93 of file `PrRectangularGrid_OP.h`.

29.384.3.21 `void PrRectangularGrid_OP::setDim ( int numcols, int numrows )`

Set the dimensions of the grid.

29.384.3.22 `virtual void PrRectangularGrid_OP::setU ( int i, double u )` [inline],[virtual]

reset the *u* parameter value of the *i*-th node.

Implements [PrOrganizedPoints](#).

Definition at line 99 of file `PrRectangularGrid_OP.h`.

29.384.3.23 void PrRectangularGrid\_OP::setUVVertices ( const double \* *grid* )

Reset all the uv points from an array.

29.384.3.24 virtual void PrRectangularGrid\_OP::setV ( int *i*, double *v* ) [inline], [virtual]

reset the v parameter value of the i-th node.

Implements [PrOrganizedPoints](#).

Definition at line 100 of file PrRectangularGrid\_OP.h.

29.384.3.25 void PrRectangularGrid\_OP::setXYZVertices ( const double \* *grid* )

Reset all the xyz points from an array.

29.384.3.26 double\* PrRectangularGrid\_OP::uvData ( ) [inline]

Definition at line 195 of file PrRectangularGrid\_OP.h.

29.384.3.27 const double\* PrRectangularGrid\_OP::uvData ( ) const [inline]

Definition at line 203 of file PrRectangularGrid\_OP.h.

29.384.3.28 double\* PrRectangularGrid\_OP::xyzData ( ) [inline]

Definition at line 199 of file PrRectangularGrid\_OP.h.

29.384.3.29 const double\* PrRectangularGrid\_OP::xyzData ( ) const [inline]

Definition at line 207 of file PrRectangularGrid\_OP.h.

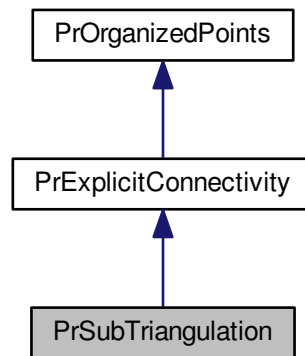
The documentation for this class was generated from the following file:

- parametrization/include/GoTools/parametrization/PrRectangularGrid\_OP.h

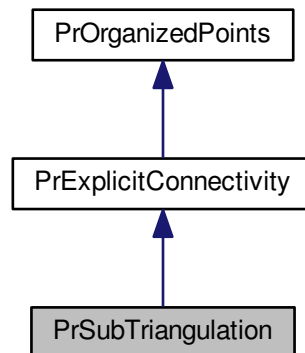
## 29.385 PrSubTriangulation Class Reference

```
#include <PrSubTriangulation.h>
```

Inheritance diagram for PrSubTriangulation:



Collaboration diagram for PrSubTriangulation:



### Public Member Functions

- [PrSubTriangulation](#) ()  
*Default constructor.*
- [PrSubTriangulation](#) ([PrTriangulation\\_OP](#) \*graph, vector< int > boundary, vector< int > interior)
- [~PrSubTriangulation](#) ()  
*Empty destructor.*



- void [attach](#) ([PrTriangulation\\_OP](#) \*graph)  
*Set the top-level triangulation to 'graph'.*
- void [initialize](#) (vector< int > boundary, vector< int > interior)

### Derived from base class

- virtual int [getNumNodes](#) () const  
*return the number of nodes in the graph.*
- virtual int [findNumBdyNodes](#) () const  
*count the number of boundary nodes in the graph.*
- virtual [Vector3D](#) [get3dNode](#) (int i) const  
*return the i-th node in the graph if the nodes are three-dimensional*
- virtual void [set3dNode](#) (int i, const [Vector3D](#) &p)
- virtual void [getNeighbours](#) (int i, vector< int > &neighbours) const
- virtual bool [isBoundary](#) (int i) const  
*is the i-th node a boundary node or interior node?*
- virtual double [getU](#) (int i) const  
*return the u parameter value of the i-th node.*
- virtual double [getV](#) (int i) const  
*return the v parameter value of the i-th node.*
- virtual void [setU](#) (int i, double u)  
*reset the u parameter value of the i-th node.*
- virtual void [setV](#) (int i, double v)  
*reset the v parameter value of the i-th node.*

### Other routines

- int [getGlobalIndex](#) (int i) const  
*get the global index of the node with local index 'i'.*
- int [getLocalIndex](#) (int i) const  
*get the local index of the node with global index 'i'.*
- bool [triangleInThis](#) (int i) const
- void [getNeighbourTriangles](#) (int i, vector< int > &neighbours) const
- void [getTriangleIndices](#) (vector< int > &indices) const

## Additional Inherited Members

### 29.385.1 Detailed Description

[PrSubTriangulation](#) - This class represents a sub-triangulation of another triangulation, using a permutation vector to access the nodes and triangles of the other triangulation It implements the virtual functions in [PrOrganizedPoints](#). This kind of triangulation can be parametrized by [PrParametrize](#).

The internal indices of the vertices are  $[0..\text{numBdy\_}-1]$  for the boundary nodes and  $[\text{numBdy\_}..\text{numBdy\_}+\text{numInt\_}-1]$  for the interior nodes. They are referred to as "new" indices, whereas the indices in the associated [PrTriangulation\\_OP](#) are called "old" indices. Therefore, "new2old\_" provides a mapping from  $[0..\text{numBdy\_}+\text{numInt\_}-1]$  to the "old" indices and "old2new\_" is its inverse.

We assume the boundary vertices to be ordered anticlockwise.

Definition at line 71 of file [PrSubTriangulation.h](#).

## 29.385.2 Constructor & Destructor Documentation

### 29.385.2.1 `PrSubTriangulation::PrSubTriangulation ( )` `[inline]`

Default constructor.

Definition at line 82 of file `PrSubTriangulation.h`.

### 29.385.2.2 `PrSubTriangulation::PrSubTriangulation ( PrTriangulation_OP * graph, vector< int > boundary, vector< int > interior )`

Constructor. Construct the [PrSubTriangulation](#) from an array with the indices of the "boundary" nodes and another carrying the indices of the "interior" nodes. All indices are relative to the indexing in the associated [PrTriangulation\\_OP](#) "graph"

### 29.385.2.3 `PrSubTriangulation::~~PrSubTriangulation ( )` `[inline]`

Empty destructor.

Definition at line 92 of file `PrSubTriangulation.h`.

## 29.385.3 Member Function Documentation

### 29.385.3.1 `void PrSubTriangulation::attach ( PrTriangulation_OP * graph )` `[inline]`

Set the top-level triangulation to 'graph'.

Definition at line 95 of file `PrSubTriangulation.h`.

### 29.385.3.2 `virtual int PrSubTriangulation::findNumBdyNodes ( ) const` `[inline],[virtual]`

count the number of boundary nodes in the graph.

Reimplemented from [PrOrganizedPoints](#).

Definition at line 108 of file `PrSubTriangulation.h`.

### 29.385.3.3 `virtual Vector3D PrSubTriangulation::get3dNode ( int i ) const` `[inline],[virtual]`

return the i-th node in the graph if the nodes are three-dimensional

Implements [PrOrganizedPoints](#).

Definition at line 109 of file `PrSubTriangulation.h`.

**29.385.3.4** `int PrSubTriangulation::getGlobalIndex ( int i ) const` `[inline]`

get the global index of the node with local index '*i*'.

Definition at line 125 of file PrSubTriangulation.h.

**29.385.3.5** `int PrSubTriangulation::getLocalIndex ( int i ) const` `[inline]`

get the local index of the node with global index '*i*'.

Definition at line 127 of file PrSubTriangulation.h.

**29.385.3.6** `virtual void PrSubTriangulation::getNeighbours ( int i, vector< int > & neighbours ) const` `[virtual]`

**29.385.3.7** `void PrSubTriangulation::getNeighbourTriangles ( int i, vector< int > & neighbours ) const`

Get all the neighbour triangles in the global triangulation of a node '*i*'. (*i* is global index).

**29.385.3.8** `virtual int PrSubTriangulation::getNumNodes ( ) const` `[inline],[virtual]`

return the number of nodes in the graph.

Implements [PrOrganizedPoints](#).

Definition at line 107 of file PrSubTriangulation.h.

**29.385.3.9** `void PrSubTriangulation::getTriangleIndices ( vector< int > & indices ) const`

Get all indices of triangles made up of interior points in the global triangulation, as well as those made up of boundary/interior points and equally belong to the sub-triangulation.

**29.385.3.10** `virtual double PrSubTriangulation::getU ( int i ) const` `[inline],[virtual]`

return the u parameter value of the *i*-th node.

Implements [PrOrganizedPoints](#).

Definition at line 116 of file PrSubTriangulation.h.

**29.385.3.11** `virtual double PrSubTriangulation::getV ( int i ) const` `[inline],[virtual]`

return the v parameter value of the *i*-th node.

Implements [PrOrganizedPoints](#).

Definition at line 117 of file PrSubTriangulation.h.

29.385.3.12 `void PrSubTriangulation::initialize ( vector< int > boundary, vector< int > interior )`

Initialize the [PrSubTriangulation](#). The `attach()` function should already have been called in order to define the [PrTriangulation\\_OP](#) that constitutes the base triangulation. The 'boundary' vector specifies the indexes of the boundary nodes, and the 'interior' vector specifies the indexes of the interior nodes. All indices are relative to the indexing in the associated [PrTriangulation\\_OP](#).

29.385.3.13 `virtual bool PrSubTriangulation::isBoundary ( int i ) const [inline],[virtual]`

is the *i*-th node a boundary node or interior node?

Implements [PrOrganizedPoints](#).

Definition at line 114 of file `PrSubTriangulation.h`.

29.385.3.14 `virtual void PrSubTriangulation::set3dNode ( int i, const Vector3D & p ) [inline],[virtual]`

set the coordinates of the *i*-th node in the graph if the nodes are three-dimensional.

Implements [PrOrganizedPoints](#).

Definition at line 111 of file `PrSubTriangulation.h`.

29.385.3.15 `virtual void PrSubTriangulation::setU ( int i, double u ) [inline],[virtual]`

reset the *u* parameter value of the *i*-th node.

Implements [PrOrganizedPoints](#).

Definition at line 118 of file `PrSubTriangulation.h`.

29.385.3.16 `virtual void PrSubTriangulation::setV ( int i, double v ) [inline],[virtual]`

reset the *v* parameter value of the *i*-th node.

Implements [PrOrganizedPoints](#).

Definition at line 119 of file `PrSubTriangulation.h`.

29.385.3.17 `bool PrSubTriangulation::triangleInThis ( int i ) const`

Check if the triangle indexed '*i*' in the global triangulation also exists in this sub-triangulation. This is true if all three nodes of the triangle can be found in the sub-triangulation.

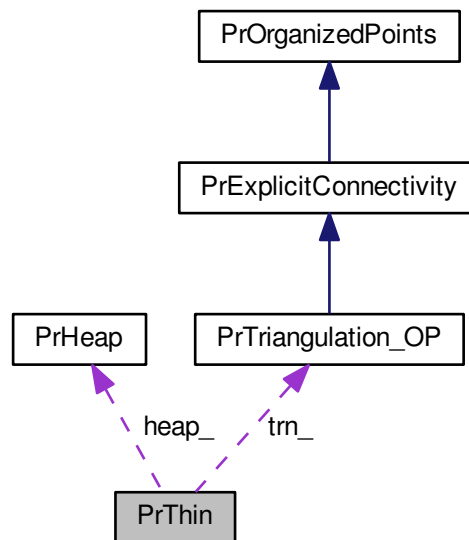
The documentation for this class was generated from the following file:

- parametrization/include/GoTools/parametrization/[PrSubTriangulation.h](#)

## 29.386 PrThin Class Reference

```
#include <PrThin.h>
```

Collaboration diagram for PrThin:



### Public Member Functions

- [PrThin \(\)](#)  
*Default constructor.*
- [virtual ~PrThin \(\)](#)  
*Empty destructor.*
- [void attach \(PrTriangulation\\_OP \\*trn\)](#)  
*Set the triangulation.*
- [void setError \(double error\)](#)  
*Set the tolerated error.*
- [void setStepNumber \(int n\)](#)  
*Set number of steps used for thinning.*
- [void findNearestNeighbour \(int i, int &tr\)](#)
- [void halfEdgeCollapse \(int i\)](#)  
*Remove the node i and retriangulate by collapsing one of its edges.*
- [void removePoint \(int i\)](#)
- [void thin \(\)](#)  
*Thin the points, until the error of further removals is above the threshold error\_.*
- [void printHeap \(\)](#)

## Protected Member Functions

- void `makeHeap` ()
- double `findError` (int i)

## Protected Attributes

- `PrTriangulation_OP` \* `trn_`
- `std::vector< PrNode >` \* `nlist_ptr`
- `std::vector< PrTriangle >` \* `tlist_ptr`
- `PrHeap` \* `heap_`
- double `error_`
- int `steps_`

### 29.386.1 Detailed Description

`PrThin` - This class implements an algorithm for thinning a triangulation.

Definition at line 50 of file `PrThin.h`.

### 29.386.2 Constructor & Destructor Documentation

#### 29.386.2.1 `PrThin::PrThin( )` [`inline`]

Default constructor.

Definition at line 68 of file `PrThin.h`.

#### 29.386.2.2 `virtual PrThin::~~PrThin( )` [`virtual`]

Empty destructor.

### 29.386.3 Member Function Documentation

#### 29.386.3.1 `void PrThin::attach( PrTriangulation_OP * trn )`

Set the triangulation.

#### 29.386.3.2 `double PrThin::findError( int i )` [`protected`]

#### 29.386.3.3 `void PrThin::findNearestNeighbour( int i, int & tr )`

Find nearest neighbour `j` to an interior node `i` (in 3D) and return the triangle `tr` which is left of the directed edge `(i,j)`. Note that `j` is then `tr.getAnticlockwiseNode(i)`.

29.386.3.4 void PrThin::halfEdgeCollapse ( int *i* )

Remove the node *i* and retriangulate by collapsing one of its edges.

29.386.3.5 void PrThin::makeHeap ( ) [protected]

29.386.3.6 void PrThin::printHeap ( )

29.386.3.7 void PrThin::removePoint ( int *i* )

Remove the node *i* from the triangulation and update heap. It is assumed that *i* is not in the heap

29.386.3.8 void PrThin::setError ( double *error* ) [inline]

Set the tolerated error.

Definition at line 76 of file PrThin.h.

29.386.3.9 void PrThin::setStepNumber ( int *n* ) [inline]

Set number of steps used for thinning.

Definition at line 79 of file PrThin.h.

29.386.3.10 void PrThin::thin ( )

Thin the points, until the error of further removals is above the threshold `error_`.

## 29.386.4 Member Data Documentation

29.386.4.1 double PrThin::error\_ [protected]

Definition at line 60 of file PrThin.h.

29.386.4.2 PrHeap\* PrThin::heap\_ [protected]

Definition at line 58 of file PrThin.h.

29.386.4.3 std::vector<PrNode>\* PrThin::nlist\_ptr [protected]

Definition at line 55 of file PrThin.h.

29.386.4.4 `int PrThin::steps_` [protected]

Definition at line 61 of file PrThin.h.

29.386.4.5 `std::vector<PrTriangle>* PrThin::tlist_ptr` [protected]

Definition at line 56 of file PrThin.h.

29.386.4.6 `PrTriangulation_OP* PrThin::trn_` [protected]

Definition at line 54 of file PrThin.h.

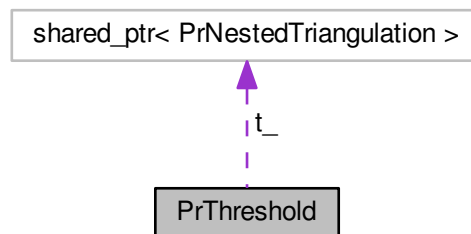
The documentation for this class was generated from the following file:

- parametrization/include/GoTools/parametrization/PrThin.h

## 29.387 PrThreshold Class Reference

```
#include <PrThreshold.h>
```

Collaboration diagram for PrThreshold:



### Public Member Functions

- [PrThreshold](#) ()  
*Default constructor.*
- virtual [~PrThreshold](#) ()  
*Empty destructor.*
- virtual void [attach](#) (shared\_ptr< [PrNestedTriangulation](#) > t)
- void [setThreshold](#) (double threshold=0.0)
- void [threshold](#) (double &comp\_rate, double &wavelet\_comp\_rate, int error\_out=0)
- double [findThreshold](#) (double &wavelet\_comp\_rate)
- void [thresholdByCompRate](#) (double &wavelet\_comp\_rate, int error\_out=0)



*Threshold by given compression rate, e.g. 0.05.*

- void [truncateLevel](#) (int jlev)  
*Set all coeffs at level jlev to zero ( $0 \leq jlev \leq \text{getFinestLevel}$ ).*
- void [leaveLevel](#) (int jlev)  
*Set all level zero coeffs to zero and all wavelet coeffs except those of level j.*
- double [maxNorm](#) ()  
*Calculate the max norm.*
- double [averageAbsValue](#) ()  
*Calculate the average absolute value.*
- double [weightedL2Norm](#) ()
- double [triangleNorm](#) (int i, int j, int k)

## Protected Attributes

- shared\_ptr< [PrNestedTriangulation](#) > t\_
- double threshold\_

### 29.387.1 Detailed Description

[PrThreshold](#) - This class implements an algorithm for thresholding a linear combination of coarse nodal functions and wavelets over several levels.

Definition at line 54 of file PrThreshold.h.

### 29.387.2 Constructor & Destructor Documentation

#### 29.387.2.1 PrThreshold::PrThreshold ( )

Default constructor.

#### 29.387.2.2 virtual PrThreshold::~PrThreshold ( ) [inline],[virtual]

Empty destructor.

Definition at line 65 of file PrThreshold.h.

### 29.387.3 Member Function Documentation

#### 29.387.3.1 virtual void PrThreshold::attach ( shared\_ptr< PrNestedTriangulation > t ) [virtual]

#### 29.387.3.2 double PrThreshold::averageAbsValue ( )

Calculate the average absolute value.

**29.387.3.3** `double PrThreshold::findThreshold ( double & wavelet_comp_rate )`

Find a threshold which will give the given compression rate. E.g. if `wavelet_comp_rate = 0.05` and there are `n` wavelets then we want a threshold so that `m = int(0.05 * n)` are retained.

**29.387.3.4** `void PrThreshold::leaveLevel ( int jlev )`

Set all level zero coeffs to zero and all wavelet coeffs except those of level `j`.

**29.387.3.5** `double PrThreshold::maxNorm ( )`

Calculate the max norm.

**29.387.3.6** `void PrThreshold::setThreshold ( double threshold = 0.0 ) [inline]`

Definition at line 69 of file `PrThreshold.h`.

**29.387.3.7** `void PrThreshold::threshold ( double & comp_rate, double & wavelet_comp_rate, int error_out = 0 )`

Threshold the wavelet coefficients by setting those whose absolute value is less than the tolerance to 0. Return the compression rate: ratio of no. of coarse hats plus non-zero wavelets over no. of coarse hats plus all wavelets. Return the wavelet compression rate: ratio of no. of non-zero over no. of wavelets.

If `error_out = 1`, do the opposite of thresholding, i.e, retain only those wavelet coefficients which are smaller in absolute value than the tolerance. The resulting piecewise linear function represents the error after thresholding and can be used to find the  $l_2$  error, max error, etc.

**29.387.3.8** `void PrThreshold::thresholdByCompRate ( double & wavelet_comp_rate, int error_out = 0 )`

Threshold by given compression rate, e.g. 0.05.

**29.387.3.9** `double PrThreshold::triangleNorm ( int i, int j, int k )`

**29.387.3.10** `void PrThreshold::truncateLevel ( int jlev )`

Set all coeffs at level `jlev` to zero ( $0 \leq jlev \leq \text{getFinestLevel}$ ).

**29.387.3.11** `double PrThreshold::weightedL2Norm ( )`

## 29.387.4 Member Data Documentation

**29.387.4.1** `shared_ptr<PrNestedTriangulation> PrThreshold::t_ [protected]`

Definition at line 58 of file `PrThreshold.h`.

29.387.4.2 `double PrThreshold::threshold_ [protected]`

Definition at line 59 of file PrThreshold.h.

The documentation for this class was generated from the following file:

- parametrization/include/GoTools/parametrization/PrThreshold.h

## 29.388 PrTriangle Class Reference

```
#include <PrTriangle.h>
```

### Public Member Functions

- [PrTriangle \(\)](#)  
*Default constructor.*
- [PrTriangle \(int n1, int n2, int n3, int t1, int t2, int t3\)](#)  
*Constructor.*
- [PrTriangle \(const PrTriangle &t\)](#)  
*Constructor.*
- [~PrTriangle \(\)](#)  
*Empty destructor.*
- [void init \(int n1, int n2, int n3, int t1, int t2, int t3\)](#)
- [const int & n1 \(\) const](#)
- [const int & n2 \(\) const](#)
- [const int & n3 \(\) const](#)
- [const int & t1 \(\) const](#)
- [const int & t2 \(\) const](#)
- [const int & t3 \(\) const](#)
- [int & n1 \(\)](#)
- [int & n2 \(\)](#)
- [int & n3 \(\)](#)
- [int & t1 \(\)](#)
- [int & t2 \(\)](#)
- [int & t3 \(\)](#)
- [int getOppositeTriangle \(int node\) const](#)  
*Return the neighbouring triangle which lies opposite to the given node.*
- [int getLeftTriangle \(int node\) const](#)  
*Return the neighbouring triangle which lies to the left of the given node.*
- [int getRightTriangle \(int node\) const](#)  
*Return the neighbouring triangle which lies to the right of the given node.*
- [int getAnticlockwiseNode \(int node\) const](#)
- [int getClockwiseNode \(int node\) const](#)  
*Return the node following "node" in a clockwise direction around the triangle.*
- [bool isVertex \(int node\) const](#)  
*Boolean routine: does "node" belong to the triangle?*
- [void replaceNode \(int n1, int n2\)](#)  
*If 'n1' is a node in the triangle, replace it with 'n2'.*
- [void replaceTriangle \(int t1, int t2\)](#)  
*If 't1' is a triangle linked to this triangle, replace it with 't2'.*
- [int getEdge \(int triangle, int &n1, int &n2\) const](#)
- [void print \(std::ostream &os\) const](#)  
*print contents to stream*
- [void scan \(std::istream &is\)](#)  
*read contents from stream*

### 29.388.1 Detailed Description

[PrTriangle](#) - This class represents a triangle for use in [PrTriangulation\\_OP](#). It has three indices of its three vertices in some anticlockwise order. Its three neighbouring triangles t1,t2,t3 are opposite to its vertices. Thus t1 is the triangle opposite n1, t2 is opposite n2 and t3 opposite n3. If any of the neighbouring triangles do not exist in the triangulation their index is -1 (which is out of range in the array of triangles in the triangulation).

Definition at line 57 of file PrTriangle.h.

### 29.388.2 Constructor & Destructor Documentation

#### 29.388.2.1 PrTriangle::PrTriangle ( ) [inline]

Default constructor.

Definition at line 66 of file PrTriangle.h.

#### 29.388.2.2 PrTriangle::PrTriangle ( int n1, int n2, int n3, int t1, int t2, int t3 ) [inline]

Constructor.

Definition at line 68 of file PrTriangle.h.

#### 29.388.2.3 PrTriangle::PrTriangle ( const PrTriangle & t ) [inline]

Constructor.

Definition at line 71 of file PrTriangle.h.

#### 29.388.2.4 PrTriangle::~PrTriangle ( ) [inline]

Empty destructor.

Definition at line 75 of file PrTriangle.h.

### 29.388.3 Member Function Documentation

#### 29.388.3.1 int PrTriangle::getAnticlockwiseNode ( int node ) const

Return the node following "node" in an anticlockwise direction around the triangle.

#### 29.388.3.2 int PrTriangle::getClockwiseNode ( int node ) const

Return the node following "node" in a clockwise direction around the triangle.

29.388.3.3 `int PrTriangle::getEdge ( int triangle, int & n1, int & n2 ) const`

if 'triangle' share an edge with this triangle, return the nodes on the shared edge.

29.388.3.4 `int PrTriangle::getLeftTriangle ( int node ) const`

Return the neighbouring triangle which lies to the left of the given node.

29.388.3.5 `int PrTriangle::getOppositeTriangle ( int node ) const`

Return the neighbouring triangle which lies opposite to the given node.

29.388.3.6 `int PrTriangle::getRightTriangle ( int node ) const`

Return the neighbouring triangle which lies to the right of the given node.

29.388.3.7 `void PrTriangle::init ( int n1, int n2, int n3, int t1, int t2, int t3 ) [inline]`

Definition at line 133 of file PrTriangle.h.

29.388.3.8 `bool PrTriangle::isVertex ( int node ) const`

Boolean routine: does "node" belong to the triangle?

29.388.3.9 `const int & PrTriangle::n1 ( ) const [inline]`

Definition at line 145 of file PrTriangle.h.

29.388.3.10 `int & PrTriangle::n1 ( ) [inline]`

Definition at line 187 of file PrTriangle.h.

29.388.3.11 `const int & PrTriangle::n2 ( ) const [inline]`

Definition at line 152 of file PrTriangle.h.

29.388.3.12 `int & PrTriangle::n2 ( ) [inline]`

Definition at line 194 of file PrTriangle.h.

29.388.3.13 `const int & PrTriangle::n3 ( ) const [inline]`

Definition at line 159 of file PrTriangle.h.

29.388.3.14 `int & PrTriangle::n3 ( ) [inline]`

Definition at line 201 of file PrTriangle.h.

29.388.3.15 `void PrTriangle::print ( std::ostream & os ) const`

print contents to stream

29.388.3.16 `void PrTriangle::replaceNode ( int n1, int n2 )`

If 'n1' is a node in the triangle, replace it with 'n2'.

29.388.3.17 `void PrTriangle::replaceTriangle ( int t1, int t2 )`

If 't1' is a triangle linked to this triangle, replace it with 't2'.

29.388.3.18 `void PrTriangle::scan ( std::istream & is )`

read contents from stream

29.388.3.19 `const int & PrTriangle::t1 ( ) const [inline]`

Definition at line 166 of file PrTriangle.h.

29.388.3.20 `int & PrTriangle::t1 ( ) [inline]`

Definition at line 208 of file PrTriangle.h.

29.388.3.21 `const int & PrTriangle::t2 ( ) const [inline]`

Definition at line 173 of file PrTriangle.h.

29.388.3.22 `int & PrTriangle::t2 ( ) [inline]`

Definition at line 215 of file PrTriangle.h.

29.388.3.23 `const int & PrTriangle::t3 ( ) const [inline]`

Definition at line 180 of file PrTriangle.h.

29.388.3.24 `int & PrTriangle::t3 ( ) [inline]`

Definition at line 222 of file PrTriangle.h.

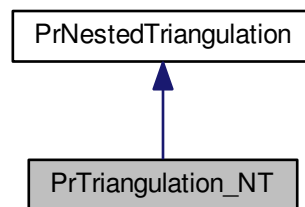
The documentation for this class was generated from the following file:

- parametrization/include/GoTools/parametrization/[PrTriangle.h](#)

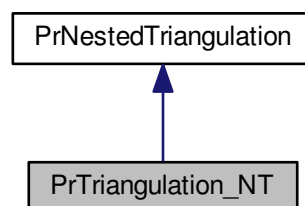
## 29.389 PrTriangulation\_NT Class Reference

```
#include <PrTriangulation_NT.h>
```

Inheritance diagram for PrTriangulation\_NT:



Collaboration diagram for PrTriangulation\_NT:



## Public Member Functions

- [PrTriangulation\\_NT](#) ()  
*Default constructor.*
- [PrTriangulation\\_NT](#) ([PrTriangulation\\_OP](#) &t)
- [~PrTriangulation\\_NT](#) ()  
*Empty destructor.*

## Derived from base class

- virtual int [getFinestLevel](#) ()
- virtual int [getNumNodes](#) (int jlev)  
*return the number of nodes in  $V^j$ ,  $j = 0, 1, \dots, k$ .*
- virtual double [getX](#) (int i)  
*return the x coeff. of the i-th node in the triangulation.*
- virtual double [getY](#) (int i)  
*return the y coeff. of the i-th node in the triangulation.*
- virtual double [getZ](#) (int i)  
*return the z coeff. of the i-th node in the triangulation.*
- virtual void [setX](#) (int i, const double &x)  
*set the x coeff. of the i-th node in the triangulation.*
- virtual void [setY](#) (int i, const double &y)  
*set the y coeff. of the i-th node in the triangulation.*
- virtual void [setZ](#) (int i, const double &z)  
*set the z coeff. of the i-th node in the triangulation.*
- virtual void [getNeighbours](#) (int i, int jlev, vector< int > &neighbours)
- virtual bool [isBoundary](#) (int i)  
*is the i-th node a boundary node or interior node?*

## Other functions

- void [refine](#) ()
- void [lift](#) ([PrParamTriangulation](#) \*pt)
- [PrLevelTriangulation\\_OP](#) \* [getLevel](#) (int i)  
*Get pointer to triangulation at level 'i'.*
- [PrLevelTriangulation\\_OP](#) \* [getTopLevel](#) ()  
*Get pointer to the finest triangulation.*

### 29.389.1 Detailed Description

[PrTriangulation\\_NT](#) - This class represents a nested sequence of triangulations of points in  $R^3$ . It implements the virtual functions in [PrNestedTriangulations](#).

Definition at line 53 of file [PrTriangulation\\_NT.h](#).

### 29.389.2 Constructor & Destructor Documentation

#### 29.389.2.1 [PrTriangulation\\_NT::PrTriangulation\\_NT](#) ( ) `[inline]`

Default constructor.

Definition at line 66 of file [PrTriangulation\\_NT.h](#).



## 29.389.2.2 PrTriangulation\_NT::PrTriangulation\_NT ( PrTriangulation\_OP &amp; t )

Constructor. Construct the [PrTriangulation\\_NT](#) from a [PrTriangulation\\_OP](#), using just one level in the hierarchy. This can later be refined by [refine\(\)](#).

## 29.389.2.3 PrTriangulation\_NT::~~PrTriangulation\_NT ( ) [inline]

Empty destructor.

Definition at line 72 of file PrTriangulation\_NT.h.

## 29.389.3 Member Function Documentation

## 29.389.3.1 virtual int PrTriangulation\_NT::getFinestLevel ( ) [inline],[virtual]

Return the finest level in the hierarchy, i.e.  $k$  if the vertices are arranged  $V^0 \subset V^1 \subset \dots \subset V^k$ .

Implements [PrNestedTriangulation](#).

Definition at line 76 of file PrTriangulation\_NT.h.

## 29.389.3.2 PrLevelTriangulation\_OP\* PrTriangulation\_NT::getLevel ( int i ) [inline]

Get pointer to triangulation at level  $i$ .

Definition at line 105 of file PrTriangulation\_NT.h.

## 29.389.3.3 virtual void PrTriangulation\_NT::getNeighbours ( int i, int jlev, vector&lt; int &gt; &amp; neighbours ) [inline],[virtual]

Return the indices of the neighbours of the  $i$ -th node with respect to  $V^j$ . This routine assumes that node  $i$  also belongs to  $V^j$ . The order of the neighbours returned will be

1. any anticlockwise order if  $i$  is an interior node
2. the unique anticlockwise order if  $i$  is a boundary node.

Implements [PrNestedTriangulation](#).

Definition at line 85 of file PrTriangulation\_NT.h.

## 29.389.3.4 virtual int PrTriangulation\_NT::getNumNodes ( int jlev ) [inline],[virtual]

return the number of nodes in  $V^j, j = 0, 1, \dots, k$ .

Implements [PrNestedTriangulation](#).

Definition at line 77 of file PrTriangulation\_NT.h.

**29.389.3.5** `PrLevelTriangulation_OP* PrTriangulation_NT::getTopLevel ( )` [inline]

Get pointer to the finest triangulation.

Definition at line 108 of file PrTriangulation\_NT.h.

**29.389.3.6** `virtual double PrTriangulation_NT::getX ( int i )` [inline],[virtual]

return the x coeff. of the i-th node in the triangulation.

Implements [PrNestedTriangulation](#).

Definition at line 79 of file PrTriangulation\_NT.h.

**29.389.3.7** `virtual double PrTriangulation_NT::getY ( int i )` [inline],[virtual]

return the y coeff. of the i-th node in the triangulation.

Implements [PrNestedTriangulation](#).

Definition at line 80 of file PrTriangulation\_NT.h.

**29.389.3.8** `virtual double PrTriangulation_NT::getZ ( int i )` [inline],[virtual]

return the z coeff. of the i-th node in the triangulation.

Implements [PrNestedTriangulation](#).

Definition at line 81 of file PrTriangulation\_NT.h.

**29.389.3.9** `virtual bool PrTriangulation_NT::isBoundary ( int i )` [inline],[virtual]

is the i-th node a boundary node or interior node?

Implements [PrNestedTriangulation](#).

Definition at line 87 of file PrTriangulation\_NT.h.

**29.389.3.10** `void PrTriangulation_NT::lift ( PrParamTriangulation * pt )`

Lift the vertices of the nested triangulation to some surface, using the parameterization defined by pt

**29.389.3.11** `void PrTriangulation_NT::refine ( )`

Construct the [PrTriangulation\\_NT](#) from a [PrTriangulation\\_OP](#), using just one level in the hierarchy. This can later be refined by [refine\(\)](#).

29.389.3.12 `virtual void PrTriangulation_NT::setX ( int i, const double & x )` `[inline],[virtual]`

set the x coeff. of the i-th node in the triangulation.

Implements [PrNestedTriangulation](#).

Definition at line 82 of file PrTriangulation\_NT.h.

29.389.3.13 `virtual void PrTriangulation_NT::setY ( int i, const double & y )` `[inline],[virtual]`

set the y coeff. of the i-th node in the triangulation.

Implements [PrNestedTriangulation](#).

Definition at line 83 of file PrTriangulation\_NT.h.

29.389.3.14 `virtual void PrTriangulation_NT::setZ ( int i, const double & z )` `[inline],[virtual]`

set the z coeff. of the i-th node in the triangulation.

Implements [PrNestedTriangulation](#).

Definition at line 84 of file PrTriangulation\_NT.h.

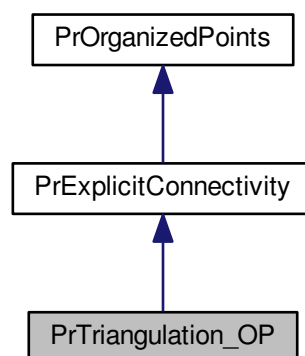
The documentation for this class was generated from the following file:

- parametrization/include/GoTools/parametrization/[PrTriangulation\\_NT.h](#)

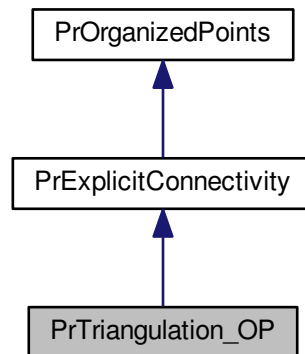
## 29.390 PrTriangulation\_OP Class Reference

```
#include <PrTriangulation_OP.h>
```

Inheritance diagram for PrTriangulation\_OP:



Collaboration diagram for PrTriangulation\_OP:



## Public Member Functions

- [PrTriangulation\\_OP](#) ()  
*Default constructor.*
- [PrTriangulation\\_OP](#) (const double \*xyz\_points, int np, const int \*triangles, int nt)
- [PrTriangulation\\_OP](#) (const double \*xyz\_points, const double \*uv\_points, int np, const int \*triangles, int nt)
- [PrTriangulation\\_OP](#) (PrExplicitConnectivity &op)
- [~PrTriangulation\\_OP](#) ()

## Derived from base class

- virtual int [getNumNodes](#) () const  
*return the number of nodes in the graph.*
- virtual Vector3D [get3dNode](#) (int i) const  
*return the i-th node in the graph if the nodes are three-dimensional*
- virtual void [set3dNode](#) (int i, const Vector3D &p)
- virtual void [getNeighbours](#) (int i, std::vector< int > &neighbours) const
- virtual bool [isBoundary](#) (int i) const  
*is the i-th node a boundary node or interior node?*
- virtual int [findNumFaces](#) () const  
*count the number of faces in the graph*
- virtual double [getU](#) (int i) const  
*return the u parameter value of the i-th node.*
- virtual double [getV](#) (int i) const  
*return the v parameter value of the i-th node.*
- virtual void [setU](#) (int i, double u)  
*reset the u parameter value of the i-th node.*
- virtual void [setV](#) (int i, double v)  
*reset the v parameter value of the i-th node.*

## Other functions

- virtual void [getTriangles](#) (int i, Go::ScratchVect< int, 20 > &triangles) const  
*Get all triangles that are incident with node 'i'.*

- [bool swapTriangles](#) (int t1, int t2)  
*swap triangle t1 and t2*
- void [swapNodes](#) (int n1, int n2)  
*swap node n1 and n2*
- [bool splitTriangles](#) (int t1, int t2, Vector3D &v)
- [bool splitVertex](#) (int i, int j, std::vector< int > &new\_nodes)
- [PrNode](#) & [getPrNode](#) (int i)  
*Get reference to node indexed 'i'.*
- [const PrNode](#) & [getPrNode](#) (int i) [const](#)
- [const PrTriangle](#) & [getPrTriangle](#) (int i) [const](#)  
*Get reference to triangle indexed 'i'.*
- [PrTriangle](#) & [getPrTriangle](#) (int i)  
*Get reference to triangle indexed 'i'.*
- std::vector< [PrTriangle](#) > & [getTriangleArray](#) ()  
*Get reference to all triangles in the triangulation.*
- std::vector< [PrNode](#) > & [getNodeArray](#) ()  
*Get reference to all nodes in the triangulation.*
- void [printXYZTriangles](#) (std::ostream &os, [bool num=false](#)) [const](#)
- void [printUVTriangles](#) (std::ostream &os, [bool num=false](#)) [const](#)
- void [print](#) (std::ostream &os) [const](#)  
*print whole class*
- void [scan](#) (std::istream &is)  
*scan whole class*
- void [printRawData](#) (std::ostream &os) [const](#)  
*print minimum data*
- void [scanRawData](#) (std::istream &is)  
*scan minimum data*
- void [printUV](#) (std::ostream &os) [const](#)  
*for converting uv to xyz with z=0*

## Additional Inherited Members

### 29.390.1 Detailed Description

[PrTriangulation\\_OP](#) - This class represents a triangulation of points in  $R^3$  with or without parameter points in  $R^2$ . It implements the virtual functions in [PrOrganizedPoints](#). This kind of triangulation can be parametrized by [PrParametrize](#).

Definition at line 55 of file [PrTriangulation\\_OP.h](#).

### 29.390.2 Constructor & Destructor Documentation

#### 29.390.2.1 [PrTriangulation\\_OP::PrTriangulation\\_OP](#) ( ) [\[inline\]](#)

Default constructor.

Definition at line 66 of file [PrTriangulation\\_OP.h](#).

### 29.390.2.2 PrTriangulation\_OP::PrTriangulation\_OP ( const double \* xyz\_points, int np, const int \* triangles, int nt )

Constructor. Construct the [PrTriangulation\\_OP](#) from an array of nodes and an array of triangles. The nodes array should contain  $x_0, y_0, z_0, x_1, y_1, z_1, x_2, y_2, z_2, x_3, \dots$  and the triangle array  $i_0, j_0, k_0, i_1, j_1, k_1, i_2, j_2, k_2, i_3, j_3, \dots$  where the  $i$ -th node is the 3D point  $(x_i, y_i, z_i)$  and the  $r$ -th triangle has the three nodes indexed  $i_r, j_r,$  and  $k_r$  in the node array. These nodes must be ordered anticlockwise (consistently throughout the triangulation). The length of the array "xyz\_points" is  $3 * np$  and the length of the array "triangles" is  $3 * nt$ . The uv points for the nodes are set to zero.

The function buildTopology is called which finds the three neighbouring triangles of each triangle and the first triangle of each node.

### 29.390.2.3 PrTriangulation\_OP::PrTriangulation\_OP ( const double \* xyz\_points, const double \* uv\_points, int np, const int \* triangles, int nt )

Constructor. Construct the [PrTriangulation\\_OP](#) from an array of nodes and an array of triangles. The xyz\_points array should contain  $x_0, y_0, z_0, x_1, y_1, z_1, x_2, y_2, z_2, x_3, \dots$ . The uv\_points array should contain  $u_0, v_0, u_1, v_1, u_2, v_2, u_3, \dots$  and the triangle array  $i_0, j_0, k_0, i_1, j_1, k_1, i_2, j_2, k_2, i_3, j_3, \dots$  where the  $i$ -th node is the 3D point  $(x_i, y_i, z_i)$  and the  $r$ -th triangle has the three nodes indexed  $i_r, j_r,$  and  $k_r$  in the node array. These nodes must be ordered anticlockwise (consistently throughout the triangulation). The length of the array "xyzpoints" is  $3 * np$  and The length of the array "uv\_points" is  $2 * np$  and the length of the array "triangles" is  $3 * nt$ .

The function buildTopology is called which finds the three neighbouring triangles of each triangle and the first triangle of each node.

### 29.390.2.4 PrTriangulation\_OP::PrTriangulation\_OP ( PrExplicitConnectivity & op )

Constructor. Construct the [PrTriangulation\\_OP](#) from a [PrExplicitConnectivity](#) class ASSUMING the [PrOrganizedPoints](#) class is a triangulation (every face has three vertices).

### 29.390.2.5 PrTriangulation\_OP::~PrTriangulation\_OP ( ) [inline]

Definition at line 109 of file PrTriangulation\_OP.h.

## 29.390.3 Member Function Documentation

### 29.390.3.1 virtual int PrTriangulation\_OP::findNumFaces ( ) const [inline], [virtual]

count the number of faces in the graph

Reimplemented from [PrExplicitConnectivity](#).

Definition at line 124 of file PrTriangulation\_OP.h.

### 29.390.3.2 virtual Vector3D PrTriangulation\_OP::get3dNode ( int i ) const [inline], [virtual]

return the  $i$ -th node in the graph if the nodes are three-dimensional

Implements [PrOrganizedPoints](#).

Definition at line 114 of file PrTriangulation\_OP.h.

**29.390.3.3** virtual void PrTriangulation\_OP::getNeighbours ( int *i*, std::vector< int > & *neighbours* ) const [virtual]

Return the indices of the neighbours of the *i*-th node in:

1. any anticlockwise order if *i* is an interior node
2. the unique anticlockwise order if *i* is a boundary node.

Implements [PrOrganizedPoints](#).

**29.390.3.4** std::vector<PrNode>& PrTriangulation\_OP::getNodeArray ( ) [inline]

Get reference to all nodes in the triangulation.

Definition at line 161 of file PrTriangulation\_OP.h.

**29.390.3.5** virtual int PrTriangulation\_OP::getNumNodes ( ) const [inline],[virtual]

return the number of nodes in the graph.

Implements [PrOrganizedPoints](#).

Definition at line 113 of file PrTriangulation\_OP.h.

**29.390.3.6** PrNode& PrTriangulation\_OP::getPrNode ( int *i* ) [inline]

Get reference to node indexed '*i*'.

Definition at line 152 of file PrTriangulation\_OP.h.

**29.390.3.7** const PrNode& PrTriangulation\_OP::getPrNode ( int *i* ) const [inline]

Definition at line 153 of file PrTriangulation\_OP.h.

**29.390.3.8** const PrTriangle& PrTriangulation\_OP::getPrTriangle ( int *i* ) const [inline]

Get reference to triangle indexed '*i*'.

Definition at line 155 of file PrTriangulation\_OP.h.

**29.390.3.9** PrTriangle& PrTriangulation\_OP::getPrTriangle ( int *i* ) [inline]

Get reference to triangle indexed '*i*'.

Definition at line 157 of file PrTriangulation\_OP.h.

29.390.3.10 `std::vector<PrTriangle>& PrTriangulation_OP::getTriangleArray ( )` [inline]

Get reference to all triangles in the triangulation.

Definition at line 159 of file PrTriangulation\_OP.h.

29.390.3.11 `virtual void PrTriangulation_OP::getTriangles ( int i, Go::ScratchVect< int, 20 > & triangles ) const`  
[virtual]

Get all triangles that are incident with node 'i'.

29.390.3.12 `virtual double PrTriangulation_OP::getU ( int i ) const` [inline],[virtual]

return the u parameter value of the i-th node.

Implements [PrOrganizedPoints](#).

Definition at line 125 of file PrTriangulation\_OP.h.

29.390.3.13 `virtual double PrTriangulation_OP::getV ( int i ) const` [inline],[virtual]

return the v parameter value of the i-th node.

Implements [PrOrganizedPoints](#).

Definition at line 126 of file PrTriangulation\_OP.h.

29.390.3.14 `virtual bool PrTriangulation_OP::isBoundary ( int i ) const` [virtual]

is the i-th node a boundary node or interior node?

Implements [PrOrganizedPoints](#).

29.390.3.15 `void PrTriangulation_OP::print ( std::ostream & os ) const`

print whole class

29.390.3.16 `void PrTriangulation_OP::printRawData ( std::ostream & os ) const`

print minimum data

29.390.3.17 `void PrTriangulation_OP::printUV ( std::ostream & os ) const`

for converting uv to xyz with z=0



29.390.3.18 `void PrTriangulation_OP::printUVTriangles ( std::ostream & os, bool num = false ) const`

Print out the uv triangles of the graph, useful for plotting If num = 1, print first the number of triangles.

29.390.3.19 `void PrTriangulation_OP::printXYZTriangles ( std::ostream & os, bool num = false ) const`

Print out the triangles of the graph, useful for plotting If num = 1, print first the number of triangles.

29.390.3.20 `void PrTriangulation_OP::scan ( std::istream & is )`

scan whole class

29.390.3.21 `void PrTriangulation_OP::scanRawData ( std::istream & is )`

scan minimum data

29.390.3.22 `virtual void PrTriangulation_OP::set3dNode ( int i, const Vector3D & p ) [inline],[virtual]`

set the coordinates of the i-th node in the graph if the nodes are three-dimensional.

Implements [PrOrganizedPoints](#).

Definition at line 115 of file PrTriangulation\_OP.h.

29.390.3.23 `virtual void PrTriangulation_OP::setU ( int i, double u ) [inline],[virtual]`

reset the u parameter value of the i-th node.

Implements [PrOrganizedPoints](#).

Definition at line 127 of file PrTriangulation\_OP.h.

29.390.3.24 `virtual void PrTriangulation_OP::setV ( int i, double v ) [inline],[virtual]`

reset the v parameter value of the i-th node.

Implements [PrOrganizedPoints](#).

Definition at line 128 of file PrTriangulation\_OP.h.

29.390.3.25 `bool PrTriangulation_OP::splitTriangles ( int t1, int t2, Vector3D & v )`

splits the common edge of triangles t1 and t2 and inserts the new vertex v there.

29.390.3.26 `bool PrTriangulation_OP::splitVertex ( int i, int j, std::vector< int > & new_nodes )`

Split the boundary vertex indexed 'i' into two, and replace the edge between 'i' and 'j' with two boundary edges. The requirements are that node 'i' should be on the boundary, and 'j' should share an edge with 'i'.

29.390.3.27 `void PrTriangulation_OP::swapNodes ( int n1, int n2 )`

swap node n1 and n2

29.390.3.28 `bool PrTriangulation_OP::swapTriangles ( int t1, int t2 )`

swap triangle t1 and t2

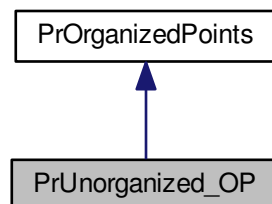
The documentation for this class was generated from the following file:

- parametrization/include/GoTools/parametrization/[PrTriangulation\\_OP.h](#)

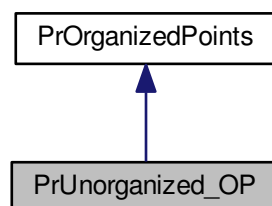
## 29.391 PrUnorganized\_OP Class Reference

```
#include <PrUnorganized_OP.h>
```

Inheritance diagram for PrUnorganized\_OP:



Collaboration diagram for PrUnorganized\_OP:



## Public Member Functions

- [PrUnorganized\\_OP](#) (int num\_cells=10)  
*Constructor.*
- [PrUnorganized\\_OP](#) (int n, int n\_int, [double](#) \*xyz\_points, int num\_cells=10)  
*Constructor.*
- [~PrUnorganized\\_OP](#) ()  
*Empty destructor.*

## Derived from base class

- virtual int [getNumNodes](#) ()  
*return the number of nodes in the graph.*
- virtual [Vector3D](#) [get3dNode](#) (int i)  
*return the i-th node in the graph if the nodes are three-dimensional*
- virtual void [set3dNode](#) (int i, [const](#) [Vector3D](#) &p)
- virtual void [getNeighbours](#) (int i, [vector](#)< int > &neighbours)
- virtual [bool](#) [isBoundary](#) (int i)  
*is the i-th node a boundary node or interior node?*
- virtual [double](#) [getU](#) (int i)  
*return the u parameter value of the i-th node.*
- virtual [double](#) [getV](#) (int i)  
*return the v parameter value of the i-th node.*
- virtual void [setU](#) (int i, [double](#) u)  
*reset the u parameter value of the i-th node.*
- virtual void [setV](#) (int i, [double](#) v)  
*reset the v parameter value of the i-th node.*

## Other functions

- [double](#) [getRadius](#) ()  
*get the specified radius for defining neighbours by distance*
- void [setRadius](#) ([double](#) radius)  
*specify the radius for defining neighbourhood by distance*
- void [setKNearest](#) (int knearest)
- void [useK](#) ()
- void [useRadius](#) ()  
*Use an euclidean distance to define the neighbourhood of a point.*
- void [print](#) (std::ostream &os)  
*Write contents to stream.*
- void [scan](#) (std::istream &is)  
*read contents from stream*
- void [printRawData](#) (std::ostream &os)  
*print raw data (no parameterization)*
- void [scanRawData](#) (std::istream &is, int num\_cells, [double](#) noise)  
*scan raw data (no parameterization)*

## Additional Inherited Members

### 29.391.1 Detailed Description

[PrUnorganized\\_OP](#) - This class represents a set of points in three dimensions. All we know about them is their boundary in sequence. No other topology is given. This kind of data can be parametrized by [PrParametrizeBdy](#) and [PrParametrizeInt](#).

Definition at line 54 of file [PrUnorganized\\_OP.h](#).

## 29.391.2 Constructor & Destructor Documentation

### 29.391.2.1 PrUnorganized\_OP::PrUnorganized\_OP ( int num\_cells = 10 )

Constructor.

### 29.391.2.2 PrUnorganized\_OP::PrUnorganized\_OP ( int n, int n\_int, double \* xyz\_points, int num\_cells = 10 )

Constructor.

### 29.391.2.3 PrUnorganized\_OP::~~PrUnorganized\_OP ( ) [inline]

Empty destructor.

Definition at line 73 of file PrUnorganized\_OP.h.

## 29.391.3 Member Function Documentation

### 29.391.3.1 virtual Vector3D PrUnorganized\_OP::get3dNode ( int i ) [inline],[virtual]

return the i-th node in the graph if the nodes are three-dimensional

Definition at line 82 of file PrUnorganized\_OP.h.

### 29.391.3.2 virtual void PrUnorganized\_OP::getNeighbours ( int i, vector< int > & neighbours ) [virtual]

Return the indices of the neighbours of the i-th node. (here there is no ordering: it's not a planar graph).

### 29.391.3.3 virtual int PrUnorganized\_OP::getNumNodes ( ) [inline],[virtual]

return the number of nodes in the graph.

Definition at line 79 of file PrUnorganized\_OP.h.

### 29.391.3.4 double PrUnorganized\_OP::getRadius ( ) [inline]

get the specified radius for defining neighbours by distance

Definition at line 110 of file PrUnorganized\_OP.h.

### 29.391.3.5 virtual double PrUnorganized\_OP::getU ( int i ) [inline],[virtual]

return the u parameter value of the i-th node.

Definition at line 94 of file PrUnorganized\_OP.h.

29.391.3.6 `virtual double PrUnorganized_OP::getV ( int i ) [inline],[virtual]`

return the *v* parameter value of the *i*-th node.

Definition at line 97 of file PrUnorganized\_OP.h.

29.391.3.7 `virtual bool PrUnorganized_OP::isBoundary ( int i ) [inline],[virtual]`

is the *i*-th node a boundary node or interior node?

Definition at line 91 of file PrUnorganized\_OP.h.

29.391.3.8 `void PrUnorganized_OP::print ( std::ostream & os )`

Write contents to stream.

29.391.3.9 `void PrUnorganized_OP::printRawData ( std::ostream & os )`

print raw data (no parameterization)

29.391.3.10 `void PrUnorganized_OP::scan ( std::istream & is )`

read contents from stream

29.391.3.11 `void PrUnorganized_OP::scanRawData ( std::istream & is, int num_cells, double noise )`

scan raw data (no parameterization)

29.391.3.12 `virtual void PrUnorganized_OP::set3dNode ( int i, const Vector3D & p ) [inline],[virtual]`

set the coordinates of the *i*-th node in the graph if the nodes are three-dimensional.

Implements [PrOrganizedPoints](#).

Definition at line 84 of file PrUnorganized\_OP.h.

29.391.3.13 `void PrUnorganized_OP::setKNearest ( int knearest ) [inline]`

specify a number for defining neighbourhoods by a fixed number of neighbours.

Definition at line 122 of file PrUnorganized\_OP.h.

29.391.3.14 `void PrUnorganized_OP::setRadius ( double radius ) [inline]`

specify the radius for defining neighbourhood by distance

Definition at line 119 of file PrUnorganized\_OP.h.

29.391.3.15 `virtual void PrUnorganized_OP::setU ( int i, double u ) [inline],[virtual]`

reset the u parameter value of the i-th node.

Implements [PrOrganizedPoints](#).

Definition at line 100 of file PrUnorganized\_OP.h.

29.391.3.16 `virtual void PrUnorganized_OP::setV ( int i, double v ) [inline],[virtual]`

reset the v parameter value of the i-th node.

Implements [PrOrganizedPoints](#).

Definition at line 103 of file PrUnorganized\_OP.h.

29.391.3.17 `void PrUnorganized_OP::useK ( ) [inline]`

Use a fixed number of neighbours to define the neighbourhood of a point.

Definition at line 125 of file PrUnorganized\_OP.h.

29.391.3.18 `void PrUnorganized_OP::useRadius ( ) [inline]`

Use an euclidean distance to define the neighbourhood of a point.

Definition at line 127 of file PrUnorganized\_OP.h.

The documentation for this class was generated from the following file:

- parametrization/include/GoTools/parametrization/[PrUnorganized\\_OP.h](#)

## 29.392 PrVec Class Reference

```
#include <PrVec.h>
```

## Public Member Functions

- [PrVec](#) ()  
*Default constructor.*
- [PrVec](#) (int n, double fill\_with=0.0)
- `template<typename InputIterator >`  
[PrVec](#) (InputIterator begin, InputIterator end)
- void [redim](#) (int n, double fill\_with=0.0)  
*Change size of vector.*
- int [size](#) () const  
*Query size of vector.*
- double & [operator](#)() (int i)  
*Element access.*
- const double & [operator](#)() (int i) const  
*Element access.*
- double & [operator](#)[] (int i)  
*Element access.*
- const double & [operator](#)[] (int i) const  
*Element access.*
- double [inner](#) (const [PrVec](#) &x)  
*Compute inner product with another vector of the same length.*
- void [read](#) (std::istream &is)  
*Read vector elements from stream 'is'.*
- void [print](#) (std::ostream &os)  
*Write vector elements, separated with spaces, to stream 'os'.*

## Protected Attributes

- std::vector< double > [a\\_](#)

### 29.392.1 Detailed Description

[PrVec](#) - This class represents a vector.

Definition at line 51 of file [PrVec.h](#).

### 29.392.2 Constructor & Destructor Documentation

#### 29.392.2.1 [PrVec::PrVec](#) ( ) [inline]

Default constructor.

Definition at line 59 of file [PrVec.h](#).

#### 29.392.2.2 [PrVec::PrVec](#) ( int n, double fill\_with = 0.0 ) [inline]

Constructor. Constructs a vector with n elements initialized by fill\_with.

Definition at line 63 of file [PrVec.h](#).

29.392.2.3 `template<typename InputIterator > PrVec::PrVec ( InputIterator begin, InputIterator end ) [inline]`

Constructor generating av vector from a range of elements specified with two iterators.

Definition at line 70 of file PrVec.h.

### 29.392.3 Member Function Documentation

29.392.3.1 `double PrVec::inner ( const PrVec & x )`

Compute inner product with another vector of the same length.

29.392.3.2 `double& PrVec::operator()( int i ) [inline]`

Element access.

Definition at line 81 of file PrVec.h.

29.392.3.3 `const double& PrVec::operator()( int i ) const [inline]`

Element access.

Definition at line 83 of file PrVec.h.

29.392.3.4 `double& PrVec::operator[]( int i ) [inline]`

Element access.

Definition at line 85 of file PrVec.h.

29.392.3.5 `const double& PrVec::operator[]( int i ) const [inline]`

Element access.

Definition at line 87 of file PrVec.h.

29.392.3.6 `void PrVec::print ( std::ostream & os )`

Write vector elements, separated with spaces, to stream '*os*'.

29.392.3.7 `void PrVec::read ( std::istream & is )`

Read vector elements from stream '*is*'.



29.392.3.8 `void PrVec::redim ( int n, double fill_with = 0.0 )`

Change size of vector.

29.392.3.9 `int PrVec::size ( ) const [inline]`

Query size of vector.

Definition at line 78 of file PrVec.h.

#### 29.392.4 Member Data Documentation

29.392.4.1 `std::vector<double> PrVec::a_ [protected]`

Definition at line 55 of file PrVec.h.

The documentation for this class was generated from the following file:

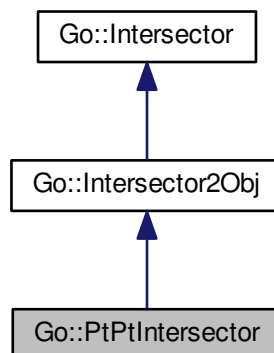
- parametrization/include/GoTools/parametrization/[PrVec.h](#)

### 29.393 Go::PtPtIntersector Class Reference

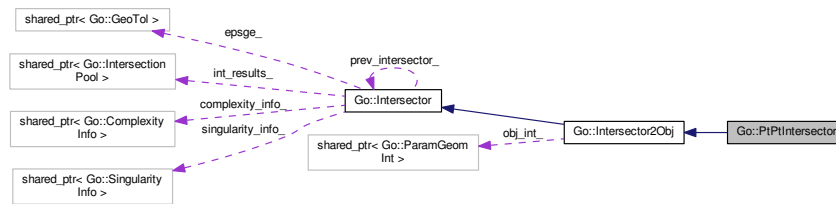
This class performs intersection between two points.

```
#include <PtPtIntersector.h>
```

Inheritance diagram for Go::PtPtIntersector:



Collaboration diagram for Go::PtPtIntersector:



## Public Member Functions

- `PtPtIntersector` (`shared_ptr< ParamGeomInt > point1`, `shared_ptr< ParamGeomInt > point2`, `shared_ptr< GeoTol > epsge`, `Intersector *prev=0`, `int eliminated_parameter=-1`, `double eliminated_value=0`)
- `PtPtIntersector` (`shared_ptr< ParamGeomInt > point1`, `shared_ptr< ParamGeomInt > point2`, `double epsge`, `Intersector *prev=0`, `int eliminated_parameter=-1`, `double eliminated_value=0`)
- virtual `~PtPtIntersector` ()  
*Destructor.*
- virtual `int numParams` () `const`

## Protected Member Functions

- virtual `shared_ptr< Intersector > lowerOrderIntersector` (`shared_ptr< ParamGeomInt > obj1`, `shared_ptr< ParamGeomInt > obj2`, `Intersector *prev=0`, `int eliminated_parameter=-1`, `double eliminated_value=0`)
- virtual `int checkCoincidence` ()
- virtual `void microCase` ()
- virtual `int updateIntersections` ()
- virtual `int repairIntersections` ()
- virtual `int linearCase` ()
- virtual `int doSubdivide` ()

## Additional Inherited Members

### 29.393.1 Detailed Description

This class performs intersection between two points.

Definition at line 52 of file PtPtIntersector.h.

### 29.393.2 Constructor & Destructor Documentation

29.393.2.1 `Go::PtPtIntersector::PtPtIntersector` (`shared_ptr< ParamGeomInt > point1`, `shared_ptr< ParamGeomInt > point2`, `shared_ptr< GeoTol > epsge`, `Intersector * prev = 0`, `int eliminated_parameter = -1`, `double eliminated_value = 0` )

Constructor. The last two variables are relevant only if the parent has one more parameter than the `Intersector` to be constructed.

## Parameters

<i>point1</i>	of type <a href="#">ParamPointInt</a> .
<i>point2</i>	of type <a href="#">ParamPointInt</a> .
<i>epsge</i>	the associated tolerance.
<i>prev</i>	the "parent" <a href="#">Intersector</a> (0 if there is no parent).
<i>eliminated_parameter</i>	the index (0) of the parameter that was removed from the parent <i>prev</i> .
<i>eliminated_value</i>	the value of the parameter that was removed from the parent <i>prev</i> .

29.393.2.2 `Go::PtPtIntersector::PtPtIntersector ( shared_ptr< ParamGeomInt > point1, shared_ptr< ParamGeomInt > point2, double epsge, Intersector * prev = 0, int eliminated_parameter = -1, double eliminated_value = 0 )`

Constructor. The last two variables are relevant only if the parent has one more parameter than the [Intersector](#) to be constructed.

## Parameters

<i>point1</i>	of type <a href="#">ParamPointInt</a> .
<i>point2</i>	of type <a href="#">ParamPointInt</a> .
<i>epsge</i>	the associated tolerance.
<i>prev</i>	the "parent" <a href="#">Intersector</a> (0 if there is no parent).
<i>eliminated_parameter</i>	the index (0) of the parameter that was removed from the parent <i>prev</i> .
<i>eliminated_value</i>	the value of the parameter that was removed from the parent <i>prev</i> .

29.393.2.3 `virtual Go::PtPtIntersector::~~PtPtIntersector ( ) [virtual]`

Destructor.

## 29.393.3 Member Function Documentation

29.393.3.1 `virtual int Go::PtPtIntersector::checkCoincidence ( ) [protected],[virtual]`

Implements [Go::Intersector2Obj](#).

29.393.3.2 `virtual int Go::PtPtIntersector::doSubdivide ( ) [protected],[virtual]`

Implements [Go::Intersector2Obj](#).

29.393.3.3 `virtual int Go::PtPtIntersector::linearCase ( ) [protected],[virtual]`

Implements [Go::Intersector2Obj](#).

29.393.3.4 `virtual shared_ptr<Intersector> Go::PtPtIntersector::lowerOrderIntersector ( shared_ptr< ParamGeomInt > obj1, shared_ptr< ParamGeomInt > obj2, Intersector * prev = 0, int eliminated_parameter = -1, double eliminated_value = 0 )` [protected],[virtual]

Implements [Go::Intersector2Obj](#).

29.393.3.5 `virtual void Go::PtPtIntersector::microCase ( )` [protected],[virtual]

Implements [Go::Intersector2Obj](#).

29.393.3.6 `virtual int Go::PtPtIntersector::numParams ( ) const` [inline],[virtual]

Return the number of parameter directions for the object.

#### Returns

the number of parameter directions

Implements [Go::Intersector](#).

Definition at line 102 of file PtPtIntersector.h.

29.393.3.7 `virtual int Go::PtPtIntersector::repairIntersections ( )` [inline],[protected],[virtual]

Implements [Go::Intersector](#).

Definition at line 121 of file PtPtIntersector.h.

29.393.3.8 `virtual int Go::PtPtIntersector::updateIntersections ( )` [protected],[virtual]

Implements [Go::Intersector2Obj](#).

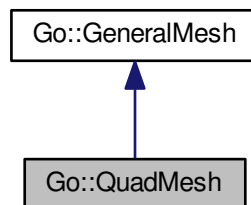
The documentation for this class was generated from the following file:

- [intersections/include/GoTools/intersections/PtPtIntersector.h](#)

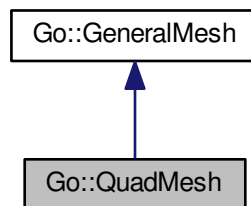
## 29.394 Go::QuadMesh Class Reference

```
#include <QuadMesh.h>
```

Inheritance diagram for Go::QuadMesh:



Collaboration diagram for Go::QuadMesh:



### Public Member Functions

- [QuadMesh](#) (int num\_vert=0, int num\_quads=0, [bool](#) use\_normals=true, [bool](#) use\_texcoords=false)
- [bool useNormals](#) ()  
*Check if normals will be computed.*
- [bool useTexCoords](#) ()  
*Check if texture coordinates will be computed.*
- int [numQuads](#) ()  
*The total number of quads.*
- virtual int [numTriangles](#) ()  
*The total number of triangles. Not used in this context.*
- virtual int [numVertices](#) ()  
*Number of nodes in mesh.*
- void [resize](#) (int num\_vert, int num\_quads)  
*Change mesh size.*

- virtual `double * vertexArray ()`  
*Fetch all nodes.*
- `double * normalArray ()`  
*Normal information.*
- `double * texcoordArray ()`  
*Texture coordinate information.*
- `unsigned int * quadIndexArray ()`  
*Indices to quads.*
- virtual `unsigned int * triangleIndexArray ()`  
*Indices to triangle. Not used in this context.*
- virtual `double * paramArray ()`  
*Fetch parameter values corresponding to the nodes.*
- virtual `int atBoundary (int idx)`  
*Check if a given node lies at the boundary.*

## Additional Inherited Members

### 29.394.1 Detailed Description

Based on `gvGenericTriMesh`, designed to store data for GL visualizing.

Definition at line 52 of file `QuadMesh.h`.

### 29.394.2 Constructor & Destructor Documentation

**29.394.2.1** `Go::QuadMesh::QuadMesh ( int num_vert = 0, int num_quads = 0, bool use_normals = true, bool use_texcoords = false ) [inline]`

Constructor. Give mesh size. Set whether or not normals and texture coordinates should be computed

Definition at line 57 of file `QuadMesh.h`.

### 29.394.3 Member Function Documentation

**29.394.3.1** `virtual int Go::QuadMesh::atBoundary ( int idx ) [inline],[virtual]`

Check if a given node lies at the boundary.

Implements [Go::GeneralMesh](#).

Definition at line 106 of file `QuadMesh.h`.

**29.394.3.2** `double* Go::QuadMesh::normalArray ( ) [inline]`

Normal information.

Definition at line 98 of file `QuadMesh.h`.

29.394.3.3 `int Go::QuadMesh::numQuads ( ) [inline]`

The total number of quads.

Definition at line 75 of file QuadMesh.h.

29.394.3.4 `virtual int Go::QuadMesh::numTriangles ( ) [inline],[virtual]`

The total number of triangles. Not used in this context.

Reimplemented from [Go::GeneralMesh](#).

Definition at line 79 of file QuadMesh.h.

29.394.3.5 `virtual int Go::QuadMesh::numVertices ( ) [inline],[virtual]`

Number of nodes in mesh.

Implements [Go::GeneralMesh](#).

Definition at line 82 of file QuadMesh.h.

29.394.3.6 `virtual double* Go::QuadMesh::paramArray ( ) [inline],[virtual]`

Fetch parameter values corresponding to the nodes.

Implements [Go::GeneralMesh](#).

Definition at line 105 of file QuadMesh.h.

29.394.3.7 `unsigned int* Go::QuadMesh::quadIndexArray ( ) [inline]`

Indices to quads.

Definition at line 102 of file QuadMesh.h.

29.394.3.8 `void Go::QuadMesh::resize ( int num_vert, int num_quads ) [inline]`

Change mesh size.

Definition at line 86 of file QuadMesh.h.

29.394.3.9 `double* Go::QuadMesh::texcoordArray ( ) [inline]`

Texture coordinate information.

Definition at line 100 of file QuadMesh.h.

**29.394.3.10** `virtual unsigned int* Go::QuadMesh::triangleIndexArray ( ) [inline],[virtual]`

Indices to triangle. Not used in this context.

Implements [Go::GeneralMesh](#).

Definition at line 104 of file QuadMesh.h.

**29.394.3.11** `bool Go::QuadMesh::useNormals ( ) [inline]`

Check if normals will be computed.

Definition at line 67 of file QuadMesh.h.

**29.394.3.12** `bool Go::QuadMesh::useTexCoords ( ) [inline]`

Check if texture coordinates will be computed.

Definition at line 71 of file QuadMesh.h.

**29.394.3.13** `virtual double* Go::QuadMesh::vertexArray ( ) [inline],[virtual]`

Fetch all nodes.

Implements [Go::GeneralMesh](#).

Definition at line 96 of file QuadMesh.h.

The documentation for this class was generated from the following file:

- [gtools-core/include/GoTools/tessellator/QuadMesh.h](#)

## 29.395 Go::QualityResults Class Reference

```
#include <QualityResults.h>
```

### Public Member Functions

- [~QualityResults \(\)](#)

### Friends

- class [ModelQuality](#)
- class [FaceSetQuality](#)
- class [ModelRepair](#)
- class [FaceSetRepair](#)



### 29.395.1 Detailed Description

Definition at line 56 of file QualityResults.h.

### 29.395.2 Constructor & Destructor Documentation

29.395.2.1 `Go::QualityResults::~~QualityResults ( )`

### 29.395.3 Friends And Related Function Documentation

29.395.3.1 `friend class FaceSetQuality` [`friend`]

Definition at line 59 of file QualityResults.h.

29.395.3.2 `friend class FaceSetRepair` [`friend`]

Definition at line 61 of file QualityResults.h.

29.395.3.3 `friend class ModelQuality` [`friend`]

Definition at line 58 of file QualityResults.h.

29.395.3.4 `friend class ModelRepair` [`friend`]

Definition at line 60 of file QualityResults.h.

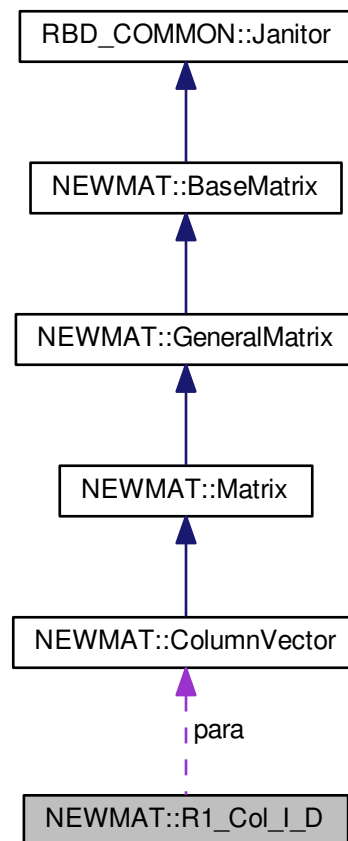
The documentation for this class was generated from the following file:

- `qualitymodule/include/GoTools/qualitymodule/QualityResults.h`

## 29.396 NEWMAT::R1\_Col\_I\_D Class Reference

```
#include <newmatnl.h>
```

Collaboration diagram for NEWMAT::R1\_Col\_I\_D:



### Public Member Functions

- virtual [bool IsValid](#) ()
- virtual [Real operator](#)() (int i)=0
- virtual void [Set](#) (const [ColumnVector](#) &X)
- [bool IsValid](#) (const [ColumnVector](#) &X)
- [Real operator](#)() (int i, const [ColumnVector](#) &X)
- virtual [ReturnMatrix Derivatives](#) ()=0
- virtual [~R1\\_Col\\_I\\_D](#) ()

### Protected Attributes

- [ColumnVector](#) para

### 29.396.1 Detailed Description

Definition at line 181 of file newmatnl.h.

### 29.396.2 Constructor & Destructor Documentation

29.396.2.1 virtual NEWMAT::R1\_Col\_I\_D::~~R1\_Col\_I\_D ( ) [inline],[virtual]

Definition at line 207 of file newmatnl.h.

### 29.396.3 Member Function Documentation

29.396.3.1 virtual ReturnMatrix NEWMAT::R1\_Col\_I\_D::Derivatives ( ) [pure virtual]

29.396.3.2 virtual bool NEWMAT::R1\_Col\_I\_D::IsValid ( ) [inline],[virtual]

Definition at line 194 of file newmatnl.h.

29.396.3.3 bool NEWMAT::R1\_Col\_I\_D::IsValid ( const ColumnVector & X ) [inline]

Definition at line 199 of file newmatnl.h.

29.396.3.4 virtual Real NEWMAT::R1\_Col\_I\_D::operator()( int i ) [pure virtual]

29.396.3.5 Real NEWMAT::R1\_Col\_I\_D::operator()( int i, const ColumnVector & X ) [inline]

Definition at line 202 of file newmatnl.h.

29.396.3.6 virtual void NEWMAT::R1\_Col\_I\_D::Set ( const ColumnVector & X ) [inline],[virtual]

Definition at line 197 of file newmatnl.h.

### 29.396.4 Member Data Documentation

29.396.4.1 ColumnVector NEWMAT::R1\_Col\_I\_D::para [protected]

Definition at line 191 of file newmatnl.h.

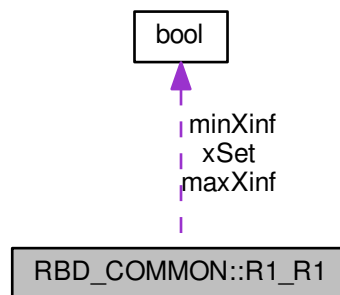
The documentation for this class was generated from the following file:

- [newmat/include/newmatnl.h](#)

## 29.397 RBD\_COMMON::R1\_R1 Class Reference

```
#include <solution.h>
```

Collaboration diagram for RBD\_COMMON::R1\_R1:



### Public Member Functions

- [R1\\_R1 \(\)](#)
- virtual [Real operator\(\) \(\)=0](#)
- virtual void [Set \(Real X\)](#)
- [Real operator\(\) \(Real X\)](#)
- virtual [bool IsValid \(Real X\)](#)
- [operator Real \(\)](#)
- virtual [~R1\\_R1 \(\)](#)

### Public Attributes

- [Real minX](#)
- [Real maxX](#)
- [bool minXinf](#)
- [bool maxXinf](#)

### Protected Attributes

- [Real x](#)
- [bool xSet](#)

### 29.397.1 Detailed Description

Definition at line 19 of file solution.h.

## 29.397.2 Constructor & Destructor Documentation

29.397.2.1 `RBD_COMMON::R1_R1( )` [inline]

Definition at line 34 of file solution.h.

29.397.2.2 `virtual RBD_COMMON::R1_R1::~~R1_R1( )` [inline],[virtual]

Definition at line 42 of file solution.h.

## 29.397.3 Member Function Documentation

29.397.3.1 `bool R1_R1::IsValid( Real X )` [virtual]

Definition at line 191 of file solution.cpp.

29.397.3.2 `R1_R1::operator Real( )`

Definition at line 27 of file solution.cpp.

29.397.3.3 `virtual Real RBD_COMMON::R1_R1::operator()( )` [pure virtual]

29.397.3.4 `Real RBD_COMMON::R1_R1::operator()( Real X )` [inline]

Definition at line 38 of file solution.h.

29.397.3.5 `void R1_R1::Set( Real X )` [virtual]

Definition at line 20 of file solution.cpp.

## 29.397.4 Member Data Documentation

29.397.4.1 `Real RBD_COMMON::R1_R1::maxX`

Definition at line 32 of file solution.h.

29.397.4.2 `bool RBD_COMMON::R1_R1::maxXinf`

Definition at line 33 of file solution.h.

#### 29.397.4.3 Real RBD\_COMMON::R1\_R1::minX

Definition at line 32 of file solution.h.

#### 29.397.4.4 bool RBD\_COMMON::R1\_R1::minXinf

Definition at line 33 of file solution.h.

#### 29.397.4.5 Real RBD\_COMMON::R1\_R1::x [protected]

Definition at line 28 of file solution.h.

#### 29.397.4.6 bool RBD\_COMMON::R1\_R1::xSet [protected]

Definition at line 29 of file solution.h.

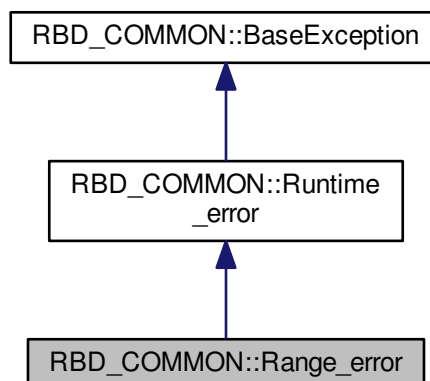
The documentation for this class was generated from the following files:

- newmat/include/solution.h
- newmat/src/solution.cpp

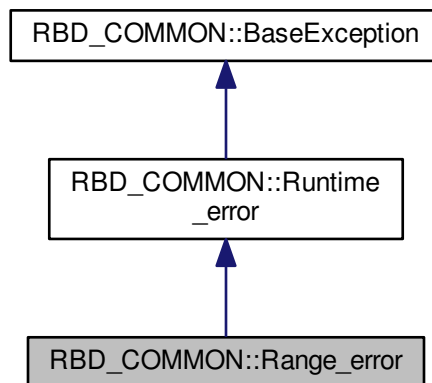
## 29.398 RBD\_COMMON::Range\_error Class Reference

```
#include <myexcept.h>
```

Inheritance diagram for RBD\_COMMON::Range\_error:



Collaboration diagram for RBD\_COMMON::Range\_error:



- static unsigned long [Select](#)
- [Range\\_error](#) (const char \*a\_what=0)

### Additional Inherited Members

#### 29.398.1 Detailed Description

Definition at line 413 of file myexcept.h.

#### 29.398.2 Constructor & Destructor Documentation

##### 29.398.2.1 Range\_error::Range\_error ( const char \* a\_what = 0 )

Definition at line 454 of file myexcept.cpp.

#### 29.398.3 Member Data Documentation

##### 29.398.3.1 unsigned long Range\_error::Select [static]

Definition at line 416 of file myexcept.h.

The documentation for this class was generated from the following files:

- [newmat/include/myexcept.h](#)
- [newmat/src/myexcept.cpp](#)

## 29.399 rank\_info Struct Reference

```
#include <sisIP.h>
```

### Public Attributes

- int \* [prio](#)
- int \* [groups](#)
- int [antgr](#)
- int [antrem](#)

### 29.399.1 Detailed Description

Definition at line 346 of file sisIP.h.

### 29.399.2 Member Data Documentation

#### 29.399.2.1 int rank\_info::antgr

Definition at line 350 of file sisIP.h.

#### 29.399.2.2 int rank\_info::antrem

Definition at line 351 of file sisIP.h.

#### 29.399.2.3 int\* rank\_info::groups

Definition at line 349 of file sisIP.h.

#### 29.399.2.4 int\* rank\_info::prio

Definition at line 348 of file sisIP.h.

The documentation for this struct was generated from the following file:

- sis/include/[sisIP.h](#)

## 29.400 Go::Rational Class Reference

```
#include <Rational.h>
```



## Public Member Functions

- [Rational](#) ()  
*Construct a rational number whose value is 0.*
- [Rational](#) (int p)  
*Construct a rational number whose value is 'p' (integer)*
- [Rational](#) (int p, int q)  
*Construct a rational number whose value is 'p'/q' (fraction)*
- [Rational](#) & [operator+=](#) (const [Rational](#) &other)  
*Add a different rational number to this rational number.*
- [Rational](#) & [operator-=](#) (const [Rational](#) &other)  
*Subtract a different rational number from this rational number.*
- [Rational](#) & [operator\\*=](#) (const [Rational](#) &other)  
*Multiply this rational number with a different rational number.*
- [Rational](#) & [operator/=](#) (const [Rational](#) &other)  
*Divide this rational number by a different rational number.*
- [Rational](#) [operator-](#) () const  
*Return the additive inverse of this rational number.*
- [bool](#) [operator==](#) (const [Rational](#) r)  
*Test this rational number for equality with another rational number.*
- [bool](#) [operator!=](#) (const [Rational](#) r)  
*Test this rational number for difference with another rational number.*
- void [write](#) (std::ostream &os) const  
*Write this rational number to a stream.*
- void [simplify](#) ()  
*Simplify the internal fractional expression of this rational number.*

### 29.400.1 Detailed Description

Class representing rational numbers

Definition at line 53 of file Rational.h.

### 29.400.2 Constructor & Destructor Documentation

#### 29.400.2.1 [Go::Rational::Rational](#) ( ) [inline]

Construct a rational number whose value is 0.

Definition at line 58 of file Rational.h.

#### 29.400.2.2 [Go::Rational::Rational](#) ( int p ) [inline]

Construct a rational number whose value is 'p' (integer)

Definition at line 61 of file Rational.h.

**29.400.2.3** `Go::Rational::Rational ( int p, int q ) [inline]`

Construct a rational number whose value is 'p/q' (fraction)

Definition at line 64 of file Rational.h.

### 29.400.3 Member Function Documentation

**29.400.3.1** `bool Go::Rational::operator!=( const Rational r ) [inline]`

Test this rational number for difference with another rational number.

Definition at line 114 of file Rational.h.

**29.400.3.2** `Rational& Go::Rational::operator*=( const Rational & other ) [inline]`

Multiply this rational number with a different rational number.

Definition at line 84 of file Rational.h.

**29.400.3.3** `Rational& Go::Rational::operator+=( const Rational & other ) [inline]`

Add a different rational number to this rational number.

Definition at line 67 of file Rational.h.

**29.400.3.4** `Rational Go::Rational::operator-( ) const [inline]`

Return the additive inverse of this rational number.

Definition at line 102 of file Rational.h.

**29.400.3.5** `Rational& Go::Rational::operator-=( const Rational & other ) [inline]`

Subtract a different rational number from this rational number.

Definition at line 76 of file Rational.h.

**29.400.3.6** `Rational& Go::Rational::operator/=( const Rational & other ) [inline]`

Divide this rational number by a different rational number.

Definition at line 93 of file Rational.h.

29.400.3.7 `bool Go::Rational::operator==( const Rational r ) [inline]`

Test this rational number for equality with another rational number.

Definition at line 108 of file Rational.h.

29.400.3.8 `void Go::Rational::simplify( ) [inline]`

Simplify the internal fractional expression of this rational number.

Definition at line 126 of file Rational.h.

29.400.3.9 `void Go::Rational::write( std::ostream & os ) const [inline]`

Write this rational number to a stream.

Definition at line 120 of file Rational.h.

The documentation for this class was generated from the following file:

- [gootools-core/include/GoTools/utils/Rational.h](#)

## 29.401 Go::raw\_pointer\_comp< T > Struct Template Reference

```
#include <IntersectionPoolUtils.h>
```

### Public Member Functions

- [bool operator\(\)](#) (shared\_ptr< T > A, shared\_ptr< T > B)

### 29.401.1 Detailed Description

```
template<class T>
struct Go::raw_pointer_comp< T >
```

Definition at line 116 of file IntersectionPoolUtils.h.

### 29.401.2 Member Function Documentation

29.401.2.1 `template<class T> bool Go::raw_pointer_comp< T >::operator()( shared_ptr< T > A, shared_ptr< T > B ) [inline]`

Definition at line 119 of file IntersectionPoolUtils.h.

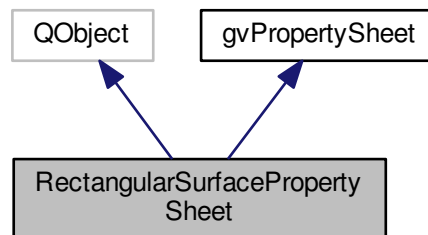
The documentation for this struct was generated from the following file:

- [intersections/include/GoTools/intersections/IntersectionPoolUtils.h](#)

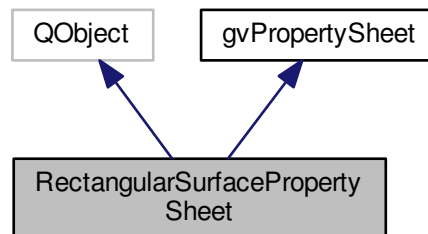
## 29.402 RectangularSurfacePropertySheet Class Reference

```
#include <RectangularSurfacePropertySheet.h>
```

Inheritance diagram for RectangularSurfacePropertySheet:



Collaboration diagram for RectangularSurfacePropertySheet:



### Public Slots

- void [apply](#) ()

### Public Member Functions

- [RectangularSurfacePropertySheet](#) ([Go::RectangularSurfaceTesselator](#) \*tess, [gvRectangularSurfacePaintable](#) \*pable, [shared\\_ptr< Go::ParamSurface >](#) &surf)
- virtual [~RectangularSurfacePropertySheet](#) ()
- virtual void [createSheet](#) ([QWidget](#) \*parent, [gvObserver](#) \*obs)

### 29.402.1 Detailed Description

Documentation ... etc

Definition at line 61 of file RectangularSurfacePropertySheet.h.

### 29.402.2 Constructor & Destructor Documentation

29.402.2.1 `RectangularSurfacePropertySheet::RectangularSurfacePropertySheet ( Go::RectangularSurfaceTesselator * tess, gvRectangularSurfacePaintable * pable, shared_ptr< Go::ParamSurface > & surf )`  
[inline]

Definition at line 67 of file RectangularSurfacePropertySheet.h.

29.402.2.2 `virtual RectangularSurfacePropertySheet::~~RectangularSurfacePropertySheet ( )` [virtual]

### 29.402.3 Member Function Documentation

29.402.3.1 `void RectangularSurfacePropertySheet::apply ( )` [slot]

29.402.3.2 `virtual void RectangularSurfacePropertySheet::createSheet ( QWidget * parent, gvObserver * obs )`  
[virtual]

Implements [gvPropertySheet](#).

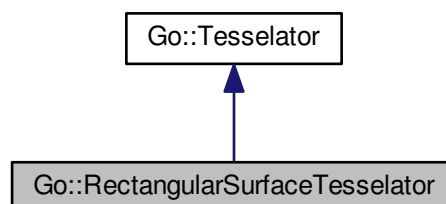
The documentation for this class was generated from the following file:

- [viewlib/include/GoTools/viewlib/RectangularSurfacePropertySheet.h](#)

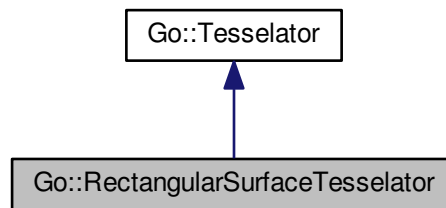
## 29.403 Go::RectangularSurfaceTesselator Class Reference

```
#include <RectangularSurfaceTesselator.h>
```

Inheritance diagram for Go::RectangularSurfaceTesselator:



Collaboration diagram for Go::RectangularSurfaceTesselator:



## Public Member Functions

- [RectangularSurfaceTesselator](#) (`const ParamSurface &surf`, `int ures=20`, `int vres=20`, `bool iso=false`, `int uiso=15`, `int viso=15`, `int isoires=300`)
- virtual `~RectangularSurfaceTesselator` ()  
*Destructor.*
- virtual void `tesselate` ()  
*Perform tessellation.*
- `shared_ptr< RegularMesh >` `getMesh` ()  
*Fetch the computed mesh.*
- `std::vector< LineStrip > &` `getIsolineStrips` ()  
*Fetch tessellation of iso parametric curves.*
- void `changeRes` (`int m`, `int n`)  
*Change mesh size.*
- void `getRes` (`int &m`, `int &n`)  
*Fetch mesh size.*
- void `changeIsolines` (`bool isolines`)  
*Retesselate iso parametric curves.*
- void `getIsolines` (`bool &isolines`)  
*Fetch mesh of iso parametric curves.*
- void `changeIsolineNumRes` (`int m`, `int n`, `int res`)  
*Change the number of iso parametric curves to compute.*
- void `getIsolineNumRes` (`int &m`, `int &n`, `int &res`)  
*Fetch the number of iso parametric curves to compute.*

### 29.403.1 Detailed Description

[RectangularSurfaceTesselator](#): create a mesh for a boundary trimmed parametric surface. Visualization purposes.

Definition at line 56 of file `RectangularSurfaceTesselator.h`.

## 29.403.2 Constructor & Destructor Documentation

29.403.2.1 `Go::RectangularSurfaceTesselator::RectangularSurfaceTesselator ( const ParamSurface & surf, int ures = 20, int vres = 20, bool iso = false, int uiso = 15, int viso = 15, int isoires = 300 ) [inline]`

Constructor. Surface and mesh size are given. The tesselator can be set to compute also iso parametric curves with a specified mesh size.

Definition at line 61 of file RectangularSurfaceTesselator.h.

29.403.2.2 `virtual Go::RectangularSurfaceTesselator::~~RectangularSurfaceTesselator ( ) [virtual]`

Destructor.

## 29.403.3 Member Function Documentation

29.403.3.1 `void Go::RectangularSurfaceTesselator::changelineNumRes ( int m, int n, int res ) [inline]`

Change the number of iso parametric curves to compute.

Definition at line 133 of file RectangularSurfaceTesselator.h.

29.403.3.2 `void Go::RectangularSurfaceTesselator::changelines ( bool isolines ) [inline]`

Retesselate iso parametric curves.

Definition at line 118 of file RectangularSurfaceTesselator.h.

29.403.3.3 `void Go::RectangularSurfaceTesselator::changeRes ( int m, int n ) [inline]`

Change mesh size.

Definition at line 100 of file RectangularSurfaceTesselator.h.

29.403.3.4 `void Go::RectangularSurfaceTesselator::getIsolineNumRes ( int & m, int & n, int & res ) [inline]`

Fetch the number of iso parametric curves to compute.

Definition at line 141 of file RectangularSurfaceTesselator.h.

29.403.3.5 `void Go::RectangularSurfaceTesselator::getIsolines ( bool & isolines ) [inline]`

Fetch mesh of iso parametric curves.

Definition at line 127 of file RectangularSurfaceTesselator.h.

29.403.3.6 `std::vector<LineStrip>& Go::RectangularSurfaceTesselator::getIsoLineStrips ( ) [inline]`

Fetch tessellation of iso parametric curves.

Definition at line 86 of file RectangularSurfaceTesselator.h.

29.403.3.7 `shared_ptr<RegularMesh> Go::RectangularSurfaceTesselator::getMesh ( ) [inline]`

Fetch the computed mesh.

Definition at line 80 of file RectangularSurfaceTesselator.h.

29.403.3.8 `void Go::RectangularSurfaceTesselator::getRes ( int & m, int & n ) [inline]`

Fetch mesh size.

Definition at line 111 of file RectangularSurfaceTesselator.h.

29.403.3.9 `virtual void Go::RectangularSurfaceTesselator::tesselate ( ) [virtual]`

Perform tessellation.

Implements [Go::Tesselator](#).

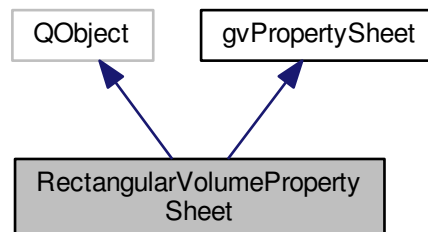
The documentation for this class was generated from the following file:

- [gotools-core/include/GoTools/tesselator/RectangularSurfaceTesselator.h](#)

## 29.404 RectangularVolumePropertySheet Class Reference

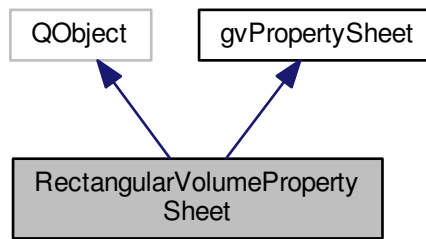
```
#include <RectangularVolumePropertySheet.h>
```

Inheritance diagram for RectangularVolumePropertySheet:





Collaboration diagram for RectangularVolumePropertySheet:



### Public Slots

- void [apply](#) ()

### Public Member Functions

- [RectangularVolumePropertySheet](#) ([Go::RectangularVolumeTesselator](#) \*tess, [gvRectangularVolumePaintable](#) \*pable, [shared\\_ptr](#)< [Go::ParamVolume](#) > &vol)
- virtual [~RectangularVolumePropertySheet](#) ()
- virtual void [createSheet](#) ([QWidget](#) \*parent, [gvObserver](#) \*obs)

#### 29.404.1 Detailed Description

Documentation ... etc

Definition at line 64 of file [RectangularVolumePropertySheet.h](#).

#### 29.404.2 Constructor & Destructor Documentation

29.404.2.1 [RectangularVolumePropertySheet::RectangularVolumePropertySheet](#) ( [Go::RectangularVolumeTesselator](#) \* tess, [gvRectangularVolumePaintable](#) \* pable, [shared\\_ptr](#)< [Go::ParamVolume](#) > & vol ) [inline]

Definition at line 70 of file [RectangularVolumePropertySheet.h](#).

29.404.2.2 virtual [RectangularVolumePropertySheet::~~RectangularVolumePropertySheet](#) ( ) [virtual]

#### 29.404.3 Member Function Documentation

29.404.3.1 void [RectangularVolumePropertySheet::apply](#) ( ) [slot]

29.404.3.2 virtual void [RectangularVolumePropertySheet::createSheet](#) ( [QWidget](#) \* parent, [gvObserver](#) \* obs ) [virtual]

Implements [gvPropertySheet](#).

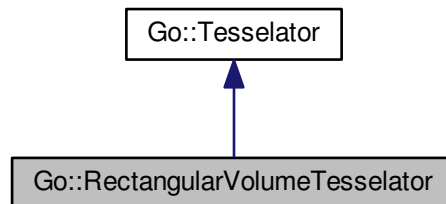
The documentation for this class was generated from the following file:

- [viewlib/include/GoTools/viewlib/vol\\_and\\_lr/RectangularVolumePropertySheet.h](#)

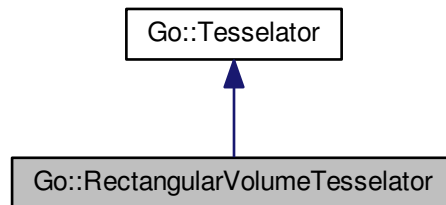
## 29.405 Go::RectangularVolumeTesselator Class Reference

```
#include <RectangularVolumeTesselator.h>
```

Inheritance diagram for Go::RectangularVolumeTesselator:



Collaboration diagram for Go::RectangularVolumeTesselator:



### Public Member Functions

- [RectangularVolumeTesselator](#) ([const ParamVolume](#) &vol, int res=50)
- virtual [~RectangularVolumeTesselator](#) ()  
*Destructor.*
- virtual void [tesselate](#) ()  
*Perform tessellation.*
- [shared\\_ptr< RegularVolMesh >](#) [getMesh](#) ()  
*Fetch the computed mesh.*
- void [changeRes](#) (int m)  
*Change mesh size.*
- void [getRes](#) (int &m)  
*Fetch mesh size.*

### 29.405.1 Detailed Description

[RectangularSurfaceTesselator](#): create a mesh for a boundary trimmed parametric surface. Visualization purposes.

Definition at line 59 of file RectangularVolumeTesselator.h.

### 29.405.2 Constructor & Destructor Documentation

29.405.2.1 `Go::RectangularVolumeTesselator::RectangularVolumeTesselator ( const ParamVolume & vol, int res = 50 )` `[inline]`

Constructor. Volume and mesh size are given. The tessellator can be set to compute also iso parametric curves with a specified mesh size.

Definition at line 64 of file RectangularVolumeTesselator.h.

29.405.2.2 `virtual Go::RectangularVolumeTesselator::~~RectangularVolumeTesselator ( )` `[virtual]`

Destructor.

### 29.405.3 Member Function Documentation

29.405.3.1 `void Go::RectangularVolumeTesselator::changeRes ( int m )` `[inline]`

Change mesh size.

Definition at line 102 of file RectangularVolumeTesselator.h.

29.405.3.2 `shared_ptr<RegularVolMesh> Go::RectangularVolumeTesselator::getMesh ( )` `[inline]`

Fetch the computed mesh.

Definition at line 88 of file RectangularVolumeTesselator.h.

29.405.3.3 `void Go::RectangularVolumeTesselator::getRes ( int & m )` `[inline]`

Fetch mesh size.

Definition at line 109 of file RectangularVolumeTesselator.h.

29.405.3.4 `virtual void Go::RectangularVolumeTesselator::tessellate ( )` `[virtual]`

Perform tessellation.

Implements [Go::Tesselator](#).

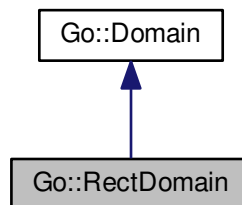
The documentation for this class was generated from the following file:

- [trivariate/include/GoTools/trivariate/RectangularVolumeTesselator.h](#)

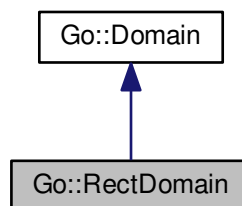
## 29.406 Go::RectDomain Class Reference

```
#include <RectDomain.h>
```

Inheritance diagram for Go::RectDomain:



Collaboration diagram for Go::RectDomain:



### Public Member Functions

- [RectDomain \(\)](#)  
*Constructs an uninitialized domain.*
- [RectDomain \(const Array< double, 2 > &corner1, const Array< double, 2 > &corner2\)](#)
- virtual [~RectDomain \(\)](#)  
*Virtual destructor, enables safe inheritance.*
- virtual [bool isInDomain \(const Array< double, 2 > &point, double tolerance\) const](#)
- virtual [int isInDomain2 \(const Array< double, 2 > &point, double tolerance\) const](#)
- virtual [bool isOnBoundary \(const Array< double, 2 > &point, double tolerance\) const](#)
- [bool isOnCorner \(const Array< double, 2 > &point, double tolerance\) const](#)
- [int whichBoundary \(const Array< double, 2 > &point1, const Array< double, 2 > &point2, double tolerance\) const](#)
- virtual [void closestInDomain \(const Array< double, 2 > &point, Array< double, 2 > &clo\\_pt, double tolerance\) const](#)

- virtual void `closestOnBoundary` (const Array< double, 2 > &point, Array< double, 2 > &clo\_bd\_pt, double tolerance) const
- void `addUnionWith` (const RectDomain &rd)
- void `intersectWith` (const RectDomain &rd)
- double `umin` () const
- double `umax` () const
- double `vmin` () const
- double `vmax` () const
- double `diagLength` ()  
*Length of diagonal.*
- Array< double, 2 > `lowerLeft` () const
- Array< double, 2 > `upperRight` () const

### 29.406.1 Detailed Description

Represents a rectangular parameter domain

Definition at line 54 of file RectDomain.h.

### 29.406.2 Constructor & Destructor Documentation

#### 29.406.2.1 Go::RectDomain::RectDomain ( ) [inline]

Constructs an uninitialized domain.

Definition at line 58 of file RectDomain.h.

#### 29.406.2.2 Go::RectDomain::RectDomain ( const Array< double, 2 > &corner1, const Array< double, 2 > &corner2 )

Constructs and defines a rectangular domain by two of its opposing corners.

##### Parameters

<i>corner1</i>	the first corner
<i>corner2</i>	the opposing corner to 'corner1'

#### 29.406.2.3 virtual Go::RectDomain::~~RectDomain ( ) [virtual]

Virtual destructor, enables safe inheritance.

### 29.406.3 Member Function Documentation

#### 29.406.3.1 void Go::RectDomain::addUnionWith ( const RectDomain & rd )

Expand the [RectDomain](#) just enough to cover the [RectDomain](#) given as argument.

## Parameters

<i>rd</i>	the <a href="#">RectDomain</a> that we want to be covered by 'this' <a href="#">RectDomain</a> .
-----------	--------------------------------------------------------------------------------------------------

29.406.3.2 `virtual void Go::RectDomain::closestInDomain ( const Array< double, 2 > & point, Array< double, 2 > & clo_pt, double tolerance ) const [virtual]`

Find the (u, v) point in the [Domain](#) that is closest (using Euclidean distance in  $R^2$ ) to a given (u, v) point. If the given point is in the domain, then the answer is obviously the same point.

Implements [Go::Domain](#).

29.406.3.3 `virtual void Go::RectDomain::closestOnBoundary ( const Array< double, 2 > & point, Array< double, 2 > & clo_bd_pt, double tolerance ) const [virtual]`

Find the (u, v) point on the boundary of the [Domain](#) that is closest (using Euclidean distance in  $R^2$ ) to a given (u, v) point. If the point is already considered *on* the boundary, then the answer is obviously the same point.

Implements [Go::Domain](#).

29.406.3.4 `double Go::RectDomain::diagLength ( ) [inline]`

Length of diagonal.

Definition at line 138 of file RectDomain.h.

29.406.3.5 `void Go::RectDomain::intersectWith ( const RectDomain & rd )`

Set 'this' [RectDomain](#) to be the intersection of its current extent and that of the argument [RectDomain](#).

## Parameters

<i>rd</i>	the <a href="#">RectDomain</a> that we want to intersect with 'this' one.
-----------	---------------------------------------------------------------------------

29.406.3.6 `virtual bool Go::RectDomain::isInDomain ( const Array< double, 2 > & point, double tolerance ) const [virtual]`

check whether a given parameter pair is located inside the domain

## Parameters

<i>point</i>	the (u,v)-pair that we want to test.
<i>tolerance</i>	the tolerance used (ruling what to do when 'point' is located very near the edge of the domain).

**Returns**

'true' if 'point' is inside the domain, or within 'tolerance' from being inside the domain, 'false' otherwise'.

Implements [Go::Domain](#).

```
29.406.3.7 virtual int Go::RectDomain::isInDomain2 (const Array< double, 2 > & point, double tolerance) const
 [virtual]
```

Query whether a given parameter pair is inside the domain or not.

**Parameters**

<i>point</i>	array containing the parameter pair
<i>tolerance</i>	the tolerance to be used. In order to be considered 'inside', the point must be located inside one of the defining <a href="#">CurveLoop</a> s, as well as being at a distance more than 'tolerance' from any point on that <a href="#">CurveLoop</a> .

**Returns**

'1' if the point is found to be inside the domain, '2' if it is found to be on the boundary '0' otherwise.

Implements [Go::Domain](#).

```
29.406.3.8 virtual bool Go::RectDomain::isOnBoundary (const Array< double, 2 > & point, double tolerance) const
 [virtual]
```

check whether a given parameter pair is located on the domain boundary

**Parameters**

<i>point</i>	the (u,v)-pair that we want to test
<i>tolerance</i>	the tolerance used (how 'far' from the boundary our (u,v) pair can be and still be considered 'on' the boundary.

**Returns**

'true' if the point is considered to be on the boundary (within 'tolerance', 'false' otherwise).

Implements [Go::Domain](#).

```
29.406.3.9 bool Go::RectDomain::isOnCorner (const Array< double, 2 > & point, double tolerance) const
```

Check if a given parameter pair lies on a corner in the domain within the given tolerance

29.406.3.10 **Array<double, 2> Go::RectDomain::lowerLeft ( ) const** [inline]

Get the 'lower left' corner of this [RectDomain](#).

**Returns**

a 2D array containing the 'lower left' corner of this [RectDomain](#)

Definition at line 145 of file RectDomain.h.

29.406.3.11 **double Go::RectDomain::umax ( ) const** [inline]

Get the [RectDomain](#)'s largest value for the first parameter

**Returns**

the [RectDomain](#)'s largest value for the first parameter

Definition at line 127 of file RectDomain.h.

29.406.3.12 **double Go::RectDomain::umin ( ) const** [inline]

Get the [RectDomain](#)'s smallest value for the first parameter

**Returns**

the [RectDomain](#)'s smallest value for the first parameter

Definition at line 123 of file RectDomain.h.

29.406.3.13 **Array<double, 2> Go::RectDomain::upperRight ( ) const** [inline]

Get the 'upper right' corner of this [RectDomain](#)

**Returns**

a 2D array containing the 'upper right' corner of this [RectDomain](#)

Definition at line 149 of file RectDomain.h.

29.406.3.14 **double Go::RectDomain::vmax ( ) const** [inline]

Get the [RectDomain](#)'s largest value for the second parameter

**Returns**

the [RectDomain](#)'s largest value for the second parameter

Definition at line 135 of file RectDomain.h.



29.406.3.15 `double Go::RectDomain::vmin ( ) const [inline]`

Get the [RectDomain](#)'s smallest value for the second parameter

#### Returns

the [RectDomain](#)'s smallest value for the second parameter

Definition at line 131 of file [RectDomain.h](#).

29.406.3.16 `int Go::RectDomain::whichBoundary ( const Array< double, 2 > & point1, const Array< double, 2 > & point2, double tolerance ) const`

Given two parameter pairs, check if they specify a domain boundary return value: -1=no boundary, 0=u<sub>min</sub>, 1=u<sub>max</sub>, 2=v<sub>min</sub>, 3=v<sub>max</sub>

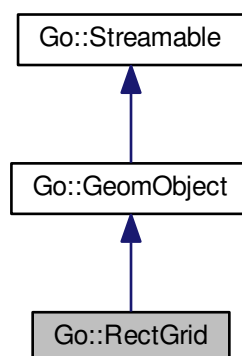
The documentation for this class was generated from the following file:

- [gotools-core/include/GoTools/geometry/RectDomain.h](#)

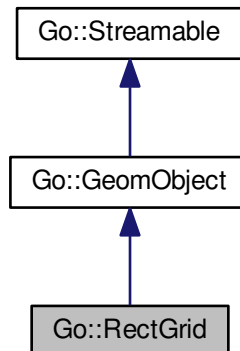
## 29.407 Go::RectGrid Class Reference

```
#include <RectGrid.h>
```

Inheritance diagram for [Go::RectGrid](#):



Collaboration diagram for Go::RectGrid:



## Public Member Functions

- [RectGrid](#) ()
- [RectGrid](#) (int numu, int numv, int dim, double \*pts)
- virtual [~RectGrid](#) ()  
*virtual destructor assures safe inheritacne*
- virtual [BoundingBox boundingBox](#) () const  
*Return the object's bounding box.*
- virtual int [dimension](#) () const  
*Return the dimension of the space in which the object lies (usually 2 or 3)*
- virtual [ClassType instanceType](#) () const  
*Return the class type identifier of a given, derived instance of [GeomObject](#).*
- virtual [GeomObject \\* clone](#) () const  
*Clone the [GeomObject](#) and return a pointer to the clone.*
- void [read](#) (std::istream &is)
- void [write](#) (std::ostream &os) const
- void [setGrid](#) (int numu, int numv, int dim, const double \*pts)
- int [numCoefs\\_u](#) () const
- int [numCoefs\\_v](#) () const
- double \* [rawData](#) ()
- const double \* [rawData](#) () const
- void [swapDirections](#) ()  
*Swap grid directions.*

## Static Public Member Functions

- static [ClassType classType](#) ()

### 29.407.1 Detailed Description

This class represent the simplest of all quadrangulations: a rectangular grid.

Definition at line 50 of file RectGrid.h.

### 29.407.2 Constructor & Destructor Documentation

#### 29.407.2.1 Go::RectGrid::RectGrid ( ) `[inline]`

Constructor that defines an empty grid, which can only be assigned or [read\(\)](#) into.

Definition at line 56 of file RectGrid.h.

#### 29.407.2.2 Go::RectGrid::RectGrid ( int *numu*, int *numv*, int *dim*, double \* *pts* ) `[inline]`

Constructor explicitly specifying a [RectGrid](#)

##### Parameters

<i>numu</i>	number of points along the first grid direction
<i>numv</i>	number of points along the second grid direction
<i>dim</i>	dimension of the points (usually 2 or 3)
<i>pts</i>	pointer to the array where the points are stored. The points should be stored so that the first grid has lowest stride (ie., coefficients are stored (u1v1, u2v1,... unv1, u1v2, u2v2, ...unv2,... unvm)

Definition at line 69 of file RectGrid.h.

#### 29.407.2.3 virtual Go::RectGrid::~~RectGrid ( ) `[virtual]`

virtual destructor assures safe inheritacne

### 29.407.3 Member Function Documentation

#### 29.407.3.1 virtual BoundingBox Go::RectGrid::boundingBox ( ) const `[virtual]`

Return the object's bounding box.

Implements [Go::GeomObject](#).

#### 29.407.3.2 static ClassType Go::RectGrid::classType ( ) `[inline], [static]`

Definition at line 87 of file RectGrid.h.

**29.407.3.3** `virtual GeomObject* Go::RectGrid::clone ( ) const [inline],[virtual]`

Clone the [GeomObject](#) and return a pointer to the clone.

Implements [Go::GeomObject](#).

Definition at line 93 of file RectGrid.h.

**29.407.3.4** `virtual int Go::RectGrid::dimension ( ) const [virtual]`

Return the dimension of the space in which the object lies (usually 2 or 3)

Implements [Go::GeomObject](#).

**29.407.3.5** `virtual ClassType Go::RectGrid::instanceType ( ) const [virtual]`

Return the class type identifier of a given, derived instance of [GeomObject](#).

Implements [Go::GeomObject](#).

**29.407.3.6** `int Go::RectGrid::numCoefs_u ( ) const [inline]`

Get number of points along the first grid direction

**Returns**

the number of points along the first grid direction

Definition at line 123 of file RectGrid.h.

**29.407.3.7** `int Go::RectGrid::numCoefs_v ( ) const [inline]`

Get number of points along the second grid direction

**Returns**

the number of points along the second grid direction

Definition at line 130 of file RectGrid.h.

**29.407.3.8** `double* Go::RectGrid::rawData ( ) [inline]`

Get a pointer to the start of the array where grid point coordinates are stored.

**Returns**

a pointer to the storage array for grid points.

Definition at line 138 of file RectGrid.h.

29.407.3.9 `const double* Go::RectGrid::rawData ( ) const` `[inline]`

Get a const pointer to the start of the array where grid point coordinates are stored.

#### Returns

a const pointer to the storage array for grid points.

Definition at line 146 of file RectGrid.h.

29.407.3.10 `void Go::RectGrid::read ( std::istream & is )` `[virtual]`

read object from stream

#### Parameters

<i>is</i>	stream from which object is read
-----------	----------------------------------

Implements [Go::Streamable](#).

29.407.3.11 `void Go::RectGrid::setGrid ( int numu, int numv, int dim, const double * pts )` `[inline]`

Set grid according to the user-supplied data

#### Parameters

<i>numu</i>	number of points along the first grid direction
<i>numv</i>	number of points along the second grid direction
<i>dim</i>	dimension of the points (usually 2 or 3)
<i>pts</i>	pointer to the array where the points are stored. The points should be stored so that the first grid has lowest stride (ie., coefficients are stored (u1v1, u2v1,... unv1, u1v2, u2v2, ...unv2,... unvm)

Definition at line 112 of file RectGrid.h.

29.407.3.12 `void Go::RectGrid::swapDirections ( )`

Swap grid directions.

29.407.3.13 `void Go::RectGrid::write ( std::ostream & os ) const` `[virtual]`

write object to stream

#### Parameters

<i>os</i>	stream to which object is written
-----------	-----------------------------------

Implements [Go::Streamable](#).

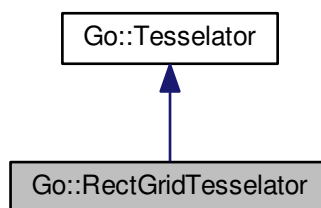
The documentation for this class was generated from the following file:

- [gotools-core/include/GoTools/geometry/RectGrid.h](#)

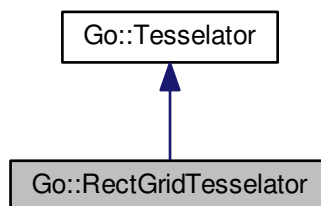
## 29.408 Go::RectGridTesselator Class Reference

```
#include <RectGridTesselator.h>
```

Inheritance diagram for Go::RectGridTesselator:



Collaboration diagram for Go::RectGridTesselator:



### Public Member Functions

- [RectGridTesselator](#) (`const RectGrid &rg`)  
*Constructor.*
- virtual [~RectGridTesselator](#) ()  
*Destructor.*
- virtual void [tesselate](#) ()  
*Perform tessellation.*
- `shared_ptr< QuadMesh >` [getMesh](#) ()  
*Fetch result.*

### 29.408.1 Detailed Description

Transfer the information in a [RectGrid](#) to a [QuadMesh](#)

Definition at line 55 of file RectGridTesselator.h.

### 29.408.2 Constructor & Destructor Documentation

#### 29.408.2.1 Go::RectGridTesselator::RectGridTesselator ( const RectGrid & rg )

Constructor.

#### 29.408.2.2 virtual Go::RectGridTesselator::~~RectGridTesselator ( ) [virtual]

Destructor.

### 29.408.3 Member Function Documentation

#### 29.408.3.1 shared\_ptr<QuadMesh> Go::RectGridTesselator::getMesh ( ) [inline]

Fetch result.

Definition at line 67 of file RectGridTesselator.h.

#### 29.408.3.2 virtual void Go::RectGridTesselator::tessellate ( ) [virtual]

Perform tessellation.

Implements [Go::Tesselator](#).

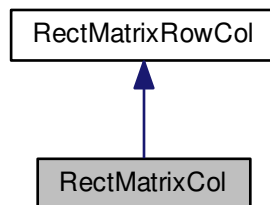
The documentation for this class was generated from the following file:

- [gotools-core/include/GoTools/tesselator/RectGridTesselator.h](#)

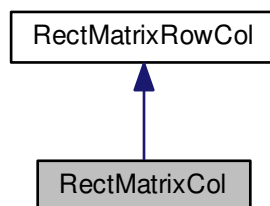
## 29.409 RectMatrixCol Class Reference

```
#include <newmatrm.h>
```

Inheritance diagram for RectMatrixCol:



Collaboration diagram for RectMatrixCol:



### Public Member Functions

- [RectMatrixCol](#) ([const Matrix &](#), [int](#), [int](#), [int](#))
- [RectMatrixCol](#) ([const Matrix &](#), [int](#))
- void [Reset](#) ([const Matrix &](#), [int](#), [int](#), [int](#))
- void [Reset](#) ([const Matrix &](#), [int](#))
- void [Down](#) ()
- void [Right](#) ()
- void [Up](#) ()
- void [Left](#) ()

### Friends

- void [ComplexScale](#) ([RectMatrixCol &](#), [RectMatrixCol &](#), [Real](#), [Real](#))
- void [Rotate](#) ([RectMatrixCol &](#), [RectMatrixCol &](#), [Real](#), [Real](#))



## Additional Inherited Members

### 29.409.1 Detailed Description

Definition at line 63 of file newmatrm.h.

### 29.409.2 Constructor & Destructor Documentation

29.409.2.1 `RectMatrixCol::RectMatrixCol ( const Matrix & M, int skip, int col, int length )` [inline]

Definition at line 99 of file newmatrm.h.

29.409.2.2 `RectMatrixCol::RectMatrixCol ( const Matrix & M, int col )` [inline]

Definition at line 102 of file newmatrm.h.

### 29.409.3 Member Function Documentation

29.409.3.1 `void RectMatrixCol::Down ( )` [inline]

Definition at line 70 of file newmatrm.h.

29.409.3.2 `void RectMatrixCol::Left ( )` [inline]

Definition at line 73 of file newmatrm.h.

29.409.3.3 `void RectMatrixCol::Reset ( const Matrix & M, int skip, int col, int length )`

Definition at line 37 of file newmatrm.cpp.

29.409.3.4 `void RectMatrixCol::Reset ( const Matrix & M, int col )`

Definition at line 44 of file newmatrm.cpp.

29.409.3.5 `void RectMatrixCol::Right ( )` [inline]

Definition at line 71 of file newmatrm.h.

29.409.3.6 `void RectMatrixCol::Up ( )` [inline]

Definition at line 72 of file newmatrm.h.

### 29.409.4 Friends And Related Function Documentation

29.409.4.1 void ComplexScale ( RectMatrixCol & U, RectMatrixCol & V, Real x, Real y ) [friend]

Definition at line 131 of file newmatrm.cpp.

29.409.4.2 void Rotate ( RectMatrixCol & U, RectMatrixCol & V, Real tau, Real s ) [friend]

Definition at line 155 of file newmatrm.cpp.

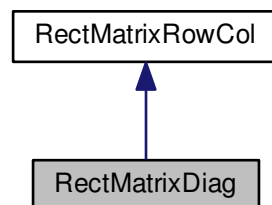
The documentation for this class was generated from the following files:

- [newmat/include/newmatrm.h](#)
- [newmat/src/newmatrm.cpp](#)

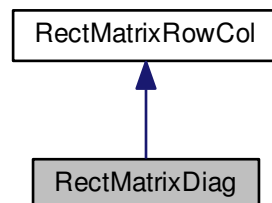
## 29.410 RectMatrixDiag Class Reference

```
#include <newmatrm.h>
```

Inheritance diagram for RectMatrixDiag:



Collaboration diagram for RectMatrixDiag:



## Public Member Functions

- [RectMatrixDiag](#) (`const DiagonalMatrix &D`)
- `Real & operator[]` (`int i`)
- `void DownDiag` ()
- `void UpDiag` ()

## Additional Inherited Members

### 29.410.1 Detailed Description

Definition at line 79 of file `newmatrm.h`.

### 29.410.2 Constructor & Destructor Documentation

29.410.2.1 `RectMatrixDiag::RectMatrixDiag ( const DiagonalMatrix & D )` [`inline`]

Definition at line 82 of file `newmatrm.h`.

### 29.410.3 Member Function Documentation

29.410.3.1 `void RectMatrixDiag::DownDiag ( )` [`inline`]

Definition at line 85 of file `newmatrm.h`.

29.410.3.2 `Real& RectMatrixDiag::operator[] ( int i )` [`inline`]

Definition at line 84 of file `newmatrm.h`.

29.410.3.3 `void RectMatrixDiag::UpDiag ( )` [`inline`]

Definition at line 86 of file `newmatrm.h`.

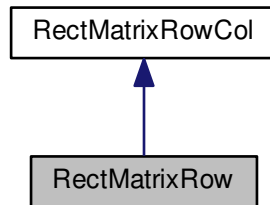
The documentation for this class was generated from the following file:

- `newmat/include/newmatrm.h`

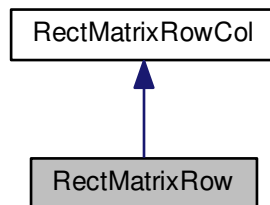
## 29.411 RectMatrixRow Class Reference

```
#include <newmatrm.h>
```

Inheritance diagram for RectMatrixRow:



Collaboration diagram for RectMatrixRow:



### Public Member Functions

- [RectMatrixRow](#) ([const](#) Matrix &, int, int, int)
- [RectMatrixRow](#) ([const](#) Matrix &, int)
- void [Reset](#) ([const](#) Matrix &, int, int, int)
- void [Reset](#) ([const](#) Matrix &, int)
- Real & [operator\[\]](#) (int i)
- void [Down](#) ()
- void [Right](#) ()
- void [Up](#) ()
- void [Left](#) ()

### Additional Inherited Members

#### 29.411.1 Detailed Description

Definition at line 48 of file newmatrm.h.

## 29.411.2 Constructor & Destructor Documentation

29.411.2.1 `RectMatrixRow::RectMatrixRow ( const Matrix & M, int row, int skip, int length )` `[inline]`

Definition at line 92 of file `newmatrm.h`.

29.411.2.2 `RectMatrixRow::RectMatrixRow ( const Matrix & M, int row )` `[inline]`

Definition at line 95 of file `newmatrm.h`.

## 29.411.3 Member Function Documentation

29.411.3.1 `void RectMatrixRow::Down ( )` `[inline]`

Definition at line 56 of file `newmatrm.h`.

29.411.3.2 `void RectMatrixRow::Left ( )` `[inline]`

Definition at line 59 of file `newmatrm.h`.

29.411.3.3 `Real& RectMatrixRow::operator[] ( int i )` `[inline]`

Definition at line 55 of file `newmatrm.h`.

29.411.3.4 `void RectMatrixRow::Reset ( const Matrix & M, int row, int skip, int length )`

Definition at line 24 of file `newmatrm.cpp`.

29.411.3.5 `void RectMatrixRow::Reset ( const Matrix & M, int row )`

Definition at line 31 of file `newmatrm.cpp`.

29.411.3.6 `void RectMatrixRow::Right ( )` `[inline]`

Definition at line 57 of file `newmatrm.h`.

29.411.3.7 `void RectMatrixRow::Up ( )` `[inline]`

Definition at line 58 of file `newmatrm.h`.

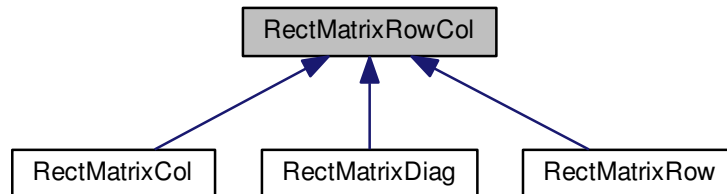
The documentation for this class was generated from the following files:

- [newmat/include/newmatrm.h](#)
- [newmat/src/newmatrm.cpp](#)

## 29.412 RectMatrixRowCol Class Reference

```
#include <newmatrm.h>
```

Inheritance diagram for RectMatrixRowCol:



### Public Member Functions

- Real [operator\\*](#) (const RectMatrixRowCol &) const
- void [AddScaled](#) (const RectMatrixRowCol &, Real)
- void [Divide](#) (const RectMatrixRowCol &, Real)
- void [Divide](#) (Real)
- void [Negate](#) ()
- void [Zero](#) ()
- Real & [operator\[\]](#) (int i)
- Real [SumSquare](#) () const
- Real & [First](#) ()
- void [DownDiag](#) ()
- void [UpDiag](#) ()

### Protected Member Functions

- [RectMatrixRowCol](#) (Real \*st, int nx, int sp, int sh)
- void [Reset](#) (Real \*st, int nx, int sp, int sh)

### Protected Attributes

- Real \* [store](#)
- int [n](#)
- int [spacing](#)
- int [shift](#)

### Friends

- void [ComplexScale](#) (RectMatrixCol &, RectMatrixCol &, Real, Real)
- void [Rotate](#) (RectMatrixCol &, RectMatrixCol &, Real, Real)

### 29.412.1 Detailed Description

Definition at line 16 of file newmatrm.h.

### 29.412.2 Constructor & Destructor Documentation

29.412.2.1 `RectMatrixRowCol::RectMatrixRowCol ( Real * st, int nx, int sp, int sh )` `[inline]`, `[protected]`

Definition at line 27 of file newmatrm.h.

### 29.412.3 Member Function Documentation

29.412.3.1 `void RectMatrixRowCol::AddScaled ( const RectMatrixRowCol & rmrc, Real r )`

Definition at line 78 of file newmatrm.cpp.

29.412.3.2 `void RectMatrixRowCol::Divide ( const RectMatrixRowCol & rmrc, Real r )`

Definition at line 93 of file newmatrm.cpp.

29.412.3.3 `void RectMatrixRowCol::Divide ( Real r )`

Definition at line 107 of file newmatrm.cpp.

29.412.3.4 `void RectMatrixRowCol::DownDiag ( )` `[inline]`

Definition at line 41 of file newmatrm.h.

29.412.3.5 `Real& RectMatrixRowCol::First ( )` `[inline]`

Definition at line 40 of file newmatrm.h.

29.412.3.6 `void RectMatrixRowCol::Negate ( )`

Definition at line 115 of file newmatrm.cpp.

29.412.3.7 `Real RectMatrixRowCol::operator* ( const RectMatrixRowCol & rmrc ) const`

Definition at line 61 of file newmatrm.cpp.

29.412.3.8 `Real& RectMatrixRowCol::operator[] ( int i )` `[inline]`

Definition at line 38 of file `newmatrm.h`.

29.412.3.9 `void RectMatrixRowCol::Reset ( Real * st, int nx, int sp, int sh )` `[inline]`, `[protected]`

Definition at line 29 of file `newmatrm.h`.

29.412.3.10 `Real RectMatrixRowCol::SumSquare ( )` `const`

Definition at line 51 of file `newmatrm.cpp`.

29.412.3.11 `void RectMatrixRowCol::UpDiag ( )` `[inline]`

Definition at line 42 of file `newmatrm.h`.

29.412.3.12 `void RectMatrixRowCol::Zero ( )`

Definition at line 123 of file `newmatrm.cpp`.

## 29.412.4 Friends And Related Function Documentation

29.412.4.1 `void ComplexScale ( RectMatrixCol & U, RectMatrixCol & V, Real x, Real y )` `[friend]`

Definition at line 131 of file `newmatrm.cpp`.

29.412.4.2 `void Rotate ( RectMatrixCol & U, RectMatrixCol & V, Real tau, Real s )` `[friend]`

Definition at line 155 of file `newmatrm.cpp`.

## 29.412.5 Member Data Documentation

29.412.5.1 `int RectMatrixRowCol::n` `[protected]`

Definition at line 24 of file `newmatrm.h`.

29.412.5.2 `int RectMatrixRowCol::shift` `[protected]`

Definition at line 26 of file `newmatrm.h`.



29.412.5.3 `int RectMatrixRowCol::spacing` [protected]

Definition at line 25 of file `newmatrm.h`.

29.412.5.4 `Real* RectMatrixRowCol::store` [protected]

Definition at line 23 of file `newmatrm.h`.

The documentation for this class was generated from the following files:

- `newmat/include/newmatrm.h`
- `newmat/src/newmatrm.cpp`

## 29.413 Go::LRSplineSurface::Refinement2D Struct Reference

```
#include <LRSplineSurface.h>
```

### Public Member Functions

- void `setVal` (`double val`, `double st`, `double e`, `Direction2D dir`, `int mult`)

### Public Attributes

- `double kval`
- `double start`
- `double end`
- `Direction2D d`
- `int multiplicity`

### 29.413.1 Detailed Description

Definition at line 71 of file `LRSplineSurface.h`.

### 29.413.2 Member Function Documentation

29.413.2.1 `void Go::LRSplineSurface::Refinement2D::setVal ( double val, double st, double e, Direction2D dir, int mult )` [inline]

Definition at line 78 of file `LRSplineSurface.h`.

### 29.413.3 Member Data Documentation

29.413.3.1 `Direction2D Go::LRSplineSurface::Refinement2D::d`

Definition at line 75 of file `LRSplineSurface.h`.

29.413.3.2 **double** Go::LRSplineSurface::Refinement2D::end

Definition at line 74 of file LRSplineSurface.h.

29.413.3.3 **double** Go::LRSplineSurface::Refinement2D::kval

Definition at line 72 of file LRSplineSurface.h.

29.413.3.4 **int** Go::LRSplineSurface::Refinement2D::multiplicity

Definition at line 76 of file LRSplineSurface.h.

29.413.3.5 **double** Go::LRSplineSurface::Refinement2D::start

Definition at line 73 of file LRSplineSurface.h.

The documentation for this struct was generated from the following file:

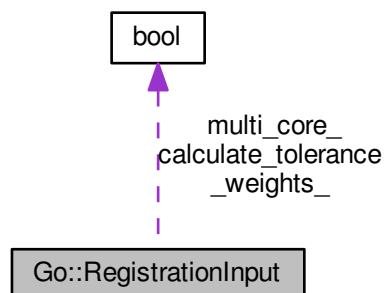
- [Irsplines2D/include/GoTools/Irsplines2D/LRSplineSurface.h](#)

## 29.414 Go::RegistrationInput Struct Reference

Struct for input to registration, either raw, fine or combined.

```
#include <RegistrationUtils.h>
```

Collaboration diagram for Go::RegistrationInput:



## Public Member Functions

- [RegistrationInput](#) ()
- void [setToleranceWeights](#) (double w\_rotation, double w\_translation, double w\_rescaling)  
*Set the tolerance weights used in Newtons method.*

## Public Attributes

- double [area\\_tolerance\\_sq\\_](#)
- int [max\\_newton\\_iterations\\_](#)  
*Maximum number of iterations in newton approach method.*
- double [newton\\_tolerance\\_](#)  
*Upper limit for when weighed sum of squares of changes in Newton's method requires us to break out of the loop.*
- bool [calculate\\_tolerance\\_weights\\_](#)
- double [tolerance\\_weight\\_rotation\\_](#)  
*The weight of the square of the change in the rotation vector, used in the Newtons method.*
- double [tolerance\\_weight\\_translation\\_](#)  
*The weight of the square of the change in the translation vector, used in the Newtons method.*
- double [tolerance\\_weight\\_rescale\\_](#)  
*The weight of the square of the change in the rescaling, used in the Newtons method.*
- int [max\\_solve\\_iterations\\_](#)  
*The maximum number of iterations used when solving the linear system in each iteration in Newtons method.*
- double [solve\\_tolerance\\_](#)  
*The tolerance used when solving the linear system in each iteration in Newtons method.*
- bool [multi\\_core\\_](#)  
*If true, and if OPENMP is included, run the fine registration in multicore.*

### 29.414.1 Detailed Description

Struct for input to registration, either raw, fine or combined.

Definition at line 60 of file RegistrationUtils.h.

### 29.414.2 Constructor & Destructor Documentation

29.414.2.1 `Go::RegistrationInput::RegistrationInput( )` `[inline]`

Definition at line 63 of file RegistrationUtils.h.

### 29.414.3 Member Function Documentation

29.414.3.1 `void Go::RegistrationInput::setToleranceWeights( double w_rotation, double w_translation, double w_rescaling )` `[inline]`

Set the tolerance weights used in Newtons method.

Definition at line 113 of file RegistrationUtils.h.

#### 29.414.4 Member Data Documentation

##### 29.414.4.1 `double Go::RegistrationInput::area_tolerance_sq_`

The lower limit of the square of the sine value of the smallest angle in a triangle, to accept the triangle as good enough for a raw registration

Definition at line 80 of file RegistrationUtils.h.

##### 29.414.4.2 `bool Go::RegistrationInput::calculate_tolerance_weights_`

If true, the result of the first iteration of Newtons method will set the weights for the calculation of the squares of changes used to check if we have reached a satisfactory result and should break out of Newtons method. The weights will be set relatively according to the changes, with 1.0 as the weight for the translation vector

Definition at line 92 of file RegistrationUtils.h.

##### 29.414.4.3 `int Go::RegistrationInput::max_newton_iterations_`

Maximum number of iterations in newton approach method.

Definition at line 83 of file RegistrationUtils.h.

##### 29.414.4.4 `int Go::RegistrationInput::max_solve_iterations_`

The maximum number of iterations used when solving the linear system in each iteration in Newtons method.

Definition at line 104 of file RegistrationUtils.h.

##### 29.414.4.5 `bool Go::RegistrationInput::multi_core_`

If true, and if OPENMP is included, run the fine registration in multicore.

Definition at line 110 of file RegistrationUtils.h.

##### 29.414.4.6 `double Go::RegistrationInput::newton_tolerance_`

Upper limit for when weighed sum of squares of changes in Newton's method requires us to break out of the loop.

Definition at line 86 of file RegistrationUtils.h.

##### 29.414.4.7 `double Go::RegistrationInput::solve_tolerance_`

The tolerance used when solving the linear system in each iteration in Newtons method.

Definition at line 107 of file RegistrationUtils.h.

29.414.4.8 `double Go::RegistrationInput::tolerance_weight_rescale_`

The weight of the square of the change in the rescaling, used in the Newtons method.

Definition at line 101 of file RegistrationUtils.h.

29.414.4.9 `double Go::RegistrationInput::tolerance_weight_rotation_`

The weight of the square of the change in the rotation vector, used in the Newtons method.

Definition at line 95 of file RegistrationUtils.h.

29.414.4.10 `double Go::RegistrationInput::tolerance_weight_translation_`

The weight of the square of the change in the translation vector, used in the Newtons method.

Definition at line 98 of file RegistrationUtils.h.

The documentation for this struct was generated from the following file:

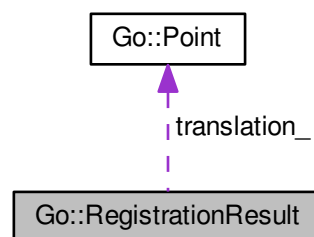
- [gotools-core/include/GoTools/Utils/RegistrationUtils.h](#)

## 29.415 Go::RegistrationResult Struct Reference

Struct for result from registration process, either raw, fine or combined.

```
#include <RegistrationUtils.h>
```

Collaboration diagram for Go::RegistrationResult:



### Public Member Functions

- `bool ok ()`

*Return wether the result of the registration was RegistrationOK.*

## Public Attributes

- [RegistrationReturnType result\\_type\\_](#)  
*The result type of the registration process, either OK or an error code.*
- `std::vector< std::vector< double > >` [rotation\\_matrix\\_](#)  
*The rotation matrix of the registration.*
- [Point translation\\_](#)  
*The translation of the registration (to be performed after rescaling and rotation)*
- [double rescaling\\_](#)  
*The rescaling factor of the registration;.*
- `int` [last\\_newton\\_iteration\\_](#)
- `double` [last\\_change\\_](#)
- `int` [solve\\_result\\_](#)

### 29.415.1 Detailed Description

Struct for result from registration process, either raw, fine or combined.

Definition at line 123 of file RegistrationUtils.h.

### 29.415.2 Member Function Documentation

#### 29.415.2.1 `bool Go::RegistrationResult::ok ( )` `[inline]`

Return wether the result of the registration was RegistrationOK.

Definition at line 154 of file RegistrationUtils.h.

### 29.415.3 Member Data Documentation

#### 29.415.3.1 `double Go::RegistrationResult::last_change_`

The last weighed sum of squares of changes in Newton's method Only used for fine registration

Definition at line 147 of file RegistrationUtils.h.

#### 29.415.3.2 `int Go::RegistrationResult::last_newton_iteration_`

The iteration number (starting at 0) of the last newton iteration, or the maximum number of iterations if the weighed sum of squares of changes never became smaller than the tolerance limit. Only used for fine registration

Definition at line 143 of file RegistrationUtils.h.

#### 29.415.3.3 `double Go::RegistrationResult::rescaling_`

The rescaling factor of the registration;.

Definition at line 137 of file RegistrationUtils.h.

**29.415.3.4 RegistrationReturnType Go::RegistrationResult::result\_type\_**

The result type of the registration process, either OK or an error code.

Definition at line 128 of file RegistrationUtils.h.

**29.415.3.5 std::vector<std::vector<double> > Go::RegistrationResult::rotation\_matrix\_**

The rotation matrix of the registration.

Definition at line 131 of file RegistrationUtils.h.

**29.415.3.6 int Go::RegistrationResult::solve\_result\_**

The return value of the last attempt to solve the linear system in the Newtons method iterations Only used for fine registration

Definition at line 151 of file RegistrationUtils.h.

**29.415.3.7 Point Go::RegistrationResult::translation\_**

The translation of the registration (to be performed after rescaling and rotation)

Definition at line 134 of file RegistrationUtils.h.

The documentation for this struct was generated from the following file:

- [gtools-core/include/GoTools/Utils/RegistrationUtils.h](#)

**29.416 Go::Registrator< T > Class Template Reference**

```
#include <Factory.h>
```

**Public Member Functions**

- [Registrator \(\)](#)

**29.416.1 Detailed Description**

```
template<class T>
class Go::Registrator< T >
```

Work around for compilation problems. On some compilers (ie., VS6), the [Register\(\)](#) function cannot be used directly. To work around this, we wrap it in the constructor of a dummy class, which we call [registrator](#). With other words, in order to register the class 'DerivedClass' (supposedly inheriting from [GeomObject](#)), we type:

```
Registrator<DerivedClass> r
```

```
.
```

Definition at line 174 of file Factory.h.

## 29.416.2 Constructor & Destructor Documentation

### 29.416.2.1 `template<class T> Go::Registrator< T >::Registrator ( ) [inline]`

Definition at line 177 of file `Factory.h`.

The documentation for this class was generated from the following file:

- `gotools-core/include/GoTools/geometry/Factory.h`

## 29.417 Go::RegularizeFace Class Reference

Split one face into a number of 4-sided domains without inner trimming. This class is intended for use in block structuring. One face with possible inner and outer trimming is split according to certain rules to result in a face set with 4 sided faces although faces with less than 4 sides can occur. A side is defined as a piece of the face boundary between two corners or between vertices where there are more than one adjacent face. The trimmed surfaces being output from this class can later be approximated by spline surfaces. The splitting procedure is recursive.

```
#include <RegularizeFace.h>
```

### Public Member Functions

- `RegularizeFace` (`shared_ptr< ftSurface > face`, `double epsge`, `double angtol`, `double tol2`, `bool split_in_↔ cand=false`)  
*Constructor.*
- `RegularizeFace` (`shared_ptr< ftSurface > face`, `double epsge`, `double angtol`, `double tol2`, `double bend`, `bool split_in_cand=false`)  
*Constructor.*
- `RegularizeFace` (`shared_ptr< ftSurface > face`, `shared_ptr< SurfaceModel > model`, `bool split_in_↔ cand=false`)  
*Constructor.*
- `~RegularizeFace` ()  
*Destructor.*
- void `setAxis` (`Point &centre`, `Point &axis`)
- void `unsetAxis` ()
- void `setCandSplit` (`std::vector< std::pair< std::pair< Point, int >, std::pair< Point, int > > > cand_split`)
- void `setNonTjointFaces` (`std::vector< shared_ptr< ftSurface > > &faces`)  
*Set information about faces not to be the cause of T-joint splitting.*
- void `setSplitMode` (`int split_mode`)
- void `classifyVertices` ()
- `std::vector< shared_ptr< ftSurface > >` `getRegularFaces` ()  
*Fetch result.*
- `std::vector< Point >` `getSeamJointInfo` () `const`  
*Fetch info about removed seams.*
- void `setDivideInT` (`bool divideInT`)  
*Decides whether T-joint splitting should be performed at face level.*
- `std::vector< std::pair< Point, Point > >` `fetchVxPntCorr` ()  
*Fetch info about point correspondance.*



### 29.417.1 Detailed Description

Split one face into a number of 4-sided domains without inner trimming. This class is intended for use in block structuring. One face with possible inner and outer trimming is split according to certain rules to result in a face set with 4 sided faces although faces with less than 4 sides can occur. A side is defined as a piece of the face boundary between two corners or between vertices where there are more than one adjacent face. The trimmed surfaces being output from this class can later be approximated by spline surfaces. The splitting procedure is recursive.

Definition at line 58 of file RegularizeFace.h.

### 29.417.2 Constructor & Destructor Documentation

29.417.2.1 `Go::RegularizeFace::RegularizeFace ( shared_ptr< ftSurface > face, double epsge, double angtol, double tol2, bool split_in_cand = false )`

Constructor.

29.417.2.2 `Go::RegularizeFace::RegularizeFace ( shared_ptr< ftSurface > face, double epsge, double angtol, double tol2, double bend, bool split_in_cand = false )`

Constructor.

29.417.2.3 `Go::RegularizeFace::RegularizeFace ( shared_ptr< ftSurface > face, shared_ptr< SurfaceModel > model, bool split_in_cand = false )`

Constructor.

29.417.2.4 `Go::RegularizeFace::~~RegularizeFace ( )`

Destructor.

### 29.417.3 Member Function Documentation

29.417.3.1 `void Go::RegularizeFace::classifyVertices ( )`

Classify vertices according to significance. Mark vertices that should not trigger splitting

29.417.3.2 `std::vector<std::pair<Point, Point> > Go::RegularizeFace::fetchVxPntCorr ( )` `[inline]`

Fetch info about point corrspondance.

Definition at line 124 of file RegularizeFace.h.

29.417.3.3 `std::vector<shared_ptr<ftSurface>> Go::RegularizeFace::getRegularFaces ( )`

Fetch result.

29.417.3.4 `std::vector<Point> Go::RegularizeFace::getSeamJointInfo ( ) const` `[inline]`

Fetch info about removed seams.

Definition at line 112 of file RegularizeFace.h.

29.417.3.5 `void Go::RegularizeFace::setAxis ( Point & centre, Point & axis )`

Set information about the centre of the current face to the regularization of sub faces

29.417.3.6 `void Go::RegularizeFace::setCandSplit ( std::vector< std::pair< std::pair< Point, int >, std::pair< Point, int >>> cand_split )` `[inline]`

Set info about splitting performed in opposite faces in a body. Used from [RegularizeFaceSet](#).

Definition at line 87 of file RegularizeFace.h.

29.417.3.7 `void Go::RegularizeFace::setDivideInT ( bool divideInT )` `[inline]`

Decides whether T-joint splitting should be performed at face level.

Definition at line 118 of file RegularizeFace.h.

29.417.3.8 `void Go::RegularizeFace::setNonTjointFaces ( std::vector< shared_ptr< ftSurface >> & faces )` `[inline]`

Set information about faces not to be the cause of T-joint splitting.

Definition at line 94 of file RegularizeFace.h.

29.417.3.9 `void Go::RegularizeFace::setSplitMode ( int split_mode )` `[inline]`

Definition at line 99 of file RegularizeFace.h.

29.417.3.10 `void Go::RegularizeFace::unsetAxis ( )`

The documentation for this class was generated from the following file:

- `compositemodel/include/GoTools/compositemodel/RegularizeFace.h`

## 29.418 Go::RegularizeFaceSet Class Reference

```
#include <RegularizeFaceSet.h>
```

### Public Member Functions

- [RegularizeFaceSet](#) (std::vector< shared\_ptr< [ftSurface](#) > > faces, double epsge, double angtol, bool split\_in\_cand=false)  
*Constructor.*
- [RegularizeFaceSet](#) (std::vector< shared\_ptr< [ftSurface](#) > > faces, double gap, double neighbour, double kink, double bend, bool split\_in\_cand=false)
- [RegularizeFaceSet](#) (shared\_ptr< [SurfaceModel](#) > model, bool split\_in\_cand=false)  
*Constructor.*
- [~RegularizeFaceSet](#) ()  
*Destructor.*
- void [setSplitMode](#) (int split\_mode)
- void [setFaceCorrespondance](#) (int idx1, int idx2)  
*Set information.*
- std::vector< shared\_ptr< [ftSurface](#) > > [getRegularFaces](#) (bool reverse\_sequence=false)  
*Fetch result.*
- shared\_ptr< [SurfaceModel](#) > [getRegularModel](#) (bool reverse\_sequence=false)  
*Return the resulting face set as a surface model.*
- std::vector< std::pair< [Point](#), [Point](#) > > [fetchVxPntCorr](#) ()  
*Fetch info about point correspondance.*
- std::vector< [SurfaceModel](#) \* > [getModifiedAdjacentModels](#) ()

### 29.418.1 Detailed Description

Split a set of faces into a number of 4-sided domains without inner trimming. This class is intended for use in block structuring. A set of faces with possible inner and outer trimming is split according to certain rules to result in a face set with 4 sided faces although faces with less than 4 sides can occur. A side is defined as a piece of the face boundary between two corners or between vertices where there are more than one adjacent face. The trimmed surfaces being output from this class can later be approximated by spline surfaces.

Definition at line 61 of file RegularizeFaceSet.h.

### 29.418.2 Constructor & Destructor Documentation

29.418.2.1 [Go::RegularizeFaceSet::RegularizeFaceSet](#) ( std::vector< shared\_ptr< [ftSurface](#) > > faces, double epsge, double angtol, bool split\_in\_cand = false )

Constructor.

29.418.2.2 [Go::RegularizeFaceSet::RegularizeFaceSet](#) ( std::vector< shared\_ptr< [ftSurface](#) > > faces, double gap, double neighbour, double kink, double bend, bool split\_in\_cand = false )

29.418.2.3 [Go::RegularizeFaceSet::RegularizeFaceSet](#) ( shared\_ptr< [SurfaceModel](#) > model, bool split\_in\_cand = false )

Constructor.

29.418.2.4 `Go::RegularizeFaceSet::~~RegularizeFaceSet ( )`

Destructor.

### 29.418.3 Member Function Documentation

29.418.3.1 `std::vector<std::pair<Point, Point> > Go::RegularizeFaceSet::fetchVxPntCorr ( ) [inline]`

Fetch info about point correspondance.

Definition at line 93 of file RegularizeFaceSet.h.

29.418.3.2 `std::vector<SurfaceModel*> Go::RegularizeFaceSet::getModifiedAdjacentModels ( ) [inline]`

Fetch info about adjacent surface models that are modified due to changes in the current model

Definition at line 100 of file RegularizeFaceSet.h.

29.418.3.3 `std::vector<shared_ptr<ftSurface> > Go::RegularizeFaceSet::getRegularFaces ( bool reverse_sequence = false )`

Fetch result.

29.418.3.4 `shared_ptr<SurfaceModel> Go::RegularizeFaceSet::getRegularModel ( bool reverse_sequence = false )`

Return the resulting face set as a surface model.

29.418.3.5 `void Go::RegularizeFaceSet::setFaceCorrespondance ( int idx1, int idx2 )`

Set information.

29.418.3.6 `void Go::RegularizeFaceSet::setSplitMode ( int split_mode ) [inline]`

Definition at line 78 of file RegularizeFaceSet.h.

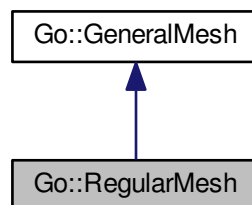
The documentation for this class was generated from the following file:

- [compositemodel/include/GoTools/compositemodel/RegularizeFaceSet.h](#)

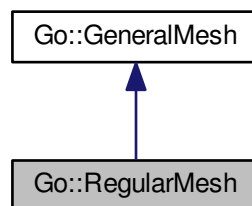
## 29.419 Go::RegularMesh Class Reference

```
#include <RegularMesh.h>
```

Inheritance diagram for Go::RegularMesh:



Collaboration diagram for Go::RegularMesh:



### Public Member Functions

- [RegularMesh](#) (int m=20, int n=20, bool use\_normals=true, bool use\_texcoords=false)
- virtual [~RegularMesh](#) ()  
*Destructor.*
- [bool useNormals](#) ()  
*Check if normals will be computed.*
- [bool useTexCoords](#) ()  
*Check if texture coordinates will be computed.*
- int [numStrips](#) ()  
*Number of strips.*
- int [stripLength](#) ()
- virtual int [numTriangles](#) ()  
*The total number of triangles.*
- virtual int [numVertices](#) ()

- Number of vertices.*

  - void [resize](#) (int m, int n)

*Change mesh size.*

  - virtual [double](#) \* [vertexArray](#) ()

*Fetch all nodes.*

  - virtual [double](#) \* [paramArray](#) ()

*Fetch parameter values corresponding to the nodes.*

  - virtual int [atBoundary](#) (int idx)

*Check if a given node lies at the boundary.*

  - [double](#) \* [normalArray](#) ()

*Normal information.*

  - [double](#) \* [texcoordArray](#) ()

*Texture coordinate information.*

  - unsigned int \* [stripArray](#) ()
  - [Triangle](#) \* [triangleArray](#) ()
  - virtual unsigned int \* [triangleIndexArray](#) ()

*Indices of nodes belonging to each triangle.*

  - virtual [RegularMesh](#) \* [asRegularMesh](#) ()

*Casting. Return as regular mesh.*

  - void [translate](#) (const std::vector< [double](#) > &vert\_translation)

*Translate all vertices by vert\_translation, wrt the geometry.*

## Additional Inherited Members

### 29.419.1 Detailed Description

Simple triangulation class, designed to store data for GL visualizing with triangle strips. Vertices are stored in x1 y1 z1 x2 y2 z2... format. Triangles are stored as startindices into the stripe array. Based on earlier work.

Definition at line 68 of file RegularMesh.h.

### 29.419.2 Constructor & Destructor Documentation

**29.419.2.1** `Go::RegularMesh::RegularMesh ( int m = 20, int n = 20, bool use_normals = true, bool use_texcoords = false )`

Constructor. Give mesh size. Set whether or not normals and texture coordinates should be computed 010613: To me it seems that 'm' and 'n' is the dimension of the (rectangular) grid of vertices, i.e., that in total there will be (m-1)\*(n-1)\*2 triangles. Trying to fix things using this assumption... (jon)

**29.419.2.2** `virtual Go::RegularMesh::~~RegularMesh ( ) [virtual]`

Destructor.

### 29.419.3 Member Function Documentation

**29.419.3.1** `virtual RegularMesh* Go::RegularMesh::asRegularMesh ( ) [virtual]`

Casting. Return as regular mesh.

Reimplemented from [Go::GeneralMesh](#).

29.419.3.2 `virtual int Go::RegularMesh::atBoundary ( int idx ) [virtual]`

Check if a given node lies at the boundary.

Implements [Go::GeneralMesh](#).

29.419.3.3 `double* Go::RegularMesh::normalArray ( ) [inline]`

Normal information.

Definition at line 119 of file RegularMesh.h.

29.419.3.4 `int Go::RegularMesh::numStrips ( ) [inline]`

Number of strips.

Definition at line 95 of file RegularMesh.h.

29.419.3.5 `virtual int Go::RegularMesh::numTriangles ( ) [inline],[virtual]`

The total number of triangles.

Reimplemented from [Go::GeneralMesh](#).

Definition at line 104 of file RegularMesh.h.

29.419.3.6 `virtual int Go::RegularMesh::numVertices ( ) [inline],[virtual]`

Number of vertices.

Implements [Go::GeneralMesh](#).

Definition at line 108 of file RegularMesh.h.

29.419.3.7 `virtual double* Go::RegularMesh::paramArray ( ) [inline],[virtual]`

Fetch parameter values corresponding to the nodes.

Implements [Go::GeneralMesh](#).

Definition at line 116 of file RegularMesh.h.

29.419.3.8 `void Go::RegularMesh::resize ( int m, int n )`

Change mesh size.

29.419.3.9 `unsigned int* Go::RegularMesh::stripArray ( ) [inline]`

Definition at line 122 of file RegularMesh.h.

29.419.3.10 `int Go::RegularMesh::stripLength ( ) [inline]`

The number of vertices in a strip (i.e the number of triangles in a strip+ 2).

Definition at line 100 of file RegularMesh.h.

29.419.3.11 `double* Go::RegularMesh::texcoordArray ( ) [inline]`

Texture coordinate information.

Definition at line 121 of file RegularMesh.h.

29.419.3.12 `void Go::RegularMesh::translate ( const std::vector< double > & vert_translation )`

Translate all vertices by `vert_translation`, wrt the geometry.

29.419.3.13 `Triangle* Go::RegularMesh::triangleArray ( ) [inline]`

Definition at line 123 of file RegularMesh.h.

29.419.3.14 `virtual unsigned int* Go::RegularMesh::triangleIndexArray ( ) [inline],[virtual]`

Indices of nodes belonging to each triangle.

Implements [Go::GeneralMesh](#).

Definition at line 124 of file RegularMesh.h.

29.419.3.15 `bool Go::RegularMesh::useNormals ( ) [inline]`

Check if normals will be computed.

Definition at line 87 of file RegularMesh.h.

29.419.3.16 `bool Go::RegularMesh::useTexCoords ( ) [inline]`

Check if texture coordinates will be computed.

Definition at line 91 of file RegularMesh.h.



29.419.3.17 `virtual double* Go::RegularMesh::vertexArray( ) [inline],[virtual]`

Fetch all nodes.

Implements [Go::GeneralMesh](#).

Definition at line 115 of file RegularMesh.h.

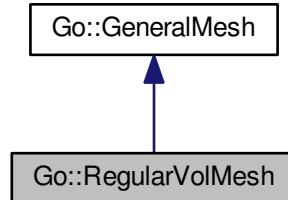
The documentation for this class was generated from the following file:

- [gotools-core/include/GoTools/tesselator/RegularMesh.h](#)

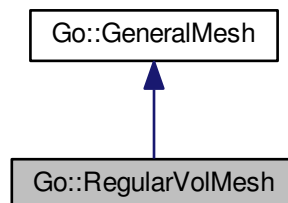
## 29.420 Go::RegularVolMesh Class Reference

```
#include <RegularVolMesh.h>
```

Inheritance diagram for Go::RegularVolMesh:



Collaboration diagram for Go::RegularVolMesh:



## Public Member Functions

- [RegularVolMesh](#) (int m=20, [bool use\\_normals=true](#), [bool use\\_texcoords=false](#))
- virtual [~RegularVolMesh](#) ()
  - Destructor.*
- [bool useNormals](#) ()
  - Check if normals will be computed.*
- [bool useTexCoords](#) ()
  - Check if texture coordinates will be computed.*
- int [numStrips](#) ()
  - Number of strips.*
- int [stripLength](#) ()
- virtual int [numTriangles](#) ()
  - The total number of triangles.*
- virtual int [numVertices](#) ()
  - Number of vertices.*
- void [resize](#) (int m)
  - Change mesh size.*
- virtual [double \\* vertexArray](#) ()
  - Fetch all nodes.*
- virtual [double \\* paramArray](#) ()
  - Fetch parameter values corresponding to the nodes.*
- virtual int [atBoundary](#) (int idx)
  - Check if a given node lies at the boundary.*
- [double \\* normalArray](#) ()
  - Normal information.*
- [double \\* texcoordArray](#) ()
  - Texture coordinate information.*
- unsigned int \* [stripArray](#) ()
- [Triangle \\* triangleArray](#) ()
- virtual unsigned int \* [triangleIndexArray](#) ()
  - Indices of nodes belonging to each triangle.*
- virtual [RegularMesh \\* asRegularMesh](#) ()
  - Casting. Return as regular mesh.*

## Additional Inherited Members

### 29.420.1 Detailed Description

Simple triangulation class, designed to store data for GL visualizing with triangle strips. Vertices are stored in x1 y1 z1 x2 y2 z2... format. Triangles are stored as startindices into the stripe array. Based on earlier work.

Definition at line 56 of file RegularVolMesh.h.

### 29.420.2 Constructor & Destructor Documentation

#### 29.420.2.1 Go::RegularVolMesh::RegularVolMesh ( int m = 20, bool use\_normals = true, bool use\_texcoords = false )

Constructor. Give mesh size. Set whether or not normals and texture coordinates should be computed 010613: To me it seems that 'm' and 'n' is the dimension of the (rectangular) grid of vertices, i.e., that in total there will be (m-1)\*(n-1)\*2 triangles. Trying to fix things using this assumption... (jon)

29.420.2.2 `virtual Go::RegularVolMesh::~~RegularVolMesh ( ) [virtual]`

Destructor.

### 29.420.3 Member Function Documentation

29.420.3.1 `virtual RegularMesh* Go::RegularVolMesh::asRegularMesh ( ) [virtual]`

Casting. Return as regular mesh.

Reimplemented from [Go::GeneralMesh](#).

29.420.3.2 `virtual int Go::RegularVolMesh::atBoundary ( int idx ) [virtual]`

Check if a given node lies at the boundary.

Implements [Go::GeneralMesh](#).

29.420.3.3 `double* Go::RegularVolMesh::normalArray ( ) [inline]`

Normal information.

Definition at line 107 of file RegularVolMesh.h.

29.420.3.4 `int Go::RegularVolMesh::numStrips ( ) [inline]`

Number of strips.

Definition at line 83 of file RegularVolMesh.h.

29.420.3.5 `virtual int Go::RegularVolMesh::numTriangles ( ) [inline],[virtual]`

The total number of triangles.

Reimplemented from [Go::GeneralMesh](#).

Definition at line 92 of file RegularVolMesh.h.

29.420.3.6 `virtual int Go::RegularVolMesh::numVertices ( ) [inline],[virtual]`

Number of vertices.

Implements [Go::GeneralMesh](#).

Definition at line 96 of file RegularVolMesh.h.

**29.420.3.7** `virtual double* Go::RegularVolMesh::paramArray ( ) [inline],[virtual]`

Fetch parameter values corresponding to the nodes.

Implements [Go::GeneralMesh](#).

Definition at line 104 of file RegularVolMesh.h.

**29.420.3.8** `void Go::RegularVolMesh::resize ( int m )`

Change mesh size.

**29.420.3.9** `unsigned int* Go::RegularVolMesh::stripArray ( ) [inline]`

Definition at line 110 of file RegularVolMesh.h.

**29.420.3.10** `int Go::RegularVolMesh::stripLength ( ) [inline]`

The number of vertices in a strip (i.e the number of triangles in a strip+ 2).

Definition at line 88 of file RegularVolMesh.h.

**29.420.3.11** `double* Go::RegularVolMesh::texcoordArray ( ) [inline]`

Texture coordinate information.

Definition at line 109 of file RegularVolMesh.h.

**29.420.3.12** `Triangle* Go::RegularVolMesh::triangleArray ( ) [inline]`

Definition at line 111 of file RegularVolMesh.h.

**29.420.3.13** `virtual unsigned int* Go::RegularVolMesh::triangleIndexArray ( ) [inline],[virtual]`

Indices of nodes belonging to each triangle.

Implements [Go::GeneralMesh](#).

Definition at line 112 of file RegularVolMesh.h.

**29.420.3.14** `bool Go::RegularVolMesh::useNormals ( ) [inline]`

Check if normals will be computed.

Definition at line 75 of file RegularVolMesh.h.

29.420.3.15 `bool Go::RegularVolMesh::useTexCoords ( ) [inline]`

Check if texture coordinates will be computed.

Definition at line 79 of file RegularVolMesh.h.

29.420.3.16 `virtual double* Go::RegularVolMesh::vertexArray ( ) [inline],[virtual]`

Fetch all nodes.

Implements [Go::GeneralMesh](#).

Definition at line 103 of file RegularVolMesh.h.

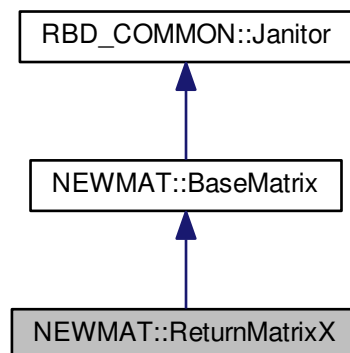
The documentation for this class was generated from the following file:

- [trivariate/include/GoTools/trivariate/RegularVolMesh.h](#)

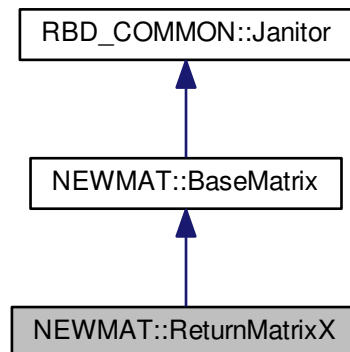
## 29.421 NEWMAT::ReturnMatrixX Class Reference

```
#include <newmat.h>
```

Inheritance diagram for NEWMAT::ReturnMatrixX:



Collaboration diagram for NEWMAT::ReturnMatrixX:



## Public Member Functions

- [~ReturnMatrixX \(\)](#)
- [GeneralMatrix \\* Evaluate \(MatrixType mt=MatrixTypeUnSp\)](#)
- [ReturnMatrixX \(const ReturnMatrixX &tm\)](#)
- [ReturnMatrixX \(const GeneralMatrix \\*gmx\)](#)
- [MatrixBandWidth BandWidth \(\) const](#)

## Friends

- class [BaseMatrix](#)

## Additional Inherited Members

### 29.421.1 Detailed Description

Definition at line 1466 of file newmat.h.

### 29.421.2 Constructor & Destructor Documentation

#### 29.421.2.1 NEWMAT::ReturnMatrixX::~ReturnMatrixX ( ) [\[inline\]](#)

Definition at line 1471 of file newmat.h.

#### 29.421.2.2 NEWMAT::ReturnMatrixX::ReturnMatrixX ( const ReturnMatrixX & tm ) [\[inline\]](#)

Definition at line 1477 of file newmat.h.

29.421.2.3 `NEWMAT::ReturnMatrixX::ReturnMatrixX ( const GeneralMatrix * gmx ) [inline]`

Definition at line 1479 of file newmat.h.

### 29.421.3 Member Function Documentation

29.421.3.1 `MatrixBandWidth ReturnMatrixX::BandWidth ( ) const [virtual]`

Reimplemented from [NEWMAT::BaseMatrix](#).

Definition at line 485 of file newmat4.cpp.

29.421.3.2 `GeneralMatrix * ReturnMatrixX::Evaluate ( MatrixType mt = MatrixTypeUnSp ) [virtual]`

Implements [NEWMAT::BaseMatrix](#).

Definition at line 376 of file newmat5.cpp.

### 29.421.4 Friends And Related Function Documentation

29.421.4.1 `friend class BaseMatrix [friend]`

Definition at line 1473 of file newmat.h.

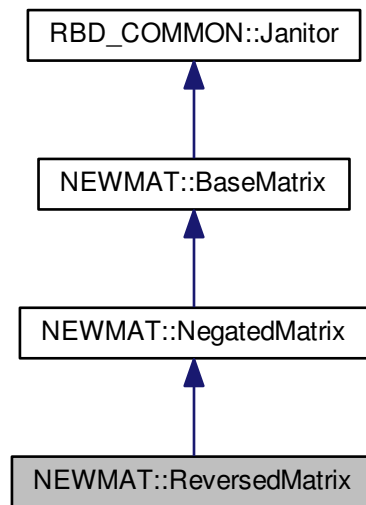
The documentation for this class was generated from the following files:

- [newmat/include/newmat.h](#)
- [newmat/src/newmat4.cpp](#)
- [newmat/src/newmat5.cpp](#)

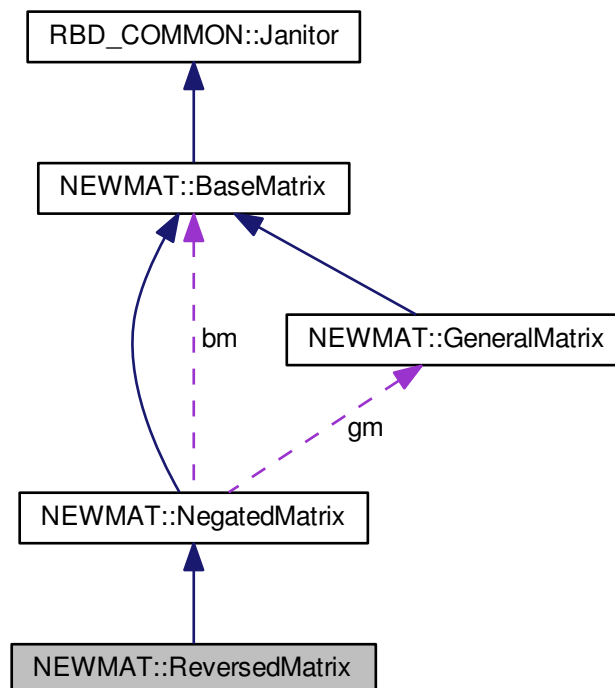
## 29.422 NEWMAT::ReversedMatrix Class Reference

```
#include <newmat.h>
```

Inheritance diagram for NEWMAT::ReversedMatrix:



Collaboration diagram for NEWMAT::ReversedMatrix:





## Public Member Functions

- [~ReversedMatrix](#) ( )
- [GeneralMatrix \\* Evaluate](#) ( [MatrixType](#) mt=[MatrixTypeUnSp](#) )

## Friends

- class [BaseMatrix](#)

## Additional Inherited Members

### 29.422.1 Detailed Description

Definition at line 1392 of file [newmat.h](#).

### 29.422.2 Constructor & Destructor Documentation

29.422.2.1 [NEWMAT::ReversedMatrix::~ReversedMatrix](#) ( ) [[inline](#)]

Definition at line 1397 of file [newmat.h](#).

### 29.422.3 Member Function Documentation

29.422.3.1 [GeneralMatrix \\* ReversedMatrix::Evaluate](#) ( [MatrixType](#) mt = [MatrixTypeUnSp](#) ) [[virtual](#)]

Reimplemented from [NEWMAT::NegatedMatrix](#).

Definition at line 251 of file [newmat5.cpp](#).

### 29.422.4 Friends And Related Function Documentation

29.422.4.1 [friend class BaseMatrix](#) [[friend](#)]

Definition at line 1395 of file [newmat.h](#).

The documentation for this class was generated from the following files:

- [newmat/include/newmat.h](#)
- [newmat/src/newmat5.cpp](#)

## 29.423 Go::RotatedBox Class Reference

```
#include <RotatedBox.h>
```

## Public Member Functions

- `template<typename RandomAccessIterator >`  
`RotatedBox` (`RandomAccessIterator start`, `int dim`, `int num_u`, `int num_v`, `const Point *axis`)
- `RotatedBox` (`const Point &low`, `const Point &high`, `const Point *axis`)
- `~RotatedBox` ()  
*Do not inherit from this class – nonvirtual destructor.*
- `template<typename RandomAccessIterator >`  
`void setFromArray` (`RandomAccessIterator start`, `int dim`, `int num_u`, `int num_v`)
- `void setFromPoints` (`const Point &low`, `const Point &high`)
- `int dimension` () `const`  
*The dimension of the rotated box.*
- `const Point & low` () `const`
- `const Point & high` () `const`
- `const Point low_rot` () `const`
- `const Point high_rot` () `const`
- `const CompositeBox & box` () `const`
- `bool containsPoint` (`const Point &pt`, `double toli=0.0`, `double tole=0.0`) `const`
- `bool overlaps` (`const RotatedBox &box`, `double toli=0.0`, `double tole=0.0`) `const`
- `bool containsBox` (`const RotatedBox &box`, `double toli=0.0`, `double tole=0.0`) `const`

### 29.423.1 Detailed Description

A rotated version of [CompositeBox](#). It works in the same way, except that the boxes are aligned with an arbitrary (given) coordinate system.

Definition at line 57 of file `RotatedBox.h`.

### 29.423.2 Constructor & Destructor Documentation

29.423.2.1 `template<typename RandomAccessIterator > Go::RotatedBox::RotatedBox ( RandomAccessIterator start, int dim, int num_u, int num_v, const Point * axis ) [inline]`

Given an array of `dim`-dimensional points and a coordinate system given by (`axis[0]`, `axis[1]`, `axis[0] x axis[1]`), construct a rotated box containing all points. If in 2D, coordinate system is (`axis[0]`, `Rot(Pi/2)*axis[0]`)

Definition at line 65 of file `RotatedBox.h`.

29.423.2.2 `Go::RotatedBox::RotatedBox ( const Point & low, const Point & high, const Point * axis ) [inline]`

Creates a [RotatedBox](#) with the [Point](#) `low` specifying the lower bound in all dimensions and `high` specifying the upper bound. The inner and edge boxes are equal.

Definition at line 80 of file `RotatedBox.h`.

29.423.2.3 `Go::RotatedBox::~~RotatedBox ( ) [inline]`

Do not inherit from this class – nonvirtual destructor.

Definition at line 90 of file `RotatedBox.h`.

### 29.423.3 Member Function Documentation

**29.423.3.1** `const CompositeBox& Go::RotatedBox::box ( ) const [inline]`

The composite box. WARNING: The coordinates of this box must be interpreted in the coordinate system given by `coordsystem()`.

Definition at line 211 of file `RotatedBox.h`.

**29.423.3.2** `bool Go::RotatedBox::containsBox ( const RotatedBox & box, double toli = 0.0, double tole = 0.0 ) const [inline]`

Returns true if this box contain the box passed as a parameter, up to tolerances. Tolerances may be specified separately for inner and edge boxes.

Definition at line 258 of file `RotatedBox.h`.

**29.423.3.3** `bool Go::RotatedBox::containsPoint ( const Point & pt, double toli = 0.0, double tole = 0.0 ) const [inline]`

Returns true if the point `pt` is inside the box, up to tolerances. Tolerances may be specified separately for inner and edge boxes.

Definition at line 219 of file `RotatedBox.h`.

**29.423.3.4** `int Go::RotatedBox::dimension ( ) const [inline]`

The dimension of the rotated box.

Definition at line 156 of file `RotatedBox.h`.

**29.423.3.5** `const Point& Go::RotatedBox::high ( ) const [inline]`

The upper bound of the bounding box, in coordinate system of rotated box.

Definition at line 170 of file `RotatedBox.h`.

**29.423.3.6** `const Point Go::RotatedBox::high_rot ( ) const [inline]`

The upper bound of the bounding box, in standard coordinate system.

Definition at line 194 of file `RotatedBox.h`.

**29.423.3.7** `const Point& Go::RotatedBox::low ( ) const [inline]`

The lower bound of the bounding box, in coordinate system of rotated box.

Definition at line 163 of file `RotatedBox.h`.

29.423.3.8 `const Point Go::RotatedBox::low_rot ( ) const` [inline]

The lower bound of the bounding box, in standard coordinate system.

Definition at line 177 of file RotatedBox.h.

29.423.3.9 `bool Go::RotatedBox::overlaps ( const RotatedBox & box, double toli = 0.0, double tole = 0.0 ) const` [inline]

Returns true if the two boxes overlap, up to tolerances. Tolerances may be specified separately for inner and edge boxes.

Definition at line 238 of file RotatedBox.h.

29.423.3.10 `template<typename RandomAccessIterator > void Go::RotatedBox::setFromArray ( RandomAccessIterator start, int dim, int num_u, int num_v )` [inline]

Given an array of dim-dimensional points stored as doubles or floats, makes the smallest composite box containing all points in the array. The array must be like a control point grid, with num\_u points in the fastest running direction, and num\_v points in the other direction. For curves, use num\_v = 1.

Definition at line 101 of file RotatedBox.h.

29.423.3.11 `void Go::RotatedBox::setFromPoints ( const Point & low, const Point & high )` [inline]

Makes the bounding box have lower bounds as specified in low and upper bounds as specified in high.

Definition at line 136 of file RotatedBox.h.

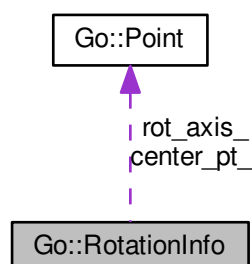
The documentation for this class was generated from the following file:

- [gotools-core/include/GoTools/Utils/RotatedBox.h](#)

## 29.424 Go::RotationInfo Struct Reference

```
#include <cmUtils.h>
```

Collaboration diagram for Go::RotationInfo:



## Public Attributes

- [Go::Point center\\_pt\\_](#)
- [Go::Point rot\\_axis\\_](#)
- [double rot\\_angle\\_](#)

*Measured in radians.*

### 29.424.1 Detailed Description

Definition at line 53 of file cmUtils.h.

### 29.424.2 Member Data Documentation

#### 29.424.2.1 [Go::Point](#) [Go::RotationInfo::center\\_pt\\_](#)

Definition at line 55 of file cmUtils.h.

#### 29.424.2.2 [double](#) [Go::RotationInfo::rot\\_angle\\_](#)

Measured in radians.

Definition at line 57 of file cmUtils.h.

#### 29.424.2.3 [Go::Point](#) [Go::RotationInfo::rot\\_axis\\_](#)

Definition at line 56 of file cmUtils.h.

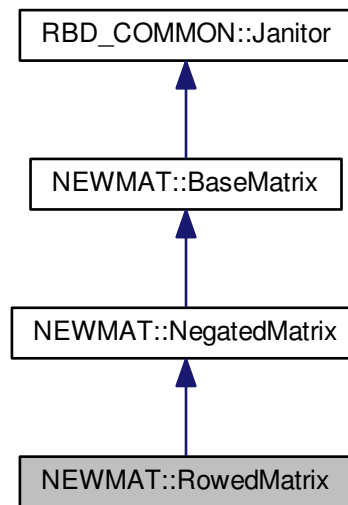
The documentation for this struct was generated from the following file:

- [compositemodel/include/GoTools/compositemodel/cmUtils.h](#)

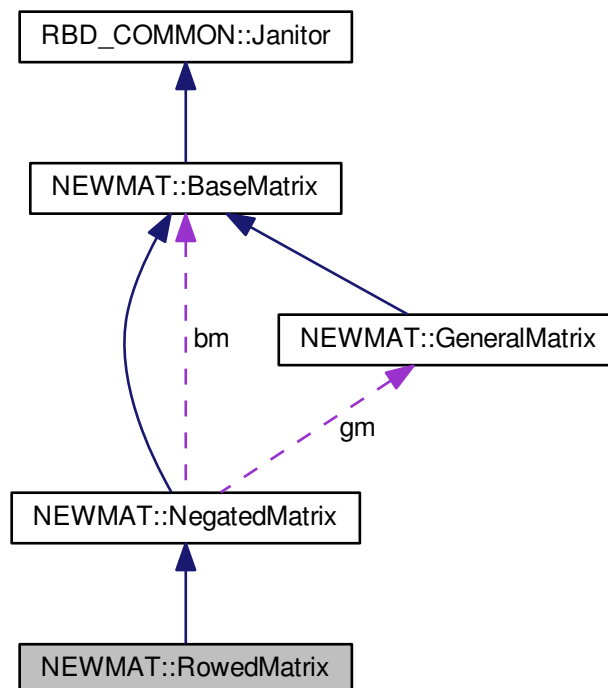
## 29.425 NEWMAT::**RowedMatrix** Class Reference

```
#include <newmat.h>
```

Inheritance diagram for NEWMAT::RowedMatrix:



Collaboration diagram for NEWMAT::RowedMatrix:



## Public Member Functions

- [~RowedMatrix](#) ()
- [GeneralMatrix \\* Evaluate](#) ([MatrixType](#) mt=[MatrixTypeUnSp](#))
- [MatrixBandWidth BandWidth](#) () const

## Friends

- class [BaseMatrix](#)

## Additional Inherited Members

### 29.425.1 Detailed Description

Definition at line 1420 of file newmat.h.

### 29.425.2 Constructor & Destructor Documentation

29.425.2.1 [NEWMAT::RowedMatrix::~RowedMatrix](#) ( ) `[inline]`

Definition at line 1425 of file newmat.h.

### 29.425.3 Member Function Documentation

29.425.3.1 [MatrixBandWidth](#) [RowedMatrix::BandWidth](#) ( ) const `[virtual]`

Reimplemented from [NEWMAT::NegatedMatrix](#).

Definition at line 481 of file newmat4.cpp.

29.425.3.2 [GeneralMatrix \\* Evaluate](#) ( [MatrixType](#) mt = [MatrixTypeUnSp](#) ) `[virtual]`

Reimplemented from [NEWMAT::NegatedMatrix](#).

Definition at line 290 of file newmat5.cpp.

### 29.425.4 Friends And Related Function Documentation

29.425.4.1 `friend class` [BaseMatrix](#) `[friend]`

Definition at line 1423 of file newmat.h.

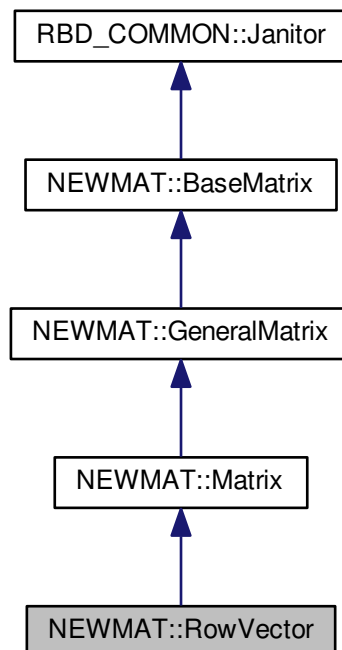
The documentation for this class was generated from the following files:

- [newmat/include/newmat.h](#)
- [newmat/src/newmat4.cpp](#)
- [newmat/src/newmat5.cpp](#)

## 29.426 NEWMAT::RowVector Class Reference

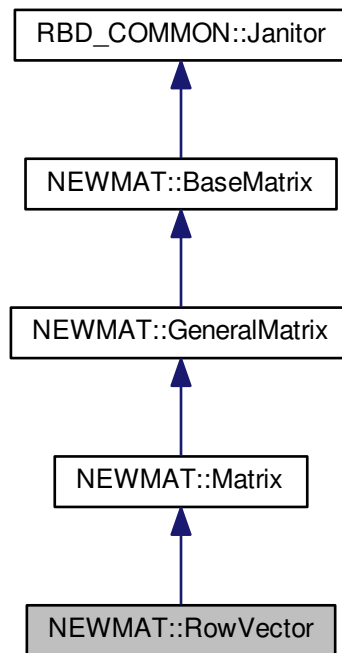
```
#include <newmat.h>
```

Inheritance diagram for NEWMAT::RowVector:





Collaboration diagram for NEWMAT::RowVector:



### Public Member Functions

- [RowVector](#) ()
- [~RowVector](#) ()
- [RowVector](#) (ArrayLengthSpecifier n)
- [RowVector](#) (const BaseMatrix &)
- [RowVector](#) (const RowVector &gm)
- void [operator=](#) (const BaseMatrix &)
- void [operator=](#) (Real f)
- void [operator=](#) (const RowVector &m)
- [Real & operator\(\)](#) (int)
- [Real & element](#) (int)
- [Real operator\(\)](#) (int) const
- [Real element](#) (int) const
- [MatrixType Type](#) () const
- void [GetCol](#) (MatrixRowCol &)
- void [GetCol](#) (MatrixColX &)
- void [NextCol](#) (MatrixRowCol &)
- void [NextCol](#) (MatrixColX &)
- void [RestoreCol](#) (MatrixRowCol &)
- void [RestoreCol](#) (MatrixColX &c)
- [GeneralMatrix \\* Transpose](#) (TransposedMatrix \*, MatrixType)
- void [ReSize](#) (int)
- void [ReSize](#) (int, int)
- void [ReSize](#) (const GeneralMatrix &A)
- [Real \\* nric](#) () const
- void [CleanUp](#) ()

## Additional Inherited Members

### 29.426.1 Detailed Description

Definition at line 792 of file newmat.h.

### 29.426.2 Constructor & Destructor Documentation

#### 29.426.2.1 `NEWMAT::RowVector::RowVector ( )` [inline]

Definition at line 796 of file newmat.h.

#### 29.426.2.2 `NEWMAT::RowVector::~~RowVector ( )` [inline]

Definition at line 797 of file newmat.h.

#### 29.426.2.3 `NEWMAT::RowVector::RowVector ( ArrayLengthSpecifier n )` [inline]

Definition at line 798 of file newmat.h.

#### 29.426.2.4 `NEWMAT::RowVector::RowVector ( const BaseMatrix & )`

#### 29.426.2.5 `NEWMAT::RowVector::RowVector ( const RowVector & gm )` [inline]

Definition at line 800 of file newmat.h.

### 29.426.3 Member Function Documentation

#### 29.426.3.1 `void RowVector::CleanUp ( )` [virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 832 of file newmat4.cpp.

#### 29.426.3.2 `Real& NEWMAT::RowVector::element ( int )`

#### 29.426.3.3 `Real NEWMAT::RowVector::element ( int ) const`

#### 29.426.3.4 `void NEWMAT::RowVector::GetCol ( MatrixRowCol & )` [virtual]

Reimplemented from [NEWMAT::Matrix](#).

29.426.3.5 void NEWMAT::RowVector::GetCol ( MatrixColX & ) [virtual]

Reimplemented from [NEWMAT::Matrix](#).

29.426.3.6 void NEWMAT::RowVector::NextCol ( MatrixRowCol & ) [virtual]

Reimplemented from [NEWMAT::Matrix](#).

29.426.3.7 void NEWMAT::RowVector::NextCol ( MatrixColX & ) [virtual]

Reimplemented from [NEWMAT::Matrix](#).

29.426.3.8 Real\* NEWMAT::RowVector::nric ( ) const [inline]

Definition at line 823 of file newmat.h.

29.426.3.9 Real& NEWMAT::RowVector::operator() ( int )

29.426.3.10 Real NEWMAT::RowVector::operator() ( int ) const

29.426.3.11 void NEWMAT::RowVector::operator= ( const BaseMatrix & )

29.426.3.12 void NEWMAT::RowVector::operator= ( Real f ) [inline]

Definition at line 802 of file newmat.h.

29.426.3.13 void NEWMAT::RowVector::operator= ( const RowVector & m ) [inline]

Definition at line 803 of file newmat.h.

29.426.3.14 void NEWMAT::RowVector::ReSize ( int )

29.426.3.15 void NEWMAT::RowVector::ReSize ( int, int ) [virtual]

Reimplemented from [NEWMAT::Matrix](#).

29.426.3.16 void RowVector::ReSize ( const GeneralMatrix & A ) [virtual]

Reimplemented from [NEWMAT::Matrix](#).

Definition at line 276 of file newmat4.cpp.

29.426.3.17 `void NEWMAT::RowVector::RestoreCol ( MatrixRowCol & ) [inline],[virtual]`

Reimplemented from [NEWMAT::Matrix](#).

Definition at line 817 of file `newmat.h`.

29.426.3.18 `void NEWMAT::RowVector::RestoreCol ( MatrixColX & c ) [virtual]`

Reimplemented from [NEWMAT::Matrix](#).

29.426.3.19 `GeneralMatrix * RowVector::Transpose ( TransposedMatrix *, MatrixType mt ) [virtual]`

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 68 of file `newmat5.cpp`.

29.426.3.20 `MatrixType RowVector::Type ( ) const [virtual]`

Reimplemented from [NEWMAT::Matrix](#).

Definition at line 390 of file `newmat4.cpp`.

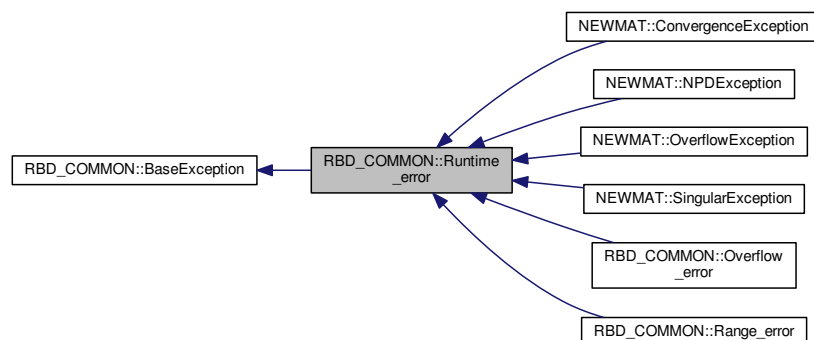
The documentation for this class was generated from the following files:

- [newmat/include/newmat.h](#)
- [newmat/src/newmat4.cpp](#)
- [newmat/src/newmat5.cpp](#)

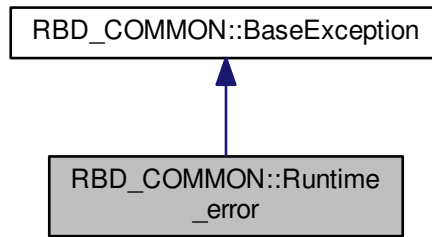
## 29.427 RBD\_COMMON::Runtime\_error Class Reference

```
#include <myexcept.h>
```

Inheritance diagram for `RBD_COMMON::Runtime_error`:



Collaboration diagram for RBD\_COMMON::Runtime\_error:



- static unsigned long [Select](#)
- [Runtime\\_error](#) (`const char *a_what=0`)

### Additional Inherited Members

#### 29.427.1 Detailed Description

Definition at line 364 of file `myexcept.h`.

#### 29.427.2 Constructor & Destructor Documentation

##### 29.427.2.1 `Runtime_error::Runtime_error ( const char * a_what = 0 )`

Definition at line 404 of file `myexcept.cpp`.

#### 29.427.3 Member Data Documentation

##### 29.427.3.1 `unsigned long Runtime_error::Select` `[static]`

Definition at line 367 of file `myexcept.h`.

The documentation for this class was generated from the following files:

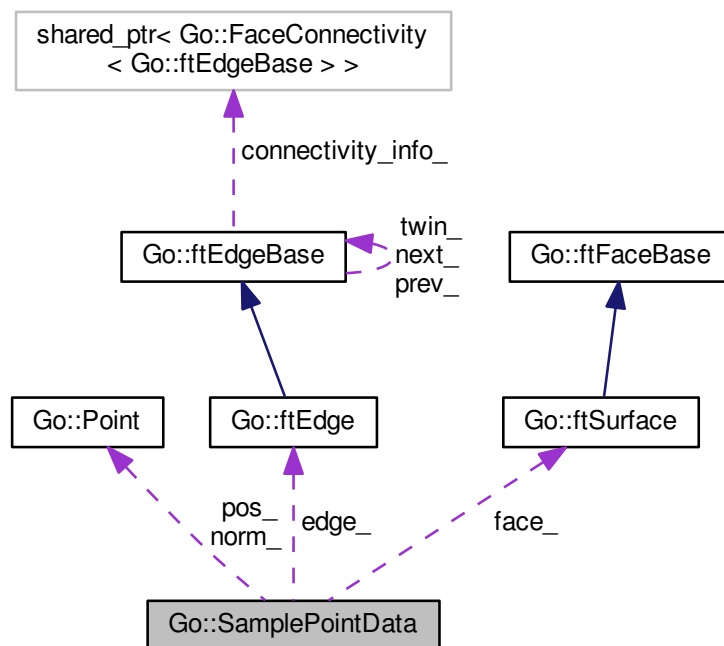
- `newmat/include/myexcept.h`
- `newmat/src/myexcept.cpp`

## 29.428 Go::SamplePointData Struct Reference

Sample data related to one face. The struct contains one point in a point set with information about position, associated surface normal and curvature. Points lying on the face boundary know about its associated edge. If the surface normal and curvature information regarding boundary points is not unique, the associated data is set to be equal to MAX\_DOUBLE.

```
#include <FaceUtilities.h>
```

Collaboration diagram for Go::SamplePointData:



### Public Member Functions

- [SamplePointData](#) ([Point](#) pos, [Point](#) norm, [double](#) curvature, [ftSurface](#) \*face, [double](#) face\_par\_u, [double](#) face\_par\_v, [ftEdge](#) \*edge, [double](#) edge\_par)
 

*Constructor for points at the face boundary.*
- [SamplePointData](#) ([Point](#) pos, [Point](#) norm, [double](#) curvature, [ftSurface](#) \*face, [double](#) face\_par\_u, [double](#) face\_par\_v)
 

*Constructor for points in the inner of the face.*

### Public Attributes

- [Point](#) pos\_
- [Point](#) norm\_
- [double](#) mean\_curvature\_
- [ftSurface](#) \* face\_
- [double](#) face\_par\_[2]
- [ftEdge](#) \* edge\_
- [double](#) edge\_par\_

### 29.428.1 Detailed Description

Sample data related to one face. The struct contains one point in a point set with information about position, associated surface normal and curvature. Points lying on the face boundary know about its associated edge. If the surface normal and curvature information regarding boundary points is not unique, the associated data is set to be equal to MAX\_DOUBLE.

Definition at line 55 of file FaceUtilities.h.

### 29.428.2 Constructor & Destructor Documentation

**29.428.2.1** `Go::SamplePointData::SamplePointData ( Point pos, Point norm, double curvature, ftSurface * face, double face_par_u, double face_par_v, ftEdge * edge, double edge_par ) [inline]`

Constructor for points at the face boundary.

Definition at line 66 of file FaceUtilities.h.

**29.428.2.2** `Go::SamplePointData::SamplePointData ( Point pos, Point norm, double curvature, ftSurface * face, double face_par_u, double face_par_v ) [inline]`

Constructor for points in the inner of the face.

Definition at line 81 of file FaceUtilities.h.

### 29.428.3 Member Data Documentation

**29.428.3.1** `ftEdge* Go::SamplePointData::edge_`

Definition at line 62 of file FaceUtilities.h.

**29.428.3.2** `double Go::SamplePointData::edge_par_`

Definition at line 63 of file FaceUtilities.h.

**29.428.3.3** `ftSurface* Go::SamplePointData::face_`

Definition at line 60 of file FaceUtilities.h.

**29.428.3.4** `double Go::SamplePointData::face_par_[2]`

Definition at line 61 of file FaceUtilities.h.

29.428.3.5 **double** Go::SamplePointData::mean\_curvature\_

Definition at line 59 of file FaceUtilities.h.

29.428.3.6 **Point** Go::SamplePointData::norm\_

Definition at line 58 of file FaceUtilities.h.

29.428.3.7 **Point** Go::SamplePointData::pos\_

Definition at line 57 of file FaceUtilities.h.

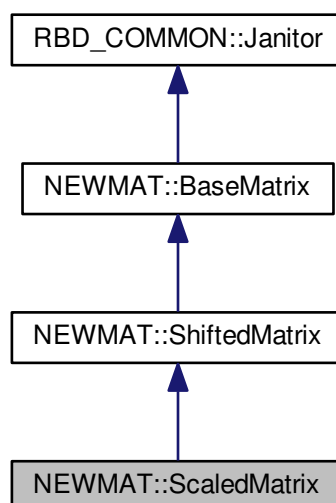
The documentation for this struct was generated from the following file:

- [compositemodel/include/GoTools/compositemodel/FaceUtilities.h](#)

## 29.429 NEWMAT::ScaledMatrix Class Reference

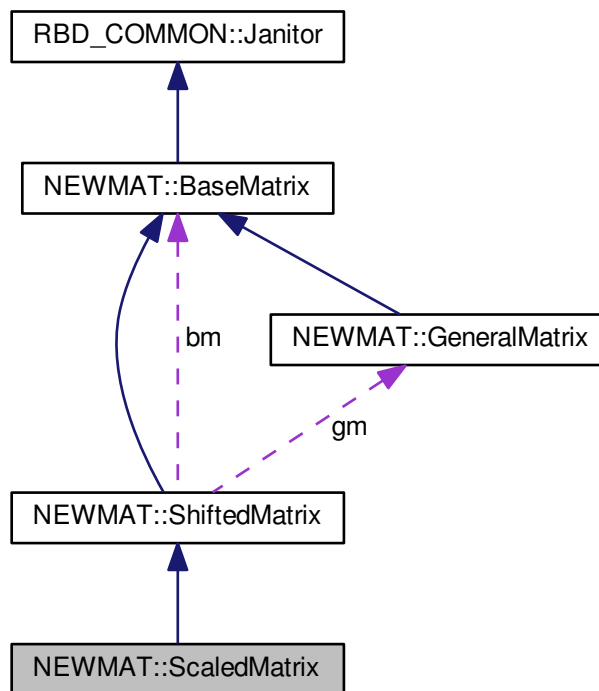
```
#include <newmat.h>
```

Inheritance diagram for NEWMAT::ScaledMatrix:





Collaboration diagram for NEWMAT::ScaledMatrix:



## Public Member Functions

- `~ScaledMatrix ()`
- `GeneralMatrix * Evaluate (MatrixType mt=MatrixTypeUnSp)`
- `MatrixBandWidth BandWidth () const`

## Friends

- class `BaseMatrix`
- class `GeneralMatrix`
- class `GenericMatrix`
- `ScaledMatrix operator* (Real f, const BaseMatrix &BM)`

## Additional Inherited Members

### 29.429.1 Detailed Description

Definition at line 1349 of file newmat.h.

## 29.429.2 Constructor & Destructor Documentation

### 29.429.2.1 `NEWMAT::ScaledMatrix::~~ScaledMatrix ( )` [inline]

Definition at line 1356 of file `newmat.h`.

## 29.429.3 Member Function Documentation

### 29.429.3.1 `MatrixBandWidth ScaledMatrix::BandWidth ( ) const` [virtual]

Reimplemented from [NEWMAT::BaseMatrix](#).

Definition at line 465 of file `newmat4.cpp`.

### 29.429.3.2 `GeneralMatrix * ScaledMatrix::Evaluate ( MatrixType mt = MatrixTypeUnSp )` [virtual]

Reimplemented from [NEWMAT::ShiftedMatrix](#).

Definition at line 170 of file `newmat5.cpp`.

## 29.429.4 Friends And Related Function Documentation

### 29.429.4.1 `friend class BaseMatrix` [friend]

Definition at line 1352 of file `newmat.h`.

### 29.429.4.2 `friend class GeneralMatrix` [friend]

Definition at line 1353 of file `newmat.h`.

### 29.429.4.3 `friend class GenericMatrix` [friend]

Definition at line 1354 of file `newmat.h`.

### 29.429.4.4 `ScaledMatrix operator* ( Real f, const BaseMatrix & BM )` [friend]

Definition at line 1778 of file `newmat.h`.

The documentation for this class was generated from the following files:

- [newmat/include/newmat.h](#)
- [newmat/src/newmat4.cpp](#)
- [newmat/src/newmat5.cpp](#)

## 29.430 Go::ScratchVect< T, N > Class Template Reference

```
#include <ScratchVect.h>
```

### Public Types

- typedef `T * iterator`
- typedef `const T * const_iterator`

### Public Member Functions

- `ScratchVect ()`  
*Create a new, empty ScratchVect.*
- `ScratchVect (size_t size)`  
*Create a new ScratchVect with 'size' (uninitialized) elements.*
- `ScratchVect (size_t size, T elem)`
- `ScratchVect (const std::vector< T > &vec)`
- `template<typename RAlter >`  
`ScratchVect (RAlter beg, RAlter end)`
- `template<int M>`  
`ScratchVect (const ScratchVect< T, M > &orig)`
- `ScratchVect & operator= (const ScratchVect &orig)`  
*Assignment operator.*
- `void resize (size_t new_size)`  
*resize the ScratchVect*
- `template<typename RAlter >`  
`void assign (RAlter beg, RAlter end)`
- `iterator begin ()`  
*Get iterator to beginning of range.*
- `const_iterator begin () const`  
*Get constant iterator to beginning of range.*
- `iterator end ()`  
*Get iterator to one-past-end of range.*
- `const_iterator end () const`  
*Get constant iterator to one-past-end of range.*
- `size_t size () const`  
*Get the number of elements in the ScratchVect.*
- `const T & operator[] (int i) const`  
*Element access operators (const and not-const)*
- `T & operator[] (int i)`
- `void push_back (const T &e)`  
*Add an element to the back of the range.*
- `void clear ()`  
*Clear ScratchVect.*

### 29.430.1 Detailed Description

```
template<typename T, size_t N>
class Go::ScratchVect< T, N >
```

A template Vector class that behaves much like the `std::vector` template, but stores its elements on the stack rather than on the heap as long as the total size stays less than 'N' (template argument).

Definition at line 56 of file `ScratchVect.h`.

### 29.430.2 Member Typedef Documentation

29.430.2.1 `template<typename T, size_t N> typedef const T* Go::ScratchVect< T, N >::const_iterator`

Definition at line 66 of file `ScratchVect.h`.

29.430.2.2 `template<typename T, size_t N> typedef T* Go::ScratchVect< T, N >::iterator`

Definition at line 65 of file `ScratchVect.h`.

### 29.430.3 Constructor & Destructor Documentation

29.430.3.1 `template<typename T, size_t N> Go::ScratchVect< T, N >::ScratchVect ( ) [inline]`

Create a new, empty [ScratchVect](#).

Definition at line 69 of file `ScratchVect.h`.

29.430.3.2 `template<typename T, size_t N> Go::ScratchVect< T, N >::ScratchVect ( size_t size ) [inline]`

Create a new [ScratchVect](#) with 'size' (uninitialized) elements.

Definition at line 75 of file `ScratchVect.h`.

29.430.3.3 `template<typename T, size_t N> Go::ScratchVect< T, N >::ScratchVect ( size_t size, T elem ) [inline]`

Create a new [ScratchVect](#) with 'size' elements initialized to the value 'elem'.

Definition at line 88 of file `ScratchVect.h`.

29.430.3.4 `template<typename T, size_t N> Go::ScratchVect< T, N >::ScratchVect ( const std::vector< T > & vec ) [inline]`

Set the contents of this [ScratchVect](#) to be equal to that of the STL vector 'vec'.

Definition at line 104 of file `ScratchVect.h`.

29.430.3.5 `template<typename T, size_t N> template<typename RAlter > Go::ScratchVect< T, N >::ScratchVect ( RAlter beg, RAlter end ) [inline]`

Set the contents of this [ScratchVect](#) to be equal to the contents of the range specified by 'beg' and 'end'.

Definition at line 111 of file ScratchVect.h.

29.430.3.6 `template<typename T, size_t N> template<int M> Go::ScratchVect< T, N >::ScratchVect ( const ScratchVect< T, M > & orig ) [inline]`

Set the contents of this ScratchVector to be equal to that of the ScratchVector 'orig'. (A templated copy constructor, if you like).

Definition at line 119 of file ScratchVect.h.

#### 29.430.4 Member Function Documentation

29.430.4.1 `template<typename T, size_t N> template<typename RAlter > void Go::ScratchVect< T, N >::assign ( RAlter beg, RAlter end ) [inline]`

Set the contents of this [ScratchVect](#) to be equal to the contents of the range specified by 'beg' and 'end'.

Definition at line 161 of file ScratchVect.h.

29.430.4.2 `template<typename T, size_t N> iterator Go::ScratchVect< T, N >::begin ( ) [inline]`

Get iterator to beginning of range.

Definition at line 176 of file ScratchVect.h.

29.430.4.3 `template<typename T, size_t N> const_iterator Go::ScratchVect< T, N >::begin ( ) const [inline]`

Get constant iterator to beginning of range.

Definition at line 179 of file ScratchVect.h.

29.430.4.4 `template<typename T, size_t N> void Go::ScratchVect< T, N >::clear ( ) [inline]`

Clear [ScratchVect](#).

Definition at line 217 of file ScratchVect.h.

29.430.4.5 `template<typename T, size_t N> iterator Go::ScratchVect< T, N >::end ( ) [inline]`

Get iterator to one-past-end of range.

Definition at line 182 of file ScratchVect.h.

29.430.4.6 `template<typename T, size_t N> const_iterator Go::ScratchVect< T, N >::end ( ) const [inline]`

Get constant iterator to one-past-end of range.

Definition at line 185 of file ScratchVect.h.

29.430.4.7 `template<typename T, size_t N> ScratchVect& Go::ScratchVect< T, N >::operator= ( const ScratchVect< T, N > & orig ) [inline]`

Assignment operator.

Definition at line 124 of file ScratchVect.h.

29.430.4.8 `template<typename T, size_t N> const T& Go::ScratchVect< T, N >::operator[]( int i ) const [inline]`

Element access operators (const and not-const)

Definition at line 191 of file ScratchVect.h.

29.430.4.9 `template<typename T, size_t N> T& Go::ScratchVect< T, N >::operator[]( int i ) [inline]`

Definition at line 192 of file ScratchVect.h.

29.430.4.10 `template<typename T, size_t N> void Go::ScratchVect< T, N >::push_back ( const T & e ) [inline]`

Add an element to the back of the range.

Definition at line 195 of file ScratchVect.h.

29.430.4.11 `template<typename T, size_t N> void Go::ScratchVect< T, N >::resize ( size_t new_size ) [inline]`

resize the [ScratchVect](#)

Definition at line 130 of file ScratchVect.h.

29.430.4.12 `template<typename T, size_t N> size_t Go::ScratchVect< T, N >::size ( ) const [inline]`

Get the number of elements in the [ScratchVect](#).

Definition at line 188 of file ScratchVect.h.

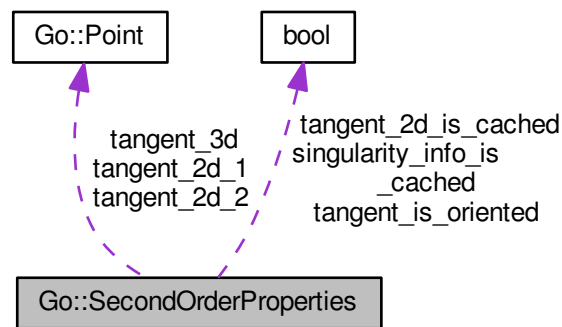
The documentation for this class was generated from the following file:

- [gotools-core/include/GoTools/utis/ScratchVect.h](#)

## 29.431 Go::SecondOrderProperties Struct Reference

```
#include <SecondOrderProperties.h>
```

Collaboration diagram for Go::SecondOrderProperties:



### Public Member Functions

- [SecondOrderProperties](#) ()
- void [clear](#) ()

### Public Attributes

- [bool](#) [singularity\\_info\\_is\\_cached](#)
- [Point](#) [tangent\\_3d](#) [2]
- [SingularityType](#) [singularity\\_type](#)
- [bool](#) [tangent\\_is\\_oriented](#)
- [bool](#) [tangent\\_2d\\_is\\_cached](#)
- [Point](#) [tangent\\_2d\\_1](#) [2]
- [Point](#) [tangent\\_2d\\_2](#) [2]

#### 29.431.1 Detailed Description

Definition at line 59 of file SecondOrderProperties.h.

#### 29.431.2 Constructor & Destructor Documentation

29.431.2.1 [Go::SecondOrderProperties::SecondOrderProperties](#) ( ) [[inline](#)]

Definition at line 61 of file SecondOrderProperties.h.

### 29.431.3 Member Function Documentation

29.431.3.1 `void Go::SecondOrderProperties::clear ( ) [inline]`

Definition at line 64 of file `SecondOrderProperties.h`.

### 29.431.4 Member Data Documentation

29.431.4.1 `bool Go::SecondOrderProperties::singularity_info_is_cached`

Definition at line 74 of file `SecondOrderProperties.h`.

29.431.4.2 `SingularityType Go::SecondOrderProperties::singularity_type`

Definition at line 76 of file `SecondOrderProperties.h`.

29.431.4.3 `Point Go::SecondOrderProperties::tangent_2d_1[2]`

Definition at line 82 of file `SecondOrderProperties.h`.

29.431.4.4 `Point Go::SecondOrderProperties::tangent_2d_2[2]`

Definition at line 83 of file `SecondOrderProperties.h`.

29.431.4.5 `bool Go::SecondOrderProperties::tangent_2d_is_cached`

Definition at line 81 of file `SecondOrderProperties.h`.

29.431.4.6 `Point Go::SecondOrderProperties::tangent_3d[2]`

Definition at line 75 of file `SecondOrderProperties.h`.

29.431.4.7 `bool Go::SecondOrderProperties::tangent_is_oriented`

Definition at line 78 of file `SecondOrderProperties.h`.

The documentation for this struct was generated from the following file:

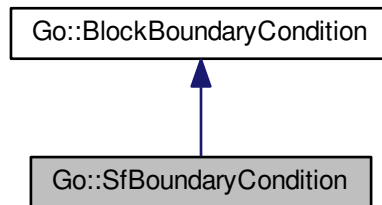
- [intersections/include/GoTools/intersections/SecondOrderProperties.h](#)



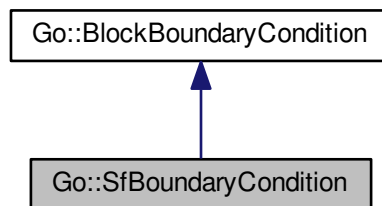
## 29.432 Go::SfBoundaryCondition Class Reference

```
#include <SfBoundaryCondition.h>
```

Inheritance diagram for Go::SfBoundaryCondition:



Collaboration diagram for Go::SfBoundaryCondition:



### Public Member Functions

- [SfBoundaryCondition](#) (int edge\_nmb, [BdConditionType](#) type, [BdCondFuncor](#) \*fbd, std::pair< [double](#), [double](#) > end\_par, [SfSolution](#) \*solution)
- [SfBoundaryCondition](#) (int edge\_nmb, [BdConditionType](#) type, [const Point](#) &const\_val, std::pair< [double](#), [double](#) > end\_par, [SfSolution](#) \*solution)
- virtual [~SfBoundaryCondition](#) ()
- virtual void [getCoefficientsEnumeration](#) (std::vector< int > &local\_enumeration)
- virtual void [getCoefficientsEnumeration](#) (std::vector< int > &local\_enumeration\_bd, std::vector< int > &local\_enumeration\_bd2)
- virtual void [getBdCoefficients](#) (std::vector< std::pair< int, [Point](#) > > &coefs)
- virtual void [getBdCoefficients](#) (std::vector< std::pair< int, [Point](#) > > &coefs\_bd, std::vector< std::pair< int, [Point](#) > > &coefs\_bd2)
- virtual void [update](#) ()
- void [getBasisFunctions](#) (int index\_of\_Gauss\_point, [bool](#) &u\_dir, std::vector< [double](#) > &basisValues, std::vector< [double](#) > &basisDerivs) [const](#)
- [shared\\_ptr](#)< [SplineCurve](#) > [getSplineApproximation](#) () [const](#)
- virtual void [updateBoundaryValue](#) ([BdCondFuncor](#) \*fbd)
- int [edgeNumber](#) () [const](#)
- virtual [tpTolerances](#) [getTolerances](#) () [const](#)

## Additional Inherited Members

### 29.432.1 Detailed Description

Definition at line 68 of file SfBoundaryCondition.h.

### 29.432.2 Constructor & Destructor Documentation

29.432.2.1 `Go::SfBoundaryCondition::SfBoundaryCondition ( int edge_nmb, BdConditionType type, BdCondFuncor * fbf, std::pair< double, double > end_par, SfSolution * solution )`

29.432.2.2 `Go::SfBoundaryCondition::SfBoundaryCondition ( int edge_nmb, BdConditionType type, const Point & const_val, std::pair< double, double > end_par, SfSolution * solution )`

29.432.2.3 `virtual Go::SfBoundaryCondition::~~SfBoundaryCondition ( ) [virtual]`

### 29.432.3 Member Function Documentation

29.432.3.1 `int Go::SfBoundaryCondition::edgeNumber ( ) const`

29.432.3.2 `void Go::SfBoundaryCondition::getBasisFunctions ( int index_of_Gauss_point, bool & u_dir, std::vector< double > & basisValues, std::vector< double > & basisDerivs ) const`

29.432.3.3 `virtual void Go::SfBoundaryCondition::getBdCoefficients ( std::vector< std::pair< int, Point > > & coefs ) [virtual]`

#### Parameters

<i>coefs</i>	the boundary coefficients. Each pair consists the coefficient and its index when considered a curve.
--------------	------------------------------------------------------------------------------------------------------

Implements [Go::BlockBoundaryCondition](#).

29.432.3.4 `virtual void Go::SfBoundaryCondition::getBdCoefficients ( std::vector< std::pair< int, Point > > & coefs_bd, std::vector< std::pair< int, Point > > & coefs_bd2 ) [virtual]`

#### Parameters

<i>coefs_bd</i>	the boundary coefficients. Each pair consists the coefficient and its index when considered a curve.
<i>coefs_bd2</i>	the row of coefficients next to the boundary. Each pair consists the coefficient and its index when considered a curve.

Implements [Go::BlockBoundaryCondition](#).

29.432.3.5 virtual void Go::SfBoundaryCondition::getCoefficientsEnumeration ( std::vector< int > & local\_enumeration )  
[virtual]

Implements [Go::BlockBoundaryCondition](#).

29.432.3.6 virtual void Go::SfBoundaryCondition::getCoefficientsEnumeration ( std::vector< int > & local\_enumeration\_bd,  
std::vector< int > & local\_enumeration\_bd2 ) [virtual]

Implements [Go::BlockBoundaryCondition](#).

29.432.3.7 shared\_ptr<SplineCurve> Go::SfBoundaryCondition::getSplineApproximation ( ) const

29.432.3.8 virtual tpTolerances Go::SfBoundaryCondition::getTolerances ( ) const [virtual]

Implements [Go::BlockBoundaryCondition](#).

29.432.3.9 virtual void Go::SfBoundaryCondition::update ( ) [virtual]

Implements [Go::BlockBoundaryCondition](#).

29.432.3.10 virtual void Go::SfBoundaryCondition::updateBoundaryValue ( BdCondFuncor \* fbd ) [virtual]

Implements [Go::BlockBoundaryCondition](#).

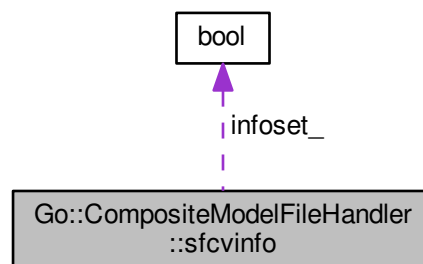
The documentation for this class was generated from the following file:

- [isogeometric\\_model/include/GoTools/isogeometric\\_model/SfBoundaryCondition.h](#)

## 29.433 Go::CompositeModelFileHandler::sfvinfo Struct Reference

```
#include <CompositeModelFileHandler.h>
```

Collaboration diagram for Go::CompositeModelFileHandler::sfvinfo:



## Public Member Functions

- [sfcvinfo](#) (int *ccm*, int *constdir*, double *constpar*, int *bd*, int *orientation*)
- [sfcvinfo](#) ()

## Public Attributes

- [bool](#) *infoset\_*
- [int](#) *ccm\_*
- [int](#) *constdir\_*
- [int](#) *bd\_*
- [int](#) *orientation\_*
- [double](#) *constpar\_*

### 29.433.1 Detailed Description

Definition at line 102 of file CompositeModelFileHandler.h.

### 29.433.2 Constructor & Destructor Documentation

29.433.2.1 `Go::CompositeModelFileHandler::sfcvinfo::sfcvinfo ( int ccm, int constdir, double constpar, int bd, int orientation ) [inline]`

Definition at line 104 of file CompositeModelFileHandler.h.

29.433.2.2 `Go::CompositeModelFileHandler::sfcvinfo::sfcvinfo ( ) [inline]`

Definition at line 113 of file CompositeModelFileHandler.h.

### 29.433.3 Member Data Documentation

29.433.3.1 `int Go::CompositeModelFileHandler::sfcvinfo::bd_`

Definition at line 119 of file CompositeModelFileHandler.h.

29.433.3.2 `int Go::CompositeModelFileHandler::sfcvinfo::ccm_`

Definition at line 119 of file CompositeModelFileHandler.h.

29.433.3.3 `int Go::CompositeModelFileHandler::sfcvinfo::constdir_`

Definition at line 119 of file CompositeModelFileHandler.h.

29.433.3.4 double Go::CompositeModelFileHandler::sfvinfo::constpar\_

Definition at line 120 of file CompositeModelFileHandler.h.

29.433.3.5 bool Go::CompositeModelFileHandler::sfvinfo::infoset\_

Definition at line 118 of file CompositeModelFileHandler.h.

29.433.3.6 int Go::CompositeModelFileHandler::sfvinfo::orientation\_

Definition at line 119 of file CompositeModelFileHandler.h.

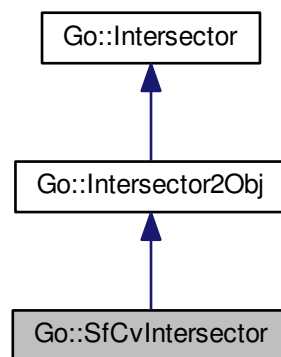
The documentation for this struct was generated from the following file:

- [compositemodel/include/GoTools/compositemodel/CompositeModelFileHandler.h](#)

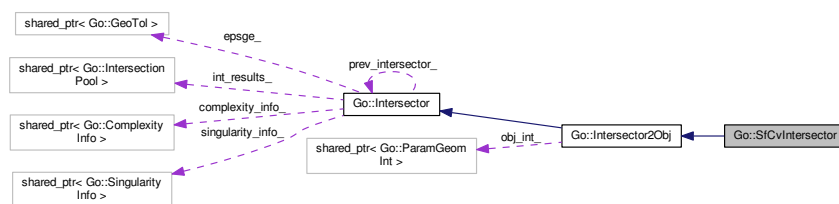
## 29.434 Go::SfCvIntersector Class Reference

```
#include <SfCvIntersector.h>
```

Inheritance diagram for Go::SfCvIntersector:



Collaboration diagram for Go::SfCvIntersector:



## Public Member Functions

- [SfCvIntersector](#) (shared\_ptr< [ParamGeomInt](#) > obj1, shared\_ptr< [ParamGeomInt](#) > obj2, double epsge, [Intersector](#) \*prev=0, int eliminated\_parameter=-1, double eliminated\_value=0)
- [SfCvIntersector](#) (shared\_ptr< [ParamGeomInt](#) > obj1, shared\_ptr< [ParamGeomInt](#) > obj2, shared\_ptr< [GeoTol](#) > epsge, [Intersector](#) \*prev=0, int eliminated\_parameter=-1, double eliminated\_value=0)
- virtual [~SfCvIntersector](#) ()  
*Destructor.*
- virtual int [numParams](#) () const
- void [postIterateBd](#) ()

## Protected Member Functions

- virtual shared\_ptr< [Intersector](#) > [lowerOrderIntersector](#) (shared\_ptr< [ParamGeomInt](#) > obj1, shared\_ptr< [ParamGeomInt](#) > obj2, [Intersector](#) \*prev=0, int eliminated\_parameter=-1, double eliminated\_value=0)
- virtual int [performRotatedBoxTest](#) (double eps1, double eps2)
- virtual bool [foundIntersectionNearBoundary](#) ()
- virtual int [simpleCase](#) ()
- virtual int [simpleCase2](#) ([Point](#) &axis1, [Point](#) &axis2)
- virtual int [checkCoincidence](#) ()
- virtual void [microCase](#) ()
- virtual int [updateIntersections](#) ()
- virtual int [repairIntersections](#) ()
- virtual int [linearCase](#) ()
- virtual int [doSubdivide](#) ()
- virtual void [postIterate](#) (int nmb\_orig, int dir=-1, bool keep\_endpt=true)

## Additional Inherited Members

### 29.434.1 Detailed Description

This class performs intersection between a parametric surface and a parametric curve.

Definition at line 53 of file [SfCvIntersector.h](#).

### 29.434.2 Constructor & Destructor Documentation

29.434.2.1 `Go::SfCvIntersector::SfCvIntersector ( shared_ptr< ParamGeomInt > obj1, shared_ptr< ParamGeomInt > obj2, double epsge, Intersector * prev = 0, int eliminated_parameter = -1, double eliminated_value = 0 )`

Constructor. One of the objects should refer to a surface, the other a curve (this is not checked compile-time, so we rely on the user to obey this rule). The last two variables are relevant only if the parent has one more parameter than the [Intersector](#) to be constructed.

#### Parameters

<i>obj1</i>	either of type <a href="#">ParamSurfaceInt</a> or <a href="#">ParamCurveInt</a> .
<i>obj2</i>	either of type <a href="#">ParamCurveInt</a> or <a href="#">ParamSurfaceInt</a> (not the same type as <i>obj1</i> ).
<i>epsgge</i>	the associated tolerance.
<i>prev</i>	the "parent" <a href="#">Intersector</a> (0 if there is no parent).
<i>eliminated_parameter</i>	the index of the parameter that was removed from the parent <i>prev</i> .
<i>eliminated_value</i>	the value of the parameter that was removed from the parent <i>prev</i> .

29.434.2.2 `Go::SfCvIntersector::SfCvIntersector ( shared_ptr< ParamGeomInt > obj1, shared_ptr< ParamGeomInt > obj2, shared_ptr< GeoTol > epsge, Intersector * prev = 0, int eliminated_parameter = -1, double eliminated_value = 0 )`

Constructor. One of the objects should refer to a surface, the other a curve (this is not checked compile-time, so we rely on the user to obey this rule). The last two variables are relevant only if the parent has one more parameter than the [Intersector](#) to be constructed.

#### Parameters

<i>obj1</i>	either of type <a href="#">ParamSurfaceInt</a> or <a href="#">ParamCurveInt</a> .
<i>obj2</i>	either of type <a href="#">ParamCurveInt</a> or <a href="#">ParamSurfaceInt</a> (not the same type as <i>obj1</i> ).
<i>epsg</i>	the associated tolerance.
<i>prev</i>	the "parent" <a href="#">Intersector</a> (0 if there is no parent).
<i>eliminated_parameter</i>	the index of the parameter that was removed from the parent <i>prev</i> .
<i>eliminated_value</i>	the value of the parameter that was removed from the parent <i>prev</i> .

29.434.2.3 `virtual Go::SfCvIntersector::~~SfCvIntersector ( ) [virtual]`

Destructor.

### 29.434.3 Member Function Documentation

29.434.3.1 `virtual int Go::SfCvIntersector::checkCoincidence ( ) [protected],[virtual]`

Implements [Go::Intersector2Obj](#).

29.434.3.2 `virtual int Go::SfCvIntersector::doSubdivide ( ) [protected],[virtual]`

Implements [Go::Intersector2Obj](#).

29.434.3.3 `virtual bool Go::SfCvIntersector::foundIntersectionNearBoundary ( ) [protected],[virtual]`

Reimplemented from [Go::Intersector2Obj](#).

29.434.3.4 `virtual int Go::SfCvIntersector::linearCase ( ) [protected],[virtual]`

Implements [Go::Intersector2Obj](#).

29.434.3.5 `virtual shared_ptr< Intersector > Go::SfCvIntersector::lowerOrderIntersector ( shared_ptr< ParamGeomInt > obj1, shared_ptr< ParamGeomInt > obj2, Intersector * prev = 0, int eliminated_parameter = -1, double eliminated_value = 0 ) [protected],[virtual]`

Implements [Go::Intersector2Obj](#).

29.434.3.6 `virtual void Go::SfCvIntersector::microCase ( ) [protected],[virtual]`

Implements [Go::Intersector2Obj](#).

29.434.3.7 `virtual int Go::SfCvIntersector::numParams ( ) const [inline],[virtual]`

Return the number of parameter directions for the object.

#### Returns

the number of parameter directions

Implements [Go::Intersector](#).

Definition at line 110 of file `SfCvIntersector.h`.

29.434.3.8 `virtual int Go::SfCvIntersector::performRotatedBoxTest ( double eps1, double eps2 ) [protected],[virtual]`

Reimplemented from [Go::Intersector2Obj](#).

29.434.3.9 `virtual void Go::SfCvIntersector::postIterate ( int nmb_orig, int dir = -1, bool keep_endpt = true ) [protected],[virtual]`

Reimplemented from [Go::Intersector2Obj](#).

29.434.3.10 `void Go::SfCvIntersector::postIterateBd ( )`

29.434.3.11 `virtual int Go::SfCvIntersector::repairIntersections ( ) [inline],[protected],[virtual]`

Implements [Go::Intersector](#).

Definition at line 139 of file `SfCvIntersector.h`.

29.434.3.12 `virtual int Go::SfCvIntersector::simpleCase ( ) [protected],[virtual]`

Reimplemented from [Go::Intersector2Obj](#).

29.434.3.13 `virtual int Go::SfCvIntersector::simpleCase2 ( Point & axis1, Point & axis2 ) [protected],[virtual]`

Reimplemented from [Go::Intersector2Obj](#).



29.434.3.14 `virtual int Go::SfCvIntersector::updateIntersections ( ) [protected],[virtual]`

Implements [Go::Intersector2Obj](#).

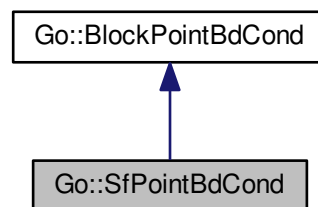
The documentation for this class was generated from the following file:

- [intersections/include/GoTools/intersections/SfCvIntersector.h](#)

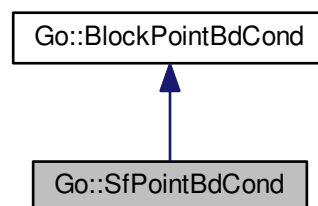
## 29.435 Go::SfPointBdCond Class Reference

```
#include <SfPointBdCond.h>
```

Inheritance diagram for Go::SfPointBdCond:



Collaboration diagram for Go::SfPointBdCond:



### Public Member Functions

- `SfPointBdCond` (int edge\_nmb, double param, Point &condition\_value)
- `virtual ~SfPointBdCond` ()
- `virtual void getCoefficientsEnumeration` (std::vector< int > &local\_enumeration) const
- `virtual Point getConditionValue` () const
- `double getParam` () const
- `virtual void getInterpolationFactors` (std::vector< std::pair< int, double > > &factors) const
- `int edgeNumber` () const

### 29.435.1 Detailed Description

Definition at line 56 of file SfPointBdCond.h.

### 29.435.2 Constructor & Destructor Documentation

29.435.2.1 `Go::SfPointBdCond::SfPointBdCond ( int edge_nmb, double param, Point & condition_value )`

29.435.2.2 `virtual Go::SfPointBdCond::~~SfPointBdCond ( ) [virtual]`

### 29.435.3 Member Function Documentation

29.435.3.1 `int Go::SfPointBdCond::edgeNumber ( ) const`

29.435.3.2 `virtual void Go::SfPointBdCond::getCoefficientsEnumeration ( std::vector< int > & local_enumeration ) const [virtual]`

Implements [Go::BlockPointBdCond](#).

29.435.3.3 `virtual Point Go::SfPointBdCond::getConditionValue ( ) const [virtual]`

Implements [Go::BlockPointBdCond](#).

29.435.3.4 `virtual void Go::SfPointBdCond::getInterpolationFactors ( std::vector< std::pair< int, double > > & factors ) const [virtual]`

Implements [Go::BlockPointBdCond](#).

29.435.3.5 `double Go::SfPointBdCond::getParam ( ) const`

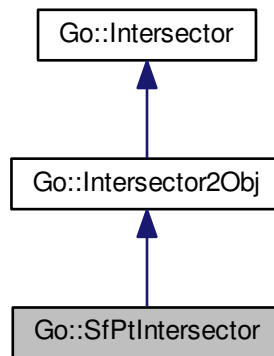
The documentation for this class was generated from the following file:

- [isogeometric\\_model/include/GoTools/isogeometric\\_model/SfPointBdCond.h](#)

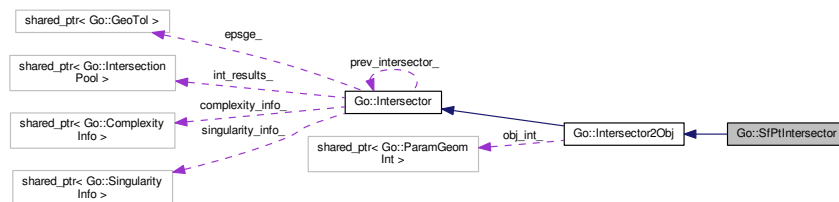
## 29.436 Go::SfPtIntersector Class Reference

```
#include <SfPtIntersector.h>
```

Inheritance diagram for Go::SfPtIntersector:



Collaboration diagram for Go::SfPtIntersector:



### Public Member Functions

- `SfPtIntersector` (`shared_ptr< ParamGeomInt > obj1`, `shared_ptr< ParamGeomInt > obj2`, `shared_ptr< GeoTol > epsge`, `Intersector *prev=0`, `int eliminated_parameter=-1`, `double eliminated_value=0`)
- virtual `~SfPtIntersector` ()  
*Destructor.*
- virtual `int numParams` () `const`

### Protected Member Functions

- virtual `shared_ptr< Intersector > lowerOrderIntersector` (`shared_ptr< ParamGeomInt > obj1`, `shared_ptr< ParamGeomInt > obj2`, `Intersector *prev=0`, `int eliminated_parameter=-1`, `double eliminated_value=0`)
- virtual `int checkCoincidence` ()
- virtual `void microCase` ()
- virtual `int updateIntersections` ()
- virtual `int repairIntersections` ()
- virtual `int linearCase` ()
- virtual `int doSubdivide` ()
- virtual `int performRotatedBoxTest` (`double eps1`, `double eps2`)

## Additional Inherited Members

### 29.436.1 Detailed Description

This class performs intersection between a parametric surface and a point

Definition at line 53 of file SfPtIntersector.h.

### 29.436.2 Constructor & Destructor Documentation

**29.436.2.1** `Go::SfPtIntersector::SfPtIntersector ( shared_ptr< ParamGeomInt > obj1, shared_ptr< ParamGeomInt > obj2, shared_ptr< GeoTol > epsge, Intersector * prev = 0, int eliminated_parameter = -1, double eliminated_value = 0 )`

Constructor. One of the objects should refer to a surface, the other a point (this is not checked compile-time, so we rely on the user to obey this rule). The last two variables are relevant only if the parent has one more parameter than the [Intersector](#) to be constructed.

#### Parameters

<i>obj1</i>	either of type <a href="#">ParamSurfaceInt</a> or <a href="#">ParamPointInt</a> .
<i>obj2</i>	either of type <a href="#">ParamPointInt</a> or <a href="#">ParamSurfaceInt</a> (not the same type as <i>obj1</i> ).
<i>epsge</i>	the associated tolerance.
<i>prev</i>	the "parent" <a href="#">Intersector</a> (0 if there is no parent).
<i>eliminated_parameter</i>	the index of the parameter that was removed from the parent <i>prev</i> .
<i>eliminated_value</i>	the value of the parameter that was removed from the parent <i>prev</i> .

**29.436.2.2** `virtual Go::SfPtIntersector::~~SfPtIntersector ( ) [virtual]`

Destructor.

### 29.436.3 Member Function Documentation

**29.436.3.1** `virtual int Go::SfPtIntersector::checkCoincidence ( ) [protected],[virtual]`

Implements [Go::Intersector2Obj](#).

**29.436.3.2** `virtual int Go::SfPtIntersector::doSubdivide ( ) [protected],[virtual]`

Implements [Go::Intersector2Obj](#).

**29.436.3.3** `virtual int Go::SfPtIntersector::linearCase ( ) [protected],[virtual]`

Implements [Go::Intersector2Obj](#).

29.436.3.4 `virtual shared_ptr<Intersector> Go::SfPtIntersector::lowerOrderIntersector ( shared_ptr< ParamGeomInt > obj1, shared_ptr< ParamGeomInt > obj2, Intersector * prev = 0, int eliminated_parameter = -1, double eliminated_value = 0 )` [protected],[virtual]

Implements [Go::Intersector2Obj](#).

29.436.3.5 `virtual void Go::SfPtIntersector::microCase ( )` [protected],[virtual]

Implements [Go::Intersector2Obj](#).

29.436.3.6 `virtual int Go::SfPtIntersector::numParams ( ) const` [inline],[virtual]

Return the number of parameter directions for the object.

#### Returns

the number of parameter directions

Implements [Go::Intersector](#).

Definition at line 87 of file SfPtIntersector.h.

29.436.3.7 `virtual int Go::SfPtIntersector::performRotatedBoxTest ( double eps1, double eps2 )` [protected],[virtual]

Reimplemented from [Go::Intersector2Obj](#).

29.436.3.8 `virtual int Go::SfPtIntersector::repairIntersections ( )` [inline],[protected],[virtual]

Implements [Go::Intersector](#).

Definition at line 106 of file SfPtIntersector.h.

29.436.3.9 `virtual int Go::SfPtIntersector::updateIntersections ( )` [protected],[virtual]

Implements [Go::Intersector2Obj](#).

The documentation for this class was generated from the following file:

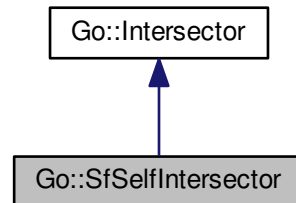
- [intersections/include/GoTools/intersections/SfPtIntersector.h](#)

## 29.437 Go::SfSelfIntersector Class Reference

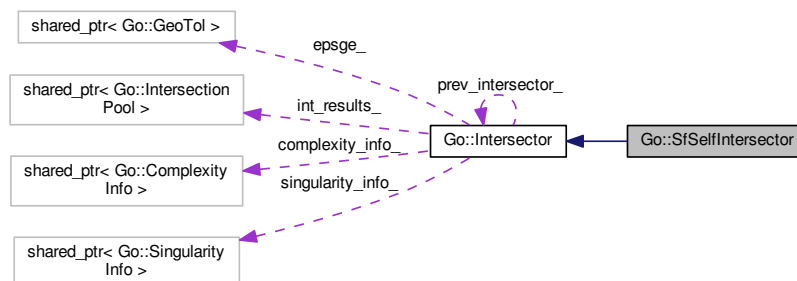
This class finds self-intersections for a parametric surface.

```
#include <SfSelfIntersector.h>
```

Inheritance diagram for Go::SfSelfIntersector:



Collaboration diagram for Go::SfSelfIntersector:



### Public Member Functions

- [SfSelfIntersector](#) (shared\_ptr< [ParamSurfaceInt](#) > surf, double epsge, [Intersector](#) \*prev=NULL)
- [SfSelfIntersector](#) (shared\_ptr< [ParamSurfaceInt](#) > surf, shared\_ptr< [GeoTol](#) > epsge, [Intersector](#) \*prev=NULL)
- virtual [~SfSelfIntersector](#) ()

*Destructor.*

- virtual void [compute](#) (bool compute\_at\_boundary=true)
- virtual int [numParams](#) () const
- void [setMaxRec](#) (int max\_rec)
- int [getNmbComplexDomain](#) ()
- virtual bool [isSelfIntersection](#) ()

*Verifies that this computation is a self intersection problem.*

## Protected Member Functions

- [bool computeG1 \(\)](#)
- [std::vector< shared\\_ptr< ParamSurfaceInt > > getNonSelfintersecting \(\)](#)
- [virtual void print\\_objs \(\)](#)
- [virtual int getBoundaryIntersections \(\)](#)
- [virtual int performInterception \(\)](#)
- [virtual int simpleCase \(\)](#)
- [virtual bool isLinear \(\)](#)
- [virtual bool complexityReduced \(\)](#)
- [virtual void handleComplexity \(\)](#)
- [virtual int checkCoincidence \(\)](#)
- [virtual void microCase \(\)](#)
- [virtual int updateIntersections \(\)](#)
- [virtual int repairIntersections \(\)](#)
- [virtual int linearCase \(\)](#)
- [virtual int doSubdivide \(\)](#)
- [virtual void printDebugInfo \(\)](#)
- [virtual void addComplexDomain \(RectDomain dom\)](#)

## Additional Inherited Members

### 29.437.1 Detailed Description

This class finds self-intersections for a parametric surface.

Definition at line 96 of file SfSelfIntersector.h.

### 29.437.2 Constructor & Destructor Documentation

**29.437.2.1** `Go::SfSelfIntersector::SfSelfIntersector ( shared_ptr< ParamSurfaceInt > surf, double epsge, Intersector * prev = NULL )`

Constructor.

#### Parameters

<i>surf</i>	the parametric surface.
<i>epsge</i>	the associated tolerance.
<i>prev</i>	the "parent" <a href="#">Intersector</a> (0 if there is no parent).

**29.437.2.2** `Go::SfSelfIntersector::SfSelfIntersector ( shared_ptr< ParamSurfaceInt > surf, shared_ptr< GeoTol > epsge, Intersector * prev = NULL )`

Constructor.

## Parameters

<i>surf</i>	the parametric surface.
<i>epsge</i>	the associated tolerance.
<i>prev</i>	the "parent" <a href="#">Intersector</a> (0 if there is no parent).

29.437.2.3 `virtual Go::SfSelfIntersector::~~SfSelfIntersector ( ) [virtual]`

Destructor.

### 29.437.3 Member Function Documentation

29.437.3.1 `virtual void Go::SfSelfIntersector::addComplexDomain ( RectDomain dom ) [inline],[protected],[virtual]`

Reimplemented from [Go::Intersector](#).

Definition at line 190 of file `SfSelfIntersector.h`.

29.437.3.2 `virtual int Go::SfSelfIntersector::checkCoincidence ( ) [inline],[protected],[virtual]`

Implements [Go::Intersector](#).

Definition at line 172 of file `SfSelfIntersector.h`.

29.437.3.3 `virtual bool Go::SfSelfIntersector::complexityReduced ( ) [inline],[protected],[virtual]`

Implements [Go::Intersector](#).

Definition at line 167 of file `SfSelfIntersector.h`.

29.437.3.4 `virtual void Go::SfSelfIntersector::compute ( bool compute_at_boundary = true ) [virtual]`

Compute topology of self-intersection.

## Parameters

<i>compute_at_boundary</i>	indicate if we want to compute at the boundary
----------------------------	------------------------------------------------

Reimplemented from [Go::Intersector](#).

29.437.3.5 `bool Go::SfSelfIntersector::computeG1 ( ) [protected]`



29.437.3.6 `virtual int Go::SfSelfIntersector::doSubdivide ( ) [inline],[protected],[virtual]`

Implements [Go::Intersector](#).

Definition at line 185 of file SfSelfIntersector.h.

29.437.3.7 `virtual int Go::SfSelfIntersector::getBoundaryIntersections ( ) [inline],[protected],[virtual]`

Implements [Go::Intersector](#).

Definition at line 155 of file SfSelfIntersector.h.

29.437.3.8 `int Go::SfSelfIntersector::getNmbComplexDomain ( ) [inline]`

Count the number of complex domains in the object.

#### Returns

The number of complex domains in the object.

Definition at line 134 of file SfSelfIntersector.h.

29.437.3.9 `std::vector<shared_ptr<ParamSurfaceInt>> Go::SfSelfIntersector::getNonSelfintersecting ( ) [inline],[protected]`

Definition at line 150 of file SfSelfIntersector.h.

29.437.3.10 `virtual void Go::SfSelfIntersector::handleComplexity ( ) [inline],[protected],[virtual]`

Implements [Go::Intersector](#).

Definition at line 170 of file SfSelfIntersector.h.

29.437.3.11 `virtual bool Go::SfSelfIntersector::isLinear ( ) [inline],[protected],[virtual]`

Implements [Go::Intersector](#).

Definition at line 164 of file SfSelfIntersector.h.

29.437.3.12 `virtual bool Go::SfSelfIntersector::isSelfIntersection ( ) [inline],[virtual]`

Verifies that this computation is a self intersection problem.

Reimplemented from [Go::Intersector](#).

Definition at line 138 of file SfSelfIntersector.h.

29.437.3.13 `virtual int Go::SfSelfIntersector::linearCase ( ) [inline],[protected],[virtual]`

Implements [Go::Intersector](#).

Definition at line 182 of file SfSelfIntersector.h.

29.437.3.14 `virtual void Go::SfSelfIntersector::microCase ( ) [inline],[protected],[virtual]`

Implements [Go::Intersector](#).

Definition at line 175 of file SfSelfIntersector.h.

29.437.3.15 `virtual int Go::SfSelfIntersector::numParams ( ) const [inline],[virtual]`

Get the number of parameter directions for the object.

Returns

the number of parameter directions (i.e. 2)

Implements [Go::Intersector](#).

Definition at line 124 of file SfSelfIntersector.h.

29.437.3.16 `virtual int Go::SfSelfIntersector::performInterception ( ) [inline],[protected],[virtual]`

Implements [Go::Intersector](#).

Definition at line 158 of file SfSelfIntersector.h.

29.437.3.17 `virtual void Go::SfSelfIntersector::print_objs ( ) [inline],[protected],[virtual]`

Implements [Go::Intersector](#).

Definition at line 153 of file SfSelfIntersector.h.

29.437.3.18 `virtual void Go::SfSelfIntersector::printDebugInfo ( ) [inline],[protected],[virtual]`

Implements [Go::Intersector](#).

Definition at line 188 of file SfSelfIntersector.h.

29.437.3.19 `virtual int Go::SfSelfIntersector::repairIntersections ( ) [protected],[virtual]`

Implements [Go::Intersector](#).

29.437.3.20 `void Go::SfSelfIntersector::setMaxRec ( int max_rec ) [inline]`

Set the maximum number of recursion levels.

## Parameters

<code>max_rec</code>	the number of levels
----------------------	----------------------

Definition at line 129 of file SfSelfIntersector.h.

**29.437.3.21** `virtual int Go::SfSelfIntersector::simpleCase ( ) [inline],[protected],[virtual]`

Implements [Go::Intersector](#).

Definition at line 161 of file SfSelfIntersector.h.

**29.437.3.22** `virtual int Go::SfSelfIntersector::updateIntersections ( ) [inline],[protected],[virtual]`

Implements [Go::Intersector](#).

Definition at line 177 of file SfSelfIntersector.h.

The documentation for this class was generated from the following file:

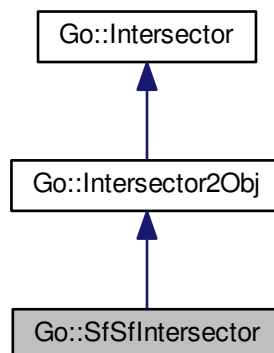
- [intersections/include/GoTools/intersections/SfSelfIntersector.h](#)

## 29.438 Go::SfSfIntersector Class Reference

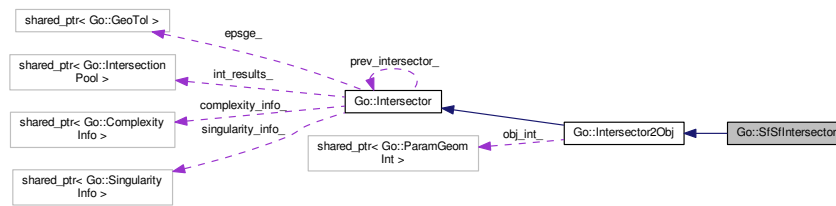
This class performs intersection between two parametric surfaces.

```
#include <SfSfIntersector.h>
```

Inheritance diagram for Go::SfSfIntersector:



Collaboration diagram for `Go::SfSfIntersector`:



## Public Member Functions

- [SfSfIntersector](#) ()  
*Default constructor.*
- [SfSfIntersector](#) (shared\_ptr< [ParamGeomInt](#) > obj1, shared\_ptr< [ParamGeomInt](#) > obj2, shared\_ptr< [GeoTol](#) > epsge, [Intersector](#) \*prev=0)
- [SfSfIntersector](#) (shared\_ptr< [ParamGeomInt](#) > obj1, shared\_ptr< [ParamGeomInt](#) > obj2, double epsge, [Intersector](#) \*prev=0)
- virtual [~SfSfIntersector](#) ()  
*Destructor.*
- virtual int [numParams](#) () const
- void [postIterate3](#) (int nmb\_orig, int dir)

## Protected Member Functions

- virtual shared\_ptr< [Intersector](#) > [lowerOrderIntersector](#) (shared\_ptr< [ParamGeomInt](#) > obj1, shared\_ptr< [ParamGeomInt](#) > obj2, [Intersector](#) \*prev=0, int eliminated\_parameter=-1, double eliminated\_value=0)
- virtual int [performRotatedBoxTest](#) (double eps1, double eps2)
- virtual bool [foundIntersectionNearBoundary](#) ()
- virtual int [performInterceptionByImplicitization](#) ()
- virtual int [interceptionBySeparationSurface](#) ()
- virtual int [simpleCase2](#) ([Point](#) &axis1, [Point](#) &axis2)
- virtual int [simpleCaseByImplicitization](#) ()
- virtual bool [complexityReduced](#) ()
- virtual void [handleComplexity](#) ()
- virtual int [checkCoincidence](#) ()
- virtual void [microCase](#) ()
- virtual bool [degTriangleSimple](#) ()
- virtual int [updateIntersections](#) ()
- virtual int [repairIntersections](#) ()
- void [repairSingularityBox](#) ()
- void [repairFalseBranchPoints](#) ()
- void [removeIsolatedPoints](#) ()
- void [repairMissingLinks](#) ()
- bool [connectIfPossible](#) (shared\_ptr< [IntersectionPoint](#) > pt1, shared\_ptr< [IntersectionPoint](#) > pt2)
- void [fixCrossingLinks](#) ()  
*Fix crossing intersection links.*
- bool [checkCloseEndpoint](#) (shared\_ptr< [IntersectionPoint](#) > pnt, shared\_ptr< [IntersectionLink](#) > link)
- void [iterateOnIntersectionPoints](#) ()
- void [getSingularityBox](#) (shared\_ptr< [IntersectionPoint](#) > sing, double frompar[], double topar[])

- `shared_ptr< SfSfIntersector > getSubIntersector (double frompar[], double topar[])`
- virtual int `linearCase ()`
- virtual int `doSubdivide ()`
- void `getApproxImplicit (std::vector< std::vector< shared_ptr< Param2FunctionInt > > > &approx_implicit, std::vector< double > &approx_implicit_err, std::vector< double > &approx_implicit_gradsize, std::vector< double > &approx_implicit_gradvar)`

## Friends

- class `SfSelfIntersector`
- class `IntersectionPool`

## Additional Inherited Members

### 29.438.1 Detailed Description

This class performs intersection between two parametric surfaces.

Definition at line 57 of file `SfSfIntersector.h`.

### 29.438.2 Constructor & Destructor Documentation

#### 29.438.2.1 Go::SfSfIntersector::SfSfIntersector ( ) [inline]

Default constructor.

Definition at line 63 of file `SfSfIntersector.h`.

#### 29.438.2.2 Go::SfSfIntersector::SfSfIntersector ( shared\_ptr< ParamGeomInt > obj1, shared\_ptr< ParamGeomInt > obj2, shared\_ptr< GeoTol > epsge, Intersector \* prev = 0 )

Constructor. Both objects should refer to surfaces (this is not checked compile-time, so we rely on the user to obey this rule).

#### Parameters

<i>obj1</i>	of type <code>ParamSurfaceInt</code> .
<i>obj2</i>	of type <code>ParamSurfaceInt</code> .
<i>epsg</i>	the associated tolerance.
<i>prev</i>	the "parent" <code>Intersector</code> (0 if there is no parent).

#### 29.438.2.3 Go::SfSfIntersector::SfSfIntersector ( shared\_ptr< ParamGeomInt > obj1, shared\_ptr< ParamGeomInt > obj2, double epsge, Intersector \* prev = 0 )

Constructor. Both objects should refer to surfaces (this is not checked compile-time, so we rely on the user to obey this rule).

## Parameters

<i>obj1</i>	of type <a href="#">ParamSurfaceInt</a> .
<i>obj2</i>	of type <a href="#">ParamSurfaceInt</a> .
<i>epsge</i>	the associated tolerance.
<i>prev</i>	the "parent" <a href="#">Intersector</a> (0 if there is no parent).

29.438.2.4 `virtual Go::SfSfIntersector::~~SfSfIntersector ( ) [virtual]`

Destructor.

### 29.438.3 Member Function Documentation

29.438.3.1 `bool Go::SfSfIntersector::checkCloseEndpoint ( shared_ptr< IntersectionPoint > pnt, shared_ptr< IntersectionLink > link ) [protected]`

29.438.3.2 `virtual int Go::SfSfIntersector::checkCoincidence ( ) [protected],[virtual]`

Implements [Go::Intersector2Obj](#).

29.438.3.3 `virtual bool Go::SfSfIntersector::complexityReduced ( ) [protected],[virtual]`

Reimplemented from [Go::Intersector2Obj](#).

29.438.3.4 `bool Go::SfSfIntersector::connectIfPossible ( shared_ptr< IntersectionPoint > pt1, shared_ptr< IntersectionPoint > pt2 ) [protected]`

29.438.3.5 `virtual bool Go::SfSfIntersector::degTriangleSimple ( ) [protected],[virtual]`

Reimplemented from [Go::Intersector](#).

29.438.3.6 `virtual int Go::SfSfIntersector::doSubdivide ( ) [protected],[virtual]`

Implements [Go::Intersector2Obj](#).

29.438.3.7 `void Go::SfSfIntersector::fixCrossingLinks ( ) [protected]`

Fix crossing intersection links.

29.438.3.8 `virtual bool Go::SfSfIntersector::foundIntersectionNearBoundary ( ) [protected],[virtual]`

Reimplemented from [Go::Intersector2Obj](#).

29.438.3.9 void Go::SfSfIntersector::getApproxImplicit ( std::vector< std::vector< shared\_ptr< Param2FunctionInt > > > & *approx\_implicit*, std::vector< double > & *approx\_implicit\_err*, std::vector< double > & *approx\_implicit\_gradsize*, std::vector< double > & *approx\_implicit\_gradvar* ) [protected]

29.438.3.10 void Go::SfSfIntersector::getSingularityBox ( shared\_ptr< IntersectionPoint > *sing*, double *frompar*[], double *topar*[] ) [protected]

29.438.3.11 shared\_ptr<SfSfIntersector> Go::SfSfIntersector::getSubIntersector ( double *frompar*[], double *topar*[] ) [protected]

29.438.3.12 virtual void Go::SfSfIntersector::handleComplexity ( ) [protected],[virtual]

Reimplemented from [Go::Intersector2Obj](#).

29.438.3.13 virtual int Go::SfSfIntersector::interceptionBySeparationSurface ( ) [protected],[virtual]

Reimplemented from [Go::Intersector2Obj](#).

29.438.3.14 void Go::SfSfIntersector::iterateOnIntersectionPoints ( ) [protected]

29.438.3.15 virtual int Go::SfSfIntersector::linearCase ( ) [protected],[virtual]

Implements [Go::Intersector2Obj](#).

29.438.3.16 virtual shared\_ptr<Intersector> Go::SfSfIntersector::lowerOrderIntersector ( shared\_ptr< ParamGeomInt > *obj1*, shared\_ptr< ParamGeomInt > *obj2*, Intersector \* *prev* = 0, int *eliminated\_parameter* = -1, double *eliminated\_value* = 0 ) [protected],[virtual]

Implements [Go::Intersector2Obj](#).

29.438.3.17 virtual void Go::SfSfIntersector::microCase ( ) [protected],[virtual]

Implements [Go::Intersector2Obj](#).

29.438.3.18 virtual int Go::SfSfIntersector::numParams ( ) const [inline],[virtual]

Return the number of parameter directions for the object.

#### Returns

the number of parameter directions

Implements [Go::Intersector](#).

Definition at line 95 of file SfSfIntersector.h.

29.438.3.19 `virtual int Go::SfSfIntersector::performInterceptionByImplicitization ( )` [protected],[virtual]

Reimplemented from [Go::Intersector2Obj](#).

29.438.3.20 `virtual int Go::SfSfIntersector::performRotatedBoxTest ( double eps1, double eps2 )` [protected],[virtual]

Reimplemented from [Go::Intersector2Obj](#).

29.438.3.21 `void Go::SfSfIntersector::postIterate3 ( int nmb_orig, int dir )` [inline]

Definition at line 98 of file SfSfIntersector.h.

29.438.3.22 `void Go::SfSfIntersector::removelsolatedPoints ( )` [protected]

29.438.3.23 `void Go::SfSfIntersector::repairFalseBranchPoints ( )` [protected]

29.438.3.24 `virtual int Go::SfSfIntersector::repairIntersections ( )` [protected],[virtual]

Implements [Go::Intersector](#).

29.438.3.25 `void Go::SfSfIntersector::repairMissingLinks ( )` [protected]

29.438.3.26 `void Go::SfSfIntersector::repairSingularityBox ( )` [protected]

29.438.3.27 `virtual int Go::SfSfIntersector::simpleCase2 ( Point & axis1, Point & axis2 )` [protected],[virtual]

Reimplemented from [Go::Intersector2Obj](#).

29.438.3.28 `virtual int Go::SfSfIntersector::simpleCaseByImplicitization ( )` [protected],[virtual]

Reimplemented from [Go::Intersector2Obj](#).

29.438.3.29 `virtual int Go::SfSfIntersector::updateIntersections ( )` [protected],[virtual]

Implements [Go::Intersector2Obj](#).

## 29.438.4 Friends And Related Function Documentation

29.438.4.1 `friend class IntersectionPool` [friend]

Definition at line 103 of file SfSfIntersector.h.



29.438.4.2 friend class SfSelfIntersector [friend]

Definition at line 60 of file SfSfIntersector.h.

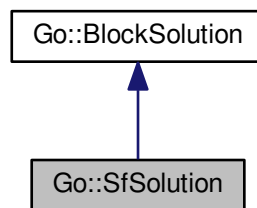
The documentation for this class was generated from the following file:

- intersections/include/GoTools/intersections/SfSfIntersector.h

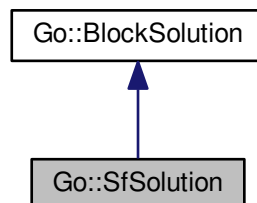
## 29.439 Go::SfSolution Class Reference

```
#include <SfSolution.h>
```

Inheritance diagram for Go::SfSolution:



Collaboration diagram for Go::SfSolution:



## Public Member Functions

- [SfSolution](#) ([IsogeometricSfBlock](#) \*parent, [shared\\_ptr](#)< [SplineSurface](#) > sol\_surf)
- virtual [~SfSolution](#) ()
- virtual [SfSolution](#) \* asSfSolution ()
- void [addBoundaryCondition](#) (int edge\_nmb, [BdConditionType](#) type, [BdCondFuncor](#) \*fbd, [std::pair](#)< [double](#), [double](#) > end\_par)
- void [addBoundaryCondition](#) (int edge\_nmb, [BdConditionType](#) type, [const Point](#) &const\_val, [std::pair](#)< [double](#), [double](#) > end\_par)
- void [addDirichletPointBdCond](#) ([double](#) param[], [Point](#) &condition\_value)
- int [getNmbOfBoundaryConditions](#) () const
- [shared\\_ptr](#)< [SfBoundaryCondition](#) > [getBoundaryCondition](#) (int index) const
- void [getEdgeBoundaryConditions](#) (int edge\_number, [std::vector](#)< [shared\\_ptr](#)< [SfBoundaryCondition](#) > > &bd\_cond) const
- void [getEdgeBoundaryConditions](#) ([std::vector](#)< [shared\\_ptr](#)< [SfBoundaryCondition](#) > > &bd\_cond) const
- virtual int [getNmbOfPointBdConditions](#) () const
- [shared\\_ptr](#)< [SfPointBdCond](#) > [getPointBdCondition](#) (int index) const
- void [getEdgePointBdConditions](#) (int edge\_number, [std::vector](#)< [shared\\_ptr](#)< [SfPointBdCond](#) > > &bd\_← cond) const
- void [getPointBdCond](#) ([std::vector](#)< [shared\\_ptr](#)< [SfPointBdCond](#) > > &bd\_cond) const
- virtual [bool](#) [matchingSplineSpace](#) ([BlockSolution](#) \*other) const
- virtual void [getMatchingCoefficients](#) ([BlockSolution](#) \*other, [std::vector](#)< [std::pair](#)< int, int > > &enumeration, int match\_pos=0) const
- virtual void [getBoundaryCoefficients](#) (int boundary, [std::vector](#)< int > &enumeration) const
- virtual void [getBoundaryCoefficients](#) (int boundary, [std::vector](#)< int > &enumeration\_bd, [std::vector](#)< int > &enumeration\_bd2) const
- virtual void [makeMatchingSplineSpace](#) ([BlockSolution](#) \*other)
- virtual void [increaseDegree](#) (int new\_degree, int pardir)
- virtual void [insertKnots](#) (const [std::vector](#)< int > &knot\_intervals, int pardir)
- virtual void [insertKnots](#) (const [std::vector](#)< [double](#) > &knots, int pardir)
- virtual void [erasePreEvaluatedBasisFunctions](#) ()
- virtual void [performPreEvaluation](#) ([std::vector](#)< [std::vector](#)< [double](#) > > &Gauss\_par)
- void [getBasisFunctions](#) (int index\_of\_Gauss\_point1, int index\_of\_Gauss\_point2, [std::vector](#)< [double](#) > &basisValues, [std::vector](#)< [double](#) > &basisDerivs\_u, [std::vector](#)< [double](#) > &basisDerivs\_v) const
- void [getBasisFunctions](#) ([double](#) param1, [double](#) param2, [std::vector](#)< [double](#) > &basisValues, [std::vector](#)< [double](#) > &basisDerivs\_u, [std::vector](#)< [double](#) > &basisDerivs\_v) const
- void [getBasisFunctionValues](#) (int basis\_func\_id\_u, int basis\_func\_id\_v, [std::vector](#)< int > &index\_of\_← Gauss\_points1, [std::vector](#)< int > &index\_of\_Gauss\_points2, [std::vector](#)< [double](#) > &basisValues, [std::vector](#)< [double](#) > &basisDerivs\_u, [std::vector](#)< [double](#) > &basisDerivs\_v) const
- void [getBasisFunctionValues](#) (int basis\_func\_id\_u, int basis\_func\_id\_v, int knot\_ind\_u, int knot\_ind\_v, [std::vector](#)< int > &index\_of\_Gauss\_points1, [std::vector](#)< int > &index\_of\_Gauss\_points2, [std::vector](#)< [double](#) > &basisValues, [std::vector](#)< [double](#) > &basisDerivs\_u, [std::vector](#)< [double](#) > &basisDerivs\_v) const
- virtual [double](#) [getJacobian](#) ([std::vector](#)< int > &index\_of\_Gauss\_point) const
- virtual void [valuesInGaussPoint](#) (const [std::vector](#)< int > &index\_of\_Gauss\_point, [std::vector](#)< [Point](#) > &derivs) const
- virtual void [setSolutionCoefficients](#) (const [std::vector](#)< [double](#) > &coefs)
- [shared\\_ptr](#)< [SplineSurface](#) > [getSolutionSurface](#) () const
- [shared\\_ptr](#)< [SplineSurface](#) > [getGeometrySurface](#) () const
- void [setMinimumDegree](#) (int degree)
- void [refineToGeometry](#) (int pardir)
- virtual int [nmbCoefs](#) () const
- virtual int [nmbCoefs](#) (int pardir) const
- virtual int [degree](#) (int pardir) const
- virtual [std::vector](#)< [double](#) > [knots](#) (int pardir) const
- virtual [std::vector](#)< [double](#) > [distinctKnots](#) (int pardir) const
- virtual [BsplineBasis](#) [basis](#) (int pardir) const

- virtual int [dimension](#) ( ) const
- virtual void [updateConditions](#) ( )
- double [getGaussParameter](#) (int index\_of\_Gauss\_point, int pardir) const
- virtual [tpTolerances](#) [getTolerances](#) ( ) const

### 29.439.1 Detailed Description

Definition at line 79 of file SfSolution.h.

### 29.439.2 Constructor & Destructor Documentation

29.439.2.1 `Go::SfSolution::SfSolution ( IsogeometricSfBlock * parent, shared_ptr< SplineSurface > sol_surf )`

29.439.2.2 `virtual Go::SfSolution::~~SfSolution ( ) [virtual]`

### 29.439.3 Member Function Documentation

29.439.3.1 `void Go::SfSolution::addBoundaryCondition ( int edge_nmb, BdConditionType type, BdCondFuncor * fbd, std::pair< double, double > end_par )`

29.439.3.2 `void Go::SfSolution::addBoundaryCondition ( int edge_nmb, BdConditionType type, const Point & const_val, std::pair< double, double > end_par )`

29.439.3.3 `void Go::SfSolution::addDirichletPointBdCond ( double param[], Point & condition_value )`

29.439.3.4 `virtual SfSolution* Go::SfSolution::asSfSolution ( ) [virtual]`

Reimplemented from [Go::BlockSolution](#).

29.439.3.5 `virtual BsplineBasis Go::SfSolution::basis ( int pardir ) const [virtual]`

Implements [Go::BlockSolution](#).

29.439.3.6 `virtual int Go::SfSolution::degree ( int pardir ) const [virtual]`

Implements [Go::BlockSolution](#).

29.439.3.7 `virtual int Go::SfSolution::dimension ( ) const [virtual]`

Implements [Go::BlockSolution](#).

29.439.3.8 `virtual std::vector<double> Go::SfSolution::distinctKnots ( int pardir ) const [virtual]`

Implements [Go::BlockSolution](#).

29.439.3.9 virtual void Go::SfSolution::erasePreEvaluatedBasisFunctions ( ) [virtual]

Implements [Go::BlockSolution](#).

29.439.3.10 void Go::SfSolution::getBasisFunctions ( int *index\_of\_Gauss\_point1*, int *index\_of\_Gauss\_point2*, std::vector< double > & *basisValues*, std::vector< double > & *basisDerivs\_u*, std::vector< double > & *basisDerivs\_v* ) const

29.439.3.11 void Go::SfSolution::getBasisFunctions ( double *param1*, double *param2*, std::vector< double > & *basisValues*, std::vector< double > & *basisDerivs\_u*, std::vector< double > & *basisDerivs\_v* ) const

29.439.3.12 void Go::SfSolution::getBasisFunctionValues ( int *basis\_func\_id\_u*, int *basis\_func\_id\_v*, std::vector< int > & *index\_of\_Gauss\_points1*, std::vector< int > & *index\_of\_Gauss\_points2*, std::vector< double > & *basisValues*, std::vector< double > & *basisDerivs\_u*, std::vector< double > & *basisDerivs\_v* ) const

29.439.3.13 void Go::SfSolution::getBasisFunctionValues ( int *basis\_func\_id\_u*, int *basis\_func\_id\_v*, int *knot\_ind\_u*, int *knot\_ind\_v*, std::vector< int > & *index\_of\_Gauss\_points1*, std::vector< int > & *index\_of\_Gauss\_points2*, std::vector< double > & *basisValues*, std::vector< double > & *basisDerivs\_u*, std::vector< double > & *basisDerivs\_v* ) const

29.439.3.14 virtual void Go::SfSolution::getBoundaryCoefficients ( int *boundary*, std::vector< int > & *enumeration* ) const [virtual]

Implements [Go::BlockSolution](#).

29.439.3.15 virtual void Go::SfSolution::getBoundaryCoefficients ( int *boundary*, std::vector< int > & *enumeration\_bd*, std::vector< int > & *enumeration\_bd2* ) const [virtual]

Implements [Go::BlockSolution](#).

29.439.3.16 shared\_ptr<SfBoundaryCondition> Go::SfSolution::getBoundaryCondition ( int *index* ) const

29.439.3.17 void Go::SfSolution::getEdgeBoundaryConditions ( int *edge\_number*, std::vector< shared\_ptr< SfBoundaryCondition > > & *bd\_cond* ) const

29.439.3.18 void Go::SfSolution::getEdgeBoundaryConditions ( std::vector< shared\_ptr< SfBoundaryCondition > > & *bd\_cond* ) const

29.439.3.19 void Go::SfSolution::getEdgePointBdConditions ( int *edge\_number*, std::vector< shared\_ptr< SfPointBdCond > > & *bd\_cond* ) const

29.439.3.20 double Go::SfSolution::getGaussParameter ( int *index\_of\_Gauss\_point*, int *pardir* ) const

29.439.3.21 shared\_ptr<SplineSurface> Go::SfSolution::getGeometrySurface ( ) const

29.439.3.22 virtual double Go::SfSolution::getJacobian ( std::vector< int > & *index\_of\_Gauss\_point* ) const [virtual]

Implements [Go::BlockSolution](#).

29.439.3.23 virtual void Go::SfSolution::getMatchingCoefficients ( **BlockSolution** \* *other*, std::vector< std::pair< int, int > > & *enumeration*, int *match\_pos* = 0 ) const [virtual]

Implements [Go::BlockSolution](#).

29.439.3.24 int Go::SfSolution::getNmbOfBoundaryConditions ( ) const

29.439.3.25 virtual int Go::SfSolution::getNmbOfPointBdConditions ( ) const [virtual]

Implements [Go::BlockSolution](#).

29.439.3.26 void Go::SfSolution::getPointBdCond ( std::vector< shared\_ptr< **SfPointBdCond** > > & *bd\_cond* ) const

29.439.3.27 shared\_ptr<**SfPointBdCond**> Go::SfSolution::getPointBdCondition ( int *index* ) const

29.439.3.28 shared\_ptr<**SplineSurface**> Go::SfSolution::getSolutionSurface ( ) const

29.439.3.29 virtual tpTolerances Go::SfSolution::getTolerances ( ) const [virtual]

Implements [Go::BlockSolution](#).

29.439.3.30 virtual void Go::SfSolution::increaseDegree ( int *new\_degree*, int *pardir* ) [virtual]

Implements [Go::BlockSolution](#).

29.439.3.31 virtual void Go::SfSolution::insertKnots ( const std::vector< int > & *knot\_intervals*, int *pardir* ) [virtual]

Implements [Go::BlockSolution](#).

29.439.3.32 virtual void Go::SfSolution::insertKnots ( const std::vector< double > & *knots*, int *pardir* ) [virtual]

Implements [Go::BlockSolution](#).

29.439.3.33 virtual std::vector<double> Go::SfSolution::knots ( int *pardir* ) const [virtual]

Implements [Go::BlockSolution](#).

29.439.3.34 virtual void Go::SfSolution::makeMatchingSplineSpace ( **BlockSolution** \* *other* ) [virtual]

Implements [Go::BlockSolution](#).

29.439.3.35 `virtual bool Go::SfSolution::matchingSplineSpace ( BlockSolution * other ) const` [virtual]

Implements [Go::BlockSolution](#).

29.439.3.36 `virtual int Go::SfSolution::nmbCoefs ( ) const` [virtual]

Implements [Go::BlockSolution](#).

29.439.3.37 `virtual int Go::SfSolution::nmbCoefs ( int pardir ) const` [virtual]

Implements [Go::BlockSolution](#).

29.439.3.38 `virtual void Go::SfSolution::performPreEvaluation ( std::vector< std::vector< double > > & Gauss_par )`  
[virtual]

Implements [Go::BlockSolution](#).

29.439.3.39 `void Go::SfSolution::refineToGeometry ( int pardir )`

29.439.3.40 `void Go::SfSolution::setMinimumDegree ( int degree )`

29.439.3.41 `virtual void Go::SfSolution::setSolutionCoefficients ( const std::vector< double > & coefs )` [virtual]

Implements [Go::BlockSolution](#).

29.439.3.42 `virtual void Go::SfSolution::updateConditions ( )` [virtual]

Implements [Go::BlockSolution](#).

29.439.3.43 `virtual void Go::SfSolution::valuesInGaussPoint ( const std::vector< int > & index_of_Gauss_point,`  
`std::vector< Point > & derivs ) const` [virtual]

Implements [Go::BlockSolution](#).

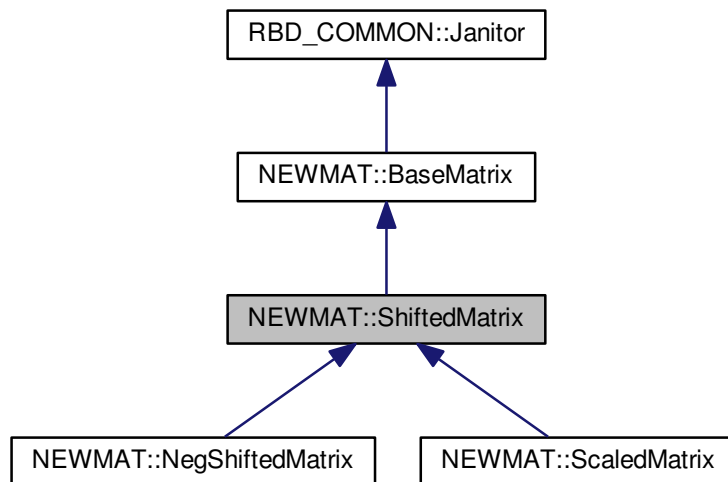
The documentation for this class was generated from the following file:

- [isogeometric\\_model/include/GoTools/isogeometric\\_model/SfSolution.h](#)

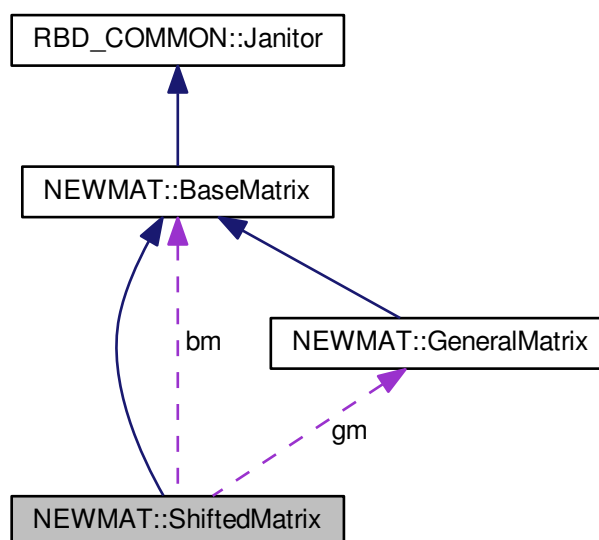
## 29.440 NEWMAT::ShiftedMatrix Class Reference

```
#include <newmat.h>
```

Inheritance diagram for NEWMAT::ShiftedMatrix:



Collaboration diagram for NEWMAT::ShiftedMatrix:



## Public Member Functions

- [~ShiftedMatrix](#) ()
- [GeneralMatrix \\* Evaluate](#) ([MatrixType](#) mt=[MatrixTypeUnSp](#))

## Protected Member Functions

- [ShiftedMatrix](#) ([const BaseMatrix \\* bmx](#), [Real](#) fx)
- [int search](#) ([const BaseMatrix \\*](#)) [const](#)

## Protected Attributes

- [union](#) {  
     [const BaseMatrix \\* bm](#)  
     [GeneralMatrix \\* gm](#)  
 };
- [Real](#) f

## Friends

- [class BaseMatrix](#)
- [class GeneralMatrix](#)
- [class GenericMatrix](#)
- [ShiftedMatrix operator+](#) ([Real](#) f, [const BaseMatrix &BM](#))

### 29.440.1 Detailed Description

Definition at line 1311 of file newmat.h.

### 29.440.2 Constructor & Destructor Documentation

29.440.2.1 [NEWMAT::ShiftedMatrix::ShiftedMatrix](#) ([const BaseMatrix \\* bmx](#), [Real](#) fx) [[inline](#)], [[protected](#)]

Definition at line 1316 of file newmat.h.

29.440.2.2 [NEWMAT::ShiftedMatrix::~~ShiftedMatrix](#) ( ) [[inline](#)]

Definition at line 1322 of file newmat.h.

### 29.440.3 Member Function Documentation

29.440.3.1 [GeneralMatrix \\* ShiftedMatrix::Evaluate](#) ([MatrixType](#) mt = [MatrixTypeUnSp](#)) [[virtual](#)]

Implements [NEWMAT::BaseMatrix](#).

Reimplemented in [NEWMAT::ScaledMatrix](#), and [NEWMAT::NegShiftedMatrix](#).

Definition at line 94 of file newmat5.cpp.



29.440.3.2 `int ShiftedMatrix::search ( const BaseMatrix * s ) const` [protected], [virtual]

Implements [NEWMAT::BaseMatrix](#).

Definition at line 376 of file `newmat4.cpp`.

## 29.440.4 Friends And Related Function Documentation

29.440.4.1 `friend class BaseMatrix` [friend]

Definition at line 1318 of file `newmat.h`.

29.440.4.2 `friend class GeneralMatrix` [friend]

Definition at line 1319 of file `newmat.h`.

29.440.4.3 `friend class GenericMatrix` [friend]

Definition at line 1320 of file `newmat.h`.

29.440.4.4 `ShiftedMatrix operator+ ( Real f, const BaseMatrix & BM )` [friend]

Definition at line 1776 of file `newmat.h`.

## 29.440.5 Member Data Documentation

29.440.5.1 `union { ... }` [protected]

29.440.5.2 `const BaseMatrix* NEWMAT::ShiftedMatrix::bm`

Definition at line 1314 of file `newmat.h`.

29.440.5.3 `Real NEWMAT::ShiftedMatrix::f` [protected]

Definition at line 1315 of file `newmat.h`.

29.440.5.4 `GeneralMatrix* NEWMAT::ShiftedMatrix::gm`

Definition at line 1314 of file `newmat.h`.

The documentation for this class was generated from the following files:

- [newmat/include/newmat.h](#)
- [newmat/src/newmat4.cpp](#)
- [newmat/src/newmat5.cpp](#)

## 29.441 Go::sideConstraint Struct Reference

```
#include <ConstraintDefinitions.h>
```

### Public Attributes

- int [dim\\_](#)  
*Dimension of coefficient (max 3).*
- `std::vector< std::pair< int, double > >` [factor\\_](#)
- `double` [constant\\_term\\_](#) [3]  
*The constant term in the current equation.*

### 29.441.1 Detailed Description

Struct defining linear side constraints between control points in a surface. The elements of `factor_` correspond to a linear combination of control points on the left side of the equation, whilst `constant_term_` denotes the right side of the equation.

Definition at line 54 of file `ConstraintDefinitions.h`.

### 29.441.2 Member Data Documentation

#### 29.441.2.1 `double` `Go::sideConstraint::constant_term_` [3]

The constant term in the current equation.

Definition at line 63 of file `ConstraintDefinitions.h`.

#### 29.441.2.2 `int` `Go::sideConstraint::dim_`

Dimension of coefficient (max 3).

Definition at line 57 of file `ConstraintDefinitions.h`.

#### 29.441.2.3 `std::vector<std::pair<int, double> >` `Go::sideConstraint::factor_`

For each coefficient involved in the constraint, the index of the coefficient is given and the factor corresponding to the coefficient in the equation.

Definition at line 61 of file `ConstraintDefinitions.h`.

The documentation for this struct was generated from the following file:

- `gotools-core/include/GoTools/creators/ConstraintDefinitions.h`

## 29.442 Go::sideConstraintSet Struct Reference

```
#include <ConstraintDefinitions.h>
```

### Public Member Functions

- [sideConstraintSet](#) ()  
*Default constructor.*
- [sideConstraintSet](#) (int *dim*)

### Public Attributes

- int *dim\_*  
*Dimension of coefficient (max 3).*
- `std::vector< std::pair< std::pair< int, int >, double > >` *factor\_*
- `double` *constant\_term\_* [3]

### 29.442.1 Detailed Description

Struct defining linear side constraints between control points in a set of surfaces. The elements of *factor\_* correspond to a linear combination of control points on the left side of the equation, whilst *constant\_term\_* denotes the right side of the equation.

Definition at line 72 of file `ConstraintDefinitions.h`.

### 29.442.2 Constructor & Destructor Documentation

**29.442.2.1** `Go::sideConstraintSet::sideConstraintSet ( )` [*inline*]

Default constructor.

Definition at line 84 of file `ConstraintDefinitions.h`.

**29.442.2.2** `Go::sideConstraintSet::sideConstraintSet ( int dim )` [*inline*]

Constructor.

#### Parameters

<i>dim</i>	dimension of geometric space.
------------	-------------------------------

Definition at line 92 of file `ConstraintDefinitions.h`.

### 29.442.3 Member Data Documentation

### 29.442.3.1 `double Go::sideConstraintSet::constant_term_[3]`

The constant term in the current equation, on the right side of the equation. For  $\text{dim} < 3$  not all elements are used.

Definition at line 82 of file `ConstraintDefinitions.h`.

### 29.442.3.2 `int Go::sideConstraintSet::dim_`

Dimension of coefficient (max 3).

Definition at line 75 of file `ConstraintDefinitions.h`.

### 29.442.3.3 `std::vector<std::pair<std::pair<int,int>, double> > Go::sideConstraintSet::factor_`

For each coefficient involved in the constraint, the index of the surface, the index of the coefficient and the factor corresponding to the coefficient in the equation is given.

Definition at line 79 of file `ConstraintDefinitions.h`.

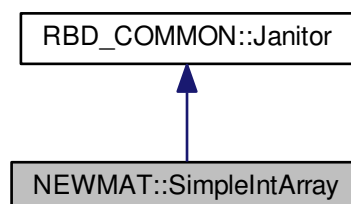
The documentation for this struct was generated from the following file:

- [gotools-core/include/GoTools/creators/ConstraintDefinitions.h](#)

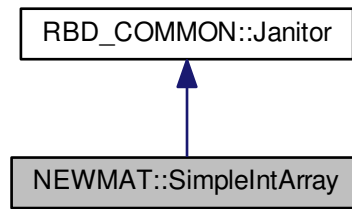
## 29.443 NEWMAT::SimpleIntArray Class Reference

```
#include <newmat.h>
```

Inheritance diagram for `NEWMAT::SimpleIntArray`:



Collaboration diagram for NEWMAT::SimpleIntArray:



### Public Member Functions

- [SimpleIntArray](#) (int xn)
- [~SimpleIntArray](#) ()
- int & [operator\[\]](#) (int i)
- int [operator\[\]](#) (int i) const
- void [operator=](#) (int ai)
- void [operator=](#) (const SimpleIntArray &b)
- [SimpleIntArray](#) (const SimpleIntArray &b)
- int [Size](#) () const
- int \* [Data](#) ()
- const int \* [Data](#) () const
- void [ReSize](#) (int i, bool keep=false)
- void [CleanUp](#) ()

### Protected Attributes

- int \* [a](#)
- int [n](#)

#### 29.443.1 Detailed Description

Definition at line 1566 of file newmat.h.

#### 29.443.2 Constructor & Destructor Documentation

##### 29.443.2.1 SimpleIntArray::SimpleIntArray ( int xn )

Definition at line 856 of file newmat4.cpp.

##### 29.443.2.2 SimpleIntArray::~~SimpleIntArray ( )

Definition at line 865 of file newmat4.cpp.

29.443.2.3 `SimpleIntArray::SimpleIntArray ( const SimpleIntArray & b )`

Definition at line 907 of file newmat4.cpp.

### 29.443.3 Member Function Documentation

29.443.3.1 `void NEWMAT::SimpleIntArray::CleanUp ( ) [inline], [virtual]`

Reimplemented from [RBD\\_COMMON::Janitor](#).

Definition at line 1589 of file newmat.h.

29.443.3.2 `int* NEWMAT::SimpleIntArray::Data ( ) [inline]`

Definition at line 1584 of file newmat.h.

29.443.3.3 `const int* NEWMAT::SimpleIntArray::Data ( ) const [inline]`

Definition at line 1585 of file newmat.h.

29.443.3.4 `void SimpleIntArray::operator= ( int ai )`

Definition at line 891 of file newmat4.cpp.

29.443.3.5 `void SimpleIntArray::operator= ( const SimpleIntArray & b )`

Definition at line 897 of file newmat4.cpp.

29.443.3.6 `int & SimpleIntArray::operator[] ( int i )`

Definition at line 872 of file newmat4.cpp.

29.443.3.7 `int NEWMAT::SimpleIntArray::operator[] ( int i ) const`

29.443.3.8 `void SimpleIntArray::ReSize ( int i, bool keep = false )`

Definition at line 921 of file newmat4.cpp.

29.443.3.9 `int NEWMAT::SimpleIntArray::Size ( ) const [inline]`

Definition at line 1582 of file newmat.h.

### 29.443.4 Member Data Documentation

29.443.4.1 `int*` `NEWMAT::SimpleIntArray::a` `[protected]`

Definition at line 1569 of file `newmat.h`.

29.443.4.2 `int` `NEWMAT::SimpleIntArray::n` `[protected]`

Definition at line 1570 of file `newmat.h`.

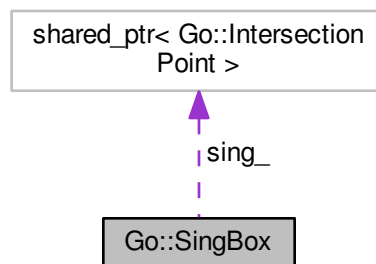
The documentation for this class was generated from the following files:

- [newmat/include/newmat.h](#)
- [newmat/src/newmat4.cpp](#)

## 29.444 Go::SingBox Struct Reference

```
#include <SfSelfIntersector.h>
```

Collaboration diagram for `Go::SingBox`:



### Public Member Functions

- [SingBox](#) (`std::vector< std::pair< double, int > >` `box`, `shared_ptr< IntersectionPoint >` `sing`)

### Public Attributes

- `std::vector< std::pair< double, int > >` `box_limit_`
- `shared_ptr< IntersectionPoint >` `sing_`

### 29.444.1 Detailed Description

Definition at line 54 of file SfSelfIntersector.h.

### 29.444.2 Constructor & Destructor Documentation

29.444.2.1 `Go::SingBox::SingBox ( std::vector< std::pair< double, int > > box, shared_ptr< IntersectionPoint > sing ) [inline]`

Definition at line 59 of file SfSelfIntersector.h.

### 29.444.3 Member Data Documentation

29.444.3.1 `std::vector<std::pair<double, int> > Go::SingBox::box_limit_`

Definition at line 56 of file SfSelfIntersector.h.

29.444.3.2 `shared_ptr<IntersectionPoint> Go::SingBox::sing_`

Definition at line 57 of file SfSelfIntersector.h.

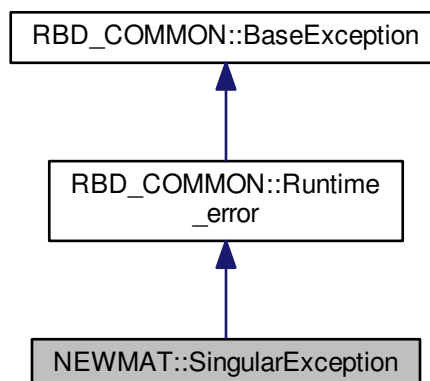
The documentation for this struct was generated from the following file:

- [intersections/include/GoTools/intersections/SfSelfIntersector.h](#)

## 29.445 NEWMAT::SingularException Class Reference

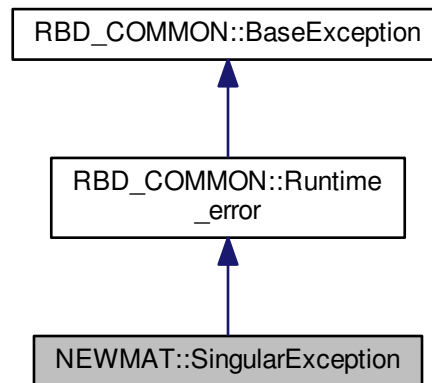
```
#include <newmat.h>
```

Inheritance diagram for NEWMAT::SingularException:





Collaboration diagram for NEWMAT::SingularException:



### Public Member Functions

- [SingularException](#) (const GeneralMatrix &A)

### Static Public Attributes

- static unsigned long [Select](#)

### Additional Inherited Members

#### 29.445.1 Detailed Description

Definition at line 1610 of file newmat.h.

#### 29.445.2 Constructor & Destructor Documentation

##### 29.445.2.1 SingularException::SingularException ( const GeneralMatrix & A )

Definition at line 54 of file newmatex.cpp.

#### 29.445.3 Member Data Documentation

##### 29.445.3.1 unsigned long SingularException::Select [static]

Definition at line 1613 of file newmat.h.

The documentation for this class was generated from the following files:

- newmat/include/newmat.h
- newmat/src/newmatex.cpp

## 29.446 Go::SingularityInfo Class Reference

```
#include <SingularityInfo.h>
```

### Public Member Functions

- [SingularityInfo](#) ()  
*Default constructor.*
- [SingularityInfo](#) (shared\_ptr< [SingularityInfo](#) > previous, bool use\_previous=false)
- [SingularityInfo](#) (shared\_ptr< [SingularityInfo](#) > previous, int missing\_dir)
- bool [hasPoint](#) ()
- int [getNmbPoints](#) (int nmb\_par)
- double [getParam](#) (int dir)
- void [addIterationCount](#) ()  
*Increment the iteration count.*
- void [addSimpleCount](#) (bool simple1, bool simple2)
- void [setSingularPoint](#) (double \*par, int nmb\_par)
- void [addSingularPoint](#) (double \*par, int nmb\_par)
- [Point](#) [getPoint](#) (int idx, int nmb\_par)
- bool [iterationDone](#) ()
- bool [iterationSucceed](#) ()
- int [nmbSimple1](#) ()
- int [nmbSimple2](#) ()
- void [setHighPriSing](#) (double \*par, int nmb\_par)
- void [setHighPriSingType](#) ([SingularityClassification](#) type)
- [SingularityClassification](#) [hasHighPriSing](#) ()
- double [getHighPriSing](#) (int idx)
- std::vector< double > [getHighPriSing](#) ()
- void [cleanUp](#) (std::vector< double > start, std::vector< double > end, double tol)

### 29.446.1 Detailed Description

This class contains information about singularities in an intersection problem

Definition at line 55 of file SingularityInfo.h.

### 29.446.2 Constructor & Destructor Documentation

#### 29.446.2.1 Go::SingularityInfo::SingularityInfo ( )

Default constructor.

#### 29.446.2.2 Go::SingularityInfo::SingularityInfo ( shared\_ptr< SingularityInfo > previous, bool use\_previous = false )

Inheritance constructor

## Parameters

<i>previous</i>	shared pointer to a <a href="#">SingularityInfo</a> object
<i>use_previous</i>	flag indicating if <i>previous</i> comes from a previous intersection problem

29.446.2.3 `Go::SingularityInfo::SingularityInfo ( shared_ptr< SingularityInfo > previous, int missing_dir )`

Inheritance constructor

## Parameters

<i>previous</i>	shared pointer to a <a href="#">SingularityInfo</a> object
<i>missing_dir</i>	if the dimension of the intersection problem has been reduced from previous, the index tells us which parameter direction that was removed. Indexing starts at 0.

## 29.446.3 Member Function Documentation

29.446.3.1 `void Go::SingularityInfo::addIterationCount ( ) [inline]`

Increment the iteration count.

Definition at line 94 of file SingularityInfo.h.

29.446.3.2 `void Go::SingularityInfo::addSimpleCount ( bool simple1, bool simple2 ) [inline]`

Increment the number of times the intersection objects are characterized as simple.

## Parameters

<i>simple1</i>	if <code>true</code> , increment for the first intersection object
<i>simple2</i>	if <code>true</code> , increment for the second intersection object

Definition at line 106 of file SingularityInfo.h.

29.446.3.3 `void Go::SingularityInfo::addSingularPoint ( double * par, int nmb_par )`

Add a singular point

## Parameters

<i>par</i>	parameter array of the singular point
<i>nmb_par</i>	number of parameters in <i>par</i>

29.446.3.4 `void Go::SingularityInfo::cleanUp ( std::vector< double > start, std::vector< double > end, double tol )`

29.446.3.5 `double Go::SingularityInfo::getHighPriSing ( int idx )` `[inline]`

Get the high priority singularity

#### Parameters

<i>idx</i>	index specifying a parameter of the singular point
------------	----------------------------------------------------

#### Returns

the parameter of the singular point in the *idx* direction

Definition at line 171 of file SingularityInfo.h.

29.446.3.6 `std::vector<double> Go::SingularityInfo::getHighPriSing ( )` `[inline]`

Get the high priority singularity

#### Returns

the vector of parameters of the singular point

Definition at line 176 of file SingularityInfo.h.

29.446.3.7 `int Go::SingularityInfo::getNmbPoints ( int nmb_par )` `[inline]`

Get number of singular points

#### Parameters

<i>nmb_par</i>	the number of parameters
----------------	--------------------------

#### Returns

the number of singular points

Definition at line 85 of file SingularityInfo.h.

29.446.3.8 `double Go::SingularityInfo::getParam ( int dir )`

Get a parameter for the last found singularity or closest point

## Parameters

<i>dir</i>	the index of the parameter direction
------------	--------------------------------------

## Returns

the parameter value in the *dir* direction

29.446.3.9 Point Go::SingularityInfo::getPoint ( int *idx*, int *nmb\_par* )

Get a singular point

## Parameters

<i>idx</i>	index to the singular point in question
<i>nmb_par</i>	number of parameters

## Returns

the singular [Point](#)

## 29.446.3.10 SingularityClassification Go::SingularityInfo::hasHighPriSing ( ) [inline]

Get info on high priority singularities

## Returns

the singularity classification

Definition at line 164 of file SingularityInfo.h.

## 29.446.3.11 bool Go::SingularityInfo::hasPoint ( ) [inline]

Check if singular point has been found

## Returns

`true` if a singular point has been found, `false` otherwise

Definition at line 79 of file SingularityInfo.h.

## 29.446.3.12 bool Go::SingularityInfo::iterationDone ( ) [inline]

Check if the iteration is done

## Returns

`true` if the iteration is done, `false` otherwise

Definition at line 132 of file SingularityInfo.h.

29.446.3.13 `bool Go::SingularityInfo::iterationSucceed ( ) [inline]`

Check if the iteration succeeded

#### Returns

`true` if the iteration succeeded, `false` otherwise

Definition at line 137 of file SingularityInfo.h.

29.446.3.14 `int Go::SingularityInfo::nmbSimple1 ( ) [inline]`

Get the number of times the first intersection object has been characterized as simple.

#### Returns

the number of times

Definition at line 143 of file SingularityInfo.h.

29.446.3.15 `int Go::SingularityInfo::nmbSimple2 ( ) [inline]`

Get the number of times the second intersection object has been characterized as simple.

#### Returns

the number of times

Definition at line 149 of file SingularityInfo.h.

29.446.3.16 `void Go::SingularityInfo::setHighPriSing ( double * par, int nmb_par )`

Set a high priority singular point

#### Parameters

<i>par</i>	parameter array of the singular point
<i>nmb_par</i>	number of parameters in <i>par</i>

29.446.3.17 `void Go::SingularityInfo::setHighPriSingType ( SingularityClassification type ) [inline]`

Set the high priority singular point type

#### Parameters

<i>type</i>	the type of the singular point
-------------	--------------------------------

Definition at line 159 of file SingularityInfo.h.

29.446.3.18 void Go::SingularityInfo::setSingularPoint ( double \* *par*, int *nmb\_par* )

Set a singular point

#### Parameters

<i>par</i>	parameter array of the singular point
<i>nmb_par</i>	number of parameters in <i>par</i>

The documentation for this class was generated from the following file:

- [intersections/include/GoTools/intersections/SingularityInfo.h](#)

## 29.447 Go::SingUnion Struct Reference

```
#include <SfSelfIntersector.h>
```

### Public Member Functions

- [SingUnion](#) (double box[], std::vector< int > *idx*)
- [bool isInside](#) (double \**mima*)

### Public Attributes

- [double limit\\_](#) [4]
- [std::vector< int > singbox\\_idx\\_](#)

### 29.447.1 Detailed Description

Definition at line 67 of file SfSelfIntersector.h.

### 29.447.2 Constructor & Destructor Documentation

29.447.2.1 Go::SingUnion::SingUnion ( double *box*[], std::vector< int > *idx* ) [inline]

Definition at line 72 of file SfSelfIntersector.h.

### 29.447.3 Member Function Documentation

29.447.3.1 bool Go::SingUnion::isInside ( double \* *mima* ) [inline]

Definition at line 79 of file SfSelfIntersector.h.

## 29.447.4 Member Data Documentation

### 29.447.4.1 `double` `Go::SingUnion::limit_[4]`

Definition at line 69 of file `SfSelfIntersector.h`.

### 29.447.4.2 `std::vector<int>` `Go::SingUnion::singbox_idx_`

Definition at line 70 of file `SfSelfIntersector.h`.

The documentation for this struct was generated from the following file:

- [intersections/include/GoTools/intersections/SfSelfIntersector.h](#)

## 29.448 SISLbox Struct Reference

```
#include <sisl.h>
```

### Public Attributes

- `double *` `emax`
- `double *` `emin`
- `int` `imin`
- `int` `imax`
- `double *` `e2max` [3]
- `double *` `e2min` [3]
- `double` `etol` [3]

### 29.448.1 Detailed Description

Definition at line 107 of file `sisl.h`.

## 29.448.2 Member Data Documentation

### 29.448.2.1 `double*` `SISLbox::e2max[3]`

Definition at line 114 of file `sisl.h`.

### 29.448.2.2 `double*` `SISLbox::e2min[3]`

Definition at line 115 of file `sisl.h`.



**29.448.2.3** double\* SISLbox::emax

Definition at line 109 of file sisl.h.

**29.448.2.4** double\* SISLbox::emin

Definition at line 110 of file sisl.h.

**29.448.2.5** double SISLbox::etol[3]

Definition at line 116 of file sisl.h.

**29.448.2.6** int SISLbox::imax

Definition at line 112 of file sisl.h.

**29.448.2.7** int SISLbox::imin

Definition at line 111 of file sisl.h.

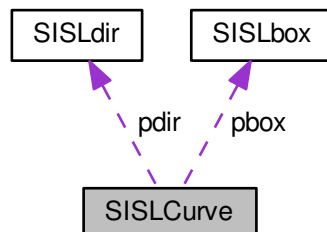
The documentation for this struct was generated from the following file:

- [sisl/include/sisl.h](#)

**29.449** SISLCurve Struct Reference

```
#include <sisl.h>
```

Collaboration diagram for SISLCurve:



## Public Attributes

- int [ik](#)
- int [in](#)
- double \* [et](#)
- double \* [ecoef](#)
- double \* [rcoef](#)
- int [ikind](#)
- int [idim](#)
- int [icopy](#)
- SISLdir \* [pdir](#)
- SISLbox \* [pbox](#)
- int [cuopen](#)

### 29.449.1 Detailed Description

Definition at line 122 of file sisl.h.

### 29.449.2 Member Data Documentation

#### 29.449.2.1 int SISLCurve::cuopen

Definition at line 148 of file sisl.h.

#### 29.449.2.2 double\* SISLCurve::ecoef

Definition at line 127 of file sisl.h.

#### 29.449.2.3 double\* SISLCurve::et

Definition at line 126 of file sisl.h.

#### 29.449.2.4 int SISLCurve::icopy

Definition at line 137 of file sisl.h.

#### 29.449.2.5 int SISLCurve::idim

Definition at line 135 of file sisl.h.

#### 29.449.2.6 int SISLCurve::ik

Definition at line 124 of file sisl.h.

29.449.2.7 int SISLCurve::ikind

Definition at line 130 of file sisl.h.

29.449.2.8 int SISLCurve::in

Definition at line 125 of file sisl.h.

29.449.2.9 SISLbox\* SISLCurve::pbox

Definition at line 146 of file sisl.h.

29.449.2.10 SISLdir\* SISLCurve::pdir

Definition at line 144 of file sisl.h.

29.449.2.11 double\* SISLCurve::rcoef

Definition at line 128 of file sisl.h.

The documentation for this struct was generated from the following file:

- [sisl/include/sisl.h](#)

## 29.450 SISLdir Struct Reference

```
#include <sisl.h>
```

### Public Attributes

- int [igtpi](#)
- double \* [ecoef](#)
- double [aang](#)
- double \* [esmooth](#)

### 29.450.1 Detailed Description

Definition at line 78 of file sisl.h.

## 29.450.2 Member Data Documentation

### 29.450.2.1 double SISLdir::aang

Definition at line 90 of file sisl.h.

### 29.450.2.2 double\* SISLdir::ecoef

Definition at line 89 of file sisl.h.

### 29.450.2.3 double\* SISLdir::esmooth

Definition at line 92 of file sisl.h.

### 29.450.2.4 int SISLdir::igtpt

Definition at line 80 of file sisl.h.

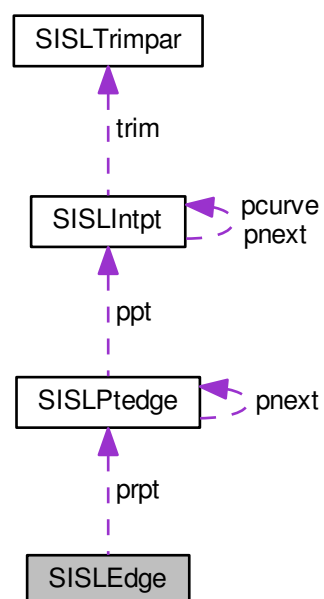
The documentation for this struct was generated from the following file:

- [sisl/include/sisl.h](#)

## 29.451 SISLEdge Struct Reference

```
#include <sislP.h>
```

Collaboration diagram for SISLEdge:



**Public Attributes**

- int [iedge](#)
- int [ipoint](#)
- [SISLPtedge](#) \*\* [prpt](#)

**29.451.1 Detailed Description**

Definition at line 323 of file [sisIP.h](#).

**29.451.2 Member Data Documentation****29.451.2.1 int SISLEdge::iedge**

Definition at line 325 of file [sisIP.h](#).

**29.451.2.2 int SISLEdge::ipoint**

Definition at line 326 of file [sisIP.h](#).

**29.451.2.3 SISLPtedge\*\* SISLEdge::prpt**

Definition at line 328 of file [sisIP.h](#).

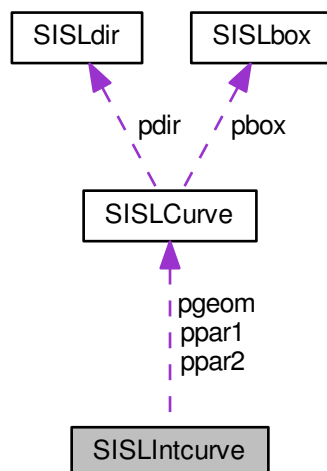
The documentation for this struct was generated from the following file:

- [sisl/include/sisIP.h](#)

**29.452 SISLIntcurve Struct Reference**

```
#include <sisl.h>
```

Collaboration diagram for SISLIntcurve:



## Public Attributes

- int [ipoint](#)
- int [ipar1](#)
- int [ipar2](#)
- double \* [epar1](#)
- double \* [epar2](#)
- SISLCurve \* [pgeom](#)
- SISLCurve \* [ppar1](#)
- SISLCurve \* [ppar2](#)
- int [itype](#)
- int [pretop](#) [4]

### 29.452.1 Detailed Description

Definition at line 205 of file sisl.h.

### 29.452.2 Member Data Documentation

#### 29.452.2.1 double\* SISLIntcurve::epar1

Definition at line 212 of file sisl.h.

#### 29.452.2.2 double\* SISLIntcurve::epar2

Definition at line 215 of file sisl.h.

#### 29.452.2.3 int SISLIntcurve::ipar1

Definition at line 208 of file sisl.h.

#### 29.452.2.4 int SISLIntcurve::ipar2

Definition at line 210 of file sisl.h.

#### 29.452.2.5 int SISLIntcurve::ipoint

Definition at line 207 of file sisl.h.

#### 29.452.2.6 int SISLIntcurve::itype

Definition at line 231 of file sisl.h.

## 29.452.2.7 SISLCurve\* SISLIntcurve::pgeom

Definition at line 220 of file sisl.h.

## 29.452.2.8 SISLCurve\* SISLIntcurve::ppar1

Definition at line 223 of file sisl.h.

## 29.452.2.9 SISLCurve\* SISLIntcurve::ppar2

Definition at line 227 of file sisl.h.

## 29.452.2.10 int SISLIntcurve::pretop[4]

Definition at line 247 of file sisl.h.

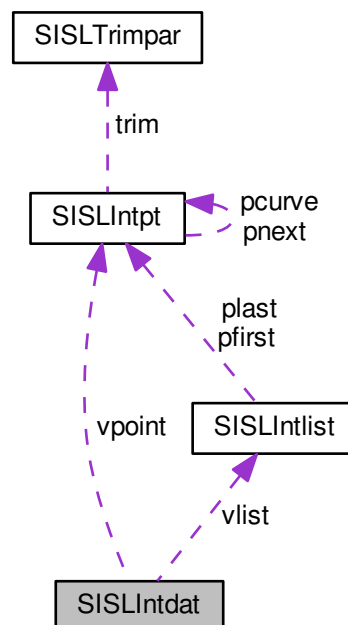
The documentation for this struct was generated from the following file:

- [sisl/include/sisl.h](#)

## 29.453 SISLIntdat Struct Reference

```
#include <sislP.h>
```

Collaboration diagram for SISLIntdat:



## Public Attributes

- [SISLIntpt](#) \*\* [vpoint](#)
- int [ipoint](#)
- int [ipmax](#)
- [SISLIntlist](#) \*\* [vlist](#)
- int [ilist](#)
- int [ilmax](#)

### 29.453.1 Detailed Description

Definition at line 336 of file [sisIP.h](#).

### 29.453.2 Member Data Documentation

#### 29.453.2.1 int SISLIntdat::ilist

Definition at line 342 of file [sisIP.h](#).

#### 29.453.2.2 int SISLIntdat::ilmax

Definition at line 343 of file [sisIP.h](#).

#### 29.453.2.3 int SISLIntdat::ipmax

Definition at line 340 of file [sisIP.h](#).

#### 29.453.2.4 int SISLIntdat::ipoint

Definition at line 339 of file [sisIP.h](#).

#### 29.453.2.5 [SISLIntlist](#)\*\* SISLIntdat::vlist

Definition at line 341 of file [sisIP.h](#).

#### 29.453.2.6 [SISLIntpt](#)\*\* SISLIntdat::vpoint

Definition at line 338 of file [sisIP.h](#).

The documentation for this struct was generated from the following file:

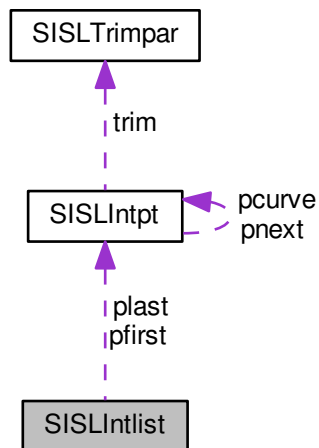
- [sisl/include/sisIP.h](#)



## 29.454 SISLIntlist Struct Reference

```
#include <sislP.h>
```

Collaboration diagram for SISLIntlist:



### Public Attributes

- [SISLIntpt](#) \* [pfirst](#)
- [SISLIntpt](#) \* [plast](#)
- int [ind\\_first](#)
- int [ind\\_last](#)
- int [itype](#)
- int [inumb](#)
- int [pretop](#) [4]

### 29.454.1 Detailed Description

Definition at line 289 of file sislP.h.

### 29.454.2 Member Data Documentation

#### 29.454.2.1 int SISLIntlist::ind\_first

Definition at line 293 of file sislP.h.

29.454.2.2 int SISLIntlist::ind\_last

Definition at line 294 of file sislP.h.

29.454.2.3 int SISLIntlist::inumb

Definition at line 304 of file sislP.h.

29.454.2.4 int SISLIntlist::itype

Definition at line 295 of file sislP.h.

29.454.2.5 SISLIntpt\* SISLIntlist::pfirst

Definition at line 291 of file sislP.h.

29.454.2.6 SISLIntpt\* SISLIntlist::plast

Definition at line 292 of file sislP.h.

29.454.2.7 int SISLIntlist::pretop[4]

Definition at line 305 of file sislP.h.

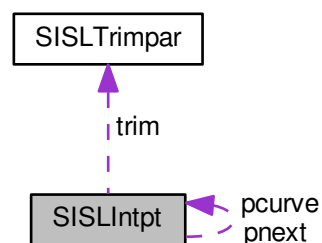
The documentation for this struct was generated from the following file:

- sisl/include/[sislP.h](#)

## 29.455 SISLIntpt Struct Reference

```
#include <sislP.h>
```

Collaboration diagram for SISLIntpt:



## Public Attributes

- int [ipar](#)
- double \* [epar](#)
- double [adist](#)
- struct [SISLIntpt](#) \* [pcurve](#)
- int [iinter](#)
- struct [SISLIntpt](#) \*\* [pnext](#)
- int \* [curve\\_dir](#)
- int [no\\_of\\_curves](#)
- int [no\\_of\\_curves\\_alloc](#)
- int \* [left\\_obj\\_1](#)
- int \* [left\\_obj\\_2](#)
- int \* [right\\_obj\\_1](#)
- int \* [right\\_obj\\_2](#)
- int [size\\_1](#)
- int [size\\_2](#)
- double \* [geo\\_data\\_1](#)
- double \* [geo\\_data\\_2](#)
- double [geo\\_track\\_3d](#) [10]
- double [geo\\_track\\_2d\\_1](#) [7]
- double [geo\\_track\\_2d\\_2](#) [7]
- int [edge\\_1](#)
- int [edge\\_2](#)
- int [marker](#)
- int [evaluated](#)
- struct [SISLTrimpar](#) \* [trim](#) [2]
- int [iside\\_1](#)
- int [iside\\_2](#)

### 29.455.1 Detailed Description

Definition at line 203 of file [sisIP.h](#).

### 29.455.2 Member Data Documentation

#### 29.455.2.1 double [SISLIntpt::adist](#)

Definition at line 209 of file [sisIP.h](#).

#### 29.455.2.2 int\* [SISLIntpt::curve\\_dir](#)

Definition at line 221 of file [sisIP.h](#).

#### 29.455.2.3 int [SISLIntpt::edge\\_1](#)

Definition at line 241 of file [sisIP.h](#).

29.455.2.4 `int SISLIntpt::edge_2`

Definition at line 242 of file sisIP.h.

29.455.2.5 `double* SISLIntpt::epar`

Definition at line 207 of file sisIP.h.

29.455.2.6 `int SISLIntpt::evaluated`

Definition at line 244 of file sisIP.h.

29.455.2.7 `double* SISLIntpt::geo_data_1`

Definition at line 235 of file sisIP.h.

29.455.2.8 `double* SISLIntpt::geo_data_2`

Definition at line 236 of file sisIP.h.

29.455.2.9 `double SISLIntpt::geo_track_2d_1[7]`

Definition at line 239 of file sisIP.h.

29.455.2.10 `double SISLIntpt::geo_track_2d_2[7]`

Definition at line 240 of file sisIP.h.

29.455.2.11 `double SISLIntpt::geo_track_3d[10]`

Definition at line 238 of file sisIP.h.

29.455.2.12 `int SISLIntpt::iinter`

Definition at line 213 of file sisIP.h.

29.455.2.13 `int SISLIntpt::ipar`

Definition at line 205 of file sisIP.h.

29.455.2.14 `int SISLIntpt::iside_1`

Definition at line 246 of file sisIP.h.

29.455.2.15 `int SISLIntpt::iside_2`

Definition at line 247 of file sisIP.h.

29.455.2.16 `int* SISLIntpt::left_obj_1`

Definition at line 225 of file sisIP.h.

29.455.2.17 `int* SISLIntpt::left_obj_2`

Definition at line 227 of file sisIP.h.

29.455.2.18 `int SISLIntpt::marker`

Definition at line 243 of file sisIP.h.

29.455.2.19 `int SISLIntpt::no_of_curves`

Definition at line 223 of file sisIP.h.

29.455.2.20 `int SISLIntpt::no_of_curves_alloc`

Definition at line 224 of file sisIP.h.

29.455.2.21 `struct SISLIntpt* SISLIntpt::pcurve`

Definition at line 211 of file sisIP.h.

29.455.2.22 `struct SISLIntpt** SISLIntpt::pnext`

Definition at line 219 of file sisIP.h.

29.455.2.23 `int* SISLIntpt::right_obj_1`

Definition at line 229 of file sisIP.h.

29.455.2.24 `int* SISLIntpt::right_obj_2`

Definition at line 231 of file `sisIP.h`.

29.455.2.25 `int SISLIntpt::size_1`

Definition at line 233 of file `sisIP.h`.

29.455.2.26 `int SISLIntpt::size_2`

Definition at line 234 of file `sisIP.h`.

29.455.2.27 `struct SISLTrimpar* SISLIntpt::trim[2]`

Definition at line 245 of file `sisIP.h`.

The documentation for this struct was generated from the following file:

- [sisl/include/sisIP.h](#)

## 29.456 SISLIntsurf Struct Reference

```
#include <sisIP.h>
```

### Public Attributes

- `int ipoint`
- `int ipar`
- `double * epar`
- `int * const_par`

### 29.456.1 Detailed Description

Definition at line 178 of file `sisIP.h`.

### 29.456.2 Member Data Documentation

29.456.2.1 `int* SISLIntsurf::const_par`

Definition at line 184 of file `sisIP.h`.

29.456.2.2 `double*` SISLIntsurf::epar

Definition at line 182 of file sislP.h.

29.456.2.3 `int` SISLIntsurf::ipar

Definition at line 181 of file sislP.h.

29.456.2.4 `int` SISLIntsurf::ipoint

Definition at line 180 of file sislP.h.

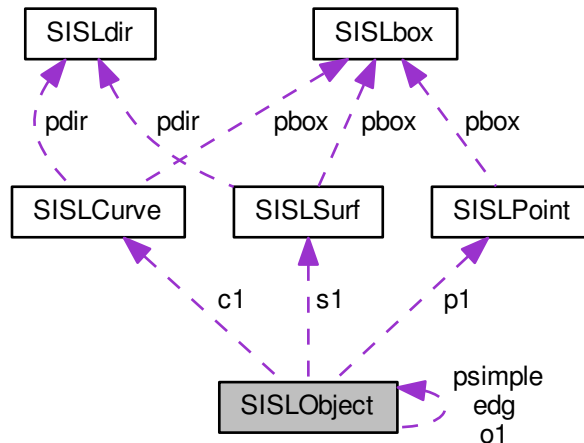
The documentation for this struct was generated from the following file:

- sisl/include/sislP.h

## 29.457 SISLObject Struct Reference

```
#include <sislP.h>
```

Collaboration diagram for SISLObject:



### Public Attributes

- `int` `iobj`
- `SISLPoint` \* `p1`
- `SISLCurve` \* `c1`
- `SISLSurf` \* `s1`
- `struct SISLObject` \* `o1`
- `struct SISLObject` \* `edg` [4]
- `struct SISLObject` \* `psimple`

### 29.457.1 Detailed Description

Definition at line 158 of file sisIP.h.

### 29.457.2 Member Data Documentation

#### 29.457.2.1 **SISLCurve\*** SISLObject::c1

Definition at line 166 of file sisIP.h.

#### 29.457.2.2 **struct SISLObject\*** SISLObject::edg[4]

Definition at line 172 of file sisIP.h.

#### 29.457.2.3 **int** SISLObject::iobj

Definition at line 160 of file sisIP.h.

#### 29.457.2.4 **struct SISLObject\*** SISLObject::o1

Definition at line 170 of file sisIP.h.

#### 29.457.2.5 **SISLPoint\*** SISLObject::p1

Definition at line 165 of file sisIP.h.

#### 29.457.2.6 **struct SISLObject\*** SISLObject::psimple

Definition at line 174 of file sisIP.h.

#### 29.457.2.7 **SISLSurf\*** SISLObject::s1

Definition at line 168 of file sisIP.h.

The documentation for this struct was generated from the following file:

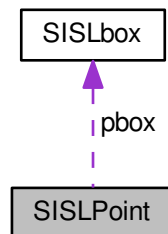
- [sisl/include/sisIP.h](#)



## 29.458 SISLPoint Struct Reference

```
#include <sislP.h>
```

Collaboration diagram for SISLPoint:



### Public Attributes

- `double ec [3]`
- `int idim`
- `double * ecoef`
- `int icopy`
- `SISLbox * pbox`

### 29.458.1 Detailed Description

Definition at line 134 of file `sislP.h`.

### 29.458.2 Member Data Documentation

#### 29.458.2.1 `double SISLPoint::ec[3]`

Definition at line 136 of file `sislP.h`.

#### 29.458.2.2 `double* SISLPoint::ecoef`

Definition at line 138 of file `sislP.h`.

#### 29.458.2.3 `int SISLPoint::icopy`

Definition at line 140 of file `sislP.h`.

#### 29.458.2.4 int SISLPoint::idim

Definition at line 137 of file sisIP.h.

#### 29.458.2.5 SISLbox\* SISLPoint::pbox

Definition at line 148 of file sisIP.h.

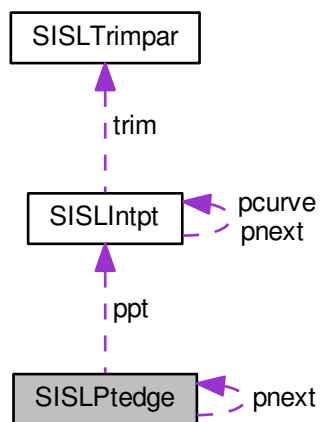
The documentation for this struct was generated from the following file:

- sisl/include/[sisIP.h](#)

## 29.459 SISLPtedge Struct Reference

```
#include <sisIP.h>
```

Collaboration diagram for SISLPtedge:



### Public Attributes

- [SISLIntpt](#) \* ppt
- struct [SISLPtedge](#) \* pnext

### 29.459.1 Detailed Description

Definition at line 312 of file sisIP.h.

## 29.459.2 Member Data Documentation

### 29.459.2.1 struct SISLPtedge\* SISLPtedge::pNext

Definition at line 315 of file sisIP.h.

### 29.459.2.2 SISLIntpt\* SISLPtedge::ppt

Definition at line 314 of file sisIP.h.

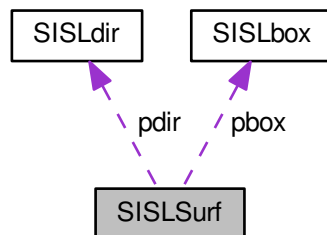
The documentation for this struct was generated from the following file:

- [sisl/include/sisIP.h](#)

## 29.460 SISLSurf Struct Reference

```
#include <sisl.h>
```

Collaboration diagram for SISLSurf:



### Public Attributes

- int [ik1](#)
- int [ik2](#)
- int [in1](#)
- int [in2](#)
- double \* [et1](#)
- double \* [et2](#)
- double \* [ecoef](#)
- double \* [rcoef](#)
- int [ikind](#)
- int [idim](#)
- int [icopy](#)
- [SISLdir](#) \* [pdir](#)
- [SISLbox](#) \* [pbox](#)
- int [use\\_count](#)
- int [cuopen\\_1](#)
- int [cuopen\\_2](#)

### 29.460.1 Detailed Description

Definition at line 154 of file sisl.h.

### 29.460.2 Member Data Documentation

#### 29.460.2.1 int SISLSurf::cuopen\_1

Definition at line 195 of file sisl.h.

#### 29.460.2.2 int SISLSurf::cuopen\_2

Definition at line 196 of file sisl.h.

#### 29.460.2.3 double\* SISLSurf::ecoeff

Definition at line 168 of file sisl.h.

#### 29.460.2.4 double\* SISLSurf::et1

Definition at line 164 of file sisl.h.

#### 29.460.2.5 double\* SISLSurf::et2

Definition at line 166 of file sisl.h.

#### 29.460.2.6 int SISLSurf::icopy

Definition at line 182 of file sisl.h.

#### 29.460.2.7 int SISLSurf::idim

Definition at line 180 of file sisl.h.

#### 29.460.2.8 int SISLSurf::ik1

Definition at line 156 of file sisl.h.

#### 29.460.2.9 int SISLSurf::ik2

Definition at line 158 of file sisl.h.

**29.460.2.10** int SISLSurf::ikind

Definition at line 171 of file sisl.h.

**29.460.2.11** int SISLSurf::in1

Definition at line 160 of file sisl.h.

**29.460.2.12** int SISLSurf::in2

Definition at line 162 of file sisl.h.

**29.460.2.13** SISLbox\* SISLSurf::pbox

Definition at line 191 of file sisl.h.

**29.460.2.14** SISLdir\* SISLSurf::pdir

Definition at line 189 of file sisl.h.

**29.460.2.15** double\* SISLSurf::rcoef

Definition at line 169 of file sisl.h.

**29.460.2.16** int SISLSurf::use\_count

Definition at line 193 of file sisl.h.

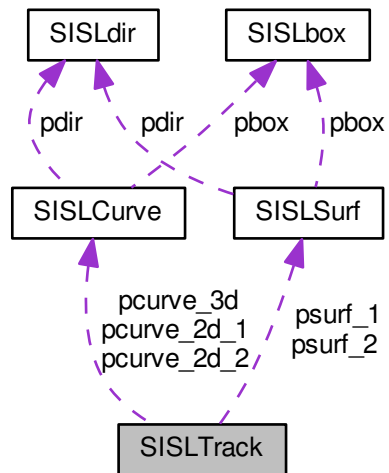
The documentation for this struct was generated from the following file:

- [sisl/include/sisl.h](#)

## 29.461 SISLTrack Struct Reference

```
#include <sislP.h>
```

Collaboration diagram for SISLTrack:



### Public Attributes

- [SISLSurf](#) \* [psurf\\_1](#)
- [SISLSurf](#) \* [psurf\\_2](#)
- [SISLCurve](#) \* [pcurve\\_3d](#)
- [SISLCurve](#) \* [pcurve\\_2d\\_1](#)
- [SISLCurve](#) \* [pcurve\\_2d\\_2](#)
- [int](#) [ideg](#)
- [double](#) [eimpli](#) [16]
- [int](#) [turned](#)
- [int](#) [exact](#)
- [int](#) [pretop](#) [4]
- [int](#) [sing\\_start](#)
- [int](#) [sing\\_end](#)

### 29.461.1 Detailed Description

Definition at line 253 of file sislP.h.

### 29.461.2 Member Data Documentation

#### 29.461.2.1 [double](#) [SISLTrack::eimpli](#)[16]

Definition at line 274 of file sislP.h.

29.461.2.2 int SISLTrack::exact

Definition at line 280 of file sisIP.h.

29.461.2.3 int SISLTrack::ideg

Definition at line 262 of file sisIP.h.

29.461.2.4 SISLCurve\* SISLTrack::pcurve\_2d\_1

Definition at line 258 of file sisIP.h.

29.461.2.5 SISLCurve\* SISLTrack::pcurve\_2d\_2

Definition at line 260 of file sisIP.h.

29.461.2.6 SISLCurve\* SISLTrack::pcurve\_3d

Definition at line 257 of file sisIP.h.

29.461.2.7 int SISLTrack::pretop[4]

Definition at line 281 of file sisIP.h.

29.461.2.8 SISLSurf\* SISLTrack::psurf\_1

Definition at line 255 of file sisIP.h.

29.461.2.9 SISLSurf\* SISLTrack::psurf\_2

Definition at line 256 of file sisIP.h.

29.461.2.10 int SISLTrack::sing\_end

Definition at line 283 of file sisIP.h.

29.461.2.11 int SISLTrack::sing\_start

Definition at line 282 of file sisIP.h.

#### 29.461.2.12 int SISLTrack::turned

Definition at line 275 of file sisIP.h.

The documentation for this struct was generated from the following file:

- [sisl/include/sisIP.h](#)

### 29.462 SISLTrimpar Struct Reference

```
#include <sisIP.h>
```

#### Public Attributes

- int [ptindex](#)
- int [parindex](#)

#### 29.462.1 Detailed Description

Definition at line 189 of file sisIP.h.

#### 29.462.2 Member Data Documentation

##### 29.462.2.1 int SISLTrimpar::parindex

Definition at line 193 of file sisIP.h.

##### 29.462.2.2 int SISLTrimpar::ptindex

Definition at line 191 of file sisIP.h.

The documentation for this struct was generated from the following file:

- [sisl/include/sisIP.h](#)

### 29.463 Go::SmoothCurve Class Reference

```
#include <SmoothCurve.h>
```



## Public Member Functions

- [SmoothCurve](#) (int *dim*=3)
- [~SmoothCurve](#) ()
  - Destructor.*
- int [attach](#) (const shared\_ptr< [SplineCurve](#) > &*incurve*, int *coef\_known*[], int *numSideConstraints*=0)
  - Specify the spline space we want to use,.*
- void [setOptim](#) (const double *weight1*, const double *weight2*, const double *weight3*)
- void [setLeastSquares](#) (const std::vector< double > &*pnts*, const std::vector< double > &*param\_pnts*, const std::vector< double > &*pnt\_weights*, double *weight*)
- void [setPeriodicity](#) (int *cont*, double *weight*)
- void [setSideConstraints](#) (std::vector< [sideConstraint](#) > &*constraints*)
- void [equationSolve](#) (shared\_ptr< [SplineCurve](#) > &*curve*)

### 29.463.1 Detailed Description

This class is used to generate a spline curve from a given spline space that approximates a set of weighed data points. In addition, user can set other conditions on things like curve smoothness or periodicity. To use this object, you must first instantiate it, then run the '[attach\(\)](#)' function, the '[setOptim\(\)](#)' function, the '[setLeastSquares\(\)](#)' function and finally the '[equationSolve\(\)](#)' function. Optionally, the functions '[setPeriodicity\(\)](#)' and '[setSideConstraints\(\)](#)' can be run before '[equationSolve\(\)](#)', if the corresponding conditions should be specified.

Definition at line 87 of file [SmoothCurve.h](#).

### 29.463.2 Constructor & Destructor Documentation

#### 29.463.2.1 Go::SmoothCurve::SmoothCurve ( int *dim* = 3 )

Constructor. Initialises the object with a given spatial dimension (default is 3).

##### Parameters

<i>dim</i>	dimension of the geometry space.
------------	----------------------------------

#### 29.463.2.2 Go::SmoothCurve::~~SmoothCurve ( )

Destructor.

### 29.463.3 Member Function Documentation

#### 29.463.3.1 int Go::SmoothCurve::attach ( const shared\_ptr< [SplineCurve](#) > & *incurve*, int *coef\_known*[], int *numSideConstraints* = 0 )

Specify the spline space we want to use,.

Initializes data given by an intermediate curve.

## Parameters

<i>incurve</i>	curve defining the initial spline curve.
<i>coef_known</i>	array defining the free coefficients. Size equal to the number of coefficients in <i>incurve</i> . Value 0 = not known, value 1 = known.
<i>numSideConstraints</i>	the number of linear side constraints for the approximation.

## Returns

status value: 0 = OK, negative not OK.

29.463.3.2 void Go::SmoothCurve::equationSolve ( shared\_ptr< SplineCurve > & curve )

Solve equation system, and produce output curve.

## Parameters

<i>curve</i>	the curve resulting from the smoothing process.
--------------	-------------------------------------------------

29.463.3.3 void Go::SmoothCurve::setLeastSquares ( const std::vector< double > & pnts, const std::vector< double > & param\_pnts, const std::vector< double > & pnt\_weights, double weight )

Compute matrices for least squares approximation.

## Parameters

<i>pnts</i>	the input points (for 3D: (x0, y0, z0, x1, y1, z1, ...)) defining the approximation part of the smoothing problem.
<i>param_pnts</i>	the parameters for the input pnts.
<i>pnt_weights</i>	weight attached to each point to be approximated. Should lie in the unit interval. Typically they are all 1.0.
<i>weight</i>	multiplier of all weights in the pnt_weights vector

29.463.3.4 void Go::SmoothCurve::setOptim ( const double weight1, const double weight2, const double weight3 )

Compute the smoothing part of the equation system. The sum of the weights should lie in the unit interval.

## Parameters

<i>weight1</i>	weight for smoothing with respect to the 1st derivative.
<i>weight2</i>	weight for smoothing with respect to the 2nd derivative.
<i>weight3</i>	weight for smoothing with respect to the 3rd derivative.

29.463.3.5 void Go::SmoothCurve::setPeriodicity ( int *cont*, double *weight* )

Set periodicity constraints in one par. dir.

#### Parameters

<i>cont</i>	the wanted continuity across the seam.
<i>weight</i>	weight given to approximative continuity conditions. Should lie inside the unit interval.

29.463.3.6 void Go::SmoothCurve::setSideConstraints ( std::vector< sideConstraint > & *constraints* )

Set linear side constraints (linear equation involving the free coefficients) to the minimization problem. The problem is solved using the method of Lagrange multipliers.

#### Parameters

<i>constraints</i>	the linear side constraints.
--------------------	------------------------------

The documentation for this class was generated from the following file:

- [gotools-core/include/GoTools/creators/SmoothCurve.h](#)

## 29.464 Go::SmoothCurveSet Class Reference

```
#include <SmoothCurveSet.h>
```

### Public Member Functions

- [SmoothCurveSet](#) ()  
*Default constructor to the class.*
- [~SmoothCurveSet](#) ()  
*Destructor.*
- int [attach](#) (std::vector< shared\_ptr< [SplineCurve](#) > > &incvs, std::vector< int > &seem, std::vector< std::vector< int > > &coef\_known, int numSideConstraints=0)
- int [setOptimize](#) (double weight1, double weight2, double weight3)  
*Compute the smoothing part of the equation system.*
- int [setLeastSquares](#) (const std::vector< std::vector< double > > &pnts, const std::vector< std::vector< double > > &param\_pnts, const std::vector< std::vector< double > > &pnt\_weights, double weight)
- void [setApproxOrig](#) (double weight)  
*Add term for approximation of the original curves.*
- void [setInterpolationConditions](#) (const std::vector< std::vector< double > > &pnts, const std::vector< std::vector< double > > &param\_pnts, const std::vector< std::vector< int > > &der, bool appr\_constraints, double appr\_wgt, int \*jstat)
- int [setCvSetConstraints](#) (const std::vector< shared\_ptr< [cvSetConstraint](#) > > &cv\_set\_constraints, bool appr\_constraints, double appr\_wgt)
- int [setApproxSideConstraints](#) (std::vector< shared\_ptr< [sideConstraintSet](#) > > &constraints, double weight)
- int [equationSolve](#) (std::vector< shared\_ptr< [SplineCurve](#) > > &curves)  
*Solve equation system, and produce output curves.*
- int [setOrthCond](#) (const std::vector< std::vector< double > > &pnts, const std::vector< std::vector< double > > &param\_pnts, double weight)
- void [setSideConstraints](#) (std::vector< shared\_ptr< [sideConstraintSet](#) > > &constraints, bool replace\_constraints)

### 29.464.1 Detailed Description

Smoothing, point interpolation and point approximation applied to a set of curves while maintaining a set of continuity conditions between the curves, exact or approximative.

Definition at line 84 of file SmoothCurveSet.h.

### 29.464.2 Constructor & Destructor Documentation

#### 29.464.2.1 Go::SmoothCurveSet::SmoothCurveSet ( )

Default constructor to the class.

#### 29.464.2.2 Go::SmoothCurveSet::~~SmoothCurveSet ( )

Destructor.

### 29.464.3 Member Function Documentation

#### 29.464.3.1 int Go::SmoothCurveSet::attach ( std::vector< shared\_ptr< SplineCurve > > & incvs, std::vector< int > & seem, std::vector< std::vector< int > > & coef\_known, int numSideConstraints = 0 )

Initializes data given by an intermediate set of curves. For each curve there exists a vector coef\_known (of size equal to the number of coefficients in the curve) Input is array of iterators to first element.

#### 29.464.3.2 int Go::SmoothCurveSet::equationSolve ( std::vector< shared\_ptr< SplineCurve > > & curves )

Solve equation system, and produce output curves.

#### 29.464.3.3 void Go::SmoothCurveSet::setApproxOrig ( double weight )

Add term for approximation of the original curves.

#### 29.464.3.4 int Go::SmoothCurveSet::setApproxSideConstraints ( std::vector< shared\_ptr< sideConstraintSet > > & constraints, double weight )

We may have side constraints which are not suitable for exact equality as spline solution space may not be large enough. We therefore allow using least squares to minimize the error. This applies in particular to constraint involving higher order derivatives. Assuming input is preprocessed (all coefs in constraints are free).

#### 29.464.3.5 int Go::SmoothCurveSet::setCvSetConstraints ( const std::vector< shared\_ptr< cvSetConstraint > > & cv\_set\_constraints, bool appr\_constraints, double appr\_wgt )

Set linear side constraints between the coefs in (possibly different) input cvs.

29.464.3.6 void Go::SmoothCurveSet::setInterpolationConditions ( const std::vector< std::vector< double > > & *pnts*, const std::vector< std::vector< double > > & *param\_pnts*, const std::vector< std::vector< int > > & *der*, bool *appr\_constraints*, double *appr\_wgt*, int \* *jstat* )

We add the interpolation conditions as linear side constraints. Assuming the degrees of freedom are sufficient (i.e. that the input curve provided by the user has enough knots).

29.464.3.7 int Go::SmoothCurveSet::setLeastSquares ( const std::vector< std::vector< double > > & *pnts*, const std::vector< std::vector< double > > & *param\_pnts*, const std::vector< std::vector< double > > & *pnt\_weights*, double *weight* )

Compute matrices for least squares approximation. Each curve is assigned a number of data points with corresponding parameter values and weights

29.464.3.8 int Go::SmoothCurveSet::setOptimize ( double *weight1*, double *weight2*, double *weight3* )

Compute the smoothing part of the equation system.

29.464.3.9 int Go::SmoothCurveSet::setOrthCond ( const std::vector< std::vector< double > > & *pnts*, const std::vector< std::vector< double > > & *param\_pnts*, double *weight* )

The contribution to the equation system from the approximation of normal directions.

29.464.3.10 void Go::SmoothCurveSet::setSideConstraints ( std::vector< shared\_ptr< sideConstraintSet > > & *constraints*, bool *replace\_constraints* )

Add side constraints to the functional (Lagrange multiplier). Assuming input is preprocessed (all coefs in constraints are free). If `replace_constraints==true` the old constraints are removed prior to adding new constraints.

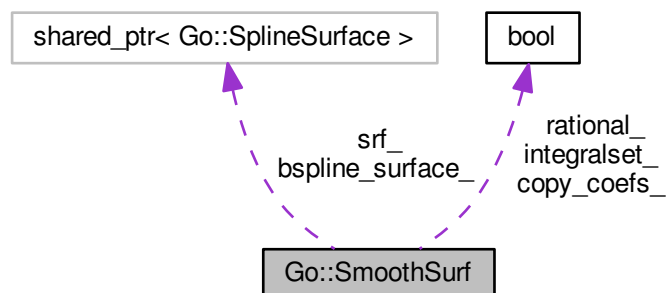
The documentation for this class was generated from the following file:

- [gtools-core/include/GoTools/creators/SmoothCurveSet.h](#)

## 29.465 Go::SmoothSurf Class Reference

```
#include <SmoothSurf.h>
```

Collaboration diagram for Go::SmoothSurf:



## Public Member Functions

- [SmoothSurf](#) ()  
*Default constructor. Initializes class variable.*
- [SmoothSurf](#) (bool copy\_coefs)
- virtual [~SmoothSurf](#) ()  
*Destructor.*
- void [attach](#) (shared\_ptr< [SplineSurface](#) > &insf, int seem[], int coef\_known[], int num\_side\_constraints=0, int has\_normal\_cond=0)
- virtual void [setOptimize](#) (const double weight1, const double weight2, const double weight3)
- void [setLeastSquares](#) (const std::vector< double > &pnts, const std::vector< double > &param\_pnts, const std::vector< double > &pnt\_weights, const double weight)
- int [setNormalCond](#) (const std::vector< double > &pnts, const std::vector< double > &param\_pnts, const std::vector< double > &pnt\_weights, const double weight)
- void [approxOrig](#) (double weight)
- virtual void [setPeriodicity](#) (int paddir, int cont, double weight1, double weight2)
- void [setSideConstraints](#) (std::vector< [sideConstraint](#) > &constraints)
- int [equationSolve](#) (shared\_ptr< [SplineSurface](#) > &surf)
- void [setRelaxParam](#) (double omega)  
*Set the relaxation parameter for the RILU preconditioner.*

## Protected Member Functions

- virtual void [releaseScratch](#) ()  
*Free all memory allocated for class members.*
- virtual void [prepareIntegral](#) ()  
*Prepare storage for integrals of inner products of basis functions.*
- virtual void [getBasis](#) (const double \*sb1, const double \*sb2, int kleft1, int kleft2, int ider, double \*sbasis)
- void [preparePeriodicity](#) (int seem[])
- void [setOptimizeNonrational](#) (const double weight1, const double weight2, const double weight3)
- void [setOptimizeRational](#) (const double weight1, const double weight2, const double weight3)
- void [approxOrigNonrational](#) (double weight)
- void [approxOrigRational](#) (double weight)
- void [setC1AtSeem](#) (int paddir, double weight)
- void [setC2AtSeem](#) (int paddir, double weight)
- void [setRationalCnAtSeem](#) (int paddir, int cn, double weight1, double weight2)
- virtual int [adjustAtSeem](#) ()

## Protected Attributes

- int [norm\\_dim\\_](#)
- int [idim\\_](#)
- int [kdim\\_](#)
- int [idim1\\_](#)
- int [ider\\_](#)
- bool [integralset\\_](#)
- int [ider\\_scratch\\_](#)
- bool [rational\\_](#)
- shared\_ptr< [SplineSurface](#) > [bspline\\_surface\\_](#)
- int [cont\\_seem\\_](#) [2]
- int [kncond\\_](#)
- int [knconstraint\\_](#)

- `const int kpointer_`
- `int * coefknown_`
- `std::vector< int > pivot_`
- `shared_ptr< SplineSurface > srf_`
- `int kk1_`
- `int kk2_`
- `int kn1_`
- `int kn2_`
- `std::vector< double >::const_iterator st1_`
- `std::vector< double >::const_iterator st2_`
- `const bool copy_coefs_`
- `std::vector< double > coef_array_`
  - Parameters used to define the specific input spline surface.*
- `std::vector< double >::iterator scoef_`
- `std::vector< double > gmat_`
  - Storage of the equation system.*
- `std::vector< double > gright_`

### 29.465.1 Detailed Description

This class modifies a tensor product B-spline surface with respect to conditions on smoothness, editing constraints and boundary conditions.

Definition at line 74 of file SmoothSurf.h.

### 29.465.2 Constructor & Destructor Documentation

#### 29.465.2.1 Go::SmoothSurf::SmoothSurf ( )

Default constructor. Initializes class variable.

#### 29.465.2.2 Go::SmoothSurf::SmoothSurf ( bool copy\_coefs )

Constructor. Initializes class variable.

Parameters

<code>copy_coefs</code>	true if coefficients on attached surface are not to be modified.
-------------------------	------------------------------------------------------------------

#### 29.465.2.3 virtual Go::SmoothSurf::~SmoothSurf ( ) [virtual]

Destructor.

### 29.465.3 Member Function Documentation

29.465.3.1 `virtual int Go::SmoothSurf::adjustAtSeem ( ) [protected], [virtual]`

Ensure that the expected C1 or C2 continuity at the seem is satisfied. Only applicable if `seem_[0] > 1 || seem_[1] > 1`.

29.465.3.2 `void Go::SmoothSurf::approxOrig ( double weight )`

Compute the contribution to the equation system from the approximation of an original surface, i.e. the contribution of the original coefficients.

#### Parameters

<i>weight</i>	the relative contribution of the original coefs. Should lie in the unit interval.
---------------	-----------------------------------------------------------------------------------

29.465.3.3 `void Go::SmoothSurf::approxOrigNonrational ( double weight ) [protected]`

Compute the contribution to the equation system from the approximation of an original surface, i.e. the contribution of the original coefficients for the non-rational case

#### Parameters

<i>weight</i>	the relative contribution of the original coefs. Should lie in the unit interval.
---------------	-----------------------------------------------------------------------------------

29.465.3.4 `void Go::SmoothSurf::approxOrigRational ( double weight ) [protected]`

Compute the contribution to the equation system from the approximation of an original surface, i.e. the contribution of the original coefficients for the rational case

#### Parameters

<i>weight</i>	the relative contribution of the original coefs. Should lie in the unit interval.
---------------	-----------------------------------------------------------------------------------

29.465.3.5 `void Go::SmoothSurf::attach ( shared_ptr< SplineSurface > & insf, int seem[], int coef_known[], int num_side_constraints = 0, int has_normal_cond = 0 )`

Initializes data given by an intermediate surface.

#### Parameters

<i>insf</i>	the initial surface.
<i>seem</i>	continuity across opposite edges. 0 not specified, 1 = C0, 2 = C1, 3 = C2.
<i>coef_known</i>	whether the coefs are free to be altered. 0: not known, 1: known $\geq$ <code>kpointer_ = 3</code> : the coefficients indexed by <code>ki</code> & <code>coef_known[ki]</code> - <code>kpointer_</code> should be equal.
<i>num_side_constraints</i>	the number of linear side constraints in the system.
<i>has_normal_cond</i>	whether the system must fulfill normal conditions.



29.465.3.6 `int Go::SmoothSurf::equationSolve ( shared_ptr< SplineSurface > & surf )`

Solve equation system, and produce output surface. If failing to solve the routine may throw an exception.

#### Parameters

<i>surf</i>	the output surface.
-------------	---------------------

#### Returns

0 = OK, negative = failed solving system.

29.465.3.7 `virtual void Go::SmoothSurf::getBasis ( const double * sb1, const double * sb2, int kleft1, int kleft2, int ider, double * sbasis )` [protected],[virtual]

Given the value of non-zero B-spline functions, compute the value of the corresponding surface basis function (i.e. the products of the u- and v-basis functions).

#### Parameters

<i>sb1</i>	the basis values in the first parameter direction (u).
<i>sb2</i>	the basis values in the second parameter direction (v).
<i>kleft1</i>	index of the first knot interval in u-dir.
<i>kleft2</i>	index of the first knot interval in v-dir.
<i>ider</i>	the number of derivatives to compute.
<i>sbasis</i>	the computed basis values in sf. size = order_u()*order_v()*(ider + 1)*(ider + 1). The space must be allocated on the outside.

29.465.3.8 `virtual void Go::SmoothSurf::prepareIntegral ( )` [protected],[virtual]

Prepare storage for integrals of inner products of basis functions.

29.465.3.9 `void Go::SmoothSurf::preparePeriodicity ( int seem[] )` [protected]

Set pointers between identical coefficients at a periodic seem. If possible, update fixed coefficients at the seem.

#### Parameters

<i>seem</i>	continuity across the seem. <a href="#">Array</a> size = 2.
-------------	-------------------------------------------------------------

29.465.3.10 `virtual void Go::SmoothSurf::releaseScratch ( )` [protected],[virtual]

Free all memory allocated for class members.

29.465.3.11 void Go::SmoothSurf::setC1AtSeem ( int *parDir*, double *weight* ) [protected]

Set constraints on C1-continuity across a seem and add these to the equation system.

29.465.3.12 void Go::SmoothSurf::setC2AtSeem ( int *parDir*, double *weight* ) [protected]

Set constraints on C2-continuity across a seem and add these to the equation system.

29.465.3.13 void Go::SmoothSurf::setLeastSquares ( const std::vector< double > & *pnts*, const std::vector< double > & *param\_pnts*, const std::vector< double > & *pnt\_weights*, const double *weight* )

Compute matrices for least squares approximation.

#### Parameters

<i>pnts</i>	points on surface to be approximated. Stored as (for 3D): (x0, y0, z0, x1, y1, z1, ...)
<i>param_pnts</i>	the corresponding 2-dimensional parametric points. Stored as: (u0, v0, u1, v1, ...)
<i>pnt_weights</i>	each of the pnts is assigned a weight lying in the unit interval. 1.0 if all pnts are equally important.
<i>weight</i>	the contribution of the approximation of the pnts in the system. weight should lie in the unit interval.

29.465.3.14 int Go::SmoothSurf::setNormalCond ( const std::vector< double > & *pnts*, const std::vector< double > & *param\_pnts*, const std::vector< double > & *pnt\_weights*, const double *weight* )

Compute matrices for approximation of normal directions.

#### Parameters

<i>pnts</i>	normals in sf to be approximated. Stored as (for 3D): (x0, y0, z0, x1, y1, z1, ...)
<i>param_pnts</i>	the corresponding 2-dimensional parametric points. Stored as: (u0, v0, u1, v1, ...)
<i>pnt_weights</i>	each of the pnts is assigned a weight lying in the unit interval. 1.0 if all pnts are equally important.
<i>weight</i>	the contribution of the approximation of the normals in the system. weight should lie in the unit interval.

#### Returns

status value: 0 = OK, 1 = warning (system not prepared for normal conditions).

29.465.3.15 virtual void Go::SmoothSurf::setOptimize ( const double *weight1*, const double *weight2*, const double *weight3* ) [virtual]

Compute the smoothing part of the equation system.

#### Parameters

<i>weight1</i>	contribution weight with respect to the 1st derivative.
<i>weight2</i>	contribution weight with respect to the 2nd derivative.
<i>weight3</i>	contribution weight with respect to the 3rd derivative.

29.465.3.16 void Go::SmoothSurf::setOptimizeNonrational ( const double *weight1*, const double *weight2*, const double *weight3* ) [protected]

Compute the smoothing part of the equation system, non-rational case.

Parameters

<i>weight1</i>	contribution weight with respect to the 1st derivative.
<i>weight2</i>	contribution weight with respect to the 2nd derivative.
<i>weight3</i>	contribution weight with respect to the 3rd derivative.

29.465.3.17 void Go::SmoothSurf::setOptimizeRational ( const double *weight1*, const double *weight2*, const double *weight3* ) [protected]

Compute the smoothing part of the equation system, rational case.

Parameters

<i>weight1</i>	contribution weight with respect to the 1st derivative.
<i>weight2</i>	contribution weight with respect to the 2nd derivative.
<i>weight3</i>	contribution weight with respect to the 3rd derivative.

29.465.3.18 virtual void Go::SmoothSurf::setPeriodicity ( int *pardir*, int *cont*, double *weight1*, double *weight2* ) [virtual]

Set periodicity constraints in one par. dir.

Parameters

<i>pardir</i>	the direction of the periodicity, 1 == udir && 2 == vdir.
<i>cont</i>	the continuity across the seem, 0 = C0, 1 = C1, 2 = C2.
<i>weight1</i>	C1 continuity contribution used in approximation.
<i>weight2</i>	C2 continuity contribution used in approximation.

29.465.3.19 void Go::SmoothSurf::setRationalCnAtSeem ( int *pardir*, int *cn*, double *weight1*, double *weight2* ) [protected]

Set constraints on C1- and C2-continuity across a seem and add these to the equation system when surface is rational.

29.465.3.20 void Go::SmoothSurf::setRelaxParam ( double *omega* )

Set the relaxation parameter for the RILU preconditioner.

29.465.3.21 void Go::SmoothSurf::setSideConstraints ( std::vector< sideConstraint > & constraints )

Add linear side constraints to the system equation.

## Parameters

<i>constraints</i>	the linear side constraints between surface coefficients.
--------------------	-----------------------------------------------------------

## 29.465.4 Member Data Documentation

**29.465.4.1** `shared_ptr<SplineSurface> Go::SmoothSurf::bspline_surface_` [protected]

Definition at line 181 of file SmoothSurf.h.

**29.465.4.2** `std::vector<double> Go::SmoothSurf::coef_array_` [protected]

Parameters used to define the specific input spline surface.

Definition at line 213 of file SmoothSurf.h.

**29.465.4.3** `int* Go::SmoothSurf::coefknown_` [protected]

Definition at line 193 of file SmoothSurf.h.

**29.465.4.4** `int Go::SmoothSurf::cont_seem_[2]` [protected]

Definition at line 186 of file SmoothSurf.h.

**29.465.4.5** `const bool Go::SmoothSurf::copy_coefs_` [protected]

Definition at line 210 of file SmoothSurf.h.

**29.465.4.6** `std::vector<double> Go::SmoothSurf::gmat_` [protected]

Storage of the equation system.

Definition at line 217 of file SmoothSurf.h.

**29.465.4.7** `std::vector<double> Go::SmoothSurf::gright_` [protected]

Definition at line 218 of file SmoothSurf.h.

**29.465.4.8** `int Go::SmoothSurf::ider_` [protected]

Definition at line 175 of file SmoothSurf.h.

29.465.4.9 `int Go::SmoothSurf::ider_scratch_` [protected]

Definition at line 177 of file SmoothSurf.h.

29.465.4.10 `int Go::SmoothSurf::idim1_` [protected]

Definition at line 174 of file SmoothSurf.h.

29.465.4.11 `int Go::SmoothSurf::idim_` [protected]

Definition at line 172 of file SmoothSurf.h.

29.465.4.12 `bool Go::SmoothSurf::integralset_` [protected]

Definition at line 176 of file SmoothSurf.h.

29.465.4.13 `int Go::SmoothSurf::kdim_` [protected]

Definition at line 173 of file SmoothSurf.h.

29.465.4.14 `int Go::SmoothSurf::kk1_` [protected]

Definition at line 203 of file SmoothSurf.h.

29.465.4.15 `int Go::SmoothSurf::kk2_` [protected]

Definition at line 203 of file SmoothSurf.h.

29.465.4.16 `int Go::SmoothSurf::kn1_` [protected]

Definition at line 204 of file SmoothSurf.h.

29.465.4.17 `int Go::SmoothSurf::kn2_` [protected]

Definition at line 204 of file SmoothSurf.h.

29.465.4.18 `int Go::SmoothSurf::kncond_` [protected]

Definition at line 188 of file SmoothSurf.h.

29.465.4.19 `int Go::SmoothSurf::knconstraint_` [protected]

Definition at line 189 of file SmoothSurf.h.

29.465.4.20 `const int Go::SmoothSurf::kpointer_` [protected]

Definition at line 190 of file SmoothSurf.h.

29.465.4.21 `int Go::SmoothSurf::norm_dim_` [protected]

Definition at line 171 of file SmoothSurf.h.

29.465.4.22 `std::vector<int> Go::SmoothSurf::pivot_` [protected]

Definition at line 198 of file SmoothSurf.h.

29.465.4.23 `bool Go::SmoothSurf::rational_` [protected]

Definition at line 180 of file SmoothSurf.h.

29.465.4.24 `std::vector<double>::iterator Go::SmoothSurf::scoef_` [protected]

Definition at line 214 of file SmoothSurf.h.

29.465.4.25 `shared_ptr<SplineSurface> Go::SmoothSurf::srf_` [protected]

Definition at line 202 of file SmoothSurf.h.

29.465.4.26 `std::vector<double>::const_iterator Go::SmoothSurf::st1_` [protected]

Definition at line 205 of file SmoothSurf.h.

29.465.4.27 `std::vector<double>::const_iterator Go::SmoothSurf::st2_` [protected]

Definition at line 207 of file SmoothSurf.h.

The documentation for this class was generated from the following file:

- [gotools-core/include/GoTools/creators/SmoothSurf.h](#)

## 29.466 Go::SmoothSurfSet Class Reference

```
#include <SmoothSurfSet.h>
```

### Public Member Functions

- [SmoothSurfSet](#) ()  
*Default constructor. Initializes class variable.*
- [SmoothSurfSet](#) (bool copy\_coefs)
- virtual [~SmoothSurfSet](#) ()  
*Destructor.*
- void [attach](#) (std::vector< shared\_ptr< [SplineSurface](#) > > &insf, std::vector< std::vector< int > > &coef\_↔ known, int num\_side\_constraints=0, int has\_normal\_cond=0)
- virtual void [setOptimize](#) (double weight1, double weight2, double weight3)
- void [setLeastSquares](#) (std::vector< std::vector< double > > &pnts, std::vector< std::vector< double > > &param\_pnts, std::vector< std::vector< double > > &pnt\_weights, double weight)
- int [setNormalCond](#) (std::vector< std::vector< double > > &pnts, std::vector< std::vector< double > > &param\_pnts, std::vector< std::vector< double > > &pnt\_weights, double weight)
- void [approxOrig](#) (double weight)
- void [setSideConstraints](#) (std::vector< [sideConstraintSet](#) > &constraints)
- void [setApproxSideConstraints](#) (std::vector< [sideConstraintSet](#) > &constraints, double weight)  
*Use least squares to minimize the error given by the constraints.*
- int [equationSolve](#) (std::vector< shared\_ptr< [SplineSurface](#) > > &surfaces)

### Protected Member Functions

- virtual void [getBasis](#) (double \*sb1, double \*sb2, int kk1, int kk2, int kleft1, int kleft2, int ider, double \*sbasis)

### Protected Attributes

- int [kdim\\_](#)
- int [idim\\_](#)
- int [idim1\\_](#)
- int [ider\\_](#)
- int [kncond\\_](#)
- int [knconstraint\\_](#)
- std::vector< std::vector< int >::iterator > [coef\\_known\\_](#)
- std::vector< std::vector< int > > [pivot\\_](#)
- std::vector< shared\_ptr< [SplineSurface](#) > > [srf\\_](#)
- const int [copy\\_coefs\\_](#)
- std::vector< std::vector< double > > [coef\\_array\\_](#)
- std::vector< double > [gmat\\_](#)
- std::vector< double > [gright\\_](#)

#### 29.466.1 Detailed Description

This class modifies a set of tensor product B-spline surfaces with respect to conditions on smoothness, editing constraints and boundary conditions.

Definition at line 70 of file SmoothSurfSet.h.



## 29.466.2 Constructor & Destructor Documentation

### 29.466.2.1 Go::SmoothSurfSet::SmoothSurfSet ( )

Default constructor. Initializes class variable.

### 29.466.2.2 Go::SmoothSurfSet::SmoothSurfSet ( bool *copy\_coefs* )

Constructor. Initializes class variable.

#### Parameters

<i>copy_coefs</i>	whether to copy coefs in input surface or work directly on them.
-------------------	------------------------------------------------------------------

### 29.466.2.3 virtual Go::SmoothSurfSet::~~SmoothSurfSet ( ) [virtual]

Destructor.

## 29.466.3 Member Function Documentation

### 29.466.3.1 void Go::SmoothSurfSet::approxOrig ( double *weight* )

Compute the contribution to the equation system from the approximation of an original surface, i.e. the contribution of the original coefficients.

#### Parameters

<i>weight</i>	the relative contribution of the original coefs. Should lie in the unit interval.
---------------	-----------------------------------------------------------------------------------

### 29.466.3.2 void Go::SmoothSurfSet::attach ( std::vector< shared\_ptr< SplineSurface > > & *insf*, std::vector< std::vector< int > > & *coef\_known*, int *num\_side\_constraints* = 0, int *has\_normal\_cond* = 0 )

Initializes data given by an intermediate surface.

#### Parameters

<i>insf</i>	input set of surfaces to be smoothed.
<i>coef_known</i>	whether the coefs are free to be altered. 0: not known, 1: known $\geq$ <i>kpointer_</i> = 3: the coefficients indexed by <i>ki</i> & <i>coef_known</i> [ <i>ki</i> ] - <i>kpointer_</i> should be equal. Size of <i>coef_known</i> equal to that of <i>insf</i> , size of <i>coef_known</i> [ <i>ki</i> ] equals the number of control points in <i>insf</i> [ <i>ki</i> ].
<i>num_side_constraints</i>	number of side constraints to the minimization problem
<i>has_normal_cond</i>	indicates if normal conditions will be given

29.466.3.3 `int Go::SmoothSurfSet::equationSolve ( std::vector< shared_ptr< SplineSurface > > & surfaces )`

Solve equation system, and produce output surfaces. If failing to solve the routine may throw an exception.

#### Parameters

<i>surfaces</i>	the output surface.
-----------------	---------------------

#### Returns

0 = OK, negative = failed solving system.

29.466.3.4 `virtual void Go::SmoothSurfSet::getBasis ( double * sb1, double * sb2, int kk1, int kk2, int kleft1, int kleft2, int nder, double * sbasis ) [protected], [virtual]`

Given the value of non-zero B-spline functions, compute the value of the corresponding surface basis function (i.e. the products of the u and v basis functions).

#### Parameters

<i>sb1</i>	the basis values in the first parameter direction (u).
<i>sb2</i>	the basis values in the second parameter direction (v).
<i>kk1</i>	order in the first (u) direction.
<i>kk2</i>	order in the second (v) direction.
<i>kleft1</i>	index of the first knot interval in u-dir.
<i>kleft2</i>	index of the first knot interval in v-dir.
<i>nder</i>	the number of derivatives to compute.
<i>sbasis</i>	the computed basis values in sf. size = order_u()*order_v()*(nder + 1)*(nder + 1). The space must be allocated on the outside.

29.466.3.5 `void Go::SmoothSurfSet::setApproxSideConstraints ( std::vector< sideConstraintSet > & constraints, double weight )`

Use least squares to minimize the error given by the constraints.

29.466.3.6 `void Go::SmoothSurfSet::setLeastSquares ( std::vector< std::vector< double > > & pnts, std::vector< std::vector< double > > & param_pnts, std::vector< std::vector< double > > & pnt_weights, double weight )`

Compute matrices for least squares approximation. First index of all vectors corresponds to the indexing of the attached surfaces.

#### Parameters

<i>pnts</i>	points on surfaces to be approximated. Stored as (for 3D): (x0, y0, z0, x1, y1, z1, ...)
<i>param_pnts</i>	the corresponding 2-dimensional parametric points. Stored as: (u0, v0, u1, v1, ...)
<i>pnt_weights</i>	each of the pnts is assigned a weight lying in the unit interval. 1.0 if all pnts are equally important.
<i>weight</i>	the contribution of the approximation of the pnts in the system. weight should lie in the unit interval.

29.466.3.7 `int Go::SmoothSurfSet::setNormalCond ( std::vector< std::vector< double > > & pnts, std::vector< std::vector< double > > & param_pnts, std::vector< std::vector< double > > & pnt_weights, double weight )`

Compute matrices for approximation of normal directions.

#### Parameters

<i>pnts</i>	normals in sf to be approximated. Stored as (for 3D): (x0, y0, z0, x1, y1, z1, ...)
<i>param_pnts</i>	the corresponding 2-dimensional parametric points. Stored as: (u0, v0, u1, v1, ...)
<i>pnt_weights</i>	each of the pnts is assigned a weight lying in the unit interval. 1.0 if all pnts are equally important.
<i>weight</i>	the contribution of the approximation of the normals in the system. weight should lie in the unit interval.

#### Returns

status value: 0 = OK, 1 = warning (system not prepared for normal conditions).

29.466.3.8 `virtual void Go::SmoothSurfSet::setOptimize ( double weight1, double weight2, double weight3 )`  
[virtual]

Compute the smoothing part of the equation system. All weights should be positive, and their sum not exceed 1.0.

#### Parameters

<i>weight1</i>	smoothing weight w.r.t. the 1st derivative.
<i>weight2</i>	smoothing weight w.r.t. the 2nd derivative.
<i>weight3</i>	smoothing weight w.r.t. the 3rd derivative.

29.466.3.9 `void Go::SmoothSurfSet::setSideConstraints ( std::vector< sideConstraintSet > & constraints )`

Add linear side constraints to the system equation.

#### Parameters

<i>constraints</i>	the linear side constraints between surface coefficients.
--------------------	-----------------------------------------------------------

## 29.466.4 Member Data Documentation

29.466.4.1 `std::vector<std::vector<double>> Go::SmoothSurfSet::coef_array_` [protected]

Definition at line 190 of file SmoothSurfSet.h.

29.466.4.2 `std::vector<std::vector<int>::iterator> Go::SmoothSurfSet::coef_known_` [protected]

Definition at line 176 of file SmoothSurfSet.h.

29.466.4.3 `const int Go::SmoothSurfSet::copy_coefs_` [protected]

Definition at line 186 of file SmoothSurfSet.h.

29.466.4.4 `std::vector<double> Go::SmoothSurfSet::gmat_` [protected]

Definition at line 195 of file SmoothSurfSet.h.

29.466.4.5 `std::vector<double> Go::SmoothSurfSet::gright_` [protected]

Definition at line 196 of file SmoothSurfSet.h.

29.466.4.6 `int Go::SmoothSurfSet::ider_` [protected]

Definition at line 171 of file SmoothSurfSet.h.

29.466.4.7 `int Go::SmoothSurfSet::idim1_` [protected]

Definition at line 170 of file SmoothSurfSet.h.

29.466.4.8 `int Go::SmoothSurfSet::idim_` [protected]

Definition at line 169 of file SmoothSurfSet.h.

29.466.4.9 `int Go::SmoothSurfSet::kdim_` [protected]

Definition at line 168 of file SmoothSurfSet.h.

29.466.4.10 `int Go::SmoothSurfSet::kncond_` [protected]

Definition at line 173 of file SmoothSurfSet.h.

29.466.4.11 `int Go::SmoothSurfSet::knconstraint_` [protected]

Definition at line 174 of file SmoothSurfSet.h.

29.466.4.12 `std::vector<std::vector<int>>` `Go::SmoothSurfSet::pivot_` [protected]

Definition at line 180 of file `SmoothSurfSet.h`.

29.466.4.13 `std::vector<shared_ptr<SplineSurface>>` `Go::SmoothSurfSet::srfs_` [protected]

Definition at line 184 of file `SmoothSurfSet.h`.

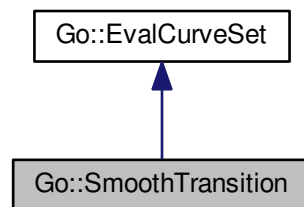
The documentation for this class was generated from the following file:

- `gotools-core/include/GoTools/creators/SmoothSurfSet.h`

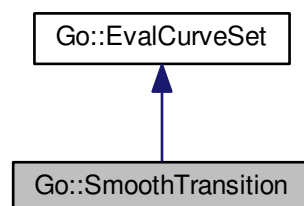
## 29.467 Go::SmoothTransition Class Reference

```
#include <SmoothTransition.h>
```

Inheritance diagram for `Go::SmoothTransition`:



Collaboration diagram for `Go::SmoothTransition`:



## Public Member Functions

- [SmoothTransition](#) (shared\_ptr< const SplineCurve > &inters\_crv, shared\_ptr< const SplineCurve > &p\_crv1, shared\_ptr< const SplineCurve > &p\_crv2, shared\_ptr< const ParamSurface > surf1, shared\_ptr< const ParamSurface > surf2, double offset\_dist1, double offset\_dist2, double epsgeo)
- virtual [~SmoothTransition](#) ()  
*Empty destructor.*
- virtual std::vector< Point > [eval](#) (double t)
- virtual void [eval](#) (double t, int n, std::vector< std::vector< Point > > &der)
- virtual double [start](#) ()
- virtual double [end](#) ()
- virtual int [dim](#) ()
- virtual bool [approximationOK](#) (double par, const std::vector< Point > &approxpos, double tol1, double tol2)
- virtual int [nmbCvs](#) ()

### 29.467.1 Detailed Description

This abstract class provides an interface to a curve that can be evaluated. Given input of curve lying on two surfaces, we create a smooth transition surface by offsetting both surfaces, approximating intersecting curve, continue with projection of intersection curve on the two surfaces. Resulting projected curves and cross tangent curves may then be lofted. We're thus computing four points in [eval\(\)](#).

Definition at line 61 of file SmoothTransition.h.

### 29.467.2 Constructor & Destructor Documentation

**29.467.2.1** `Go::SmoothTransition::SmoothTransition ( shared_ptr< const SplineCurve > &inters_crv, shared_ptr< const SplineCurve > &p_crv1, shared_ptr< const SplineCurve > &p_crv2, shared_ptr< const ParamSurface > surf1, shared_ptr< const ParamSurface > surf2, double offset_dist1, double offset_dist2, double epsgeo )`

Constructor.

#### Parameters

<i>inters_crv</i>	intersection curve between two surfaces
<i>p_crv1</i>	associated parameter curve in first surface
<i>p_crv2</i>	associated parameter curve in second surface
<i>surf1</i>	first surface
<i>surf2</i>	second surface
<i>offset_dist1</i>	offset distance related to first surface
<i>offset_dist2</i>	offset distance related to second surface
<i>epsgeo</i>	approximation tolerance and tolerance used in intersection computations

**29.467.2.2** `virtual Go::SmoothTransition::~~SmoothTransition ( ) [virtual]`

Empty destructor.

### 29.467.3 Member Function Documentation

29.467.3.1 `virtual bool Go::SmoothTransition::approximationOK ( double par, const std::vector< Point > & approxpos, double tol1, double tol2 )` [virtual]

Whether the approximation is within tolerances in input parameter.

#### Parameters

<i>par</i>	parameter in which to evaluate.
<i>approxpos</i>	whether the input points are within tolerance from the evaluated points (as given by <a href="#">eval()</a> ).
<i>tol1</i>	tolerance used to decide approximation accuracy.
<i>tol2</i>	tolerance used to decide approximation accuracy.

#### Returns

whether the approximation is within tolerances in input parameter.

Implements [Go::EvalCurveSet](#).

29.467.3.2 `virtual int Go::SmoothTransition::dim ( )` [virtual]

The geometric dimension of the spline curves.

#### Returns

geometric dimension of the space.

Implements [Go::EvalCurveSet](#).

29.467.3.3 `virtual double Go::SmoothTransition::end ( )` [virtual]

End parameter of domain.

#### Returns

end parameter of the spline space.

Implements [Go::EvalCurveSet](#).

29.467.3.4 `virtual std::vector<Point> Go::SmoothTransition::eval ( double t )` [virtual]

Evaluate the curves.

#### Parameters

<i>t</i>	parameter in which to evaluate.
----------	---------------------------------

**Returns**

the evaluated points for the curve set.

Implements [Go::EvalCurveSet](#).

```
29.467.3.5 virtual void Go::SmoothTransition::eval (double t, int n, std::vector< std::vector< Point > > & der)
 [virtual]
```

Evaluate the curve derivatives.

**Parameters**

<i>t</i>	parameter in which to evaluate.
<i>n</i>	number of derivatives to compute.
<i>der</i>	the evaluated points up to the n'th derivative for the curve set.

Implements [Go::EvalCurveSet](#).

```
29.467.3.6 virtual int Go::SmoothTransition::nmbCvs () [inline],[virtual]
```

The number of curves in the curve set.

**Returns**

the number of curves in the curve set.

Implements [Go::EvalCurveSet](#).

Definition at line 96 of file SmoothTransition.h.

```
29.467.3.7 virtual double Go::SmoothTransition::start () [virtual]
```

Start parameter of domain.

**Returns**

start parameter of the spline space.

Implements [Go::EvalCurveSet](#).

The documentation for this class was generated from the following file:

- [gotools-core/include/GoTools/creators/SmoothTransition.h](#)



## 29.468 Go::SmoothVolume Class Reference

This class modifies a NURBS volume with respect to smoothness, editing constraints and boundary conditions. Specified coefficients are adjusted to minimize a functional combining the different modification conditions.

```
#include <SmoothVolume.h>
```

### Public Member Functions

- [SmoothVolume](#) ()  
*Default constructor. Initializes class variable.*
- [SmoothVolume](#) (bool copy\_coefs)
- virtual [~SmoothVolume](#) ()
- void [attach](#) (shared\_ptr< [SplineVolume](#) > &in\_vol, std::vector< [CoefStatus](#) > status)
- void [reset](#) ()  
*Reset all smoothing parameters, but keep the input volume.*
- void [setOptimize](#) (const double weight1, const double weight2, const double weight3)
- void [setLeastSquares](#) (std::vector< double > &pnts, std::vector< double > &param\_pnts, std::vector< double > &pnt\_weights, const double weight)
- void [setPeriodicity](#) (int pardir, int cont, double weight1, double weight2)
- int [equationSolve](#) (shared\_ptr< [SplineVolume](#) > &vol)

### 29.468.1 Detailed Description

This class modifies a NURBS volume with respect to smoothness, editing constraints and boundary conditions. Specified coefficients are adjusted to minimize a functional combining the different modification conditions.

Definition at line 68 of file SmoothVolume.h.

### 29.468.2 Constructor & Destructor Documentation

#### 29.468.2.1 Go::SmoothVolume::SmoothVolume ( )

Default constructor. Initializes class variable.

#### 29.468.2.2 Go::SmoothVolume::SmoothVolume ( bool copy\_coefs )

Constructor. Initializes class variable.

#### Parameters

<i>copy_coefs</i>	true if coefficients on attached volume are not to be modified.
-------------------	-----------------------------------------------------------------

#### 29.468.2.3 virtual Go::SmoothVolume::~~SmoothVolume ( ) [virtual]

### 29.468.3 Member Function Documentation

29.468.3.1 `void Go::SmoothVolume::attach ( shared_ptr< SplineVolume > & in_vol, std::vector< CoefStatus > status )`

Initializes data given by an intermediate volume.

#### Parameters

<i>in_vol</i>	the initial volume.
<i>status</i>	array with the freedom for each coefficient of the volume CoefFree: Not known, CoefKnown: Known, CoefAvoid: Not of interest

29.468.3.2 `int Go::SmoothVolume::equationSolve ( shared_ptr< SplineVolume > & vol )`

Set up and solve equation system, and produce output volume. If failing to solve the routine may throw an exception.

#### Parameters

<i>vol</i>	the output surface.
------------	---------------------

#### Returns

0 = OK, negative = failed solving system.

29.468.3.3 `void Go::SmoothVolume::reset ( )`

Reset all smoothing parameters, but keep the input volume.

29.468.3.4 `void Go::SmoothVolume::setLeastSquares ( std::vector< double > & pnts, std::vector< double > & param_pnts, std::vector< double > & pnt_weights, const double weight )`

Add data for the least squares approximation.

#### Parameters

<i>pnts</i>	points in volume to be approximated. Stored as (for 3D): (x0, y0, z0, x1, y1, z1, ...)
<i>param_pnts</i>	the corresponding 3-dimensional parametric points. Stored as: (u0, v0, w0, u1, v1, w1, ...)
<i>pnt_weights</i>	each of the pnts is assigned a weight lying in the unit interval. 1.0 if all pnts are equally important.
<i>weight</i>	the contribution of the approximation of the pnts in the system. weight should lie in the unit interval.

29.468.3.5 `void Go::SmoothVolume::setOptimize ( const double weight1, const double weight2, const double weight3 )`

Add weights for the smoothing part of the equation system.

## Parameters

<i>weight1</i>	contribution weight with respect to the 1st derivative.
<i>weight2</i>	contribution weight with respect to the 2nd derivative.
<i>weight3</i>	contribution weight with respect to the 3rd derivative.

29.468.3.6 void Go::SmoothVolume::setPeriodicity ( int *parDir*, int *cont*, double *weight1*, double *weight2* )

Add periodicity constraints in one par. dir.

## Parameters

<i>parDir</i>	the direction of the periodicity, 0, 1 or 2
<i>cont</i>	the continuity across the seam, 0 = C0, 1 = C1, 2 = C2.
<i>weight1</i>	C1 continuity contribution used in approximation.
<i>weight2</i>	C2 continuity contribution used in approximation.

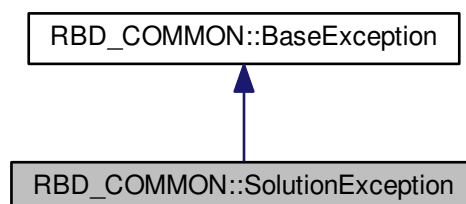
The documentation for this class was generated from the following file:

- [trivariate/include/GoTools/trivariate/SmoothVolume.h](#)

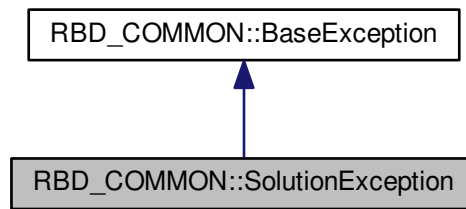
## 29.469 RBD\_COMMON::SolutionException Class Reference

```
#include <solution.h>
```

Inheritance diagram for RBD\_COMMON::SolutionException:



Collaboration diagram for RBD\_COMMON::SolutionException:



## Public Member Functions

- [SolutionException](#) (`const char *a_what=0`)

## Static Public Attributes

- static unsigned long [Select](#)

## Additional Inherited Members

### 29.469.1 Detailed Description

Definition at line 45 of file `solution.h`.

### 29.469.2 Constructor & Destructor Documentation

#### 29.469.2.1 `SolutionException::SolutionException ( const char * a_what = 0 )`

Definition at line 36 of file `solution.cpp`.

### 29.469.3 Member Data Documentation

#### 29.469.3.1 `unsigned long SolutionException::Select [static]`

Definition at line 48 of file `solution.h`.

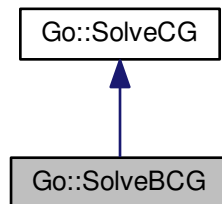
The documentation for this class was generated from the following files:

- `newmat/include/solution.h`
- `newmat/src/solution.cpp`

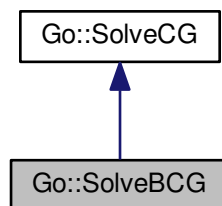
## 29.470 Go::SolveBCG Class Reference

```
#include <SolveBCG.h>
```

Inheritance diagram for Go::SolveBCG:



Collaboration diagram for Go::SolveBCG:



### Public Member Functions

- [SolveBCG](#) (int conv\_type, bool symm)
- virtual [~SolveBCG](#) ()  
*Destructor.*
- virtual void [precond](#) (double relaxfac)  
*Prepare for preconditioning.*
- virtual int [solve](#) (double \*ex, double \*eb, int nn)  
*Solve the equation system.*

### Additional Inherited Members

#### 29.470.1 Detailed Description

Using the biconjugate gradient method to solve sparse and positive definite matrix systems.

Definition at line 52 of file SolveBCG.h.

## 29.470.2 Constructor & Destructor Documentation

### 29.470.2.1 `Go::SolveBCG::SolveBCG ( int conv_type, bool symm )`

Constructor.

Parameters

<code><i>conv_type</i></code>	in the range 1 to 4. Convergence estimate. The selection of preconditioner is based on this number
-------------------------------	----------------------------------------------------------------------------------------------------

### 29.470.2.2 `virtual Go::SolveBCG::~~SolveBCG ( ) [virtual]`

Destructor.

## 29.470.3 Member Function Documentation

### 29.470.3.1 `virtual void Go::SolveBCG::precond ( double relaxfac ) [virtual]`

Prepare for preconditioning.

### 29.470.3.2 `virtual int Go::SolveBCG::solve ( double * ex, double * eb, int nn ) [virtual]`

Solve the equation system.

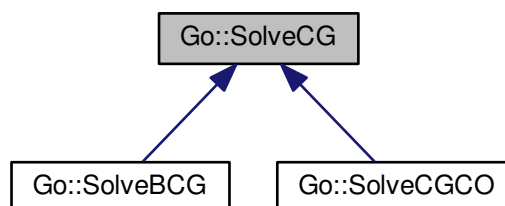
The documentation for this class was generated from the following file:

- [gotools-core/include/GoTools/creators/SolveBCG.h](#)

## 29.471 `Go::SolveCG` Class Reference

```
#include <SolveCG.h>
```

Inheritance diagram for `Go::SolveCG`:



## Public Member Functions

- [SolveCG](#) ()  
*Default constructor.*
- virtual [~SolveCG](#) ()  
*Destructor.*
- void [attachMatrix](#) (double \*gmat, int nn)
- virtual void [precondRILU](#) (double relaxfac)
- int [solve](#) (double \*ex, double \*eb, int nn)
- void [setTolerance](#) (double tolerance=1.0e-6)
- void [setMaxIterations](#) (int max\_ iterations)

## Protected Member Functions

- template<typename RandomIterator1 , typename RandomIterator2 >  
void [matrixProduct](#) (RandomIterator1 sx, RandomIterator2 sy)
- int [getIndex](#) (int ki, int kj)  
*Given an index in the full equation system, get the index in A\_.*
- void [forwBack](#) (double \*r, double \*s)
- void [transposedMatrixProduct](#) (double \*sx, double \*sy)
- int [solveRILU](#) (double \*ex, double \*eb, int nn)
- int [solveStd](#) (double \*ex, double \*eb, int nn)
- void [printPrecond](#) ()  
*Print to file ("fM.m") the preconditioning matrix.*

## Protected Attributes

- std::vector< double > [A\\_](#)
- int [nn\\_](#)
- int [np\\_](#)
- std::vector< int > [irow\\_](#)
- std::vector< int > [jcol\\_](#)
- double [tolerance\\_](#)
- int [max\\_ iterations\\_](#)
- std::vector< double > [M\\_](#)
- double [omega\\_](#)
- std::vector< int > [diagonal\\_](#)
- int [diagset\\_](#)

### 29.471.1 Detailed Description

Solve the equation system  $Ax=b$  where  $A$  is a symmetric positive definite matrix using the Conjugate Gradient Method.

Definition at line 63 of file SolveCG.h.

### 29.471.2 Constructor & Destructor Documentation

#### 29.471.2.1 Go::SolveCG::SolveCG ( )

Default constructor.

29.471.2.2 virtual Go::SolveCG::~~SolveCG ( ) [virtual]

Destructor.

### 29.471.3 Member Function Documentation

29.471.3.1 void Go::SolveCG::attachMatrix ( double \* *gmat*, int *nn* )

Attach the left side of the equation system to the current object and represent the matrix as a sparse matrix. No test is applied on whether the matrix really is symmetric and positive definite.

#### Parameters

<i>gmat</i>	the system matrix for the linear equations. Size is $nn \times nn$ , stored columnwise.
<i>nn</i>	the number of unknowns in the system.

29.471.3.2 void Go::SolveCG::forwBack ( double \* *r*, double \* *s* ) [protected]

Apply preconditioning matrix, i.e. solve the equation system  $M_*s = r$ , where  $M_*$  stores an LU-factorized matrix.

#### Parameters

<i>r</i>	the input (right side) vector.
<i>s</i>	the output (unknown) vector.

29.471.3.3 int Go::SolveCG::getIndex ( int *ki*, int *kj* ) [protected]

Given an index in the full equation system, get the index in  $A_*$ .

29.471.3.4 template<typename Randomlterator1 , typename Randomlterator2 > void Go::SolveCG::matrixProduct ( Randomlterator1 *sx*, Randomlterator2 *sy* ) [inline], [protected]

Compute the matrix product  $sy = A_* * sx$ .

#### Parameters

<i>sx</i>	the vector to be multiplied by the matrix.
<i>sy</i>	the resulting vector.

Definition at line 130 of file SolveCG.h.

29.471.3.5 virtual void Go::SolveCG::precondRILU ( double *relaxfac* ) [virtual]

Prepare for preconditioning.



## Parameters

<i>relaxfac</i>	relaxation parameter. Range: [0,0, 1.0].
-----------------	------------------------------------------

Reimplemented in [Go::SolveCGCO](#).

29.471.3.6 void Go::SolveCG::printPrecond ( ) [protected]

Print to file ("fM.m") the preconditioning matrix.

29.471.3.7 void Go::SolveCG::setMaxIterations ( int *max\_iterations* ) [inline]

Set the maximal number of iterations to be used by the solver.

## Parameters

<i>max_iterations</i>	the maximal number of iterations.
-----------------------	-----------------------------------

Definition at line 101 of file SolveCG.h.

29.471.3.8 void Go::SolveCG::setTolerance ( double *tolerance* = 1.0e-6 ) [inline]

Set numerical tolerance used by the solver.

## Parameters

<i>tolerance</i>	numerical tolerance.
------------------	----------------------

Definition at line 96 of file SolveCG.h.

29.471.3.9 int Go::SolveCG::solve ( double \* *ex*, double \* *eb*, int *nn* )

Solve the equation system by conjugate gradient method.

## Parameters

<i>ex</i>	the solution vector. The input should be the initial guess. Size is equal to nn.
<i>eb</i>	the right side of the equation. Size is equal to nn.
<i>nn</i>	the number of unknowns in the system.

## Returns

0: success, 1: iterationcount exceeded, < 0: error.

29.471.3.10 `int Go::SolveCG::solveRILU ( double * ex, double * eb, int nn )` [protected]

Solve the equation system by conjugate gradient method using a given RILU (Relaxed Incomplete LU) preconditioner

#### Parameters

<i>ex</i>	the solution vector. The input should be the initial guess. Size is equal to <i>nn</i> .
<i>eb</i>	the right side of the equation. Size is equal to <i>nn</i> .
<i>nn</i>	the number of unknowns in the system.

#### Returns

0: success, 1: iterationcount exceeded, < 0: error.

29.471.3.11 `int Go::SolveCG::solveStd ( double * ex, double * eb, int nn )` [protected]

#### Parameters

<i>ex</i>	the solution vector. The input should be the initial guess. Size is equal to <i>nn</i> .
<i>eb</i>	the right side of the equation. Size is equal to <i>nn</i> .
<i>nn</i>	the number of unknowns in the system.

#### Returns

0: success, 1: iterationcount exceeded, < 0: error.

29.471.3.12 `void Go::SolveCG::transposedMatrixProduct ( double * sx, double * sy )` [protected]

## 29.471.4 Member Data Documentation

29.471.4.1 `std::vector<double> Go::SolveCG::A_` [protected]

Definition at line 107 of file SolveCG.h.

29.471.4.2 `std::vector<int> Go::SolveCG::diagonal_` [protected]

Definition at line 123 of file SolveCG.h.

29.471.4.3 `int Go::SolveCG::diagset_` [protected]

Definition at line 124 of file SolveCG.h.

29.471.4.4 `std::vector<int> Go::SolveCG::irow_` [protected]

Definition at line 111 of file SolveCG.h.

29.471.4.5 `std::vector<int> Go::SolveCG::jcol_` [protected]

Definition at line 113 of file SolveCG.h.

29.471.4.6 `std::vector<double> Go::SolveCG::M_` [protected]

Definition at line 120 of file SolveCG.h.

29.471.4.7 `int Go::SolveCG::max_iterations_` [protected]

Definition at line 116 of file SolveCG.h.

29.471.4.8 `int Go::SolveCG::nn_` [protected]

Definition at line 109 of file SolveCG.h.

29.471.4.9 `int Go::SolveCG::np_` [protected]

Definition at line 110 of file SolveCG.h.

29.471.4.10 `double Go::SolveCG::omega_` [protected]

Definition at line 121 of file SolveCG.h.

29.471.4.11 `double Go::SolveCG::tolerance_` [protected]

Definition at line 115 of file SolveCG.h.

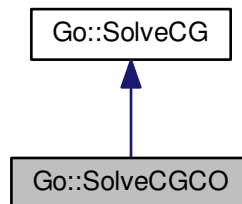
The documentation for this class was generated from the following file:

- [gotools-core/include/GoTools/creators/SolveCG.h](#)

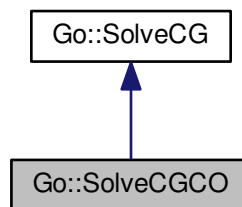
## 29.472 Go::SolveCGCO Class Reference

```
#include <SolveCGCO.h>
```

Inheritance diagram for Go::SolveCGCO:



Collaboration diagram for Go::SolveCGCO:



### Public Member Functions

- [SolveCGCO](#) (int m, int n)
- virtual void [precondRILU](#) (double relaxfac)
- virtual [~SolveCGCO](#) ()

### Additional Inherited Members

#### 29.472.1 Detailed Description

Definition at line 69 of file SolveCGCO.h.

## 29.472.2 Constructor & Destructor Documentation

29.472.2.1 `Go::SolveCGCO::SolveCGCO ( int m, int n )`

29.472.2.2 `virtual Go::SolveCGCO::~~SolveCGCO ( )` [virtual]

## 29.472.3 Member Function Documentation

29.472.3.1 `virtual void Go::SolveCGCO::precondRILU ( double relaxfac )` [virtual]

Prepare for preconditioning.

### Parameters

<i>relaxfac</i>	relaxation parameter. Range: [0,0, 1.0].
-----------------	------------------------------------------

Reimplemented from [Go::SolveCG](#).

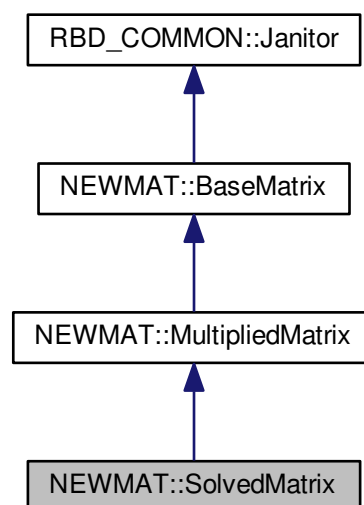
The documentation for this class was generated from the following file:

- `gtools-core/include/GoTools/creators/SolveCGCO.h`

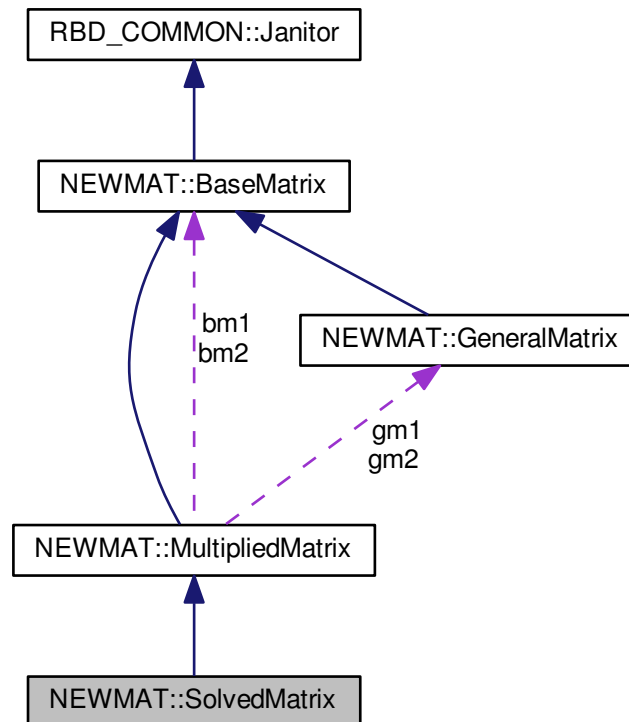
## 29.473 NEWMAT::SolvedMatrix Class Reference

```
#include <newmat.h>
```

Inheritance diagram for NEWMAT::SolvedMatrix:



Collaboration diagram for NEWMAT::SolvedMatrix:



## Public Member Functions

- [~SolvedMatrix \(\)](#)
- [GeneralMatrix \\* Evaluate \(MatrixType mt=MatrixTypeUnSp\)](#)
- [MatrixBandWidth BandWidth \(\) const](#)

## Friends

- class [BaseMatrix](#)
- class [InvertedMatrix](#)

## Additional Inherited Members

### 29.473.1 Detailed Description

Definition at line 1285 of file newmat.h.

## 29.473.2 Constructor & Destructor Documentation

29.473.2.1 `NEWMAT::SolvedMatrix::~~SolvedMatrix ( )` `[inline]`

Definition at line 1292 of file `newmat.h`.

## 29.473.3 Member Function Documentation

29.473.3.1 `MatrixBandWidth SolvedMatrix::BandWidth ( ) const` `[virtual]`

Reimplemented from [NEWMAT::MultipliedMatrix](#).

Definition at line 458 of file `newmat4.cpp`.

29.473.3.2 `GeneralMatrix * SolvedMatrix::Evaluate ( MatrixType mt = MatrixTypeUnSp )` `[virtual]`

Reimplemented from [NEWMAT::MultipliedMatrix](#).

Definition at line 115 of file `newmat7.cpp`.

## 29.473.4 Friends And Related Function Documentation

29.473.4.1 `friend class BaseMatrix` `[friend]`

Definition at line 1289 of file `newmat.h`.

29.473.4.2 `friend class InvertedMatrix` `[friend]`

Definition at line 1290 of file `newmat.h`.

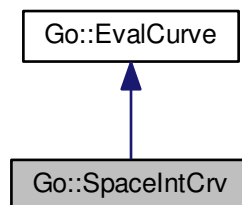
The documentation for this class was generated from the following files:

- [newmat/include/newmat.h](#)
- [newmat/src/newmat4.cpp](#)
- [newmat/src/newmat7.cpp](#)

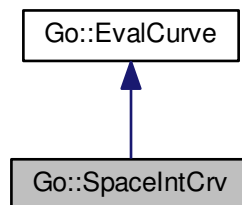
## 29.474 Go::SpaceIntCrv Class Reference

```
#include <SpaceIntCrv.h>
```

Inheritance diagram for Go::SpaceIntCrv:



Collaboration diagram for Go::SpaceIntCrv:



### Public Member Functions

- [SpaceIntCrv](#) (shared\_ptr< [ParamCurve](#) > init\_crv, int paddir, std::vector< shared\_ptr< [CurveOnSurface](#) > &sfcv1, std::vector< double > start1, std::vector< double > end1, std::vector< shared\_ptr< [CurveOnSurface](#) > > &sfcv2, std::vector< double > start2, std::vector< double > end2, std::vector< bool > opposite, bool same\_orient)
- virtual [~SpaceIntCrv](#) ()  
*Destructor.*
- virtual [Point eval](#) (double t) const
- virtual void [eval](#) (double t, int n, [Point](#) der[]) const
- virtual [double start](#) () const
- virtual [double end](#) () const
- virtual [int dim](#) () const  
*The geometric dimension of the spline curves.*
- virtual [bool approximationOK](#) (double par, [Point](#) approxpos, double tol1, double tol2) const



### 29.474.1 Detailed Description

This class represents an intersection curve approximated by two curve on surface instances

Definition at line 52 of file SpaceIntCrv.h.

### 29.474.2 Constructor & Destructor Documentation

29.474.2.1 `Go::SpaceIntCrv::SpaceIntCrv ( shared_ptr< ParamCurve > init_crv, int pardir, std::vector< shared_ptr< CurveOnSurface > > & sfcv1, std::vector< double > start1, std::vector< double > end1, std::vector< shared_ptr< CurveOnSurface > > & sfcv2, std::vector< double > start2, std::vector< double > end2, std::vector< bool > opposite, bool same_orient )`

Constructor

Parameters

<i>init_crv</i>	approximation to true intersection curve
<i>pardir</i>	assumes monotonicity of intersection curve in this parameter direction
<i>sfcv1</i>	set of curves joining into the intersection curve representation related to first surface
<i>start1</i>	startparameter of the curves sfcv1. May restrict the extent of each curve.
<i>end1</i>	endparameter of the curves sfcv1. May restrict the extent of each curve.
<i>sfcv2</i>	set of curves joining into intersection curve representation related to second surface
<i>start2</i>	startparameter of the curves sfcv2. May restrict the extent of each curve.
<i>end2</i>	endparameter of the curves sfcv2. May restrict the extent of each curve.
<i>opposite</i>	pairwise comparisment of orientation from the curves in sfcv1 and sfcv2
<i>same_orient</i>	whether the curves in sfcv1 have the same orientation as the underlying surface

29.474.2.2 `virtual Go::SpaceIntCrv::~SpaceIntCrv ( ) [virtual]`

Destructor.

### 29.474.3 Member Function Documentation

29.474.3.1 `virtual bool Go::SpaceIntCrv::approximationOK ( double par, Point approxpos, double tol1, double tol2 ) const [virtual]`

Whether the approximation is within tolerances in input parameter.

Parameters

<i>par</i>	parameter in which to evaluate.
<i>approxpos</i>	whether the input point are within tolerance from the evaluated points (as given by <a href="#">eval()</a> ).
<i>tol1</i>	tolerance used to decide approximation accuracy.
<i>tol2</i>	tolerance used to decide approximation accuracy.

**Returns**

whether the approximation is within tolerances in input parameter.

Implements [Go::EvalCurve](#).

29.474.3.2 `virtual int Go::SpaceIntCrv::dim ( ) const [virtual]`

The geometric dimension of the spline curves.

Implements [Go::EvalCurve](#).

29.474.3.3 `virtual double Go::SpaceIntCrv::end ( ) const [virtual]`

End parameter of domain.

**Returns**

end parameter of the spline space.

Implements [Go::EvalCurve](#).

29.474.3.4 `virtual Point Go::SpaceIntCrv::eval ( double t ) const [virtual]`

Evaluate the curves.

**Parameters**

<i>t</i>	parameter in which to evaluate.
----------	---------------------------------

**Returns**

the evaluated point for the curve.

Implements [Go::EvalCurve](#).

29.474.3.5 `virtual void Go::SpaceIntCrv::eval ( double t, int n, Point der[] ) const [virtual]`

Evaluate the curve derivatives.

**Parameters**

<i>t</i>	parameter in which to evaluate.
<i>n</i>	number of derivatives to compute.
<i>der</i>	the evaluated points up to the n'th derivative for the curve.

Implements [Go::EvalCurve](#).

29.474.3.6 `virtual double Go::SpaceIntCrv::start ( ) const [virtual]`

Start parameter of domain.

#### Returns

start parameter of the spline space.

Implements [Go::EvalCurve](#).

The documentation for this class was generated from the following file:

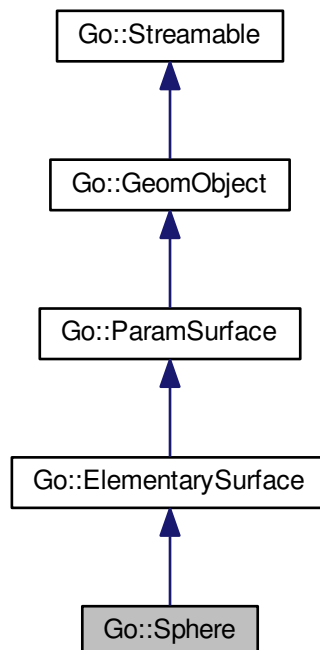
- [gotools-core/include/GoTools/creators/SpaceIntCrv.h](#)

## 29.475 Go::Sphere Class Reference

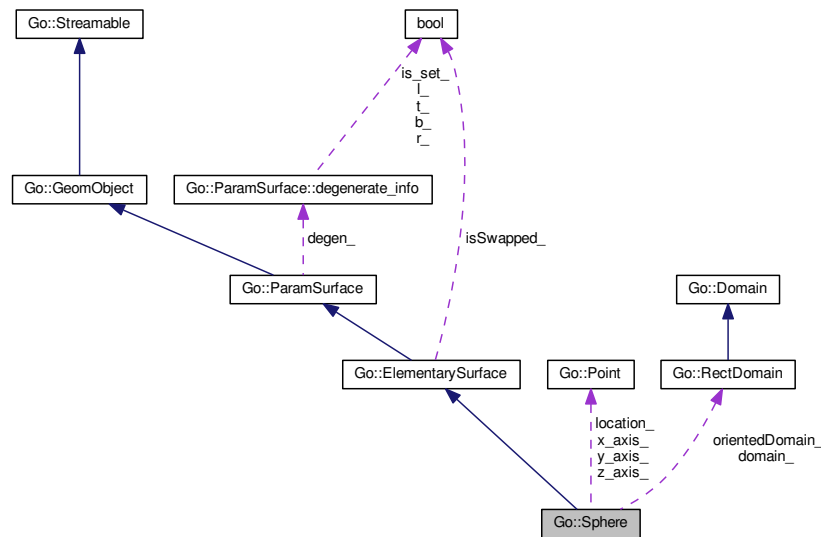
Class that represents a sphere. It is a subclass of [ElementarySurface](#), and thus has a parametrization and is non-selfintersecting.

```
#include <Sphere.h>
```

Inheritance diagram for Go::Sphere:



Collaboration diagram for Go::Sphere:



## Public Member Functions

- [Sphere](#) ()
- [Sphere](#) (double radius, [Point](#) location, [Point](#) z\_axis, [Point](#) x\_axis, bool isSwapped=false)
- virtual [~Sphere](#) ()
  - Virtual destructor - ensures safe inheritance.*
- virtual void [read](#) (std::istream &is)
- virtual void [write](#) (std::ostream &os) const
- virtual int [dimension](#) () const
  - Return the dimension of the space in which the object lies (usually 2 or 3)*
- virtual [ClassType](#) [instanceType](#) () const
  - Return the class type identifier of a given, derived instance of [GeomObject](#).*
- virtual [BoundingBox](#) [boundingBox](#) () const
  - Return the object's bounding box.*
- virtual [Sphere](#) \* [clone](#) () const
- const [RectDomain](#) & [parameterDomain](#) () const
- std::vector< [CurveLoop](#) > [allBoundaryLoops](#) (double degenerate\_epsilon=DEFAULT\_SPACE\_EPSILON) const
- [DirectionCone](#) [normalCone](#) () const
- [DirectionCone](#) [tangentCone](#) (bool paddir\_is\_u) const
- void [point](#) ([Point](#) &pt, double upar, double vpar) const
- void [point](#) (std::vector< [Point](#) > &pts, double upar, double vpar, int derivs, bool u\_from\_right=true, bool v\_from\_right=true, double resolution=1.0e-12) const
- void [normal](#) ([Point](#) &n, double upar, double vpar) const
- std::vector< shared\_ptr< [ParamCurve](#) > > [constParamCurves](#) (double parameter, bool paddir\_is\_u) const
- std::vector< shared\_ptr< [ParamSurface](#) > > [subSurfaces](#) (double from\_upar, double from\_vpar, double to\_upar, double to\_vpar, double fuzzy=DEFAULT\_PARAMETER\_EPSILON) const
- double [nextSegmentVal](#) (int dir, double par, bool forward, double tol) const
- void [closestPoint](#) (const [Point](#) &pt, double &clo\_u, double &clo\_v, [Point](#) &clo\_pt, double &clo\_dist, double epsilon, const [RectDomain](#) \*domain\_of\_interest=NULL, double \*seed=0) const

- void [closestBoundaryPoint](#) (const [Point](#) &pt, double &clo\_u, double &clo\_v, [Point](#) &clo\_pt, double &clo\_dist, double epsilon, const [RectDomain](#) \*rd=NULL, double \*seed=0) const
- void [getBoundaryInfo](#) ([Point](#) &pt1, [Point](#) &pt2, double epsilon, [SplineCurve](#) \*&cv, [SplineCurve](#) \*&crosscv, double knot\_tol=1e-05) const
- virtual shared\_ptr< [ElementaryCurve](#) > [getElementaryParamCurve](#) ([ElementaryCurve](#) \*space\_crv, double tol, const [Point](#) \*start\_par\_pt=NULL, const [Point](#) \*end\_par\_pt=NULL) const
- [bool](#) [isDegenerate](#) ([bool](#) &b, [bool](#) &r, [bool](#) &t, [bool](#) &l, double tolerance) const
- [bool](#) [isBounded](#) () const
- [bool](#) [isClosed](#) ([bool](#) &closed\_dir\_u, [bool](#) &closed\_dir\_v) const  
*Check if the sphere is closed.*
- virtual void [getDegenerateCorners](#) (std::vector< [Point](#) > &deg\_corners, double tol) const  
*Check for paralell and anti paralell partial derivatives in surface corners.*
- double [getRadius](#) () const  
*Sphere radius.*
- [Point](#) [getLocation](#) () const  
*Sphere centre.*
- void [getCoordinateAxes](#) ([Point](#) &x\_axis, [Point](#) &y\_axis, [Point](#) &z\_axis) const  
*Local coordinate axes.*
- virtual void [setParameterBounds](#) (double from\_upar, double from\_vpar, double to\_upar, double to\_vpar)  
*Limit the surface by limiting the parameter domain.*
- [Sphere](#) \* [subSurface](#) (double from\_upar, double from\_vpar, double to\_upar, double to\_vpar, double fuzzy=DEFAULT\_PARAMETER\_EPSILON) const  
*Pick part of sphere limited by parameter domain information.*
- virtual [SplineSurface](#) \* [geometrySurface](#) () const  
*A SplineSurface representation of the sphere.*
- virtual [SplineSurface](#) \* [createSplineSurface](#) () const  
*Create a SplineSurface representation of the sphere.*
- shared\_ptr< [Circle](#) > [getLatitudinalCircle](#) (double vpar) const
- shared\_ptr< [Circle](#) > [getLongitudinalCircle](#) (double upar) const
- virtual [bool](#) [isAxisRotational](#) ([Point](#) &centre, [Point](#) &axis, [Point](#) &vec, double &angle)

### Static Public Member Functions

- static [ClassType](#) [classType](#) ()

### Protected Member Functions

- void [setCoordinateAxes](#) ()

### Protected Attributes

- double [radius\\_](#)
- [Point](#) [location\\_](#)
- [Point](#) [z\\_axis\\_](#)
- [Point](#) [x\\_axis\\_](#)
- [Point](#) [y\\_axis\\_](#)
- [RectDomain](#) [domain\\_](#)
- [RectDomain](#) [orientedDomain\\_](#)

### 29.475.1 Detailed Description

Class that represents a sphere. It is a subclass of [ElementarySurface](#), and thus has a parametrization and is non-selfintersecting.

A [Sphere](#) has a natural parametrization in terms of the angles  $u$  and  $v$ :  $\mathbf{p}(u, v) = \mathbf{C} + R \cos v(\cos u) \mathbf{x} + \sin(u) \mathbf{y} + R(\sin v) \mathbf{z}$ , where  $\mathbf{C}$  is a position vector,  $R$  is the radius, and  $\mathbf{x}$ ,  $\mathbf{y}$  and  $\mathbf{z}$  are the (local) axes. The parametrization is bounded by:  $0 \leq u \leq 2\pi$ ,  $-\frac{\pi}{2} < v < \frac{\pi}{2}$ . The dimension is 3.

Definition at line 67 of file Sphere.h.

### 29.475.2 Constructor & Destructor Documentation

#### 29.475.2.1 `Go::Sphere::Sphere( ) [inline]`

Default constructor. Constructs an uninitialized [Sphere](#) which can only be assigned to or read into.

Definition at line 72 of file Sphere.h.

#### 29.475.2.2 `Go::Sphere::Sphere( double radius, Point location, Point z_axis, Point x_axis, bool isSwapped = false )`

Constructor. Input is the radius, the location, the direction of the z-axis and the (possibly approximate) x-axis. The local coordinate axes are normalized even if `z_axis` and/or `x_axis` are not unit vectors.

#### 29.475.2.3 `virtual Go::Sphere::~Sphere( ) [virtual]`

Virtual destructor - ensures safe inheritance.

### 29.475.3 Member Function Documentation

#### 29.475.3.1 `std::vector<CurveLoop> Go::Sphere::allBoundaryLoops( double degenerate_epsilon = DEFAULT_SPACE_EPSILON ) const [virtual]`

Returns the anticlockwise outer boundary loop of the surface, together with clockwise loops of any interior boundaries, such that the surface always is 'to the left of' the loops.

#### Parameters

<code>degenerate_epsilon</code>	edges whose length is smaller than this value are ignored.
---------------------------------	------------------------------------------------------------

#### Returns

a vector containing `CurveLoops`. The first of these describe the outer boundary of the surface (clockwise), whereas the others describe boundaries of interior holes (clockwise).

Implements [Go::ParamSurface](#).

29.475.3.2 virtual **BoundingBox** Go::Sphere::boundingBox ( ) const [virtual]

Return the object's bounding box.

Implements [Go::GeomObject](#).

29.475.3.3 static **ClassType** Go::Sphere::classType ( ) [inline],[static]

Definition at line 99 of file Sphere.h.

29.475.3.4 virtual **Sphere\*** Go::Sphere::clone ( ) const [virtual]

make a clone of this surface and return a pointer to it (user is responsible for clearing up memory afterwards).

Returns

pointer to cloned object

Implements [Go::ElementarySurface](#).

29.475.3.5 void Go::Sphere::closestBoundaryPoint ( const **Point** & *pt*, double & *clo\_u*, double & *clo\_v*, **Point** & *clo\_pt*, double & *clo\_dist*, double *epsilon*, const **RectDomain** \* *rd* = NULL, double \* *seed* = 0 ) const [virtual]

Iterates to the closest point to *pt* on the boundary of the surface.

See also

[closestPoint\(\)](#)

Implements [Go::ParamSurface](#).

29.475.3.6 void Go::Sphere::closestPoint ( const **Point** & *pt*, double & *clo\_u*, double & *clo\_v*, **Point** & *clo\_pt*, double & *clo\_dist*, double *epsilon*, const **RectDomain** \* *domain\_of\_interest* = NULL, double \* *seed* = 0 ) const [virtual]

Iterates to the closest point to *pt* on the surface.

Parameters

<i>pt</i>	the point to find the closest point to
<i>clo_u</i>	u parameter of the closest point
<i>clo_v</i>	v parameter of the closest point
<i>clo_pt</i>	the geometric position of the closest point
<i>clo_dist</i>	the distance between <i>pt</i> and <i>clo_pt</i>
<i>epsilon</i>	parameter tolerance (will in any case not be higher than sqrt(machine_precision) x magnitude of solution)
<i>domain_of_interest</i>	pointer to parameter domain in which to search for closest point. If a NULL pointer is used, the entire surface is searched.
<b>Generated by Doxygen</b>	
<i>seed</i>	pointer to parameter values where iteration starts.

Reimplemented from [Go::ParamSurface](#).

```
29.475.3.7 std::vector<shared_ptr<ParamCurve>> Go::Sphere::constParamCurves (double parameter, bool
 pardir_is_u) const [virtual]
```

Get the curve(s) obtained by intersecting the surface with one of its constant parameter curves. For surfaces without holes, this will be the parameter curve itself; for surfaces with interior holes this may be a collection of several, disjoint curves.

#### Parameters

<i>parameter</i>	parameter value for the constant parameter (either u or v)
<i>pardir_is_u</i>	specify whether the <i>moving</i> parameter (as opposed to the <i>constant</i> parameter) is the first ('true') or the second ('false') one.

#### Returns

a vector containing shared pointers to the obtained, newly constructed constant-parameter curves.

Implements [Go::ParamSurface](#).

```
29.475.3.8 virtual SplineSurface* Go::Sphere::createSplineSurface () const [virtual]
```

Create a [SplineSurface](#) representation of the sphere.

Implements [Go::ElementarySurface](#).

```
29.475.3.9 virtual int Go::Sphere::dimension () const [virtual]
```

Return the dimension of the space in which the object lies (usually 2 or 3)

Implements [Go::GeomObject](#).

```
29.475.3.10 virtual SplineSurface* Go::Sphere::geometrySurface () const [virtual]
```

A [SplineSurface](#) representation of the sphere.

Implements [Go::ElementarySurface](#).

```
29.475.3.11 void Go::Sphere::getBoundaryInfo (Point & pt1, Point & pt2, double epsilon, SplineCurve *& cv,
 SplineCurve *& crosscv, double knot_tol = 1e-05) const [virtual]
```

Get the boundary curve segment between two points on the boundary, as well as the cross-tangent curve. If the given points are not positioned on the same boundary (within a certain tolerance), no curves will be created.



## Parameters

<i>pt1</i>	the first point on the boundary, given by the user
<i>pt2</i>	the second point on the boundary, given by the user
<i>epsilon</i>	the tolerance used when determining whether the given points are lying on a boundary, and if they do, whether they both lie on the <i>same</i> boundary.
<i>cv</i>	upon return, this will point to a newly created curve representing the boundary curve between 'pt1' and 'pt2'. The user assumes ownership of this object and is responsible for its deletion. No curve is created if the given points are not found to lie on the same boundary.
<i>crosscv</i>	upon return, this will point to a newly created curve representing the cross-boundary curve between 'pt1' and 'pt2'. The user assumes ownership of this object and is responsible for its deletion. The direction is outwards from the surface. No curve is created if the given points are not found to lie on the same boundary.
<i>knot_tol</i>	tolerance used when working with the knot-vector, to specify how close a parameter value must be to a knot in order to be considered 'on' the knot.

Implements [Go::ParamSurface](#).

```
29.475.3.12 void Go::Sphere::getCoordinateAxes (Point & x_axis, Point & y_axis, Point & z_axis) const [inline]
```

Local coordinate axes.

Definition at line 194 of file Sphere.h.

```
29.475.3.13 virtual void Go::Sphere::getDegenerateCorners (std::vector< Point > & deg_corners, double tol) const [virtual]
```

Check for paralell and anti paralell partial derivatives in surface corners.

Implements [Go::ParamSurface](#).

```
29.475.3.14 virtual shared_ptr<ElementaryCurve> Go::Sphere::getElementaryParamCurve (ElementaryCurve * space_crv, double tol, const Point * start_par_pt = NULL, const Point * end_par_pt = NULL) const [virtual]
```

Fetch the parameter curve in the domain of the elementary surface corresponding to a given elementary curve in geometry space if this curve has a simpler elementary representation. Otherwise, nothing is returned

Reimplemented from [Go::ElementarySurface](#).

```
29.475.3.15 shared_ptr<Circle> Go::Sphere::getLatitudinalCircle (double vpar) const
```

Get the circle along the latitude for a given v parameter.

## Parameters

<i>vpar</i>	v parameter
-------------	-------------

**Returns**

A circle for the corresponding *v* parameter. If the *v* parameter is bounded, only a segment of a full circle is returned.

29.475.3.16 **Point** Go::Sphere::getLocation ( ) const [inline]

[Sphere](#) centre.

Definition at line 190 of file Sphere.h.

29.475.3.17 **shared\_ptr<Circle>** Go::Sphere::getLongitudinalCircle ( **double** *upar* ) const

Get the circle along a longitude for a given *u* parameter.

**Parameters**

<i>upar</i>	<i>u</i> parameter
-------------	--------------------

**Returns**

A circle for the corresponding *u* parameter. If the *u* parameter is bounded, only a segment of a full circle is returned.

29.475.3.18 **double** Go::Sphere::getRadius ( ) const [inline]

[Sphere](#) radius.

Definition at line 186 of file Sphere.h.

29.475.3.19 **virtual ClassType** Go::Sphere::instanceType ( ) const [virtual]

Return the class type identifier of a given, derived instance of [GeomObject](#).

Implements [Go::GeomObject](#).

29.475.3.20 **virtual bool** Go::Sphere::isAxisRotational ( **Point & centre**, **Point & axis**, **Point & vec**, **double & angle** )  
[virtual]

Check if the surface is axis rotational. Only true if a connection to an axis rotational elementary surface exist The axis and rotational angle is only specified if the surface is actually rotational

Reimplemented from [Go::ParamSurface](#).

29.475.3.21 `bool Go::Sphere::isBounded ( ) const [virtual]`

Query if parametrization is bounded. All four parameter bounds must be finite for this to be true.

#### Returns

*true* if bounded, *false* otherwise

Reimplemented from [Go::ElementarySurface](#).

29.475.3.22 `bool Go::Sphere::isClosed ( bool & closed_dir_u, bool & closed_dir_v ) const [virtual]`

Check if the sphere is closed.

Reimplemented from [Go::ElementarySurface](#).

29.475.3.23 `bool Go::Sphere::isDegenerate ( bool & b, bool & r, bool & t, bool & l, double tolerance ) const [virtual]`

The order of the edge indicators (bottom, right, top, left) matches the `edge_number` of `edgeCurve()`. Query whether any of the four boundary curves are degenerate (zero length) within a certain tolerance. In the below, we refer to 'u' as the first parameter and 'v' as the second.

#### Parameters

<i>b</i>	'true' upon return of function if the boundary ( $v = v_{\min}$ ) is degenerate
<i>r</i>	'true' upon return of function if the boundary ( $v = v_{\max}$ ) is degenerate
<i>t</i>	'true' upon return of function if the boundary ( $u = u_{\min}$ ) is degenerate
<i>l</i>	'true' upon return of function if the boundary ( $u = u_{\max}$ ) is degenerate
<i>tolerance</i>	boundaries are considered degenerate if their length is shorter than this value, given by the user

#### Returns

'true' if at least one boundary curve was found to be degenerate, 'false' otherwise.

Reimplemented from [Go::ParamSurface](#).

29.475.3.24 `double Go::Sphere::nextSegmentVal ( int dir, double par, bool forward, double tol ) const [virtual]`

Determine the parameter value of the start of the 'next segment' from a parameter value, along a given parameter direction. A 'segment' is here defined as a parameter interval in which there will be no discontinuities in derivatives or other artifacts. For spline objects, a segment will typically be the interval between two consecutive, non-coincident knots.

#### Parameters

<i>dir</i>	the parameter direction in which we search for the next segment (0 or 1)
<i>par</i>	the parameter value starting from which we search for the start value of the next segment
<i>forward</i>	define whether we shall move forward ('true') or backwards when searching along this parameter
<i>tolerance</i>	tolerance used for determining whether the 'par' is already located on the next segment value

**Returns**

the value of the start value of the next segment (or the end of the previous segment, if we are moving backwards...)

Implements [Go::ParamSurface](#).

29.475.3.25 `void Go::Sphere::normal ( Point & n, double upar, double vpar ) const` [virtual]

Evaluates the surface normal for a given parameter pair

**Parameters**

<i>n</i>	the computed normal will be written to this variable
<i>upar</i>	the first parameter
<i>vpar</i>	the second parameter

Implements [Go::ParamSurface](#).

29.475.3.26 `DirectionCone Go::Sphere::normalCone ( ) const` [virtual]

Creates a [DirectionCone](#) covering all normals to this surface.

**Returns**

a [DirectionCone](#) (not necessarily the smallest) containing all normals to this surface.

Implements [Go::ParamSurface](#).

29.475.3.27 `const RectDomain& Go::Sphere::parameterDomain ( ) const` [virtual]

Return the parameter domain of the surface. This may be a simple rectangular domain ([RectDomain](#)) or any other subclass of [Domain](#) (such as [GoCurveBoundedDomain](#), found in the `sisl_dependent` module).

**Returns**

a [Domain](#) object describing the parametric domain of the surface

Implements [Go::ParamSurface](#).

29.475.3.28 `void Go::Sphere::point ( Point & pt, double upar, double vpar ) const` [virtual]

Evaluates the surface's position for a given parameter pair.

**Parameters**

<i>pt</i>	the result of the evaluation is written here
<i>upar</i>	the first parameter
<i>vpar</i>	the second parameter

Implements [Go::ParamSurface](#).

**29.475.3.29** `void Go::Sphere::point ( std::vector< Point > & pts, double upar, double vpar, int derivs, bool u_from_right = true, bool v_from_right = true, double resolution = 1.0e-12 ) const` [virtual]

Evaluates the surface's position and a certain number of derivatives for a given parameter pair.

#### Parameters

<i>pts</i>	the vector containing the evaluated values. Its size must be set by the user prior to calling this function, and should be equal to $(derivs+1) * (derivs+2) / 2$ . Upon completion of the function, its first entry is the surface's position at the given parameter pair. Then, if 'derivs' > 0, the two next entries will be the surface tangents along the first and second parameter direction. The next three entries are the second- and cross derivatives, in the order (du2, dudv, dv2), and similar for even higher derivatives.
<i>upar</i>	the first parameter
<i>vpar</i>	the second parameter
<i>derivs</i>	number of requested derivatives
<i>u_from_right</i>	specify whether derivatives along the first parameter are to be calculated from the right ('true', default) or from the left ('false')
<i>v_from_right</i>	specify whether derivatives along the second parameter are to be calculated from the right ('true', default) or from the left ('false')
<i>resolution</i>	tolerance used when determining whether parameters are located at special values of the parameter domain (in particular; knot values in case of spline objects).

Implements [Go::ParamSurface](#).

**29.475.3.30** `virtual void Go::Sphere::read ( std::istream & is )` [virtual]

read object from stream

#### Parameters

<i>is</i>	stream from which object is read
-----------	----------------------------------

Implements [Go::Streamable](#).

**29.475.3.31** `void Go::Sphere::setCoordinateAxes ( )` [protected]

**29.475.3.32** `virtual void Go::Sphere::setParameterBounds ( double from_upar, double from_vpar, double to_upar, double to_vpar )` [virtual]

Limit the surface by limiting the parameter domain.

Implements [Go::ElementarySurface](#).

29.475.3.33 **Sphere\*** `Go::Sphere::subSurface ( double from_upar, double from_vpar, double to_upar, double to_vpar, double fuzzy = DEFAULT_PARAMETER_EPSILON ) const`

Pick part of sphere limited by parameter domain information.

29.475.3.34 `std::vector<shared_ptr<ParamSurface> > Go::Sphere::subSurfaces ( double from_upar, double from_vpar, double to_upar, double to_vpar, double fuzzy = DEFAULT_PARAMETER_EPSILON ) const`  
[virtual]

Get the surface(s) obtained by cropping the parameter domain of this surface between given values for the first and second parameter. In general, for surfaces with no interior holes, the result will be *one* surface; however, for surfaces with interior holes, the result might be *several disjoint* surfaces.

#### Parameters

<i>from_upar</i>	lower value for the first parameter in the subdomain
<i>from_vpar</i>	lower value for the second parameter in the subdomain
<i>to_upar</i>	upper value for the first parameter in the subdomain
<i>to_vpar</i>	upper value for the second parameter in the subdomain
<i>fuzzy</i>	tolerance used when determining intersection with interior boundaries

#### Returns

a vector contained shared pointers to the obtained, newly constructed sub-surfaces.

Implements [Go::ParamSurface](#).

29.475.3.35 **DirectionCone** `Go::Sphere::tangentCone ( bool pardir_is_u ) const` [virtual]

Creates a [DirectionCone](#) covering all tangents to this surface along a given parameter direction.

#### Parameters

<i>pardir_is_u</i>	if 'true', then the <a href="#">DirectionCone</a> will be defined on basis of the surface's tangents along the first parameter direction. Otherwise the second parameter direction will be used.
--------------------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

#### Returns

a [DirectionCone](#) (not necessarily the smallest) containing all tangents to this surface along the specified parameter direction.

Implements [Go::ParamSurface](#).

29.475.3.36 `virtual void Go::Sphere::write ( std::ostream & os ) const` [virtual]

write object to stream

## Parameters

<i>os</i>	stream to which object is written
-----------	-----------------------------------

Implements [Go::Streamable](#).

## 29.475.4 Member Data Documentation

### 29.475.4.1 RectDomain Go::Sphere::domain\_ [protected]

Definition at line 242 of file Sphere.h.

### 29.475.4.2 Point Go::Sphere::location\_ [protected]

Definition at line 237 of file Sphere.h.

### 29.475.4.3 RectDomain Go::Sphere::orientedDomain\_ [mutable],[protected]

Definition at line 243 of file Sphere.h.

### 29.475.4.4 double Go::Sphere::radius\_ [protected]

Definition at line 236 of file Sphere.h.

### 29.475.4.5 Point Go::Sphere::x\_axis\_ [protected]

Definition at line 239 of file Sphere.h.

### 29.475.4.6 Point Go::Sphere::y\_axis\_ [protected]

Definition at line 240 of file Sphere.h.

### 29.475.4.7 Point Go::Sphere::z\_axis\_ [protected]

Definition at line 238 of file Sphere.h.

The documentation for this class was generated from the following file:

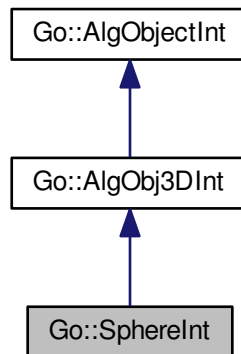
- [gotools-core/include/GoTools/geometry/Sphere.h](#)

## 29.476 Go::SphereInt Class Reference

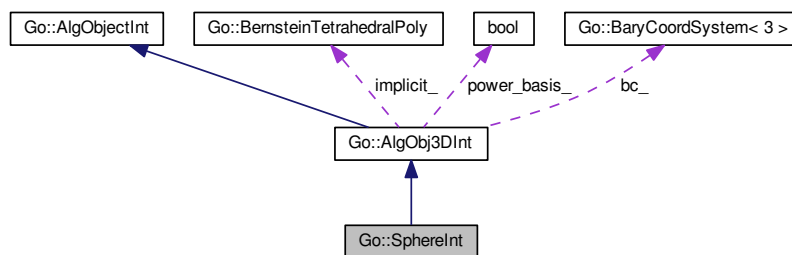
Class representing spherical algebraic intersection objects.

```
#include <SphereInt.h>
```

Inheritance diagram for Go::SphereInt:



Collaboration diagram for Go::SphereInt:



### Public Member Functions

- [SphereInt](#) ()
- [SphereInt](#) ([Point](#) center, [double](#) radius)
- [virtual](#) [~SphereInt](#) ()
- *Destructor.*
- [void](#) [read](#) ([std::istream](#) &is)
- [Point](#) center () [const](#)
- [double](#) radius () [const](#)
- [SplineSurface](#) \* [surface](#) () [const](#)



## Additional Inherited Members

### 29.476.1 Detailed Description

Class representing spherical algebraic intersection objects.

Definition at line 56 of file SphereInt.h.

### 29.476.2 Constructor & Destructor Documentation

#### 29.476.2.1 Go::SphereInt::SphereInt ( )

Constructor. Used when reading from file.

#### 29.476.2.2 Go::SphereInt::SphereInt ( Point center, double radius )

Constructor.

##### Parameters

<i>center</i>	the center point of the sphere.
<i>radius</i>	the radius of the sphere.

#### 29.476.2.3 virtual Go::SphereInt::~~SphereInt ( ) [virtual]

Destructor.

### 29.476.3 Member Function Documentation

#### 29.476.3.1 Point Go::SphereInt::center ( ) const

Get center point of the sphere.

##### Returns

The center point of the sphere.

#### 29.476.3.2 double Go::SphereInt::radius ( ) const

Get the radius of the sphere.

##### Returns

The radius of the sphere.

#### 29.476.3.3 void Go::SphereInt::read ( std::istream & is )

Read a sphere description from file.

**Parameters**

<i>is</i>	the stream containing the sphere description.
-----------	-----------------------------------------------

**29.476.3.4 SplineSurface\* Go::SphereInt::surface ( ) const**

Get a SplineSurface representing the sphere. This can be used to visualize the object.

**Returns**

The spline surface of the sphere.

The documentation for this class was generated from the following file:

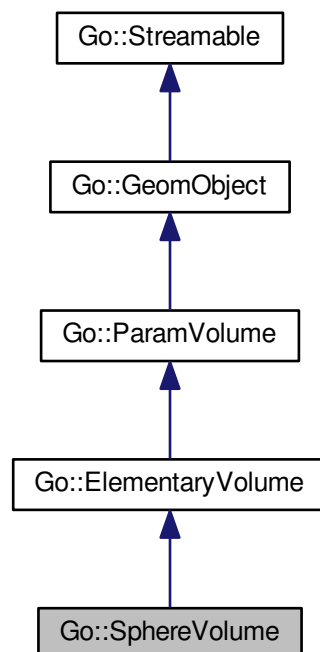
- [intersections/include/GoTools/intersections/SphereInt.h](#)

**29.477 Go::SphereVolume Class Reference**

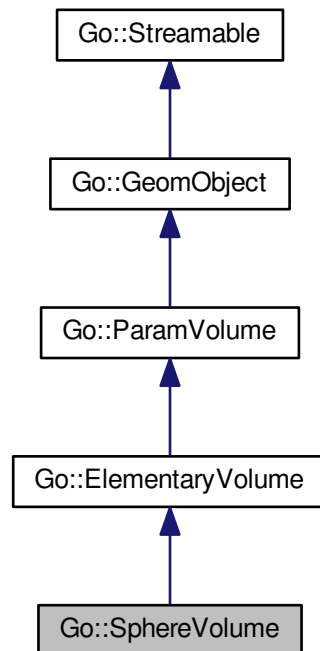
Class that represents a solid sphere. It is a subclass of [ElementaryVolume](#), and thus has a parametrization.

```
#include <SphereVolume.h>
```

Inheritance diagram for Go::SphereVolume:



Collaboration diagram for Go::SphereVolume:



## Public Member Functions

- [SphereVolume](#) ()
- [SphereVolume](#) (double radius, [Point](#) location, [Point](#) z\_axis, [Point](#) x\_axis)
- virtual [~SphereVolume](#) ()
  - Virtual destructor - ensures safe inheritance.*
- virtual void [read](#) (std::istream &is)
- virtual void [write](#) (std::ostream &os) const
- virtual int [dimension](#) () const
  - Return the dimension of the space in which the object lies (usually 2 or 3)*
- virtual [ClassType](#) [instanceType](#) () const
  - Return the class type identifier of a given, derived instance of [GeomObject](#).*
- virtual [BoundingBox](#) [boundingBox](#) () const
  - Return the object's bounding box.*
- virtual [SphereVolume](#) \* [clone](#) () const
- [DirectionCone](#) [tangentCone](#) (int paddir) const
- const [Array](#) < double, 6 > [parameterSpan](#) () const
- void [point](#) ([Point](#) &pt, double upar, double vpar, double wpar) const
- void [point](#) (std::vector< [Point](#) > &pts, double upar, double vpar, double wpar, int derivs, bool u\_from\_↔right=true, bool v\_from\_right=true, bool w\_from\_right=true, double resolution=1.0e-12) const
- double [nextSegmentVal](#) (int dir, double par, bool forward, double tol) const
- void [closestPoint](#) (const [Point](#) &pt, double &clo\_u, double &clo\_v, double &clo\_w, [Point](#) &clo\_pt, double &clo\_↔\_dist, double epsilon, double \*seed=0) const
- void [reverseParameterDirection](#) (int paddir)

- void [swapParameterDirection](#) (int paddir1, int paddir2)
- virtual std::vector< shared\_ptr< [ParamSurface](#) > > [getAllBoundarySurfaces](#) () const  
*Fetch all boundary surfaces corresponding to the volume.*
- virtual void [translate](#) (const [Point](#) &vec)  
*Translate.*
- [SplineVolume](#) \* [geometryVolume](#) () const  
*Make a NURBS representation of the object.*

## Static Public Member Functions

- static [ClassType](#) [classType](#) ()

### 29.477.1 Detailed Description

Class that represents a solid sphere. It is a subclass of [ElementaryVolume](#), and thus has a parametrization.

A [SphereVolume](#) has a natural parametrization by spherical coordinates in terms the distance  $u$  from the centre, the elevation angle  $v$  and the azimuth angle  $w$ :  $\mathbf{p}(u, v, w) = \mathbf{C} + u \cos w(\cos v) \mathbf{x} + \sin(v) \mathbf{y} + u (\sin w) \mathbf{z}$ , where  $\mathbf{C}$  is a position vector, and  $\mathbf{x}$ ,  $\mathbf{y}$  and  $\mathbf{z}$  are the (local) axes. The parametrization is bounded by:  $0 \leq u \leq R$ ,  $0 \leq v \leq 2\pi$ ,  $-\frac{\pi}{2} < w < \frac{\pi}{2}$ , where  $R$  is the radius. The dimension is 3.

Definition at line 67 of file [SphereVolume.h](#).

### 29.477.2 Constructor & Destructor Documentation

#### 29.477.2.1 [Go::SphereVolume::SphereVolume](#) ( ) [inline]

Default constructor. Constructs an uninitialized [SphereVolume](#) which can only be assigned to or read into.

Definition at line 72 of file [SphereVolume.h](#).

#### 29.477.2.2 [Go::SphereVolume::SphereVolume](#) ( double *radius*, [Point](#) *location*, [Point](#) *z\_axis*, [Point](#) *x\_axis* )

Constructor. Input is the radius, the location, the direction of the z-axis and the (possibly approximate) x-axis.

#### 29.477.2.3 virtual [Go::SphereVolume::~~SphereVolume](#) ( ) [virtual]

Virtual destructor - ensures safe inheritance.

### 29.477.3 Member Function Documentation

#### 29.477.3.1 virtual [BoundingBox](#) [Go::SphereVolume::boundingBox](#) ( ) const [virtual]

Return the object's bounding box.

Implements [Go::GeomObject](#).

29.477.3.2 `static ClassType Go::SphereVolume::classType ( ) [inline],[static]`

Definition at line 96 of file SphereVolume.h.

29.477.3.3 `virtual SphereVolume* Go::SphereVolume::clone ( ) const [inline],[virtual]`

make a clone of this volume and return a pointer to it (user is responsible for clearing up memory afterwards).

Returns

pointer to cloned object

Implements [Go::ElementaryVolume](#).

Definition at line 103 of file SphereVolume.h.

29.477.3.4 `void Go::SphereVolume::closestPoint ( const Point & pt, double & clo_u, double & clo_v, double & clo_w, Point & clo_pt, double & clo_dist, double epsilon, double * seed = 0 ) const [virtual]`

Iterates to the closest point to pt on the volume.

Parameters

<i>pt</i>	the point to find the closest point to
<i>clo_u</i>	u parameter of the closest point
<i>clo_v</i>	v parameter of the closest point
<i>clo_w</i>	w parameter of the closest point
<i>clo_pt</i>	the geometric position of the closest point
<i>clo_dist</i>	the distance between pt and clo_pt
<i>epsilon</i>	parameter tolerance (will in any case not be higher than sqrt(machine_precision) x magnitude of solution)
<i>seed</i>	pointer to parameter values where iteration starts.

Implements [Go::ParamVolume](#).

29.477.3.5 `virtual int Go::SphereVolume::dimension ( ) const [virtual]`

Return the dimension of the space in which the object lies (usually 2 or 3)

Implements [Go::GeomObject](#).

29.477.3.6 `SplineVolume* Go::SphereVolume::geometryVolume ( ) const [virtual]`

Make a NURBS representation of the object.

Implements [Go::ElementaryVolume](#).

29.477.3.7 `virtual std::vector<shared_ptr<ParamSurface>> Go::SphereVolume::getAllBoundarySurfaces ( ) const` [virtual]

Fetch all boundary surfaces corresponding to the volume.

Implements [Go::ParamVolume](#).

29.477.3.8 `virtual ClassType Go::SphereVolume::instanceType ( ) const` [virtual]

Return the class type identifier of a given, derived instance of [GeomObject](#).

Implements [Go::GeomObject](#).

29.477.3.9 `double Go::SphereVolume::nextSegmentVal ( int dir, double par, bool forward, double tol ) const` [virtual]

Determine the parameter value of the start of the 'next segment' from a parameter value, along a given parameter direction. A 'segment' is here defined as a parameter interval in which there will be no discontinuities in derivatives or other artifacts. For spline objects, a segment will typically be the interval between two consecutive, non-coincident knots.

#### Parameters

<i>dir</i>	the parameter direction in which we search for the next segment (0, 1 or 2)
<i>par</i>	the parameter value starting from which we search for the start value of the next segment
<i>forward</i>	define whether we shall move forward ('true') or backwards when searching along this parameter
<i>tol</i>	tolerance used for determining whether the 'par' is already located <i>on</i> the next segment value

#### Returns

the value of the start value of the next segment (or the end of the previous segment, if we are moving backwards...)

Implements [Go::ParamVolume](#).

29.477.3.10 `const Array<double,6> Go::SphereVolume::parameterSpan ( ) const` [virtual]

Return the parameter domain of the volume. This is an array containing the start and end parameter values. The spline's parameter *i* has its start value at the array position (2*i*) and its end parameter value at the array position (2*i*+1)

#### Returns

An array describing the parametric domain of the volume

Implements [Go::ParamVolume](#).

29.477.3.11 `void Go::SphereVolume::point ( Point & pt, double upar, double vpar, double wpar ) const` [virtual]

Evaluates the volume's position for a given parameter triple.

## Parameters

<i>pt</i>	the result of the evaluation is written here
<i>upar</i>	the first parameter
<i>vpar</i>	the second parameter
<i>wpar</i>	the third parameter

Implements [Go::ParamVolume](#).

```
29.477.3.12 void Go::SphereVolume::point (std::vector< Point > & pts, double upar, double vpar, double wpar, int
 derivs, bool u_from_right = true, bool v_from_right = true, bool w_from_right = true, double resolution =
 1.0e-12) const [virtual]
```

Evaluates the volume's position and a certain number of derivatives for a given parameter triple.

## Parameters

<i>pts</i>	the vector containing the evaluated values. Its size must be set by the user prior to calling this function. Upon completion of the function, its first entry is the volume's position at the given parameter pair. Then, if 'derivs' > 0, the two next entries will be the volume tangents along the first, second and third parameter direction. The next three entries are the second- and cross derivatives, in the order (du2, dudv, dudw, dv2, dvdw, dw2), and similar for even higher derivatives.
<i>upar</i>	the first parameter
<i>vpar</i>	the second parameter
<i>wpar</i>	the third parameter
<i>derivs</i>	number of requested derivatives
<i>u_from_right</i>	specify whether derivatives along the first parameter are to be calculated from the right ('true', default) or from the left ('false')
<i>v_from_right</i>	specify whether derivatives along the second parameter are to be calculated from the right ('true', default) or from the left ('false')
<i>w_from_right</i>	specify whether derivatives along the third parameter are to be calculated from the right ('true', default) or from the left ('false')
<i>resolution</i>	tolerance used when determining whether parameters are located at special values of the parameter domain (in particular; knot values in case of spline objects).

Implements [Go::ParamVolume](#).

```
29.477.3.13 virtual void Go::SphereVolume::read (std::istream & is) [virtual]
```

read object from stream

## Parameters

<i>is</i>	stream from which object is read
-----------	----------------------------------

Implements [Go::Streamable](#).

29.477.3.14 `void Go::SphereVolume::reverseParameterDirection ( int pardir ) [virtual]`

Reverses the direction of the basis in input direction.

#### Parameters

<i>pardir</i>	which parameter direction to reverse
---------------	--------------------------------------

Implements [Go::ParamVolume](#).

29.477.3.15 `void Go::SphereVolume::swapParameterDirection ( int pardir1, int pardir2 ) [virtual]`

Swaps two parameter directions

#### Parameters

<i>pardir1</i>	One of the parameter directions (0, 1 or 2) to be swapped
<i>pardir2</i>	The other parameter direction (0, 1 or 2) to be swapped

Implements [Go::ParamVolume](#).

29.477.3.16 `DirectionCone Go::SphereVolume::tangentCone ( int pardir ) const [virtual]`

Creates a [DirectionCone](#) covering all tangents to this volume along a given parameter direction.

#### Parameters

<i>pardir</i>	if 1, the <a href="#">DirectionCone</a> will be defined on basis of the volume's tangents along the first parameter direction. If 2, the second parameter direction will be used. If 3, the third parameter direction will be used.
---------------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

#### Returns

a [DirectionCone](#) (not necessarily the smallest) containing all tangents to this volume along the specified parameter direction.

Implements [Go::ParamVolume](#).

29.477.3.17 `virtual void Go::SphereVolume::translate ( const Point & vec ) [virtual]`

Translate.

Implements [Go::ParamVolume](#).

29.477.3.18 `virtual void Go::SphereVolume::write ( std::ostream & os ) const [virtual]`

write object to stream



## Parameters

<i>os</i>	stream to which object is written
-----------	-----------------------------------

Implements [Go::Streamable](#).

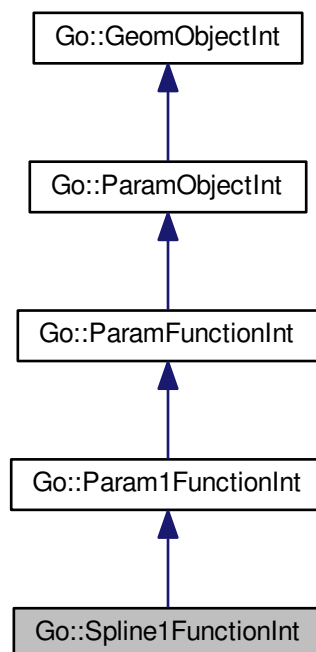
The documentation for this class was generated from the following file:

- [trivariate/include/GoTools/trivariate/SphereVolume.h](#)

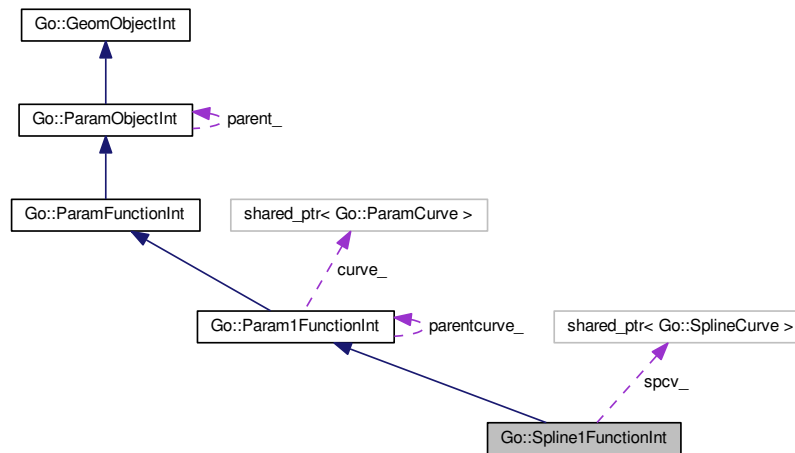
## 29.478 Go::Spline1FunctionInt Class Reference

```
#include <Spline1FunctionInt.h>
```

Inheritance diagram for Go::Spline1FunctionInt:



Collaboration diagram for Go::Spline1FunctionInt:



## Public Member Functions

- [Spline1FunctionInt](#) (shared\_ptr< [ParamCurve](#) > curve)
- [Spline1FunctionInt](#) (shared\_ptr< [ParamCurve](#) > curve, [ParamFunctionInt](#) \*parent)
- virtual [~Spline1FunctionInt](#) ()

*Destructor.*

- virtual shared\_ptr< [Param1FunctionInt](#) > [makeIntFunction](#) (shared\_ptr< [ParamCurve](#) > curve)
- virtual [bool](#) [hasInnerKnots](#) (int paddir) *const*
- virtual [bool](#) [hasCriticalValsOrKnots](#) (int paddir) *const*
- virtual std::vector< [double](#) > [getInnerKnotVals](#) (int paddir, [bool](#) sort=false) *const*
- virtual std::vector< [double](#) > [getCriticalValsAndKnots](#) (int paddir) *const*
- virtual int [getMeshSize](#) (int dir)
- virtual [double](#) [paramFromMesh](#) (int dir, int idx)
- virtual std::vector< [double](#) >::iterator [getMesh](#) ()
- virtual [bool](#) [monotone](#) ([Point](#) &dir, [double](#) tol=1.0e-15) *const*
- virtual int [knotIntervalFuzzy](#) ([double](#) &t, [double](#) tol) *const*
- virtual [double](#) [nextSegmentVal](#) ([double](#) par, [bool](#) forward) *const*

## Protected Attributes

- shared\_ptr< [SplineCurve](#) > [spcv\\_](#)

### 29.478.1 Detailed Description

Class that represents the "intersection object" of a spline curve of dimension 1.

Definition at line 56 of file [Spline1FunctionInt.h](#).

### 29.478.2 Constructor & Destructor Documentation

**29.478.2.1** [Go::Spline1FunctionInt::Spline1FunctionInt](#) ( shared\_ptr< [ParamCurve](#) > curve ) [*explicit*]

Constructor. Input curve must be of type [SplineCurve](#). This is not checked run-time, so we rely on the user to obey this rule.

## Parameters

<i>curve</i>	the parametric 1-dimensional curve defining the object.
--------------	---------------------------------------------------------

**29.478.2.2** `Go::Spline1FunctionInt::Spline1FunctionInt ( shared_ptr< ParamCurve > curve, ParamFunctionInt * parent ) [explicit]`

Constructor. Input curve must be of type [SplineCurve](#). This is not checked run-time, so we rely on the user to obey this rule.

## Parameters

<i>curve</i>	the parametric 1-dimensional curve defining the object. Can be either a curve or a surface.
<i>parent</i>	the parent object to this object.

**29.478.2.3** `virtual Go::Spline1FunctionInt::~~Spline1FunctionInt ( ) [inline],[virtual]`

Destructor.

Definition at line 75 of file Spline1FunctionInt.h.

**29.478.3 Member Function Documentation**

**29.478.3.1** `virtual std::vector<double> Go::Spline1FunctionInt::getCriticalValsAndKnots ( int pdir ) const [virtual]`

Return the critical parameter values and inner knots for object.

## Parameters

<i>pdir</i>	the parameter direction in question. Indexing starts at 0.
-------------	------------------------------------------------------------

Reimplemented from [Go::Param1FunctionInt](#).

**29.478.3.2** `virtual std::vector<double> Go::Spline1FunctionInt::getInnerKnotVals ( int pdir, bool sort = false ) const [virtual]`

Return the inner knot values in the specified direction.

## Parameters

<i>pdir</i>	the parameter direction in question. Indexing starts at 0.
<i>sort</i>	the returned values may be sorted by the function.

Reimplemented from [Go::Param1FunctionInt](#).

29.478.3.3 `virtual std::vector<double>::iterator Go::Spline1FunctionInt::getMesh ( ) [virtual]`

Return the geometric sample mesh for the spline function.

#### Returns

The geometric sample mesh for the spline function.

Reimplemented from [Go::Param1FunctionInt](#).

29.478.3.4 `virtual int Go::Spline1FunctionInt::getMeshSize ( int dir ) [virtual]`

Return the size of the geometric sample mesh in the specified direction.

#### Parameters

<i>dir</i>	the parameter direction in question. Indexing starts at 0.
------------	------------------------------------------------------------

Reimplemented from [Go::Param1FunctionInt](#).

29.478.3.5 `virtual bool Go::Spline1FunctionInt::hasCriticalValsOrKnots ( int pardir ) const [virtual]`

Return true if the object has any critical parameter values or inner knots.

#### Parameters

<i>pardir</i>	the parameter direction in question. Indexing starts at 0.
---------------	------------------------------------------------------------

Reimplemented from [Go::Param1FunctionInt](#).

29.478.3.6 `virtual bool Go::Spline1FunctionInt::hasInnerKnots ( int pardir ) const [virtual]`

Return true if the object has any inner knots in the specified parameter direction.

#### Parameters

<i>pardir</i>	the parameter direction in question. Indexing starts at 0.
---------------	------------------------------------------------------------

Reimplemented from [Go::Param1FunctionInt](#).

29.478.3.7 `virtual int Go::Spline1FunctionInt::knotIntervalFuzzy ( double & t, double tol ) const [virtual]`

Return the knot interval for which *t* lies inside, moving the value *t* if it lies close to a knot.

## Parameters

<i>t</i>	the parameter value
<i>tol</i>	the parametric tolerance deciding if the input parameter <i>t</i> should be moved.

Reimplemented from [Go::Param1FunctionInt](#).

29.478.3.8 `virtual shared_ptr<Param1FunctionInt> Go::Spline1FunctionInt::makeIntFunction ( shared_ptr<ParamCurve > curve ) [virtual]`

Return an intersection object for the input curve. Input curve must be of type [SplineCurve](#). This is not checked run-time, so we rely on the user to obey this rule. This object is used as the parent for the intersection object.

## Parameters

<i>curve</i>	the parametric curve defining the intersection object.
--------------	--------------------------------------------------------

Reimplemented from [Go::Param1FunctionInt](#).

29.478.3.9 `virtual bool Go::Spline1FunctionInt::monotone ( Point & dir, double tol = 1.0e-15 ) const [virtual]`

Return true if the curve is monotone.

## Parameters

<i>dir</i>	the direction in which the object is monotone. Is not of interest here as the curve has only 1 parameter direction.
------------	---------------------------------------------------------------------------------------------------------------------

## Returns

Whether or not the curve is monotone.

Reimplemented from [Go::Param1FunctionInt](#).

29.478.3.10 `virtual double Go::Spline1FunctionInt::nextSegmentVal ( double par, bool forward ) const [virtual]`

Return the value of the knot next to the input parameter *par*.

## Parameters

<i>par</i>	the parameter value
<i>forward</i>	if true we return the closest knot to the right, otherwise the closest knot to the left.

## Returns

The knot closest to the input parameter.

Reimplemented from [Go::Param1FunctionInt](#).

**29.478.3.11** `virtual double Go::Spline1FunctionInt::paramFromMesh ( int dir, int idx )` [virtual]

Return the corresponding mesh parameter.

#### Parameters

<i>dir</i>	the parameter direction in question. Indexing starts at 0.
<i>idx</i>	the mesh idx in the specified direction. Indexing starts at 0.

Reimplemented from [Go::Param1FunctionInt](#).

## 29.478.4 Member Data Documentation

**29.478.4.1** `shared_ptr<SplineCurve> Go::Spline1FunctionInt::spcv_` [protected]

Definition at line 157 of file Spline1FunctionInt.h.

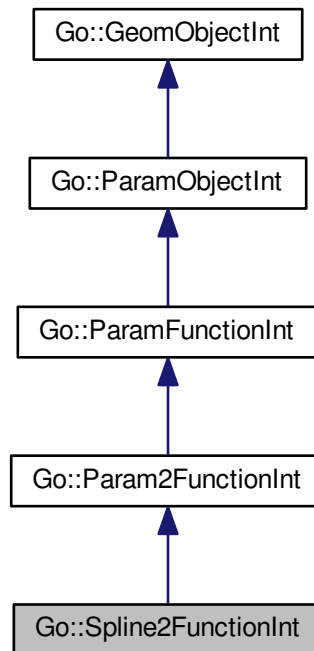
The documentation for this class was generated from the following file:

- intersections/include/GoTools/intersections/[Spline1FunctionInt.h](#)

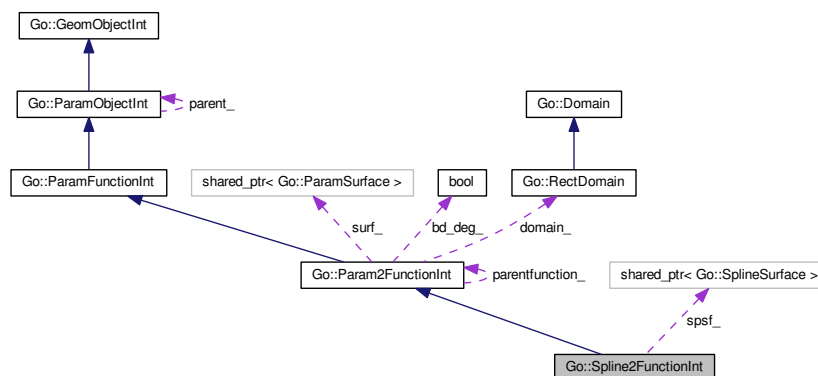
## 29.479 Go::Spline2FunctionInt Class Reference

```
#include <Spline2FunctionInt.h>
```

Inheritance diagram for Go::Spline2FunctionInt:



Collaboration diagram for Go::Spline2FunctionInt:



## Public Member Functions

- `Spline2FunctionInt` (`shared_ptr< SplineSurface > surf`)
- `Spline2FunctionInt` (`shared_ptr< SplineSurface > surf, Param2FunctionInt *parent`)
- `virtual ~Spline2FunctionInt` ()

*Destructor.*

- virtual `shared_ptr< Param2FunctionInt > makeIntFunction` (`shared_ptr< ParamSurface > surf`)
- virtual `bool hasInnerKnots` (`int pdir`) `const`
- virtual `bool hasCriticalValsOrKnots` (`int pdir`) `const`
- virtual `std::vector< double > getInnerKnotVals` (`int pdir`, `bool sort=false`) `const`
- virtual `std::vector< double > getCriticalValsAndKnots` (`int pdir`) `const`
- virtual `double startParam` (`int pdir`) `const`
- virtual `double endParam` (`int pdir`) `const`
- virtual `bool monotone` (`Point &dir`, `double tol=1.0e-15`) `const`
- virtual `int knotIntervalFuzzy` (`int pdir`, `double &t`, `double tol`) `const`
- virtual `double nextSegmentVal` (`int pdir`, `double par`, `bool forward`) `const`
- virtual `void getBoundaryObjects` (`std::vector< shared_ptr< BoundaryFunctionInt > > &bd_objs`)
- `shared_ptr< SplineSurface > surface3D` ()
- `shared_ptr< SplineSurface > createGradSurface` () `const`

### Protected Attributes

- `shared_ptr< SplineSurface > spsf_`

### Additional Inherited Members

#### 29.479.1 Detailed Description

This class represents the "intersection object" of a spline surface of dimension 1.

Definition at line 53 of file Spline2FunctionInt.h.

#### 29.479.2 Constructor & Destructor Documentation

29.479.2.1 `Go::Spline2FunctionInt::Spline2FunctionInt ( shared_ptr< SplineSurface > surf ) [explicit]`

Constructor.

##### Parameters

<i>surf</i>	the parametric 1-dimensional surface defining the object.
-------------	-----------------------------------------------------------

29.479.2.2 `Go::Spline2FunctionInt::Spline2FunctionInt ( shared_ptr< SplineSurface > surf, Param2FunctionInt * parent ) [explicit]`

Constructor.

##### Parameters

<i>surf</i>	the parametric 1-dimensional surface defining the object.
<i>parent</i>	the parent object to this object.



29.479.2.3 `virtual Go::Spline2FunctionInt::~~Spline2FunctionInt ( ) [inline],[virtual]`

Destructor.

Definition at line 68 of file Spline2FunctionInt.h.

### 29.479.3 Member Function Documentation

29.479.3.1 `shared_ptr<SplineSurface> Go::Spline2FunctionInt::createGradSurface ( ) const`

Return the gradient of the function S as a two-dimensional spline surface: (dS/du, dS/dv).

#### Returns

Pointer to the gradient spline surface

29.479.3.2 `virtual double Go::Spline2FunctionInt::endParam ( int parDir ) const [virtual]`

Return the end parameter in the specified direction.

#### Parameters

<i>parDir</i>	the parameter direction in question. Indexing starts at 0.
---------------	------------------------------------------------------------

#### Returns

The end parameter in the specified direction.

Reimplemented from [Go::Param2FunctionInt](#).

29.479.3.3 `virtual void Go::Spline2FunctionInt::getBoundaryObjects ( std::vector< shared_ptr< BoundaryFunctionInt > & bd_objs ) [virtual]`

Return the boundary objects of this object.

#### Parameters

<i>bd_objs</i>	the boundary objects of this object.
----------------	--------------------------------------

Reimplemented from [Go::Param2FunctionInt](#).

29.479.3.4 `virtual std::vector<double> Go::Spline2FunctionInt::getCriticalValsAndKnots ( int parDir ) const [virtual]`

Return the critical parameter values and inner knots for object.

## Parameters

<i>pardir</i>	the parameter direction in question. Indexing starts at 0.
---------------	------------------------------------------------------------

Reimplemented from [Go::Param2FunctionInt](#).

29.479.3.5 `virtual std::vector<double> Go::Spline2FunctionInt::getInnerKnotVals ( int pardir, bool sort = false ) const` [virtual]

Return the inner knot values in the specified direction.

## Parameters

<i>pardir</i>	the parameter direction in question. Indexing starts at 0.
<i>sort</i>	the returned values may be sorted by the function.

Reimplemented from [Go::Param2FunctionInt](#).

29.479.3.6 `virtual bool Go::Spline2FunctionInt::hasCriticalValsOrKnots ( int pardir ) const` [virtual]

Return true if the object has any critical parameter values or inner knots in the specified parameter direction.

## Parameters

<i>pardir</i>	the parameter direction in question. Indexing starts at 0.
---------------	------------------------------------------------------------

Reimplemented from [Go::Param2FunctionInt](#).

29.479.3.7 `virtual bool Go::Spline2FunctionInt::hasInnerKnots ( int pardir ) const` [virtual]

Return true if the object has any inner knots in the specified parameter direction.

## Parameters

<i>pardir</i>	the parameter direction in question. Indexing starts at 0.
---------------	------------------------------------------------------------

Reimplemented from [Go::Param2FunctionInt](#).

29.479.3.8 `virtual int Go::Spline2FunctionInt::knotIntervalFuzzy ( int pardir, double & t, double tol ) const` [virtual]

Return the knot interval for which *t* lies inside, moving the value *t* if it lies close to a knot.

## Parameters

<i>pardir</i>	the parameter direction in question. Indexing starts at 0.
<i>t</i>	the parameter value
<i>tol</i>	the parametric tolerance deciding if the input parameter <i>t</i> should be moved.

Reimplemented from [Go::Param2FunctionInt](#).

29.479.3.9 `virtual shared_ptr<Param2FunctionInt> Go::Spline2FunctionInt::makeIntFunction ( shared_ptr<ParamSurface > surf ) [virtual]`

Return an intersection object for the input surface. This object is used as the parent for the intersection object.

#### Parameters

<i>surf</i>	the parametric surface defining the intersection object.
-------------	----------------------------------------------------------

Reimplemented from [Go::Param2FunctionInt](#).

29.479.3.10 `virtual bool Go::Spline2FunctionInt::monotone ( Point & dir, double tol = 1.0e-15 ) const [virtual]`

Return true if the surface is monotone in any direction.

#### Parameters

<i>dir</i>	the direction in which the surface is monotone. Pertains only if surface is monotone.
------------	---------------------------------------------------------------------------------------

#### Returns

Whether or not the surface is monotone.

Reimplemented from [Go::Param2FunctionInt](#).

29.479.3.11 `virtual double Go::Spline2FunctionInt::nextSegmentVal ( int paddir, double par, bool forward ) const [virtual]`

Return the value of the knot next to the input parameter par.

#### Parameters

<i>paddir</i>	the parameter direction in question. Indexing starts at 0.
<i>par</i>	the parameter value
<i>forward</i>	if true we return the closest knot to the right, otherwise the closest knot to the left.

#### Returns

The knot closest to the input parameter.

Reimplemented from [Go::Param2FunctionInt](#).

29.479.3.12 `virtual double Go::Spline2FunctionInt::startParam ( int paddir ) const [virtual]`

Return the start parameter value in the specified direction.

## Parameters

<i>parDir</i>	the parameter direction in question. Indexing starts at 0.
---------------	------------------------------------------------------------

## Returns

The start parameter in the specified direction.

Reimplemented from [Go::Param2FunctionInt](#).

29.479.3.13 `shared_ptr<SplineSurface> Go::Spline2FunctionInt::surface3D ( )`

Return a 3-dimensional visualization spline surface.

## Returns

The pointer to the 3-dimensional spline surface (u, v, surf\_(u, v)).

## 29.479.4 Member Data Documentation

29.479.4.1 `shared_ptr<SplineSurface> Go::Spline2FunctionInt::spsf_ [protected]`

Definition at line 168 of file Spline2FunctionInt.h.

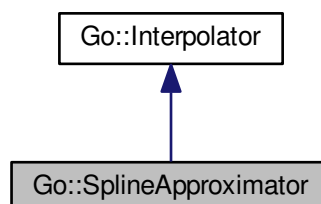
The documentation for this class was generated from the following file:

- intersections/include/GoTools/intersections/[Spline2FunctionInt.h](#)

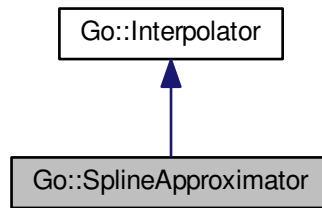
## 29.480 Go::SplineApproximator Class Reference

```
#include <SplineApproximator.h>
```

Inheritance diagram for Go::SplineApproximator:



Collaboration diagram for Go::SplineApproximator:



## Public Member Functions

- [SplineApproximator](#) ()  
*Constructor takes no arguments.*
- virtual [~SplineApproximator](#) ()  
*Virtual destructor ensures safe inheritance.*
- virtual [const BsplineBasis & basis](#) ()
- virtual void [interpolate](#) (int num\_points, int dimension, [const double](#) \*param\_start, [const double](#) \*data\_start, [std::vector< double > &coefs](#))
- void [setNumCoefs](#) (int num)
- void [setSplineSpace](#) ([const BsplineBasis &basis](#))

### 29.480.1 Detailed Description

An [Interpolator](#) that generates a spline curve approximating the given dataset in the least squares sense.

Definition at line 54 of file `SplineApproximator.h`.

### 29.480.2 Constructor & Destructor Documentation

#### 29.480.2.1 `Go::SplineApproximator::SplineApproximator ( )` [`inline`]

Constructor takes no arguments.

Definition at line 58 of file `SplineApproximator.h`.

#### 29.480.2.2 `virtual Go::SplineApproximator::~~SplineApproximator ( )` [`virtual`]

Virtual destructor ensures safe inheritance.

### 29.480.3 Member Function Documentation

29.480.3.1 `virtual const BsplineBasis& Go::SplineApproximator::basis ( )` [virtual]

after the function [interpolate\(\)](#) has been successfully run, this function can be called to get the [BsplineBasis](#) of the generated curve.

#### Returns

a constant reference to the [BsplineBasis](#) of the curve previously generated by [interpolate\(\)](#).

Implements [Go::Interpolator](#).

29.480.3.2 `virtual void Go::SplineApproximator::interpolate ( int num_points, int dimension, const double * param_start, const double * data_start, std::vector< double > & coefs )` [virtual]

The interpolating function, as inherited by [Interpolator](#). Prior to calling this function, the user must have specified:

- *either* the number of control points to use in the approximating curve by [setNumCoefs\(\)](#).
- *or/and* directly specified the [BsplineBasis](#) to use by [setSplineSpace\(\)](#). A default [BsplineBasis](#) will be generated if the user has only set the number of control points prior to calling [interpolate\(\)](#). For parameter list, see [Interpolator](#).

Implements [Go::Interpolator](#).

29.480.3.3 `void Go::SplineApproximator::setNumCoefs ( int num )` [inline]

Specify the number of basis functions / control points to use in the approximating curve.

Definition at line 86 of file [SplineApproximator.h](#).

29.480.3.4 `void Go::SplineApproximator::setSplineSpace ( const BsplineBasis & basis )` [inline]

Directly specify the spline space in which to search for the approximating function.

Definition at line 92 of file [SplineApproximator.h](#).

The documentation for this class was generated from the following file:

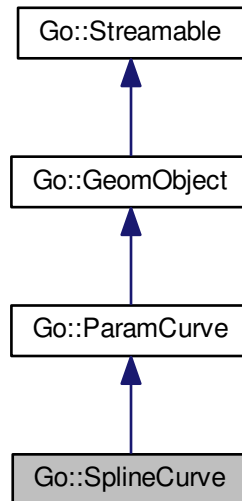
- [gotools-core/include/GoTools/geometry/SplineApproximator.h](#)

## 29.481 Go::SplineCurve Class Reference

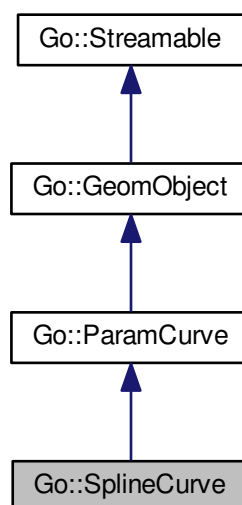
[SplineCurve](#) provides methods for storing, reading and manipulating rational and non-rational B-spline curves.

```
#include <SplineCurve.h>
```

Inheritance diagram for Go::SplineCurve:



Collaboration diagram for Go::SplineCurve:



## Public Member Functions

- [SplineCurve](#) ()
- `template<typename RandomIterator1 , typename RandomIterator2 >`  
[SplineCurve](#) (int number, int order, RandomIterator1 knotstart, RandomIterator2 coefsstart, int dim, bool rational=false)
- `template<typename RandomIterator >`  
[SplineCurve](#) (const [BsplineBasis](#) &basis, RandomIterator coefsstart, int dim, bool rational=false)
- [SplineCurve](#) (const [Point](#) &pnt1, const [Point](#) &pnt2)
- [SplineCurve](#) (const [Point](#) &pnt1, double startpar, const [Point](#) &pnt2, double endpar)
- virtual [~SplineCurve](#) ()  
*Virtual destructor, enables safe inheritance.*
- virtual void [read](#) (std::istream &is)
- virtual void [write](#) (std::ostream &os) const
- virtual [BoundingBox](#) boundingBox () const  
*Return the object's bounding box.*
- virtual int [dimension](#) () const  
*Return the dimension of the space in which the object lies (usually 2 or 3)*
- virtual [ClassType](#) instanceType () const  
*Return the class type identifier of a given, derived instance of [GeomObject](#).*
- virtual [SplineCurve](#) \* [clone](#) () const
- virtual void [point](#) ([Point](#) &pt, double tpar) const
- virtual void [point](#) (std::vector< [Point](#) > &pts, double tpar, int derivs, bool from\_right=true) const
- virtual double [startparam](#) () const
- virtual double [endparam](#) () const
- virtual void [reverseParameterDirection](#) (bool switchparam=false)
- virtual [SplineCurve](#) \* [geometryCurve](#) ()
- virtual [DirectionCone](#) directionCone () const
- virtual [CompositeBox](#) compositeBox () const
- virtual bool [isDegenerate](#) (double degenerate\_epsilon)
- [SplineCurve](#) \* [derivCurve](#) (int ider) const
- virtual [SplineCurve](#) \* [subCurve](#) (double from\_par, double to\_par, double fuzzy=DEFAULT\_PARAMETER\_EPSILON) const
- std::vector< shared\_ptr< [SplineCurve](#) > > [split](#) (std::vector< double > &param, double fuzzy=DEFAULT\_PARAMETER\_EPSILON) const  
*Split curve in specified parameter values.*
- virtual std::vector< shared\_ptr< [ParamCurve](#) > > [split](#) (double param, double fuzzy=DEFAULT\_PARAMETER\_EPSILON) const  
*Split curve in a specified parameter value.*
- virtual void [closestPoint](#) (const [Point](#) &pt, double tmin, double tmax, double &clo\_t, [Point](#) &clo\_pt, double &clo\_dist, double const \*seed=0) const
- virtual double [length](#) (double tol)
- virtual void [appendCurve](#) ([ParamCurve](#) \*other\_curve, int continuity, double &dist, bool repair=true)
- virtual void [appendCurve](#) ([ParamCurve](#) \*cv, bool repair=true)
- void [computeBasis](#) (double param, std::vector< double > &basisValues, std::vector< double > &basisDerivs) const
- void [gridEvaluator](#) (std::vector< double > &points, const std::vector< double > &par) const
- const [BsplineBasis](#) & [basis](#) () const
- [BsplineBasis](#) & [basis](#) ()
- int [numCoefs](#) () const
- int [numElem](#) () const  
*Query the number of elements in the [SplineCurve](#).*
- int [order](#) () const
- bool [rational](#) () const



- `std::vector< double >::iterator knotsBegin ()`
- `std::vector< double >::iterator knotsEnd ()`
- `std::vector< double >::const_iterator knotsBegin () const`
- `std::vector< double >::const_iterator knotsEnd () const`
- `std::vector< double >::iterator coefs_begin ()`
- `std::vector< double >::iterator coefs_end ()`
- `std::vector< double >::const_iterator coefs_begin () const`
- `std::vector< double >::const_iterator coefs_end () const`
- `std::vector< double >::iterator rcoefs_begin ()`
- `std::vector< double >::iterator rcoefs_end ()`
- `std::vector< double >::const_iterator rcoefs_begin () const`
- `std::vector< double >::const_iterator rcoefs_end () const`
- `void getWeights (std::vector< double > &weights) const`
- `void interpolate (Interpolator &interpolator, int num_points, int dim, const double *param_start, const double *data_start)`
- `void insertKnot (double apar)`
- `void insertKnot (const std::vector< double > &new_knots)`
- `void makeBernsteinKnots ()`
- `virtual void setParameterInterval (double t1, double t2)`
- `void removeKnot (double tpar)`
- `void raiseOrder (int i=1)`
- `void makeKnotStartRegular ()`
- `void makeKnotEndRegular ()`
- `void swap (SplineCurve &other)`
- `void deform (const std::vector< double > &vec, int vdim=0)`  
*Adds the given deformation vector to the coefficients.*
- `void equalBdWeights (bool at_start)`
- `void representAsRational ()`  
*Ensure that the current curve is represented as a rational curve.*
- `void setBdWeight (double wgt, bool at_start)`  
*Set the weight in one endpoint of this curve (if rational)*
- `virtual double nextSegmentVal (double par, bool forward, double tol) const`
- `void replaceEndPoint (Point pnt, bool at_start)`
- `void translateCurve (const Point &dir)`
- `void translateSwapCurve (const Point &dir, double sgn, int pdir)`
- `bool isElementaryCurve ()`  
*Query if the curve was generated from an [ElementaryCurve](#).*
- `shared_ptr< ElementaryCurve > getElementaryCurve ()`
- `void setElementaryCurve (shared_ptr< ElementaryCurve > elcurve)`
- `bool checkElementaryCurve ()`
- `void updateCoefsFromRcoefs ()`
- `virtual bool isAxisRotational (Point &centre, Point &axis, Point &vec, double &angle)`
- `virtual bool isLinear (Point &dir, double tol)`  
*Check if the curve is linear.*
- `virtual bool isInPlane (const Point &loc, const Point &axis, double eps, Point &normal) const`  
*Check if the curve lies in a plane passing through a given axis.*
- `virtual bool isInPlane (const Point &norm, double eps, Point &pos) const`  
*Check if the curve lies in a plane with a given normal.*

## Static Public Member Functions

- `static ClassType classType ()`

## Additional Inherited Members

### 29.481.1 Detailed Description

[SplineCurve](#) provides methods for storing, reading and manipulating rational and non-rational B-spline curves.

Definition at line 57 of file [SplineCurve.h](#).

### 29.481.2 Constructor & Destructor Documentation

#### 29.481.2.1 `Go::SplineCurve::SplineCurve( )` `[inline]`

Creates an uninitialized [SplineCurve](#), which can only be assigned to or [read\(\)](#) into.

Definition at line 63 of file [SplineCurve.h](#).

#### 29.481.2.2 `template<typename RandomIterator1 , typename RandomIterator2 > Go::SplineCurve::SplineCurve ( int number, int order, RandomIterator1 knotstart, RandomIterator2 coefsstart, int dim, bool rational = false )` `[inline]`

Create a [SplineCurve](#) by explicitly providing all spline-related information.

##### Parameters

<i>number</i>	number of control points
<i>order</i>	order of b-spline basis
<i>knotstart</i>	pointer to the array describing the knotvector
<i>coefsstart</i>	pointer to the array where the control points are consecutively stored.
<i>dim</i>	dimension of the space in which the curve lies (usually 2 or 3).
<i>rational</i>	Specify whether the curve is rational or not. If the curve is rational, coefficients must be in the following format: wP1 wP2 .... wPdim w. I.e., a (dim+1)-dimensional form. (This is the same form that is used within SISL).

Definition at line 83 of file [SplineCurve.h](#).

#### 29.481.2.3 `template<typename RandomIterator > Go::SplineCurve::SplineCurve ( const BsplineBasis & basis, RandomIterator coefsstart, int dim, bool rational = false )` `[inline]`

Create a [SplineCurve](#) by explicitly providing all spline-related information.

##### Parameters

<i>basis</i>	the <a href="#">BsplineBasis</a> to be used along the u-direction
<i>basis_v</i>	the <a href="#">BsplineBasis</a> to be used along the v-direction
<i>coefsstart</i>	pointer to the array where the control points are consecutively stored. The storage order is such that control points along the u-parameter have the shortest stride (stored right after each other). If the surface is rational, pay attention to the comments below.
<i>dim</i>	dimension of the space in which the surface lies (usually 3).

## Parameters

<i>rational</i>	Specify whether the surface is rational or not. If the surface is rational, coefficients must be in the following format: wP1 wP2 .... wPdim w. I.e. a (dim+1)-dimensional form. (This is the same form that is used within SISL).
-----------------	------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Definition at line 123 of file SplineCurve.h.

#### 29.481.2.4 Go::SplineCurve::SplineCurve ( const Point & *pt1*, const Point & *pt2* )

Construct a straight curve between two points (order 2 - linear). The parameter span will be from 0 to T, where T is the euclidean distance between the two points.

## Parameters

<i>pt1</i>	first of the two points defining the straight curve
<i>pt2</i>	second of the two points defining the straight curve

#### 29.481.2.5 Go::SplineCurve::SplineCurve ( const Point & *pt1*, double *startpar*, const Point & *pt2*, double *endpar* )

Make a straight curve between two points (order 2 - linear). The parameter span is also given by the user.

## Parameters

<i>pt1</i>	first of the two points defining the straight curve
<i>startpar</i>	start parameter of curve
<i>pt2</i>	second of the two points defining the straight curve
<i>endpar</i>	end parameter of curve

#### 29.481.2.6 virtual Go::SplineCurve::~~SplineCurve ( ) [virtual]

Virtual destructor, enables safe inheritance.

### 29.481.3 Member Function Documentation

#### 29.481.3.1 virtual void Go::SplineCurve::appendCurve ( ParamCurve \* *other\_curve*, int *continuity*, double & *dist*, bool *repar = true* ) [virtual]

Inherited from [ParamCurve](#). Appends a curve to end of this curve. The knotvector and control point vector of 'this' curve will be extended, and the degree will be raised if necessary. Note that there *will* be side-effects on the 'other↔\_curve' (its order might be raised, its knotvector will become k-regular and evt. reparametrized, its start point will be moved to coincide with the end point of 'this' curve, etc.)

## Parameters

<i>other_curve</i>	the curve to append to this curve
<i>continuity</i>	which level of continuity we demand at the transition between the two curves (can be from -1 to <a href="#">order()</a> ), but the higher the value the more the curves will have to be locally modified.
<i>dist</i>	upon function return, this variable will hold the estimated maximum distortion after 'smoothing' of joined curve to achieve the desired continuity.
<i>repar</i>	The parametrization of the 'other_curve' will <i>always</i> be shifted so that it starts where the parametrization of 'this' curve ends. However, if 'repar' is set to 'true', it will also be <i>scaled</i> as a function of position of control points close to the transition.

Implements [Go::ParamCurve](#).

29.481.3.2 `virtual void Go::SplineCurve::appendCurve ( ParamCurve * cv, bool repar = true ) [virtual]`

Inherited from [ParamCurve](#). Short hand function to call [appendCurve](#) with C<sup>1</sup> continuity.

## Parameters

<i>cv</i>	the curve to append to this curve
<i>repar</i>	The parametrization of the 'other_curve' will <i>always</i> be shifted so that it starts where the parametrization of 'this' curve ends. However, if 'repar' is set to 'true', it will also be <i>scaled</i> as a function of position of control points close to the transition.

Implements [Go::ParamCurve](#).

29.481.3.3 `const BsplineBasis& Go::SplineCurve::basis ( ) const [inline]`

Get a const reference to the [BsplineBasis](#) of the curve

## Returns

const reference to the curve's [BsplineBasis](#).

Definition at line 325 of file SplineCurve.h.

29.481.3.4 `BsplineBasis& Go::SplineCurve::basis ( ) [inline]`

Get a reference to the [BsplineBasis](#) of the curve

## Returns

reference to the curve's [BsplineBasis](#).

Definition at line 330 of file SplineCurve.h.

29.481.3.5 virtual **BoundingBox** Go::SplineCurve::boundingBox ( ) const [virtual]

Return the object's bounding box.

Implements [Go::GeomObject](#).

29.481.3.6 **bool** Go::SplineCurve::checkElementaryCurve ( )

Check to see if `this` corresponds to the [ElementaryCurve](#) set by `setElementaryCurve()`. NOTE: Not yet implemented!

29.481.3.7 **static ClassType** Go::SplineCurve::classType ( ) [inline],[static]

Definition at line 183 of file SplineCurve.h.

29.481.3.8 virtual **SplineCurve\*** Go::SplineCurve::clone ( ) const [inline],[virtual]

The clone-function is herited from [GeomObject](#), but overridden here to get a covariant return type (for those compilers that allow this).

Implements [Go::ParamCurve](#).

Definition at line 190 of file SplineCurve.h.

29.481.3.9 virtual void Go::SplineCurve::closestPoint ( **const Point & pt, double tmin, double tmax, double & clo\_t, Point & clo\_pt, double & clo\_dist, double const \* seed = 0** ) const [virtual]

Compute the closest point from an interval of this curve to a specified point.

#### Parameters

<i>pt</i>	point we want to find the closest point to
<i>tmin</i>	start parameter of search interval
<i>tmax</i>	end parameter of search interval
<i>clo_t</i>	upon function return, 'clo_t' will contain the parameter value of the closest point found.
<i>clo_pt</i>	upon function return, 'clo_pt' will contain the position of the closest point found.
<i>clo_dist</i>	upon function return, 'clo_dist' will contain the distance between 'pt' and the closest point found.
<i>seed</i>	pointer to initial guess value, provided by the user (can be 0, for which the algorithm will determine a (hopefully) reasonable choice).

Implements [Go::ParamCurve](#).

29.481.3.10 **std::vector<double>::iterator** Go::SplineCurve::coefs\_begin ( ) [inline]

Get an iterator to the start of the curve's internal, non-rational control point array

**Returns**

an iterator to the start of the curves non-rational control point array

Definition at line 376 of file SplineCurve.h.

**29.481.3.11** `std::vector<double>::const_iterator Go::SplineCurve::coefs_begin ( ) const` `[inline]`

Get a const iterator to the start of the curve's non-rational, internal control point array

**Returns**

a const iterator to the start of the curve's non-rational control point array.

Definition at line 388 of file SplineCurve.h.

**29.481.3.12** `std::vector<double>::iterator Go::SplineCurve::coefs_end ( )` `[inline]`

Get a one-past-end iterator to the curve's non-rational, internal control point array

**Returns**

an iterator to one-past-end of the curve's non-rational, internal control point array

Definition at line 382 of file SplineCurve.h.

**29.481.3.13** `std::vector<double>::const_iterator Go::SplineCurve::coefs_end ( ) const` `[inline]`

Get a one-past-end const iterator to the curve's non-rational, internal control point array

**Returns**

a const iterator to one-past-end of the curve's non-rational, internal control point array

Definition at line 394 of file SplineCurve.h.

**29.481.3.14** `virtual CompositeBox Go::SplineCurve::compositeBox ( ) const` `[virtual]`

Creates a composite box enclosing the curve. The composite box consists of an inner and an edge box. The inner box is supposed to be made from the interior of the curve, while the edge box is made from the start and end points. The default implementation simply makes both boxes identical to the regular bounding box.

**Returns**

the [CompositeBox](#) enclosing the curve.

Reimplemented from [Go::ParamCurve](#).

**29.481.3.15** `void Go::SplineCurve::computeBasis ( double param, std::vector< double > & basisValues, std::vector< double > & basisDerivs ) const`

Evaluate positions and first derivatives of all basis values in a given parameter For non-rationals this is an interface to [BsplineBasis::computeBasisValues](#), for rationals the routine evaluates the rational basis functions, i.e. the basis functions are divided by the denominator of the curve

## Parameters

<i>param</i>	the parameter in which to compute
<i>basisValues</i>	the value of all basis functions, size equal to (degree_u+1)
<i>basisDerivs</i>	the derivative of all basis functions, same size as previous

29.481.3.16 `void Go::SplineCurve::deform ( const std::vector< double > & vec, int vdim = 0 )`

Adds the given deformation vector to the coefficients.

29.481.3.17 `SplineCurve* Go::SplineCurve::derivCurve ( int ider ) const`

Make a curve expressing the i'th derivative of 'this' curve, and return a pointer to it.

## Parameters

<i>ider</i>	the number of the derivative of which we want to make a curve
-------------	---------------------------------------------------------------

## Returns

a pointer to the newly generated curve. User assumes ownership and is responsible for deletion.

29.481.3.18 `virtual int Go::SplineCurve::dimension ( ) const` [virtual]

Return the dimension of the space in which the object lies (usually 2 or 3)

Implements [Go::GeomObject](#).

29.481.3.19 `virtual DirectionCone Go::SplineCurve::directionCone ( ) const` [virtual]

Creates a [DirectionCone](#) which covers all tangent directions of this curve.

## Returns

the smallest [DirectionCone](#) containing all tangent directions of this curve.

Implements [Go::ParamCurve](#).

29.481.3.20 `virtual double Go::SplineCurve::endparam ( ) const` [virtual]

Query the end parameter of the curve

## Returns

the curve's end parameter

Implements [Go::ParamCurve](#).

29.481.3.21 void Go::SplineCurve::equalBdWeights ( bool *at\_start* )

Update this curve to ensure equality of weights at one end of the curve. This is a reparametrization

29.481.3.22 virtual SplineCurve\* Go::SplineCurve::geometryCurve ( ) [virtual]

If the definition of this [ParamCurve](#) contains a [SplineCurve](#) describing its spatial shape, then this function will return a pointer to this [SplineCurve](#). Otherwise it will return a null pointer. The returned curve is NEWed, so the user is responsible for deleting it. This function may have side-effects.

#### Returns

a pointer to a [SplineCurve](#) representation of the [ParamCurve](#), if it exists. Null pointer otherwise.

Implements [Go::ParamCurve](#).

29.481.3.23 shared\_ptr<ElementaryCurve> Go::SplineCurve::getElementaryCurve ( )

Get shared pointer to [ElementaryCurve](#), if it exists. If not, return empty shared pointer.

NOTE: The [ElementaryCurve](#) returned by this function should ideally be the one corresponding to the current [SplineCurve](#). However, there is no guarantee for this. One may check to see if this is the case by using [checkElementaryCurve\(\)](#).

29.481.3.24 void Go::SplineCurve::getWeights ( std::vector< double > & *weights* ) const

Return all weights corresponding to this curve, a non-rational volume has all weights equal to one

29.481.3.25 void Go::SplineCurve::gridEvaluator ( std::vector< double > & *points*, const std::vector< double > & *par* ) const

Evaluation in a number of points Does not gain effectivity compared to evaluating the points one at the time, but provides a unified interface

29.481.3.26 void Go::SplineCurve::insertKnot ( double *apar* )

Insert a new knot into the curve's knotvector

#### Parameters

<i>apar</i>	parameter value of the new knot
-------------	---------------------------------

29.481.3.27 void Go::SplineCurve::insertKnot ( const std::vector< double > & *new\_knots* )

Insert several knots into the curve's knotvector



## Parameters

<i>new_knots</i>	vector containing the parameter values of the new knots to be inserted into the knotvector.
------------------	---------------------------------------------------------------------------------------------

29.481.3.28 `virtual ClassType Go::SplineCurve::instanceType ( ) const [virtual]`

Return the class type identifier of a given, derived instance of [GeomObject](#).

Implements [Go::GeomObject](#).

29.481.3.29 `void Go::SplineCurve::interpolate ( Interpolator & interpolator, int num_points, int dim, const double * param_start, const double * data_start )`

Remake 'this' [SplineCurve](#) to interpolate (or approximate) a given sequence of data points.

## Parameters

<i>interpolator</i>	reference to the <a href="#">Interpolator</a> object specifying the interpolation method to use
<i>num_points</i>	number of points to interpolate
<i>dim</i>	dimension of the Euclidean space in which the points lie (usually 2 or 3)
<i>param_start</i>	pointer to the start of an array expressing the parameter values of the given data points.
<i>data_start</i>	pointer to the array where the coordinates of the data points are stored.

29.481.3.30 `virtual bool Go::SplineCurve::isAxisRotational ( Point & centre, Point & axis, Point & vec, double & angle ) [virtual]`

Check if the curve is axis rotational. Only true if a connection to an axis rotational elementary curve exist The axis and rotational angle is only specified if the curve is actually rotational

Reimplemented from [Go::ParamCurve](#).

29.481.3.31 `virtual bool Go::SplineCurve::isDegenerate ( double degenerate_epsilon ) [virtual]`

Query whether the curve is degenerate (collapsed into a single point).

## Parameters

<i>degenerate_epsilon</i>	the tolerance used in determine whether the curve is degenerate. A curve is considered degenerate if its total length is shorter than this value.
---------------------------	---------------------------------------------------------------------------------------------------------------------------------------------------

## Returns

`true` if the curve is degenerate, `false` otherwise.

Implements [Go::ParamCurve](#).

29.481.3.32 `bool Go::SplineCurve::isElementaryCurve ( ) [inline]`

Query if the curve was generated from an [ElementaryCurve](#).

Definition at line 521 of file `SplineCurve.h`.

29.481.3.33 `virtual bool Go::SplineCurve::isInPlane ( const Point & loc, const Point & axis, double eps, Point & normal ) const [virtual]`

Check if the curve lies in a plane passing through a given axis.

Reimplemented from [Go::ParamCurve](#).

29.481.3.34 `virtual bool Go::SplineCurve::isInPlane ( const Point & norm, double eps, Point & pos ) const [virtual]`

Check if the curve lies in a plane with a given normal.

Reimplemented from [Go::ParamCurve](#).

29.481.3.35 `virtual bool Go::SplineCurve::isLinear ( Point & dir, double tol ) [virtual]`

Check if the curve is linear.

Reimplemented from [Go::ParamCurve](#).

29.481.3.36 `std::vector<double>::iterator Go::SplineCurve::knotsBegin ( ) [inline]`

Get an iterator to the beginning of the knot vector

#### Returns

an iterator to the beginning of the knot vector

Definition at line 356 of file `SplineCurve.h`.

29.481.3.37 `std::vector<double>::const_iterator Go::SplineCurve::knotsBegin ( ) const [inline]`

Get a const iterator to the beginning of the knot vector

#### Returns

a const iterator to the beginning of the knot vector

Definition at line 364 of file `SplineCurve.h`.

29.481.3.38 `std::vector<double>::iterator Go::SplineCurve::knotsEnd ( ) [inline]`

Get a one-past-end iterator to the knot vector

**Returns**

an iterator to one-past-end of the knot vector

Definition at line 360 of file SplineCurve.h.

29.481.3.39 `std::vector<double>::const_iterator Go::SplineCurve::knotsEnd ( ) const [inline]`

Get a one-past-end const iterator to the knot vector

**Returns**

a const iterator to one-past-end of the knot vector

Definition at line 368 of file SplineCurve.h.

29.481.3.40 `virtual double Go::SplineCurve::length ( double tol ) [virtual]`

Inherited from [ParamCurve](#) Compute the total length of this curve

Implements [Go::ParamCurve](#).

29.481.3.41 `void Go::SplineCurve::makeBernsteinKnots ( )`

Insert knots into the knotvector such that all knots get multiplicity equal to the order of the b-spline basis.

29.481.3.42 `void Go::SplineCurve::makeKnotEndRegular ( )`

Make the knotvector k-regular at end. Useful when k-regularity of a knotvector is to be assumed.

29.481.3.43 `void Go::SplineCurve::makeKnotStartRegular ( )`

Make the knotvector k-regular at start. Useful when k-regularity of a knotvector is to be assumed.

29.481.3.44 `virtual double Go::SplineCurve::nextSegmentVal ( double par, bool forward, double tol ) const [virtual]`

Inherited from [ParamCurve](#). Returns the value of the next knot.

## Parameters

<i>par</i>	the parameter from which we will look for the start of the next interval.
<i>forward</i>	specify whether we will look forwards or backwards.
<i>tol</i>	a tolerance specifying how close 'par' has to be to a knot to be considered 'on' the knot.

Reimplemented from [Go::ParamCurve](#).

29.481.3.45 `int Go::SplineCurve::numCoefs ( ) const [inline]`

Query the number of control points of the curve

## Returns

the number of control points of the curve.

Definition at line 335 of file SplineCurve.h.

29.481.3.46 `int Go::SplineCurve::numElem ( ) const [inline]`

Query the number of elements in the [SplineCurve](#).

Definition at line 339 of file SplineCurve.h.

29.481.3.47 `int Go::SplineCurve::order ( ) const [inline]`

Query the order of the spline space in which the curve lies

## Returns

the order of the curve's spline space

Definition at line 346 of file SplineCurve.h.

29.481.3.48 `virtual void Go::SplineCurve::point ( Point & pt, double tpar ) const [virtual]`

Evaluate the curve's position at a given parameter

## Parameters

<i>pt</i>	the evaluated position will be written to this <a href="#">Point</a>
<i>tpar</i>	the parameter for which we wish to evaluate the curve

Implements [Go::ParamCurve](#).

29.481.3.49 `virtual void Go::SplineCurve::point ( std::vector< Point > & pts, double tpar, int derivs, bool from_right = true ) const [virtual]`

Evaluate the curve's position and a certain number of derivatives at a given parameter.

#### Parameters

<i>pts</i>	the evaluated position and derivatives (tangent, curvature vector, etc.) will be written to this vector. The first entry will be the position, the second entry will be the first derivative, etc. The size of this vector must be set to 'derivs'+ 1 prior to calling this function.
<i>tpar</i>	the parameter for which we want to evaluate the curve
<i>derivs</i>	the number of derivatives we want to have calculated
<i>from_right</i>	specify whether we should calculate derivatives 'from the right' or 'from the left' (default is from the right). This matters only when the curve presents discontinuities in its derivatives.

Implements [Go::ParamCurve](#).

29.481.3.50 `void Go::SplineCurve::raiseOrder ( int i = 1 )`

Raise the order of the curve's b-spline basis without changing the shape of the curve

#### Parameters

<i>i</i>	specifies how many times the order will be raised.
----------	----------------------------------------------------

29.481.3.51 `bool Go::SplineCurve::rational ( ) const [inline]`

Query whether or not the curve is rational.

#### Returns

'true' if the curve is rational, 'false' otherwise.

Definition at line 351 of file SplineCurve.h.

29.481.3.52 `std::vector<double>::iterator Go::SplineCurve::rcoefs_begin ( ) [inline]`

Get an iterator to the start of the curve's internal, rational control point array

#### Returns

an iterator to the start of the curves rational control point array

Definition at line 400 of file SplineCurve.h.

29.481.3.53 `std::vector<double>::const_iterator Go::SplineCurve::rcoefs_begin ( ) const` [inline]

Get a const iterator to the start of the curve's rational, internal control point array

**Returns**

a const iterator to the start of the curve's rational control point array.

Definition at line 412 of file SplineCurve.h.

29.481.3.54 `std::vector<double>::iterator Go::SplineCurve::rcoefs_end ( )` [inline]

Get a one-past-end iterator to the curve's rational, internal control point array

**Returns**

an iterator to one-past-end of the curve's rational, internal control point array

Definition at line 406 of file SplineCurve.h.

29.481.3.55 `std::vector<double>::const_iterator Go::SplineCurve::rcoefs_end ( ) const` [inline]

Get a one-past-end const iterator to the curve's rational, internal control point array

**Returns**

a const iterator to one-past-end of the curve's rational, internal control point array

Definition at line 418 of file SplineCurve.h.

29.481.3.56 `virtual void Go::SplineCurve::read ( std::istream & is )` [virtual]

read object from stream

**Parameters**

<i>is</i>	stream from which object is read
-----------	----------------------------------

Implements [Go::Streamable](#).

29.481.3.57 `void Go::SplineCurve::removeKnot ( double tpar )`

Remove a knot from the knotvector

## Parameters

<i>tpar</i>	the parameter value of the knot to be removed.
-------------	------------------------------------------------

29.481.3.58 void Go::SplineCurve::replaceEndPoint ( Point *pnt*, bool *at\_start* )

Modify the curve by changing on endpoint

## Parameters

<i>pnt</i>	new end point
<i>at_start</i>	whether or not the start coefficient should be changed

29.481.3.59 void Go::SplineCurve::representAsRational ( )

Ensure that the current curve is represented as a rational curve.

29.481.3.60 virtual void Go::SplineCurve::reverseParameterDirection ( bool *switchparam = false* ) [virtual]

Set the parameter direction of the curve. The curve's parameter interval will always remain constant, but by flipping the parameter direction, the curve will be traced the opposite way when moving a parameter over the parameter interval.

## Parameters

<i>switchparam</i>	if true, and the curve is 2D, the x and y coordinates should be swapped. This is used when turning the orientation of bounded (trimmed) surfaces.
--------------------	---------------------------------------------------------------------------------------------------------------------------------------------------

Implements [Go::ParamCurve](#).

29.481.3.61 void Go::SplineCurve::setBdWeight ( double *wgt*, bool *at\_start* )

Set the weight in one endpoint of this curve (if rational)

29.481.3.62 void Go::SplineCurve::setElementaryCurve ( shared\_ptr< ElementaryCurve > *elcurve* )

Set shared pointer to the [ElementaryCurve](#) that is represented by *this*.

NOTE: The current [SplineCurve](#) (i.e. *this*), must be created from the argument [ElementaryCurve](#) by [ElementaryCurve::createSplineCurve\(\)](#), otherwise undefined behaviour may occur. One may check to see if this is the case by using [checkElementaryCurve\(\)](#).

29.481.3.63 virtual void Go::SplineCurve::setParameterInterval ( double *t1*, double *t2* ) [virtual]

Rescale the knotvector so that the total parameter span now ranges from 't1' to 't2'.

## Parameters

<i>t1</i>	new start value of the spline's parameter span
<i>t2</i>	new end value of the spline's parameter span

Implements [Go::ParamCurve](#).

29.481.3.64 `std::vector<shared_ptr<SplineCurve>> Go::SplineCurve::split ( std::vector< double > & param, double fuzzy = DEFAULT_PARAMETER_EPSILON ) const`

Split curve in specified parameter values.

29.481.3.65 `virtual std::vector<shared_ptr<ParamCurve>> Go::SplineCurve::split ( double param, double fuzzy = DEFAULT_PARAMETER_EPSILON ) const [virtual]`

Split curve in a specified parameter value.

Reimplemented from [Go::ParamCurve](#).

29.481.3.66 `virtual double Go::SplineCurve::startparam ( ) const [virtual]`

Query the start parameter of the curve

## Returns

the curve's start parameter

Implements [Go::ParamCurve](#).

29.481.3.67 `virtual SplineCurve* Go::SplineCurve::subCurve ( double from_par, double to_par, double fuzzy = DEFAULT_PARAMETER_EPSILON ) const [virtual]`

Returns a curve which is a part of this curve. The result is NEWed, so the user is responsible for deleting it. NB: It is not guaranteed that the [ParamCurve](#) that is returned is of the same type as the curve itself. Thus, the returned curve might be a [SplineCurve](#).

## Parameters

<i>from_par</i>	start value of parameter interval that will define the subcurve
<i>to_par</i>	end value of parameter interval that will define the subcurve
<i>fuzzy</i>	since subCurve works on those curves who are spline-based, this tolerance defines how close the start and end parameter must be to an existing knot in order to be considered <i>on</i> the knot.



**Returns**

a pointer to a new subcurve which represents the part of the curve between 'from\_par' and 'to\_par'. It will be spline-based and have a k-regular knotvector. The user is responsible for deleting this subcurve when it is no longer needed.

Implements [Go::ParamCurve](#).

**29.481.3.68** void Go::SplineCurve::swap ( SplineCurve & *other* )

quick swap of 'this' [SplineCurve](#) with the 'other' one.

**Parameters**

<i>other</i>	the <a href="#">SplineCurve</a> to swap with 'this' one.
--------------	----------------------------------------------------------

**29.481.3.69** void Go::SplineCurve::translateCurve ( const Point & *dir* )

**29.481.3.70** void Go::SplineCurve::translateSwapCurve ( const Point & *dir*, double *sgn*, int *pdir* )

Modify in 1. (*pdir* == 1), 2. (*pdir* == 2) or both (*pdir* == 3) parameter directions

**29.481.3.71** void Go::SplineCurve::updateCoefsFromRcoefs ( )

Given the rational coefficients of the curve (*hx*, *hy*, *hz*, *h*), set the coefficient array where the weight is divided out (*x*, *y*, *z*).

**29.481.3.72** virtual void Go::SplineCurve::write ( std::ostream & *os* ) const [virtual]

write object to stream

**Parameters**

<i>os</i>	stream to which object is written
-----------	-----------------------------------

Implements [Go::Streamable](#).

The documentation for this class was generated from the following file:

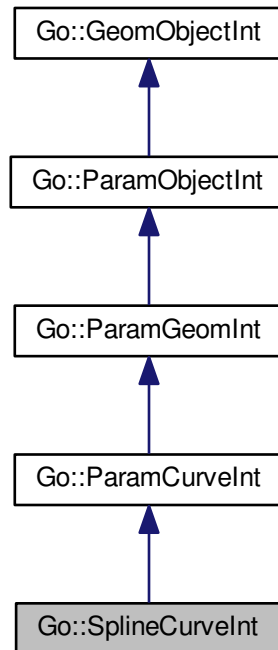
- [gotools-core/include/GoTools/geometry/SplineCurve.h](#)

## 29.482 Go::SplineCurveInt Class Reference

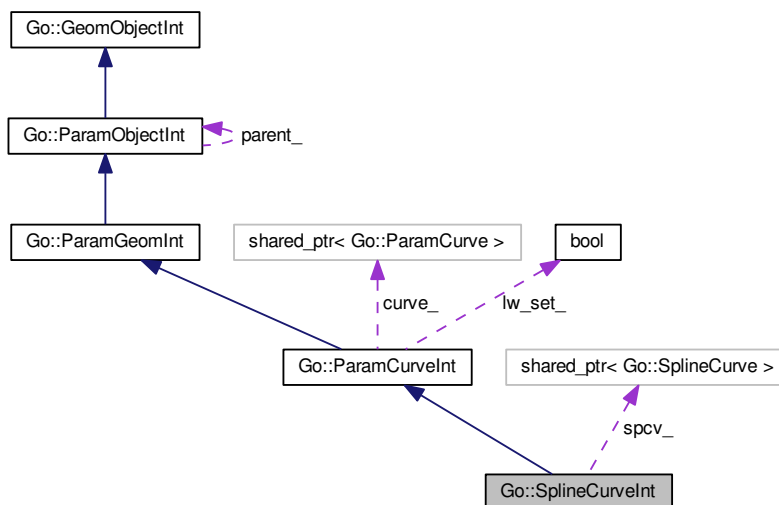
Class representing the "intersection object" of a spline curve.

```
#include <SplineCurveInt.h>
```

Inheritance diagram for Go::SplineCurveInt:



Collaboration diagram for Go::SplineCurveInt:



## Public Member Functions

- [SplineCurveInt](#) (shared\_ptr< [ParamCurve](#) > *curve*)
  - [SplineCurveInt](#) (shared\_ptr< [ParamCurve](#) > *curve*, [ParamGeomInt](#) \*parent)
  - virtual [~SplineCurveInt](#) ()
- Destructor.*
- virtual shared\_ptr< [ParamCurveInt](#) > [makeIntObject](#) (shared\_ptr< [ParamCurve](#) > *curve*)
  - virtual [bool](#) [hasInnerKnots](#) (int pdir) [const](#)
  - virtual [bool](#) [hasCriticalValsOrKnots](#) (int pdir) [const](#)
  - virtual std::vector< [double](#) > [getInnerKnotVals](#) (int pdir, [bool](#) sort=false) [const](#)
  - virtual std::vector< [double](#) > [getCriticalValsAndKnots](#) (int pdir) [const](#)
  - virtual int [getMeshSize](#) (int dir)
  - virtual [double](#) [paramFromMesh](#) (int dir, int *idx*)
  - virtual std::vector< [double](#) >::iterator [getMesh](#) ()
- Return the geometric sample mesh for the parametric function.*
- virtual int [checkPeriodicity](#) (int pdir=0) [const](#)
  - virtual int [knotIntervalFuzzy](#) ([double](#) &t, [double](#) tol) [const](#)
  - virtual [double](#) [nextSegmentVal](#) ([double](#) par, [bool](#) forward, [double](#) tol) [const](#)
  - virtual [bool](#) [isSpline](#) ()
  - virtual [const](#) [SplineCurve](#) \* [getSpline](#) ()
  - virtual [double](#) [getOptimizedConeAngle](#) ([Point](#) &axis1, [Point](#) &axis2)
  - virtual [RotatedBox](#) [getRotatedBox](#) (std::vector< [Point](#) > &axis) [const](#)

## Protected Attributes

- shared\_ptr< [SplineCurve](#) > *spcv\_*

### 29.482.1 Detailed Description

Class representing the "intersection object" of a spline curve.

Definition at line 55 of file [SplineCurveInt.h](#).

### 29.482.2 Constructor & Destructor Documentation

29.482.2.1 [Go::SplineCurveInt::SplineCurveInt](#) ( shared\_ptr< [ParamCurve](#) > *curve* ) [\[explicit\]](#)

Constructor.

Parameters

<i>curve</i>	the parametric curve defining the intersection object.
--------------	--------------------------------------------------------

29.482.2.2 [Go::SplineCurveInt::SplineCurveInt](#) ( shared\_ptr< [ParamCurve](#) > *curve*, [ParamGeomInt](#) \* *parent* ) [\[explicit\]](#)

Constructor.

## Parameters

<i>curve</i>	the parametric curve defining the intersection object.
<i>parent</i>	the parent object to this object. Can be either a curve or a surface.

29.482.2.3 virtual Go::SplineCurveInt::~~SplineCurveInt ( ) [inline],[virtual]

Destructor.

Definition at line 71 of file SplineCurveInt.h.

### 29.482.3 Member Function Documentation

29.482.3.1 virtual int Go::SplineCurveInt::checkPeriodicity ( int *pardir* = 0 ) const [virtual]

Check if the object is periodic. Analyze periodicity of curve based on number of repeating knots and control points. The return value is -1 if the curve ends are disjoint, otherwise k if cv is  $C^k$  continuous. These are sufficient but not necessary conditions for periodicity, so it is possible that a higher degree of periodicity exists. Should not be called on this layer, should be overruled by inherited class.

## Parameters

<i>pardir</i>	the parameter direction in question. Does not pertain to for a curve.
---------------	-----------------------------------------------------------------------

## Returns

-1 if the curve ends are disjoint, or k if the curve is proven to be  $C^k$  continuous.

Reimplemented from [Go::ParamCurveInt](#).

29.482.3.2 virtual std::vector<double> Go::SplineCurveInt::getCriticalValsAndKnots ( int *pardir* ) const [virtual]

Return the critical parameter values and inner knots for object.

## Parameters

<i>pardir</i>	the parameter direction in question. Indexing starts at 0.
---------------	------------------------------------------------------------

Reimplemented from [Go::ParamCurveInt](#).

29.482.3.3 virtual std::vector<double> Go::SplineCurveInt::getInnerKnotVals ( int *pardir*, bool *sort* = false ) const [virtual]

Return the inner knot values in the specified direction.

## Parameters

<i>parDir</i>	the parameter direction in question. Indexing starts at 0.
<i>sort</i>	the returned values may be sorted by the function.

Reimplemented from [Go::ParamCurveInt](#).

29.482.3.4 `virtual std::vector<double>::iterator Go::SplineCurveInt::getMesh ( ) [virtual]`

Return the geometric sample mesh for the parametric function.

Reimplemented from [Go::ParamCurveInt](#).

29.482.3.5 `virtual int Go::SplineCurveInt::getMeshSize ( int dir ) [virtual]`

Return the size of the geometric sample mesh in the specified direction.

## Parameters

<i>dir</i>	the parameter direction in question. Indexing starts at 0.
------------	------------------------------------------------------------

Reimplemented from [Go::ParamCurveInt](#).

29.482.3.6 `virtual double Go::SplineCurveInt::getOptimizedConeAngle ( Point & axis1, Point & axis2 ) [virtual]`

We try to treat problems which will never result in a simple case by shrinking the domain slightly, resulting in smaller cones. This is useful for scenarios where the normals are parallel in a boundary point.

## Parameters

<i>axis1</i>	first vector defining a projection plane
<i>axis2</i>	second vector defining a projection plane

## Returns

The optimized cone angle

Reimplemented from [Go::ParamCurveInt](#).

29.482.3.7 `virtual RotatedBox Go::SplineCurveInt::getRotatedBox ( std::vector< Point > & axis ) const [virtual]`

Create a box containing the geometric sample mesh in the input coordinate system.

## Parameters

<i>axis</i>	the axis defining the coordinate system. The size of vector <i>axis</i> is 2, the last point is given as the cross product <i>axis</i> [0] <i>axis</i> [1].
-------------	-------------------------------------------------------------------------------------------------------------------------------------------------------------

## Returns

The rotated box

Reimplemented from [Go::ParamCurveInt](#).

**29.482.3.8** `virtual const SplineCurve* Go::SplineCurveInt::getSpline ( ) [inline],[virtual]`

Reimplemented from [Go::ParamCurveInt](#).

Definition at line 154 of file *SplineCurveInt.h*.

**29.482.3.9** `virtual bool Go::SplineCurveInt::hasCriticalValsOrKnots ( int pardir ) const [virtual]`

Return true if the object has any critical parameter values or inner knots in the specified parameter direction.

## Parameters

<i>pardir</i>	the parameter direction in question. Indexing starts at 0.
---------------	------------------------------------------------------------

Reimplemented from [Go::ParamCurveInt](#).

**29.482.3.10** `virtual bool Go::SplineCurveInt::hasInnerKnots ( int pardir ) const [virtual]`

Return true if the object has any inner knots in the specified parameter direction.

## Parameters

<i>pardir</i>	the parameter direction in question. Indexing starts at 0.
---------------	------------------------------------------------------------

Reimplemented from [Go::ParamCurveInt](#).

**29.482.3.11** `virtual bool Go::SplineCurveInt::isSpline ( ) [virtual]`

Verify whether the object is a spline.

## Returns

True if the object is a spline.

Reimplemented from [Go::ParamCurveInt](#).

29.482.3.12 `virtual int Go::SplineCurveInt::knotIntervalFuzzy ( double & t, double tol ) const` [virtual]

Find the knot interval for which t lies inside, moving the value t if it lies close to a knot.

#### Parameters

<i>t</i>	the parameter value
<i>tol</i>	the parametric tolerance deciding if the input parameter t should be moved.

Reimplemented from [Go::ParamCurveInt](#).

29.482.3.13 `virtual shared_ptr<ParamCurveInt> Go::SplineCurveInt::makeIntObject ( shared_ptr< ParamCurve > curve )` [virtual]

Return an intersection object for the input curve. This object is used as the parent for the intersection object.

#### Parameters

<i>curve</i>	the parametric curve defining the intersection object.
--------------	--------------------------------------------------------

Reimplemented from [Go::ParamCurveInt](#).

29.482.3.14 `virtual double Go::SplineCurveInt::nextSegmentVal ( double par, bool forward, double tol ) const` [virtual]

Return the value of the knot next to the input parameter par.

#### Parameters

<i>par</i>	the parameter value
<i>forward</i>	if true we return the closest knot to the right, otherwise the closest knot to the left.
<i>tol</i>	the tolerance to determine if <i>par</i> is already located on the start of the next segment.

#### Returns

The knot closest to the input parameter.

Reimplemented from [Go::ParamCurveInt](#).

29.482.3.15 `virtual double Go::SplineCurveInt::paramFromMesh ( int dir, int idx )` [virtual]

Return the corresponding mesh parameter.

#### Parameters

<i>dir</i>	the parameter direction in question. Indexing starts at 0.
<i>idx</i>	the mesh idx in the specified direction. Indexing starts at 0.

Reimplemented from [Go::ParamCurveInt](#).

#### 29.482.4 Member Data Documentation

##### 29.482.4.1 `shared_ptr<SplineCurve> Go::SplineCurveInt::spcv_` `[protected]`

Definition at line 178 of file `SplineCurveInt.h`.

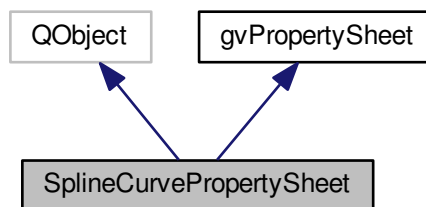
The documentation for this class was generated from the following file:

- `intersections/include/GoTools/intersections/SplineCurveInt.h`

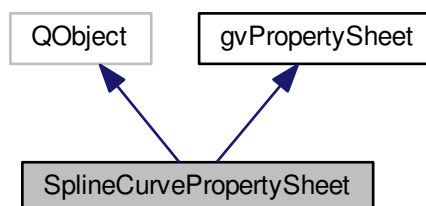
### 29.483 SplineCurvePropertySheet Class Reference

```
#include <SplineCurvePropertySheet.h>
```

Inheritance diagram for `SplineCurvePropertySheet`:



Collaboration diagram for `SplineCurvePropertySheet`:





## Public Slots

- void [apply](#) ()

## Public Member Functions

- [SplineCurvePropertySheet](#) ([Go::CurveTesselator](#) \*tess, [gvCurvePaintable](#) \*pable)
- virtual void [createSheet](#) ([QWidget](#) \*parent, [gvObserver](#) \*obs)

### 29.483.1 Detailed Description

Documentation ... etc

Definition at line 57 of file [SplineCurvePropertySheet.h](#).

### 29.483.2 Constructor & Destructor Documentation

29.483.2.1 [SplineCurvePropertySheet::SplineCurvePropertySheet](#) ( [Go::CurveTesselator](#) \* tess, [gvCurvePaintable](#) \* pable ) `[inline]`

Definition at line 63 of file [SplineCurvePropertySheet.h](#).

### 29.483.3 Member Function Documentation

29.483.3.1 void [SplineCurvePropertySheet::apply](#) ( ) `[slot]`

29.483.3.2 virtual void [SplineCurvePropertySheet::createSheet](#) ( [QWidget](#) \* parent, [gvObserver](#) \* obs ) `[virtual]`

Implements [gvPropertySheet](#).

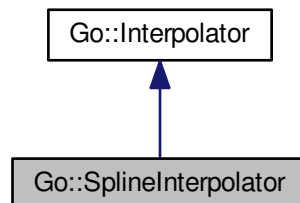
The documentation for this class was generated from the following file:

- [viewlib/include/GoTools/viewlib/SplineCurvePropertySheet.h](#)

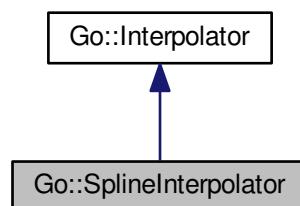
## 29.484 Go::SplineInterpolator Class Reference

```
#include <SplineInterpolator.h>
```

Inheritance diagram for Go::SplineInterpolator:



Collaboration diagram for Go::SplineInterpolator:



### Public Types

- enum `CondType` {  
`None`, `Hermite`, `Natural`, `Free`,  
`NaturalAtStart`, `NaturalAtEnd` }

### Public Member Functions

- `SplineInterpolator ()`  
*Constructor takes no arguments.*
- virtual `~SplineInterpolator ()`  
*Virtual destructor ensures safe inheritance.*
- virtual `const BsplineBasis & basis ()`
- void `interpolate (const std::vector< double > &params, const std::vector< double > &points, const std::vector< int > &tangent_index, const std::vector< double > &tangent_points, int order, std::vector< double > &coefs)`

- void `interpolate` (`const` `std::vector< double >` &params, `const` `std::vector< double >` &points, `const` `std::vector< int >` &tangent\_index, `const` `std::vector< double >` &tangent\_points, `std::vector< double >` &coefs)
- virtual void `interpolate` (int num\_points, int dimension, `const double` \*param\_start, `const double` \*data\_start, `std::vector< double >` &coefs)
- void `setHermiteConditions` (`const Point` &start\_tangent, `const Point` &end\_tangent)
- void `setNaturalConditions` ()
- void `setNaturalStartCondition` ()
- void `setNaturalEndCondition` ()
- void `setFreeConditions` ()
- void `setEndTangents` (`shared_ptr< Point >` &start\_tangent, `shared_ptr< Point >` &end\_tangent)
- `CondType` `getCondType` ()
- void `makeBasis` (`const` `std::vector< double >` &params, `const` `std::vector< int >` &tangent\_index, int order)
- void `setBasis` (`const BsplineBasis` &basis)

### 29.484.1 Detailed Description

An [Interpolator](#) that generates a spline curve interpolating the given dataset.

Definition at line 56 of file `SplineInterpolator.h`.

### 29.484.2 Member Enumeration Documentation

#### 29.484.2.1 enum Go::SplineInterpolator::CondType

CondType Enumerator specifying possible boundary conditions.

```

/// None boundary conditions have not been specified yet. This will lead
/// to an error if the virtual interpolate() function is called. (The
/// other two interpolate() functions do not care about end conditions.
/// Hermite The tangents are imposed at the start and end of the curve.
/// Natural No imposed tangents, but curvature is imposed to be zero at the
/// start and end of the curve.
/// Free No imposed conditions, neither at start nor end of curve.
/// NaturalAtStart Curvature imposed to be zero at start of curve. End of curve
/// will either have no conditions at all, or have an imposed
/// tangent (depending on whether the latter has been specified or not.
/// NaturalAtEnd Curvature imposed to be zero at end of curve. Start of curve
/// will either have no conditions, or have an imposed tangent
/// (depending on whether the latter has been specified or not).
///

```

Enumerator

***None***  
***Hermite***  
***Natural***  
***Free***  
***NaturalAtStart***  
***NaturalAtEnd***

Definition at line 141 of file `SplineInterpolator.h`.

### 29.484.3 Constructor & Destructor Documentation

#### 29.484.3.1 `Go::SplineInterpolator::SplineInterpolator ( )` [inline]

Constructor takes no arguments.

Definition at line 60 of file `SplineInterpolator.h`.

#### 29.484.3.2 `virtual Go::SplineInterpolator::~~SplineInterpolator ( )` [virtual]

Virtual destructor ensures safe inheritance.

### 29.484.4 Member Function Documentation

#### 29.484.4.1 `virtual const BsplineBasis& Go::SplineInterpolator::basis ( )` [virtual]

after the function `interpolate()` has been successfully run, this function can be called to get the `BsplineBasis` of the generated curve.

##### Returns

a constant reference to the `BsplineBasis` of the curve previously generated by `interpolate()`.

Implements `Go::Interpolator`.

#### 29.484.4.2 `CondType Go::SplineInterpolator::getCondType ( )` [inline]

Query the type of endpoint conditions currently specified.

##### Returns

the currently specified `CondType`.

Definition at line 201 of file `SplineInterpolator.h`.

#### 29.484.4.3 `void Go::SplineInterpolator::interpolate ( const std::vector< double > & params, const std::vector< double > & points, const std::vector< int > & tangent_index, const std::vector< double > & tangent_points, int order, std::vector< double > & coefs )`

This interpolating function also takes user-specified tangent information into account, but does *not* rely on the previously specified end conditions. It constructs its own basis based on the given parametrization and order, so neither does it care about any basis previously set by the user. It carries out order-1 spline interpolation.

##### Parameters

<i>params</i>	vector containing the parameters for the data points. Its size is equal to the total number of datapoints.
<i>points</i>	vector containing the coordinates of the data points. Its size is equal to the total number of datapoints multiplied with the spatial dimension. (NB: This is how the algorithm deduces the spatial dimension to use: divide the size of 'points' with the size of 'params').
<i>tangent_index</i>	a vector containing the indexes of those data points that has tangents associated with them.
<i>tangent points</i>	a vector containing the tangents associated with those data points that are referred to in

29.484.4.4 void Go::SplineInterpolator::interpolate ( const std::vector< double > & *params*, const std::vector< double > & *points*, const std::vector< int > & *tangent\_index*, const std::vector< double > & *tangent\_points*, std::vector< double > & *coefs* )

Does the same as the [interpolate\(\)](#) function above, but expects the basis to have been specified in advance (by [makeBasis\(\)](#) or [setBasis\(\)](#)). If the basis is not consistent with the given data set, an exception will be thrown. (The number of point must at least be equal to the basis' order, and the number of basis functions must be equal to the number of points plus the number of tangents). For a description of the parameter list, see [interpolate\(\)](#).

29.484.4.5 virtual void Go::SplineInterpolator::interpolate ( int *num\_points*, int *dimension*, const double \* *param\_start*, const double \* *data\_start*, std::vector< double > & *coefs* ) [virtual]

The interpolating function, as inherited by [Interpolator](#). Does cubic spline interpolation of points (does not care about tangents). It constructs its own basis based on the given parametrization, so it does not care about any basis previously set by the user. Previously specified end conditions will be taken into account. For parameter list, see [Interpolator](#).

Implements [Go::Interpolator](#).

29.484.4.6 void Go::SplineInterpolator::makeBasis ( const std::vector< double > & *params*, const std::vector< int > & *tangent\_index*, int *order* )

Given a set of interpolation conditions (currently position and tangent information) and order, specify a fitting [BsplineBasis](#) (stored internally).

#### Parameters

<i>params</i>	vector containing the parametrization of the expected datapoints (one parameter per point).
<i>tangent_index</i>	vector containing indexes of those datapoints where tangent conditions will be imposed as well.
<i>order</i>	Order of the requested <a href="#">BsplineBasis</a> .

29.484.4.7 void Go::SplineInterpolator::setBasis ( const [BsplineBasis](#) & *basis* ) [inline]

WARNING! basis must be suited to interpolation conditions! If there is no need to end up with a specific basis, [makeBasis\(\)](#) is safer. Specify a user-defined [BsplineBasis](#) to be used for interpolation.

#### Parameters

<i>basis</i>	the specified basis. NB: it must be suited to the interpolation conditions that are going to be used! If there is no need to end up with a specific basis, <a href="#">makeBasis()</a> is safer.
--------------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Definition at line 221 of file [SplineInterpolator.h](#).

29.484.4.8 void Go::SplineInterpolator::setEndTangents ( shared\_ptr< [Point](#) > & *start\_tangent*, shared\_ptr< [Point](#) > & *end\_tangent* ) [inline]

Specify tangents for endpoints of curve. One or both of the shared pointers may be zero, which means that the corresponding endpoint condition will be set to 'Natural'; otherwise it will be set to 'Hermite'. **Note:** endpoint conditions are considered by the *virtual* [interpolate\(\)](#) function. The other interpolate functions disregard this setting.

Definition at line 184 of file SplineInterpolator.h.

29.484.4.9 `void Go::SplineInterpolator::setFreeConditions ( ) [inline]`

Set the endpoint conditions to 'Free' (meaning no conditions at all). **Note:** endpoint conditions are considered by the *virtual interpolate()* function. The other interpolate functions disregard this setting.

Definition at line 177 of file SplineInterpolator.h.

29.484.4.10 `void Go::SplineInterpolator::setHermiteConditions ( const Point & start_tangent, const Point & end_tangent ) [inline]`

Set the endpoint conditions to 'Hermite'; impose tangents at start and end of curve. **Note:** endpoint conditions are considered by the *virtual interpolate()* function. The other interpolate functions disregard this setting.

#### Parameters

<i>start_tangent</i>	the imposed tangent at the start of the curve
<i>end_tangent</i>	the imposed tangent at the end of the curve.

Definition at line 148 of file SplineInterpolator.h.

29.484.4.11 `void Go::SplineInterpolator::setNaturalConditions ( ) [inline]`

Set the endpoint conditions to 'Natural'; zero curvature at start and end of curve. **Note:** endpoint conditions are considered by the *virtual interpolate()* function. The other interpolate functions disregard this setting.

Definition at line 158 of file SplineInterpolator.h.

29.484.4.12 `void Go::SplineInterpolator::setNaturalEndCondition ( ) [inline]`

Set the endpoint condition at end of curve to 'Natural'. (Condition at start of curve will be 'Hermite' if a tangent has been previously specified, or 'Free' otherwise). **Note:** endpoint conditions are considered by the *virtual interpolate()* function. The other interpolate functions disregard this setting.

Definition at line 172 of file SplineInterpolator.h.

29.484.4.13 `void Go::SplineInterpolator::setNaturalStartCondition ( ) [inline]`

Set the endpoint condition at start of curve to 'Natural'. (Condition at end of curve will be 'Hermite' if a tangent has been previously specified, or 'Free' otherwise). **Note:** endpoint conditions are considered by the *virtual interpolate()* function. The other interpolate functions disregard this setting.

Definition at line 165 of file SplineInterpolator.h.

The documentation for this class was generated from the following file:

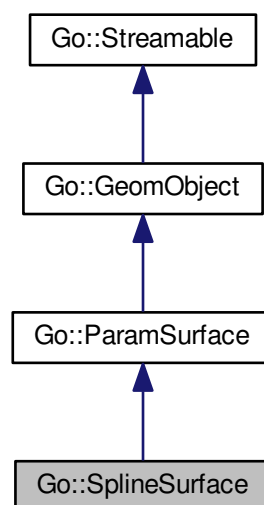
- [gotools-core/include/GoTools/geometry/SplineInterpolator.h](#)

## 29.485 Go::SplineSurface Class Reference

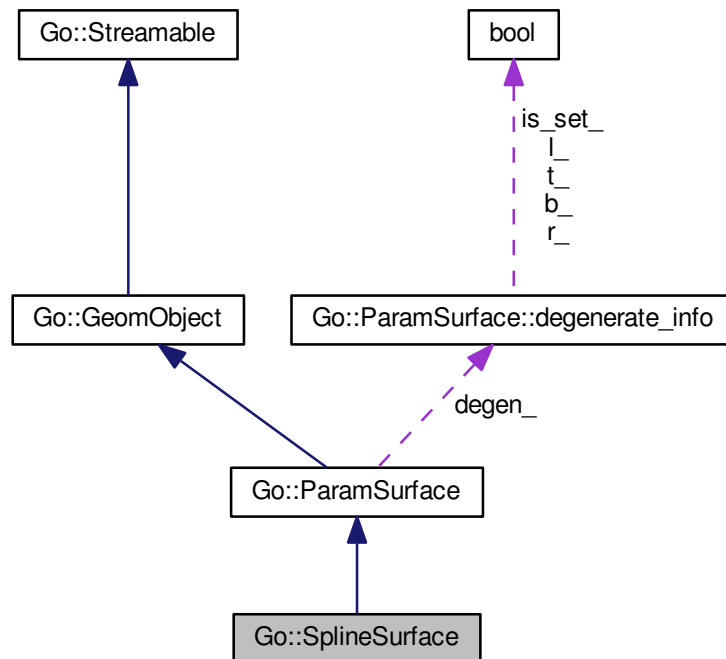
[SplineSurface](#) provides methods for storing, reading and manipulating rational and non-rational B-spline surfaces.

```
#include <SplineSurface.h>
```

Inheritance diagram for Go::SplineSurface:



Collaboration diagram for Go::SplineSurface:



## Public Types

- enum [NormalConeMethod](#) { [SederbergMeyers](#) = 0, [SMCornersFirst](#) = 1, [sislBased](#) = 2 }  
*Enumerates the method for computing the normal cone.*
- typedef `std::vector< double >` [Dvector](#)  
*Convenience to be used in computations of basis grids.*
- typedef `std::vector< Dvector >` [Dmatrix](#)  
*Convenience to be used in computations of basis grids.*

## Public Member Functions

- [SplineSurface](#) ()
- `template<typename RandomIterator1 , typename RandomIterator2 , typename RandomIterator3 >`  
[SplineSurface](#) (int number1, int number2, int order1, int order2, RandomIterator1 knot1start, RandomIterator2 knot2start, RandomIterator3 coefsstart, int dim, bool rational=false)
- `template<typename RandomIterator >`  
[SplineSurface](#) (const [BsplineBasis](#) &basis\_u, const [BsplineBasis](#) &basis\_v, RandomIterator coefsstart, int dim, bool rational=false)
- virtual `~SplineSurface` ()  
*Virtual destructor, enables safe inheritance.*
- virtual void [read](#) (std::istream &is)
- virtual void [write](#) (std::ostream &os) const
- virtual [BoundingBox](#) boundingBox () const



- Return the object's bounding box.*
- virtual int `dimension () const`
- Return the dimension of the space in which the object lies (usually 2 or 3)*
- void `swap (SplineSurface &other)`
- quick swap of two SplineSurface objects with each other*
- virtual `ClassType instanceType () const`
- Return the class type identifier of a given, derived instance of GeomObject.*
- virtual `SplineSurface * clone () const`
- virtual `SplineSurface * asSplineSurface ()`
- Return the spline surface represented by this surface, if any.*
- virtual `const RectDomain & parameterDomain () const`
- virtual `RectDomain containingDomain () const`
- virtual `bool inDomain (double u, double v, double eps=1.0e-4) const`
- Check if a parameter pair lies inside the domain of this surface.*
- virtual `int inDomain2 (double u, double v, double eps=1.0e-4) const`
- virtual `bool onBoundary (double u, double v, double eps=1.0e-4) const`
- Check if a parameter pair lies at the boundary of this surface.*
- virtual `Point closestInDomain (double u, double v) const`
- virtual `CurveLoop outerBoundaryLoop (double degenerate_epsilon=DEFAULT_SPACE_EPSILON) const`
- virtual `std::vector< CurveLoop > allBoundaryLoops (double degenerate_epsilon=DEFAULT_SPACE_EPSILON) const`
- virtual `void point (Point &pt, double upar, double vpar) const`
- virtual `void point (std::vector< Point > &pts, double upar, double vpar, int derivs, bool u_from_right=true, bool v_from_right=true, double resolution=1.0e-12) const`
- virtual `double startparam_u () const`
- virtual `double startparam_v () const`
- `double startparam (int idx) const`
- Get the start parameter value for the parameter direction idx.*
- virtual `double endparam_u () const`
- virtual `double endparam_v () const`
- `double endparam (int idx) const`
- Get the end parameter value for the parameter direction idx.*
- virtual `void normal (Point &n, double upar, double vpar) const`
- `DirectionCone normalCone (NormalConeMethod method) const`
- virtual `DirectionCone normalCone () const`
- virtual `DirectionCone tangentCone (bool pardir_is_u) const`
- virtual `CompositeBox compositeBox () const`
- `SplineSurface * normal () const`
- Not yet implemented.*
- `SplineSurface * normalSurface () const`
- `SplineSurface * derivSurface (int ider1, int ider2) const`
- `SplineSurface * subSurface (double from_upar, double from_vpar, double to_upar, double to_vpar, double fuzzy=DEFAULT_PARAMETER_EPSILON) const`
- virtual `std::vector< shared_ptr< ParamSurface > > subSurfaces (double from_upar, double from_vpar, double to_upar, double to_vpar, double fuzzy=DEFAULT_PARAMETER_EPSILON) const`
- virtual `SplineSurface * mirrorSurface (const Point &pos, const Point &norm) const`
- Mirror a surface around a specified plane.*
- virtual `void closestPoint (const Point &pt, double &clo_u, double &clo_v, Point &clo_pt, double &clo_dist, double epsilon, const RectDomain *domain_of_interest=NULL, double *seed=0) const`
- virtual `void closestBoundaryPoint (const Point &pt, double &clo_u, double &clo_v, Point &clo_pt, double &clo_dist, double epsilon, const RectDomain *rd=NULL, double *seed=0) const`
- `double appendSurface (ParamSurface *sf, int join_dir, int continuity, double &dist, bool repair=true)`
- void `appendSurface (ParamSurface *sf, int join_dir, bool repair=true)`

- virtual void [getBoundaryInfo](#) ([Point](#) &pt1, [Point](#) &pt2, [double](#) epsilon, [SplineCurve](#) \*&cv, [SplineCurve](#) \*&crosscv, [double](#) knot\_tol=1e-05) const
- void [getBoundaryInfo](#) ([double](#) par1, [double](#) par2, int bindex, [SplineCurve](#) \*&cv, [SplineCurve](#) \*&crosscv, [double](#) knot\_tol=1e-05) const
- void [getBoundaryIdx](#) ([Point](#) &pt1, [Point](#) &pt2, [double](#) epsilon, int &bindex, [double](#) &par1, [double](#) &par2, [double](#) knot\_tol=1e-05) const
- void [getBoundaryIdx](#) ([Point](#) &pt1, [double](#) epsilon, int &bindex, [double](#) knot\_tol=1e-05) const  
*Given one point on the surface boundary, find the index number of the.*
- virtual void [turnOrientation](#) ()  
*Turns the direction of the normal of the surface.*
- virtual void [swapParameterDirection](#) ()  
*Swaps the two parameter directions.*
- virtual void [reverseParameterDirection](#) ([bool](#) direction\_is\_u)
- int [boundaryIndex](#) ([Point](#) &param\_pt1, [Point](#) &param\_pt2) const
- virtual [double](#) [area](#) ([double](#) tol) const
- const [BsplineBasis](#) & [basis\\_u](#) () const
- const [BsplineBasis](#) & [basis\\_v](#) () const
- [BsplineBasis](#) & [basis\\_u](#) ()
- [BsplineBasis](#) & [basis\\_v](#) ()
- const [BsplineBasis](#) & [basis](#) (int i) const
- int [numCoefs\\_u](#) () const
- int [numCoefs\\_v](#) () const
- int [numElem](#) () const  
*Query the number of elements in the [SplineSurface](#).*
- int [numElem](#) (int pdir) const
- int [order\\_u](#) () const
- int [order\\_v](#) () const
- [bool](#) [rational](#) () const
- std::vector< [double](#) >::iterator [coefs\\_begin](#) ()
- std::vector< [double](#) >::iterator [coefs\\_end](#) ()
- std::vector< [double](#) >::const\_iterator [coefs\\_begin](#) () const
- std::vector< [double](#) >::const\_iterator [coefs\\_end](#) () const
- std::vector< [double](#) >::iterator [rcoefs\\_begin](#) ()
- std::vector< [double](#) >::iterator [rcoefs\\_end](#) ()
- std::vector< [double](#) >::const\_iterator [rcoefs\\_begin](#) () const
- std::vector< [double](#) >::const\_iterator [rcoefs\\_end](#) () const
- std::vector< [double](#) >::iterator [ctrl\\_begin](#) ()
- std::vector< [double](#) >::iterator [ctrl\\_end](#) ()
- std::vector< [double](#) >::const\_iterator [ctrl\\_begin](#) () const
- std::vector< [double](#) >::const\_iterator [ctrl\\_end](#) () const
- void [replaceCoefficient](#) (int ix, [Point](#) coef)  
*Replace one specified coefficient (local enumeration)*
- void [getWeights](#) (std::vector< [double](#) > &weights) const
- virtual [bool](#) [isDegenerate](#) ([bool](#) &b, [bool](#) &r, [bool](#) &t, [bool](#) &l, [double](#) tolerance) const
- virtual void [getDegenerateCorners](#) (std::vector< [Point](#) > &deg\_corners, [double](#) tol) const  
*Check for paralell and anti paralell partial derivatives in surface corners.*
- virtual void [getCornerPoints](#) (std::vector< std::pair< [Point](#), [Point](#) > > &corners) const
- virtual void [setParameterDomain](#) ([double](#) u1, [double](#) u2, [double](#) v1, [double](#) v2)
- void [insertKnot\\_u](#) ([double](#) apar)
- void [insertKnot\\_u](#) (const std::vector< [double](#) > &new\_knots)
- void [insertKnot\\_v](#) ([double](#) apar)
- void [insertKnot\\_v](#) (const std::vector< [double](#) > &new\_knots)
- void [removeKnot\\_u](#) ([double](#) tpar)
- void [removeKnot\\_v](#) ([double](#) tpar)

- void `makeBernsteinKnotsU` ()
- void `makeBernsteinKnotsV` ()
- void `makeSurfaceKRegular` ()
  - Ensure k-regularity of this surface in both parameter directions.*
- int `numberOfPatches_u` () const
- int `numberOfPatches_v` () const
- double `knotSpan` (int paddir, int iknot) const
- void `raiseOrder` (int raise\_u, int raise\_v)
- `SplineCurve * constParamCurve` (double parameter, bool paddir\_is\_u) const
- void `constParamCurve` (double parameter, bool paddir\_is\_u, `SplineCurve *&cv`, `SplineCurve *&crosscv`) const
- void `getConstParamCurves` (const std::vector< double > &params\_u, const std::vector< double > &params\_v, std::vector< shared\_ptr< `SplineCurve` > > &curves\_u, std::vector< shared\_ptr< `SplineCurve` > > &curves\_v)
- virtual std::vector< shared\_ptr< `ParamCurve` > > `constParamCurves` (double parameter, bool paddir\_is\_u) const
- `SplineCurve * edgeCurve` (int ccw\_edge\_number) const
- void `interpolate` (`Interpolator &interpolator1`, `Interpolator &interpolator2`, int num\_points1, int num\_points2, int dim, const double \*param1\_start, const double \*param2\_start, const double \*data\_start)
- virtual void `evalGrid` (int num\_u, int num\_v, double umin, double umax, double vmin, double vmax, std::vector< double > &points, double nodata\_val=-9999) const
- void `gridEvaluator` (int num\_u, int num\_v, std::vector< double > &points, std::vector< double > &normals, std::vector< double > &param\_u, std::vector< double > &param\_v, bool normalize=true) const
- void `gridEvaluator` (int num\_u, int num\_v, std::vector< double > &points, std::vector< double > &param\_u, std::vector< double > &param\_v) const
- void `gridEvaluator` (int num\_u, int num\_v, std::vector< double > &points, std::vector< double > &param\_u, std::vector< double > &param\_v, double start\_u, double end\_u, double start\_v, double end\_v) const
- void `gridEvaluator` (std::vector< double > &points, const std::vector< double > &param\_u, const std::vector< double > &param\_v) const
- void `gridEvaluator` (const std::vector< double > &params\_u, const std::vector< double > &params\_v, std::vector< double > &points, std::vector< double > &derivs\_u, std::vector< double > &derivs\_v, bool evaluate\_from\_right=true) const
- void `computeBasis` (double param[], std::vector< double > &basisValues, std::vector< double > &basisDerivs\_u, std::vector< double > &basisDerivs\_v, bool evaluate\_from\_right=true) const
- void `computeBasis` (const std::vector< double >::const\_iterator &bas\_vals\_u, const std::vector< double >::const\_iterator &bas\_vals\_v, int left\_u, int left\_v, std::vector< double > &basisValues, std::vector< double > &basisDerivs\_u, std::vector< double > &basisDerivs\_v) const
- void `computeBasisGrid` (const `Dvector` &param\_u, const `Dvector` &param\_v, `Dmatrix` &basisValues) const
- void `computeBasis` (double param\_u, double param\_v, `BasisPtsSf` &result) const
- void `computeBasis` (double param\_u, double param\_v, `BasisDerivsSf` &result, bool evaluate\_from\_right=true) const
- void `computeBasis` (double param\_u, double param\_v, `BasisDerivsSf2` &result, bool evaluate\_from\_right=true) const
- void `computeBasisGrid` (const `Dvector` &param\_u, const `Dvector` &param\_v, std::vector< `BasisPtsSf` > &result) const
- void `computeBasisGrid` (const `Dvector` &param\_u, const `Dvector` &param\_v, `Dmatrix` &basisValues, `Dmatrix` &basisDerivs\_u, `Dmatrix` &basisDerivs\_v, bool evaluate\_from\_right=true) const
- void `computeBasisGrid` (const `Dvector` &param\_u, const `Dvector` &param\_v, std::vector< `BasisDerivsSf` > &result, bool evaluate\_from\_right=true) const
- void `computeBasisGrid` (const `Dvector` &param\_u, const `Dvector` &param\_v, std::vector< `BasisDerivsSf2` > &result, bool evaluate\_from\_right=true) const
- virtual double `nextSegmentVal` (int dir, double par, bool forward, double tol) const
- bool `replaceBoundaryCurve` (int bd\_nmb, shared\_ptr< `SplineCurve` > bd\_crv, bool unify=true)
- virtual bool `isSpline` () const
  - Check if the surface is of type spline.*
- void `deform` (const std::vector< double > &vec, int vdim=0)

Adds the given deformation vector to the coefficients.

- void `add` (const `SplineSurface` \*other, double tol=1.0e-10)
- void `representAsRational` ()

Ensure that the current surface is represented as a rational surface.

- double `setAvBdWeight` (double wgt, int pdir, bool at\_start)
- virtual bool `isAxisRotational` (Point &centre, Point &axis, Point &vec, double &angle)
- virtual bool `isPlanar` (Point &normal, double tol)
- virtual int `ElementOnBoundary` (int elem\_ix, double eps)
- virtual int `ElementBoundaryStatus` (int elem\_ix, double eps)
- std::vector< shared\_ptr< `SplineCurve` > > `getElementBdParCvs` (int elem\_ix, double elem\_par[])
- bool `isElementarySurface` ()

Query if the surface was generated from an `ElementarySurface`.

- shared\_ptr< `ElementarySurface` > `getElementarySurface` ()
- void `setElementarySurface` (shared\_ptr< `ElementarySurface` > elsurf)
- bool `checkElementarySurface` ()
- void `pointsGrid` (int m1, int m2, int derivs, const double \*basisvals1, const double \*basisvals2, const int \*knotint1, const int \*knotint2, double \*result, double \*normals=0) const

## Static Public Member Functions

- static `ClassType classType` ()

## Additional Inherited Members

### 29.485.1 Detailed Description

`SplineSurface` provides methods for storing, reading and manipulating rational and non-rational B-spline surfaces.

Non-rational B-spline surfaces represented on the form

$$\sum_{i=1}^{n_1} \sum_{j=1}^{n_2} P_{i,j} B_{i,o_1}(u) B_{j,o_2}(v),$$

where the B-spline coefficients are stored in a vector called `coefs`. The `coefs` are stored as

$$P_{0,0}, P_{1,0}, \dots, P_{n_1,n_2},$$

where  $P_{i,j}$  is represented as `dim doubles`.

NURBS surfaces are represented on the form

$$\frac{\sum_{i=1}^{n_1} \sum_{j=1}^{n_2} w_{i,j} P_{i,j} B_{i,o_1}(u) B_{j,o_2}(v)}{\sum_{i=1}^{n_1} \sum_{j=1}^{n_2} w_{i,j} B_{i,o_1}(u) B_{j,o_2}(v)}$$

where  $B_{i,o}$  is the  $i$ 'th non-rational B-spline of order  $o$ . The Projected coefficients are stored in the `coefs` vector, i.e.

$$w_{0,0} \cdot P_{0,0}, w_{1,0} \cdot P_{1,0}, \dots, w_{n_1,n_2} \cdot P_{n_1,n_2}.$$

In addition the coefficients for the surface in projective space are kept, i.e

$$P_{0,0}, w_{0,0}, P_{1,0}, w_{1,0}, \dots, P_{n_1,n_2}, w_{n_1,n_2}.$$

With this representation surface is in the convex hull of the coefficients stored in `coefs` both for rational and non rational surfaces.

Definition at line 174 of file `SplineSurface.h`.

## 29.485.2 Member Typedef Documentation

### 29.485.2.1 typedef std::vector<Dvector> Go::SplineSurface::Dmatrix

Convenience to be used in computations of basis grids.

Definition at line 1103 of file SplineSurface.h.

### 29.485.2.2 typedef std::vector<double> Go::SplineSurface::Dvector

Convenience to be used in computations of basis grids.

Definition at line 1101 of file SplineSurface.h.

## 29.485.3 Member Enumeration Documentation

### 29.485.3.1 enum Go::SplineSurface::NormalConeMethod

Enumerates the method for computing the normal cone.

Enumerator

***SederbergMeyers***

***SMCornersFirst***

***sislBased***

Definition at line 400 of file SplineSurface.h.

## 29.485.4 Constructor & Destructor Documentation

### 29.485.4.1 Go::SplineSurface::SplineSurface ( ) `[inline]`

Creates an uninitialized [SplineSurface](#), which can only be assigned to or read(...) into.

Definition at line 179 of file SplineSurface.h.

### 29.485.4.2 template<typename Randomlterator1 , typename Randomlterator2 , typename Randomlterator3 > Go::SplineSurface::SplineSurface ( int *number1*, int *number2*, int *order1*, int *order2*, Randomlterator1 *knot1start*, Randomlterator2 *knot2start*, Randomlterator3 *coefsstart*, int *dim*, bool *rational* = false ) `[inline]`

Create a [SplineSurface](#) by explicitly providing all spline-related information.

Parameters

<i>number1</i>	number of control points along the u-parameter
<i>number2</i>	number of control points along the v-parameter
<i>order1</i>	<a href="#">BsplineBasis</a> order along the u-parameter
<i>order2</i>	<a href="#">BsplineBasis</a> order along the v-parameter
<i>knot1start</i>	pointer to the array describing the knotvector for the u-parameter
<i>knot2start</i>	pointer to the array describing the knotvector for the v-parameter
<i>coefsstart</i>	pointer to the array where the control points are consecutively stored. The storage order is such

Definition at line 210 of file SplineSurface.h.

```
29.485.4.3 template<typename RandomIterator > Go::SplineSurface::SplineSurface (const BsplineBasis & basis_u,
const BsplineBasis & basis_v, RandomIterator coefsstart, int dim, bool rational = false) [inline]
```

Create a [SplineSurface](#) by explicitly providing all spline-related information.

#### Parameters

<i>basis_u</i>	the <a href="#">BsplineBasis</a> to be used along the u-direction
<i>basis_v</i>	the <a href="#">BsplineBasis</a> to be used along the v-direction
<i>coefsstart</i>	pointer to the array where the control points are consecutively stored. The storage order is such that control points along the u-parameter have the shortest stride (stored right after each other). If the surface is rational, pay attention to the comments below.
<i>dim</i>	dimension of the space in which the surface lies (usually 3).
<i>rational</i>	Specify whether the surface is rational or not. If the surface is rational, coefficients must be in the following format: wP1 wP2 .... wPdim w. I.e. a (dim+1)-dimensional form. (This is the same form that is used within SISL).

Definition at line 254 of file SplineSurface.h.

```
29.485.4.4 virtual Go::SplineSurface::~~SplineSurface () [virtual]
```

Virtual destructor, enables safe inheritance.

## 29.485.5 Member Function Documentation

```
29.485.5.1 void Go::SplineSurface::add (const SplineSurface * other, double tol = 1.0e-10)
```

Add coefficients from another surface. Weights are not summed for rational cases Nothing is done and exception is raised if

- Spline spaces are different in any parameter direction (order or knot vectors are not identical)
- The geometry spaces of the surfaces have different dimension
- One surface is rational while the other is not
- The weights are not equal within a given tolerance (only if surfaces are rational)

#### Parameters

<i>other</i>	The other spline surface with coefficients to be added into this surface
<i>tol</i>	tolerance used to test if weights are considered equal in rational case

29.485.5.2 `virtual std::vector<CurveLoop> Go::SplineSurface::allBoundaryLoops ( double degenerate_epsilon = DEFAULT_SPACE_EPSILON ) const` [virtual]

Returns the anticlockwise outer boundary loop of the surface, together with clockwise loops of any interior boundaries, such that the surface always is 'to the left of' the loops.

#### Parameters

<i>degenerate_epsilon</i>	edges whose length is smaller than this value are ignored.
---------------------------	------------------------------------------------------------

#### Returns

a vector containing CurveLoops. The first of these describe the outer boundary of the surface (clockwise), whereas the others describe boundaries of interior holes (clockwise).

Implements [Go::ParamSurface](#).

29.485.5.3 `double Go::SplineSurface::appendSurface ( ParamSurface * sf, int join_dir, int continuity, double & dist, bool repar = true )`

Appends a surface to the end of 'this' surface along a specified parameter direction. The knotvector and control point grid of 'this' surface will be extended, and the degree will be raised if necessary. Note that there *will* be side-effects on the other surface; its order might be raised, its knotvector will become k-regular and reparametrized, one of its edges will be moved to coincide with an edge of 'this' surface, etc.

#### Parameters

<i>sf</i>	the surface to append to 'this' surface
<i>join_dir</i>	the parameter direction along which to extend 'this' surface by appending the 'sf' surface. If it is equal to 1, we will extend the surface along the u-parameter; if it equals 2, we will extend the surface along the v-parameter.
<i>continuity</i>	the desired level of continuity at the transition between the two surfaces (can be from -1 to order()). The higher the value, the more the surfaces will have to be locally modified around the seam.
<i>dist</i>	upon function return, this variable will hold the estimated maximum distortion after the 'smoothing' of the seam in order to achieve the desired continuity.
<i>repar</i>	The parametrization of the concerned parameter in the other surface will <i>always</i> be shifted so that it starts where the parametrization of 'this' surface ends. However, if 'repar' is set to 'true', it will also be <i>scaled</i> as a function of position of control points close to the transition.
<i>return</i>	Set in the rational case. Maximum adjustment of input weight

29.485.5.4 `void Go::SplineSurface::appendSurface ( ParamSurface * sf, int join_dir, bool repar = true )`

Short hand function to call [appendSurface](#) with  $C^1$  continuity.

#### Parameters

<i>sf</i>	the surface to append to 'this' surface
<i>join_dir</i>	the parameter direction along which to extend 'this' surface by appending the 'sf' surface. If it is equal to 1, we will extend the surface along the u-parameter; if it equals 2, we will extend the surface along the v-parameter.

## Parameters

<i>repar</i>	The parametrization of the concerned parameter in the other surface will <i>always</i> be shifted so that it starts where the parametrization of 'this' surface ends. However, if 'repar' is set to 'true', it will also be <i>scaled</i> as a function of position of control points close to the transition.
--------------	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

29.485.5.5 `virtual double Go::SplineSurface::area ( double tol ) const [virtual]`

Compute the total area of this surface up to some tolerance

## Parameters

<i>tol</i>	the relative tolerance when approximating the area, i.e. stop iteration when error becomes smaller than $tol/(surface\ area)$
------------	-------------------------------------------------------------------------------------------------------------------------------

## Returns

the area calculated

Implements [Go::ParamSurface](#).

29.485.5.6 `virtual SplineSurface* Go::SplineSurface::asSplineSurface ( ) [inline],[virtual]`

Return the spline surface represented by this surface, if any.

Reimplemented from [Go::ParamSurface](#).

Definition at line 312 of file SplineSurface.h.

29.485.5.7 `const BsplineBasis& Go::SplineSurface::basis ( int i ) const [inline]`

get one of the BsplineBasises of the surface

## Parameters

<i>i</i>	specify whether to return the <a href="#">BsplineBasis</a> for the first parameter (0), or for the second parameter (1).
----------	--------------------------------------------------------------------------------------------------------------------------

## Returns

const reference to the requested [BsplineBasis](#).

Definition at line 659 of file SplineSurface.h.

29.485.5.8 `const BsplineBasis& Go::SplineSurface::basis_u ( ) const [inline]`

get a const reference to the [BsplineBasis](#) for the first parameter



**Returns**

const reference to the [BsplineBasis](#) for the first parameter

Definition at line 637 of file SplineSurface.h.

**29.485.5.9 BsplineBasis& Go::SplineSurface::basis\_u( ) [inline]**

get a reference to the [BsplineBasis](#) for the first parameter

**Returns**

reference to the [BsplineBasis](#) for the first parameter

Definition at line 647 of file SplineSurface.h.

**29.485.5.10 const BsplineBasis& Go::SplineSurface::basis\_v( ) const [inline]**

get a const reference to the [BsplineBasis](#) for the second parameter

**Returns**

const reference to the [BsplineBasis](#) for the second parameter

Definition at line 642 of file SplineSurface.h.

**29.485.5.11 BsplineBasis& Go::SplineSurface::basis\_v( ) [inline]**

get a reference to the [BsplineBasis](#) for the second parameter

**Returns**

reference to the [BsplineBasis](#) for the second parameter

Definition at line 652 of file SplineSurface.h.

**29.485.5.12 int Go::SplineSurface::boundaryIndex ( Point & param\_pt1, Point & param\_pt2 ) const**

If the two points pt1 and pt2 are on the same edge the edge index is returned, otherwise -1 is returned. Edges are counted anticlockwise, and starts with the edge (upar, 0). find whether two points are lying on the same surface boundary; if this is the case, return the index of the boundary (as specified in [getBoundaryIdx\(\)](#)), else return 1

**Returns**

The boundary index on which the points lie, if they are found to both lie on the same boundary. In the opposite case, -1 is returned.

29.485.5.13 `virtual BoundingBox Go::SplineSurface::boundingBox ( ) const [virtual]`

Return the object's bounding box.

Implements [Go::GeomObject](#).

29.485.5.14 `bool Go::SplineSurface::checkElementarySurface ( )`

Check to see if `this` corresponds to the [ElementarySurface](#) set by [setElementarySurface\(\)](#). NOTE: Not yet implemented!

29.485.5.15 `static ClassType Go::SplineSurface::classType ( ) [inline],[static]`

Definition at line 301 of file `SplineSurface.h`.

29.485.5.16 `virtual SplineSurface* Go::SplineSurface::clone ( ) const [inline],[virtual]`

make a clone of this surface and return a pointer to it (user is responsible for clearing up memory afterwards).

Returns

pointer to cloned object

Implements [Go::ParamSurface](#).

Definition at line 307 of file `SplineSurface.h`.

29.485.5.17 `virtual void Go::SplineSurface::closestBoundaryPoint ( const Point & pt, double & clo_u, double & clo_v, Point & clo_pt, double & clo_dist, double epsilon, const RectDomain * rd = NULL, double * seed = 0 ) const [virtual]`

Iterates to the closest point to `pt` on the boundary of the surface.

See also

[closestPoint\(\)](#)

Implements [Go::ParamSurface](#).

29.485.5.18 `virtual Point Go::SplineSurface::closestInDomain ( double u, double v ) const [virtual]`

Fetch the parameter value in the parameter domain of the surface closest to the parameter pair (u,v)

Implements [Go::ParamSurface](#).

29.485.5.19 `virtual void Go::SplineSurface::closestPoint ( const Point & pt, double & clo_u, double & clo_v, Point & clo_pt, double & clo_dist, double epsilon, const RectDomain * domain_of_interest = NULL, double * seed = 0 ) const [virtual]`

Iterates to the closest point to `pt` on the surface.

## Parameters

<i>pt</i>	the point to find the closest point to
<i>clo_u</i>	u parameter of the closest point
<i>clo_v</i>	v parameter of the closest point
<i>clo_pt</i>	the geometric position of the closest point
<i>clo_dist</i>	the distance between pt and clo_pt
<i>epsilon</i>	parameter tolerance (will in any case not be higher than sqrt(machine_precision) x magnitude of solution)
<i>domain_of_interest</i>	pointer to parameter domain in which to search for closest point. If a NULL pointer is used, the entire surface is searched.
<i>seed</i>	pointer to parameter values where iteration starts.

Reimplemented from [Go::ParamSurface](#).

**29.485.5.20** `std::vector<double>::iterator Go::SplineSurface::coefs_begin ( ) [inline]`

Get an iterator to the start of the internal array of non-rational control points.

## Returns

an (nonconst) iterator to the start of the internal array of non-rational control points

Definition at line 705 of file SplineSurface.h.

**29.485.5.21** `std::vector<double>::const_iterator Go::SplineSurface::coefs_begin ( ) const [inline]`

Get a const iterator to the start of the internal array of non-rational control points.

## Returns

a const iterator to the start of the internal array of non-rational control points.

Definition at line 719 of file SplineSurface.h.

**29.485.5.22** `std::vector<double>::iterator Go::SplineSurface::coefs_end ( ) [inline]`

Get an iterator to the one-past-end position of the internal array of non-rational control points

## Returns

an (nonconst) iterator to the one-past-end position of the internal array of non-rational control points

Definition at line 712 of file SplineSurface.h.

29.485.5.23 `std::vector<double>::const_iterator Go::SplineSurface::coefs_end ( ) const` `[inline]`

Get a const iterator to the one-past-end position of the internal array of non-rational control points.

#### Returns

a const iterator to the one-past-end position of the internal array of non-rational control points.

Definition at line 726 of file SplineSurface.h.

29.485.5.24 `virtual CompositeBox Go::SplineSurface::compositeBox ( ) const` `[virtual]`

Creates a composite box enclosing the surface. The composite box consists of an inner and an edge box. The inner box is supposed to be made from the interior of the surface, while the edge box is made from the boundary curves. The default implementation simply makes both boxes identical to the regular bounding box.

#### Returns

the [CompositeBox](#) of the surface, as specified above

Reimplemented from [Go::ParamSurface](#).

29.485.5.25 `void Go::SplineSurface::computeBasis ( double param[], std::vector< double > & basisValues, std::vector< double > & basisDerivs_u, std::vector< double > & basisDerivs_v, bool evaluate_from_right = true ) const`

Evaluate positions and first derivatives of all basis values in a given parameter pair For non-rationals this is an interface to [BsplineBasis::computeBasisValues](#) where the basis values in each parameter direction are multiplied to compute  $B_i(u) \cdot B_j(v)$ , for rationals the routine evaluates the rational basis functions, i.e. the basis functions are divided by the denominator of the surface in the given parameter direction

#### Parameters

<i>param</i>	the parameter pair in which to compute
<i>basisValues</i>	the value of all basis functions, size equal to $(\text{degree}_u+1) \cdot (\text{degree}_v+1)$
<i>basisDerivs</i>	the derivative of all basis functions, same size as previous
<i>evaluate_from_right</i>	specifies directional derivatives, true=right, false=left, same behaviour for both parameter directions

29.485.5.26 `void Go::SplineSurface::computeBasis ( const std::vector< double >::const_iterator & bas_vals_u, const std::vector< double >::const_iterator & bas_vals_v, int left_u, int left_v, std::vector< double > & basisValues, std::vector< double > & basisDerivs_u, std::vector< double > & basisDerivs_v ) const`

Evaluate positions and first derivatives of all basis values, when the B-splines and their derivatives have all been precalculated and are given as input

## Parameters

<i>bas_vals</i> <sub>↔</sub> <i>_u</i>	the basis values and derivatives in first parameter direction
<i>bas_vals</i> <sub>↔</sub> <i>_v</i>	the basis values and derivatives in second parameter direction
<i>left_u</i>	index of first non-zero interval in first parameter direction
<i>left_v</i>	index of first non-zero interval in second parameter direction
<i>basisValues</i>	the value of all basis functions, size equal to (degree_u+1)*(degree_v+1)
<i>basisDerivs</i>	the derivative of all basis functions, same size as previous

29.485.5.27 void Go::SplineSurface::computeBasis ( double *param\_u*, double *param\_v*, BasisPtsSf & *result* ) const

Compute basis values (position) in the parameter (*param\_u*,*param\_v*). Store result in a [BasisPtsSf](#) entity

29.485.5.28 void Go::SplineSurface::computeBasis ( double *param\_u*, double *param\_v*, BasisDerivsSf & *result*, bool *evaluate\_from\_right* = true ) const

Compute basis values (position and 1. derivatives) in the parameter (*param\_u*,*param\_v*). Store result in a [BasisDerivsSf](#) entity

29.485.5.29 void Go::SplineSurface::computeBasis ( double *param\_u*, double *param\_v*, BasisDerivsSf2 & *result*, bool *evaluate\_from\_right* = true ) const

Compute basis values (position and 1. and 2. derivatives) in the parameter (*param\_u*,*param\_v*). Store result in a [BasisDerivsSf2](#) entity

29.485.5.30 void Go::SplineSurface::computeBasisGrid ( const Dvector & *param\_u*, const Dvector & *param\_v*, Dmatrix & *basisValues* ) const

Evaluate positions of all basis values in a specified grid. For non-rationals this is an interface to [BsplineBasis::computeBasisValues](#) where the basis values in each parameter direction are multiplied to compute  $B_i(u)*B_j(v)$ , for rationals the routine evaluates the rational basis functions, i.e. the basis functions are divided by the denominator of the surface in the given parameter direction

## Parameters

<i>param_u</i>	this vector holds all the numerical values for the first parameter where evaluation will take place
<i>param_v</i>	this vector holds all the numerical values for the second parameter where evaluation will take place
<i>basisValues</i>	the value of all basis functions. The first matrix dimension will correspond to all evaluation parameters, i.e. the size is <i>param_u</i> .size()* <i>param_v</i> .size(). The second dimension correspond to all coefficients, i.e. <i>nmb_coef_u</i> * <i>nmb_coef_v</i> . Most entries in the matrix will be zero

29.485.5.31 `void Go::SplineSurface::computeBasisGrid ( const Dvector & param_u, const Dvector & param_v, std::vector< BasisPtsSf > & result ) const`

Compute basis grid (position) in the parameter pairs combined from param\_u and param\_v. Store result in a vector of [BasisPtsSf](#).

29.485.5.32 `void Go::SplineSurface::computeBasisGrid ( const Dvector & param_u, const Dvector & param_v, Dmatrix & basisValues, Dmatrix & basisDerivs_u, Dmatrix & basisDerivs_v, bool evaluate_from_right = true ) const`

Evaluate positions and first derivatives of all basis values in a specified grid. For non-rationals this is an interface to [BsplineBasis::computeBasisValues](#) where the basis values in each parameter direction are multiplied to compute  $B_i(u) \cdot B_j(v)$ , for rationals the routine evaluates the rational basis functions, i.e. the basis functions are divided by the denominator of the surface in the given parameter direction

#### Parameters

<i>param_u</i>	this vector holds all the numerical values for the first parameter where evaluation will take place
<i>param_v</i>	this vector holds all the numerical values for the second parameter where evaluation will take place
<i>basisValues</i>	the value of all basis functions. The first matrix dimension will correspond to all evaluation parameters, i.e. the size is <code>param_u.size()*param_v.size()</code> . The second dimension correspond to all coefficients, i.e. <code>nmb_coef_u*nmb_coef_v</code> . Most entries in the matrix will be zero
<i>basisDerivs_u</i>	the derivative of all basis functions, size as above
<i>basisDerivs_v</i>	the derivative of all basis functions
<i>evaluate_from_right</i>	specifies directional derivatives, true=right, false=left

29.485.5.33 `void Go::SplineSurface::computeBasisGrid ( const Dvector & param_u, const Dvector & param_v, std::vector< BasisDerivsSf > & result, bool evaluate_from_right = true ) const`

Compute basis grid (position and 1. derivatives) in the parameter pairs combined from param\_u and param\_v. Store result in a vector of [BasisDerivSf](#).

29.485.5.34 `void Go::SplineSurface::computeBasisGrid ( const Dvector & param_u, const Dvector & param_v, std::vector< BasisDerivsSf2 > & result, bool evaluate_from_right = true ) const`

Compute basis grid (position and 1. and 2. derivatives) in the parameter pairs combined from param\_u and param\_v. Store result in a vector of [BasisDerivSf2](#).

29.485.5.35 `SplineCurve* Go::SplineSurface::constParamCurve ( double parameter, bool pardir_is_u ) const`

Generate and return a [SplineCurve](#) that represents a constant parameter curve on the surface

#### Parameters

<i>parameter</i>	value of the fixed parameter
<i>pardir_is_u</i>	'true' if the first parameter is the running parameter, 'false' otherwise.

## Returns

pointer to a newly constructed [SplineCurve](#) representing the specified constant parameter curve. It is the user's responsibility to delete it when it is no longer needed.

```
29.485.5.36 void Go::SplineSurface::constParamCurve (double parameter, bool pdir_is_u, SplineCurve *& cv, SplineCurve *& crosscv) const
```

Generate and return two SplineCurves, representing a constant parameter curve on the surface as well as its cross-tangent curve.

## Parameters

<i>parameter</i>	value of the fixed parameter
<i>pdir_is_u</i>	'true' if the first parameter is the running parameter, 'false' otherwise.
<i>cv</i>	upon function completion, 'cv' will point to a newly constructed <a href="#">SplineCurve</a> representing the specified constant parameter curve. It is the user's responsibility to delete it when it is no longer needed.
<i>crosscv</i>	upon function completion, 'crosscv' will point to a newly constructed <a href="#">SplineCurve</a> representing the cross-tangent curve of the specified constant parameter curve. It is the user's responsibility to delete it when it is no longer needed.

```
29.485.5.37 virtual std::vector< shared_ptr<ParamCurve> > Go::SplineSurface::constParamCurves (double parameter, bool pdir_is_u) const [virtual]
```

Get the curve(s) obtained by intersecting the surface with one of its constant parameter curves. For surfaces without holes, this will be the parameter curve itself; for surfaces with interior holes this may be a collection of several, disjoint curves.

## Parameters

<i>parameter</i>	parameter value for the constant parameter (either u or v)
<i>pdir_is_u</i>	specify whether the <i>moving</i> parameter (as opposed to the <i>constant</i> parameter) is the first ('true') or the second ('false') one.

## Returns

a vector containing shared pointers to the obtained, newly constructed constant-parameter curves.

Implements [Go::ParamSurface](#).

```
29.485.5.38 virtual RectDomain Go::SplineSurface::containingDomain () const [virtual]
```

Get a rectangular parameter domain that is guaranteed to contain the surface's [parameterDomain\(\)](#). It may be the same. There is no guarantee that this is the smallest domain containing the actual domain.

## Returns

a [RectDomain](#) that is guaranteed to include the surface's total parameter domain.

Implements [Go::ParamSurface](#).

29.485.5.39 `std::vector<double>::iterator Go::SplineSurface::ctrl_begin ( )` [inline]

Get an iterator to the start of the internal array of active control points.

**Returns**

an (nonconst) iterator to the start of the internal array of rational or non-rational control points

Definition at line 761 of file SplineSurface.h.

29.485.5.40 `std::vector<double>::const_iterator Go::SplineSurface::ctrl_begin ( ) const` [inline]

Get a const iterator to the start of the internal array of active control points.

**Returns**

a const iterator to the start of the internal array of rational or non-rational control points.

Definition at line 775 of file SplineSurface.h.

29.485.5.41 `std::vector<double>::iterator Go::SplineSurface::ctrl_end ( )` [inline]

Get an iterator to the one-past-end position of the internal array of active control points

**Returns**

an (nonconst) iterator to the one-past-end position of the internal array of rational or non-rational control points

Definition at line 768 of file SplineSurface.h.

29.485.5.42 `std::vector<double>::const_iterator Go::SplineSurface::ctrl_end ( ) const` [inline]

Get a const iterator to the one-past-end position of the internal array of active control points.

**Returns**

a const iterator to the one-past-end position of the internal array of rational or non-rational control points.

Definition at line 782 of file SplineSurface.h.

29.485.5.43 `void Go::SplineSurface::deform ( const std::vector< double > & vec, int vdim = 0 )`

Adds the given deformation vector to the coefficients.

29.485.5.44 `SplineSurface* Go::SplineSurface::derivSurface ( int ider1, int ider2 ) const`

Get the derivative surface. Expresses the i,j-th derivative of a spline surface as a spline surface. Ported from sisl routine s1386.



## Parameters

<i>ider1</i>	what derivative to use along the u parameter
<i>ider2</i>	what derivative to use along the v parameter

## Returns

pointer to a newly constructed [SplineSurface](#) which is the derivative surface. User assumes ownership of this object, and is responsible for destroying it.

**29.485.5.45** `virtual int Go::SplineSurface::dimension ( ) const` `[virtual]`

Return the dimension of the space in which the object lies (usually 2 or 3)

Implements [Go::GeomObject](#).

**29.485.5.46** `SplineCurve* Go::SplineSurface::edgeCurve ( int ccw_edge_number ) const`

Here `edge_number` means: 0 -> bottom edge, 1 -> right edge, 2 -> top edge and 3 -> left edge. The returned pointer is new'ed and the responsibility of the caller. You should either delete it, or manage it with a smart pointer. Generate and return a [SplineCurve](#) representing one of the surface's four edges.

## Parameters

<i>ccw_edge_number</i>	indicates which edge the user requests.
------------------------	-----------------------------------------

## Returns

A pointer to a newly constructed [SplineCurve](#) representing the requested edge. It is the user's responsibility to delete it when it is no longer needed.

**29.485.5.47** `virtual int Go::SplineSurface::ElementBoundaryStatus ( int elem_ix, double eps )` `[inline],[virtual]`

Check if a polynomial element (for spline surfaces) intersects the (trimming) boundaries of this [ftSurface](#), is inside or outside

## Parameters

<i>elem_ix</i>	Element index counted according to distinct knot values. Sequence of coordinates: x runs fastest, then y
<i>eps</i>	Intersection tolerance

## Returns

-1: Not a spline surface or element index out of range 0: Outside trimmed volume 1: On boundary (intersection with boundary found) 2: Internal to trimmed surfaces Note that a touch with the boundaries of the underlying surface is not considered a boundary intersection while touching a trimming curve is seen as an intersection

Reimplemented from [Go::ParamSurface](#).

Definition at line 1274 of file SplineSurface.h.

**29.485.5.48** `virtual int Go::SplineSurface::ElementOnBoundary ( int elem_ix, double eps ) [inline], [virtual]`

Check if a polynomial element (for spline surfaces) intersects the (trimming) boundaries of this surface

#### Parameters

<i>elem_ix</i>	Element index counted according to distinct knot values. Sequence of coordinates: x runs fastest, then y
<i>eps</i>	Intersection tolerance

#### Returns

-1: Not a spline surface or element index out of range  
 0: Not on boundary or touching a boundary curve  
 1: On boundary (intersection with boundary found)  
 Note that a touch with the boundaries of the underlying surfaces is not considered a boundary intersection while touching a trimming curve is seen as an intersection

Reimplemented from [Go::ParamSurface](#).

Definition at line 1257 of file SplineSurface.h.

**29.485.5.49** `double Go::SplineSurface::endparam ( int idx ) const [inline]`

Get the end parameter value for the parameter direction *idx*.

Definition at line 388 of file SplineSurface.h.

**29.485.5.50** `virtual double Go::SplineSurface::endparam_u ( ) const [virtual]`

Get the end value for the u-parameter

#### Returns

the end value for the u-parameter

**29.485.5.51** `virtual double Go::SplineSurface::endparam_v ( ) const [virtual]`

Get the end value for the v-parameter

#### Returns

the end value for the v-parameter

29.485.552 virtual void Go::SplineSurface::evalGrid ( int num\_u, int num\_v, double umin, double umax, double vmin, double vmax, std::vector< double > & points, double nodata\_val = -9999 ) const [virtual]

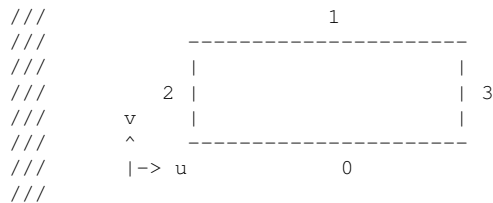
Evaluate points in a grid The nodata value is applicable for bounded surfaces and will not be used in this context

Reimplemented from [Go::ParamSurface](#).

29.485.553 void Go::SplineSurface::getBoundaryIdx ( Point & pt1, Point & pt2, double epsilon, int & bindex, double & par1, double & par2, double knot\_tol = 1e-05 ) const

Given two points on the surface boundary, find the number of the corresponding boundary and the curve parameter of the closest points on this surface boundary.

Ordering of boundaries:



**Parameters**

pt1	point in geometry space
pt2	point in geometry space
epsilon	geometric tolerance
bindex	if the two points are on a common boundary, the index of the boundary, otherwise -1.
par1	if bindex is 0 or 1, the u parameter of pt1, otherwise the v parameter.
par2	if bindex is 0 or 1, the u parameter of pt2, otherwise the v parameter.
knot_tol	tolerance used when working with the knot-vector, to specify how close a parameter value must be to a knot in order to be considered 'on' the knot.

29.485.554 void Go::SplineSurface::getBoundaryIdx ( Point & pt1, double epsilon, int & bindex, double knot\_tol = 1e-05 ) const

Given one point on the surface boundary, find the index number of the.

**Parameters**

pt1	the point we want to find the boundary for
epsilon	the geometric distance the point can have from a boundary and still be considered as laying on the boundary.
bindex	the index of the boundary on which the point lies. If the point was found not to lie on any boundary, the value will be -1.
knot_tol	tolerance used when working with the knot-vector, to specify how close a parameter value must be to a knot in order to be considered 'on' the knot.

29.485.555 `virtual void Go::SplineSurface::getBoundaryInfo ( Point & pt1, Point & pt2, double epsilon, SplineCurve *& cv, SplineCurve *& crosscv, double knot_tol = 1e-05 ) const` [virtual]

Get the boundary curve segment between two points on the boundary, as well as the cross-tangent curve. If the given points are not positioned on the same boundary (within a certain tolerance), no curves will be created.

#### Parameters

<i>pt1</i>	the first point on the boundary, given by the user
<i>pt2</i>	the second point on the boundary, given by the user
<i>epsilon</i>	the tolerance used when determining whether the given points are lying on a boundary, and if they do, whether they both lie on the <i>same</i> boundary.
<i>cv</i>	upon return, this will point to a newly created curve representing the boundary curve between 'pt1' and 'pt2'. The user assumes ownership of this object and is responsible for its deletion. No curve is created if the given points are not found to lie on the same boundary.
<i>crosscv</i>	upon return, this will point to a newly created curve representing the cross-boundary curve between 'pt1' and 'pt2'. The user assumes ownership of this object and is responsible for its deletion. The direction is outwards from the surface. No curve is created if the given points are not found to lie on the same boundary.
<i>knot_tol</i>	tolerance used when working with the knot-vector, to specify how close a parameter value must be to a knot in order to be considered 'on' the knot.

Implements [Go::ParamSurface](#).

29.485.556 `void Go::SplineSurface::getBoundaryInfo ( double par1, double par2, int bdindex, SplineCurve *& cv, SplineCurve *& crosscv, double knot_tol = 1e-05 ) const`

Get the boundary curve and the outward cross tangent curve between two parameter values on a given boundary.

#### Parameters

<i>par1</i>	first parameter along the boundary
<i>par2</i>	second parameter along the boundary
<i>bdindex</i>	index of the boundary in question. The boundaries are indexed as in <a href="#">getBoundaryIdx()</a> .
<i>cv</i>	returns a pointer to a generated <a href="#">SplineCurve</a> representing the boundary between the given parameters. The user assumes ownership and is responsible for deletion.
<i>crosscv</i>	returns a pointer to a generated <a href="#">SplineCurve</a> representing the cross tangent curve between the given parameters on the boundary. The direction is outwards from the surface. The user assumes ownership and is responsible for deletion.
<i>knot_tol</i>	tolerance used when working with the knot-vector, to specify how close a parameter value must be to a knot in order to be considered 'on' the knot.

29.485.557 `void Go::SplineSurface::getConstParamCurves ( const std::vector< double > & params_u, const std::vector< double > & params_v, std::vector< shared_ptr< SplineCurve > > & curves_u, std::vector< shared_ptr< SplineCurve > > & curves_v )`

Generate SplineCurves that represents constant parameter curves on the surface. Do this for both directions

## Parameters

<i>parameter</i>	params_u the parameter values in u-directions for curves running along the v-direction
<i>parameter</i>	params_v the parameter values in v-directions for curves running along the u-direction
<i>curves_u</i>	upon function return, holds constant parameter curves for the u-direction parameters, the curves are running along the v-direction
<i>curves_v</i>	upon function return, holds constant parameter curves for the v-direction parameters, the curves are running along the u-direction

29.485.5.58 `virtual void Go::SplineSurface::getCornerPoints ( std::vector< std::pair< Point, Point > > & corners ) const`  
[virtual]

Return surface corners, geometric and parametric points in that sequence

Implements [Go::ParamSurface](#).

29.485.5.59 `virtual void Go::SplineSurface::getDegenerateCorners ( std::vector< Point > & deg_corners, double tol ) const`  
[virtual]

Check for paralell and anti paralell partial derivatives in surface corners.

Implements [Go::ParamSurface](#).

29.485.5.60 `shared_ptr<ElementarySurface> Go::SplineSurface::getElementarySurface ( )`

Get shared pointer to [ElementarySurface](#), if it exists. If not, return empty shared pointer.

NOTE: The [ElementarySurface](#) returned by this function should ideally be the one corresponding to the current [SplineSurface](#). However, there is no guarantee for this. One may check to see if this is the case by using [checkElementarySurface\(\)](#).

29.485.5.61 `std::vector<shared_ptr<SplineCurve> > Go::SplineSurface::getElementBdParCvs ( int elem_ix, double elem_par[] )`

Fetch curves in the parameter domain surrounding the specified element elem\_par - parameter values of element boundaries, sequence umin, umax, vmin, vmax

29.485.5.62 `void Go::SplineSurface::getWeights ( std::vector< double > & weights ) const`

Return all weights corresponding to this surface, a non-rational volume has all weights equal to one

29.485.5.63 `void Go::SplineSurface::gridEvaluator ( int num_u, int num_v, std::vector< double > & points, std::vector< double > & normals, std::vector< double > & param_u, std::vector< double > & param_v, bool normalize = true ) const`

Evaluate points and normals on an entire grid, taking computational advantage over calculating all these values simultaneously rather than one-by-one.

## Parameters

<i>num_u</i>	number of values to evaluate along first parameter direction
<i>num_v</i>	number of values to evaluate along second parameter direction
<i>points</i>	upon function return, this vector holds all the evaluated points
<i>normals</i>	upon function return, this vector holds all the evaluated normals
<i>param</i> <sub>↔</sub> <i>_u</i>	upon function return, this vector holds all the numerical values for the first parameter where evaluation has taken place
<i>param</i> <sub>↔</sub> <i>_v</i>	upon function return, this vector holds all the numerical values for the second parameter where evaluation has taken place.
<i>normalize</i>	tells whether the normal vectors should be normalized

```
29.485.5.64 void Go::SplineSurface::gridEvaluator (int num_u, int num_v, std::vector< double > & points, std::vector< double > & param_u, std::vector< double > & param_v) const [inline]
```

Evaluate points on an entire grid, taking computational advantage over calculating all these values simultaneously rather than one-by-one.

## Parameters

<i>num_u</i>	number of values to evaluate along first parameter direction
<i>num_v</i>	number of values to evaluate along second parameter direction
<i>points</i>	upon function return, this vector holds all the evaluated points
<i>param</i> <sub>↔</sub> <i>_u</i>	upon function return, this vector holds all the numerical values for the first parameter where evaluation has taken place
<i>param</i> <sub>↔</sub> <i>_v</i>	upon function return, this vector holds all the numerical values for the second parameter where evaluation has taken place.

Definition at line 1003 of file SplineSurface.h.

```
29.485.5.65 void Go::SplineSurface::gridEvaluator (int num_u, int num_v, std::vector< double > & points, std::vector< double > & param_u, std::vector< double > & param_v, double start_u, double end_u, double start_v, double end_v) const
```

Evaluate points on an entire grid, taking computational advantage over calculating all these values simultaneously rather than one-by-one.

## Parameters

<i>num_u</i>	number of values to evaluate along first parameter direction
<i>num_v</i>	number of values to evaluate along second parameter direction
<i>points</i>	upon function return, this vector holds all the evaluated points
<i>param</i> <sub>↔</sub> <i>_u</i>	upon function return, this vector holds all the numerical values for the first parameter where evaluation has taken place
<i>param</i> <sub>↔</sub> <i>_v</i>	upon function return, this vector holds all the numerical values for the second parameter where evaluation has taken place.
<i>start_u</i>	start value of first parameter range
<i>end_u</i>	end value of first parameter range
<i>start_v</i>	start value of second parameter range
<i>end_v</i>	end value of second parameter range

29.485.5.66 `void Go::SplineSurface::gridEvaluator ( std::vector< double > & points, const std::vector< double > & param_u, const std::vector< double > & param_v ) const`

Evaluate points on an entire grid, taking computational advantage over calculating all these values simultaneously rather than one-by-one.

#### Parameters

<i>points</i>	upon function return, this vector holds all the evaluated points
<i>param_u</i>	this vector holds all the numerical values for the first parameter where evaluation should take place
<i>param_v</i>	this vector holds all the numerical values for the second parameter where evaluation should take place

29.485.5.67 `void Go::SplineSurface::gridEvaluator ( const std::vector< double > & params_u, const std::vector< double > & params_v, std::vector< double > & points, std::vector< double > & derivs_u, std::vector< double > & derivs_v, bool evaluate_from_right = true ) const`

Evaluate points and derivatives on an entire grid, taking computational advantage over calculating all these values simultaneously rather than one-by-one.

#### Parameters

<i>params_u</i>	the values for the first parameter where evaluation takes place
<i>params_v</i>	the values for the second parameter where evaluation takes place
<i>points</i>	upon function return, this vector holds all the evaluated points
<i>derivs_u</i>	upon function return, this vector holds all derivations w.r.t. u
<i>derivs_v</i>	upon function return, this vector holds all derivations w.r.t. v
<i>evaluate_from_right</i>	specifies directional derivatives, true=right, false=left, same behaviour for both parameter directions

29.485.5.68 `virtual bool Go::SplineSurface::inDomain ( double u, double v, double eps = 1.0e-4 ) const`  
[virtual]

Check if a parameter pair lies inside the domain of this surface.

Implements [Go::ParamSurface](#).

29.485.5.69 `virtual int Go::SplineSurface::inDomain2 ( double u, double v, double eps = 1.0e-4 ) const`  
[virtual]

Check if a parameter pair lies inside the domain of this surface return value = 0: outside = 1: internal = 2: at the boundary

Implements [Go::ParamSurface](#).

29.485.5.70 `void Go::SplineSurface::insertKnot_u ( double apar )`

Insert a new knot in the knotvector of the first parameter

## Parameters

<i>apar</i>	the parameter value at which a new knot will be inserted
-------------	----------------------------------------------------------

29.485.5.71 void Go::SplineSurface::insertKnot\_u ( const std::vector< double > & new\_knots )

Insert new knots in the knotvector of the first parameter

## Parameters

<i>new_knots</i>	a vector containing the parameter values of the new knots to insert.
------------------	----------------------------------------------------------------------

29.485.5.72 void Go::SplineSurface::insertKnot\_v ( double apar )

Insert a new knot in the knotvector of the second parameter

## Parameters

<i>apar</i>	the parameter value at which a new knot will be inserted
-------------	----------------------------------------------------------

29.485.5.73 void Go::SplineSurface::insertKnot\_v ( const std::vector< double > & new\_knots )

Insert new knots in the knotvector of the second parameter

## Parameters

<i>new_knots</i>	a vector containing the parameter values of the new knots to insert.
------------------	----------------------------------------------------------------------

29.485.5.74 virtual **ClassType** Go::SplineSurface::instanceType ( ) const [virtual]

Return the class type identifier of a given, derived instance of [GeomObject](#).

Implements [Go::GeomObject](#).

29.485.5.75 void Go::SplineSurface::interpolate ( Interpolator & interpolator1, Interpolator & interpolator2, int num\_points1, int num\_points2, int dim, const double \* param1\_start, const double \* param2\_start, const double \* data\_start )

Remake 'this'surface to interpolate (or approximate) a given point grid.

## Parameters

<i>interpolator1</i>	<a href="#">Interpolator</a> to apply to the first parameter direction
----------------------	------------------------------------------------------------------------



## Parameters

<i>interpolator2</i>	<a href="#">Interpolator</a> to apply to the second parameter direction
<i>num_points1</i>	number of points to interpolate along the first parameter direction
<i>num_points2</i>	number of points to interpolate along the second parameter direction
<i>dim</i>	dimension of the points to interpolate (usually 3)
<i>param1_start</i>	pointer to the start of the input grid's parametrization along the first parameter
<i>param2_start</i>	pointer to the start of the input grid's parametrization along the second parameter
<i>data_start</i>	pointer to the start of the storage array for the data point grid to interpolate

29.485.5.76 `virtual bool Go::SplineSurface::isAxisRotational ( Point & centre, Point & axis, Point & vec, double & angle )`  
`[virtual]`

Check if the surface is axis rotational. Only true if a connection to an axis rotational elementary surface exist

Reimplemented from [Go::ParamSurface](#).

29.485.5.77 `virtual bool Go::SplineSurface::isDegenerate ( bool & b, bool & r, bool & t, bool & l, double tolerance )`  
`const [virtual]`

The order of the edge indicators (bottom, right, top, left) matches the `edge_number` of [edgeCurve\(\)](#). Query whether any of the four boundary curves are degenerate (zero length) within a certain tolerance. In the below, we refer to 'u' as the first parameter and 'v' as the second.

## Parameters

<i>b</i>	'true' upon return of function if the boundary ( $v = v_{\min}$ ) is degenerate
<i>r</i>	'true' upon return of function if the boundary ( $v = v_{\max}$ ) is degenerate
<i>t</i>	'true' upon return of function if the boundary ( $u = u_{\min}$ ) is degenerate
<i>l</i>	'true' upon return of function if the boundary ( $u = u_{\max}$ ) is degenerate
<i>tolerance</i>	boundaries are considered degenerate if their length is shorter than this value, given by the user

## Returns

'true' if at least one boundary curve was found to be degenerate, 'false' otherwise.

Reimplemented from [Go::ParamSurface](#).

29.485.5.78 `bool Go::SplineSurface::isElementarySurface ( )` `[inline]`

Query if the surface was generated from an [ElementarySurface](#).

Definition at line 1286 of file `SplineSurface.h`.

29.485.5.79 `virtual bool Go::SplineSurface::isPlanar ( Point & normal, double tol )` `[virtual]`

This surface is planar if it represents a plane or the spline surface is linear in both parameter direction and planar

Reimplemented from [Go::ParamSurface](#).

29.485.5.80 `virtual bool Go::SplineSurface::isSpline ( ) const [inline],[virtual]`

Check if the surface is of type spline.

Reimplemented from [Go::ParamSurface](#).

Definition at line 1210 of file SplineSurface.h.

29.485.5.81 `double Go::SplineSurface::knotSpan ( int pardir, int iknot ) const`

Returns the size of the knot interval (knot[*iknot*],knot[*iknot*+1]) in the specified direction. An index outside the legal range will result in a zero knot span

29.485.5.82 `void Go::SplineSurface::makeBernsteinKnotsU ( )`

Inserts knots in the u knot vector, such that all knots have multiplicity order

29.485.5.83 `void Go::SplineSurface::makeBernsteinKnotsV ( )`

Inserts knots in the v knot vector, such that all knots have multiplicity order

29.485.5.84 `void Go::SplineSurface::makeSurfaceKRegular ( )`

Ensure k-regularity of this surface in both parameter directions.

29.485.5.85 `virtual SplineSurface* Go::SplineSurface::mirrorSurface ( const Point & pos, const Point & norm ) const [virtual]`

Mirror a surface around a specified plane.

Reimplemented from [Go::ParamSurface](#).

29.485.5.86 `virtual double Go::SplineSurface::nextSegmentVal ( int dir, double par, bool forward, double tol ) const [virtual]`

Determine the parameter value of the start of the 'next segment' from a parameter value, along a given parameter direction. A 'segment' is here defined as a parameter interval in which there will be no discontinuities in derivatives or other artifacts. For spline objects, a segment will typically be the interval between two consecutive, non-coincident knots.

#### Parameters

<i>dir</i>	the parameter direction in which we search for the next segment (0 or 1)
<i>par</i>	the parameter value starting from which we search for the start value of the next segment
<i>forward</i>	define whether we shall move forward ('true') or backwards when searching along this parameter
<i>tol</i>	tolerance used for determining whether the ' <i>par</i> ' is already located <i>on</i> the next segment value

**Returns**

the value of the start value of the next segment (or the end of the previous segment, if we are moving backwards...)

Implements [Go::ParamSurface](#).

29.485.5.87 `virtual void Go::SplineSurface::normal ( Point & n, double upar, double vpar ) const` [virtual]

Evaluates the surface normal for a given parameter pair

**Parameters**

<i>n</i>	the computed normal will be written to this variable
<i>upar</i>	the first parameter
<i>vpar</i>	the second parameter

Implements [Go::ParamSurface](#).

29.485.5.88 `SplineSurface* Go::SplineSurface::normal ( ) const`

Not yet implemented.

29.485.5.89 `DirectionCone Go::SplineSurface::normalCone ( NormalConeMethod method ) const`

Returns a cone that contains the convex hull of all normalized tangents of the surface. Note: It is an overestimate.

**Returns**

the normal cone of the surface

29.485.5.90 `virtual DirectionCone Go::SplineSurface::normalCone ( ) const` [virtual]

Function that calls `normalCone(NormalConeMethod)` with `method = SederbergMeyers`. Needed because `normalCone()` is virtual! (Inherited from [ParamSurface](#)).

**Returns**

a [DirectionCone](#) (not necessarily the smallest) containing all normals to this surface.

Implements [Go::ParamSurface](#).

**29.485.5.91 SplineSurface\* Go::SplineSurface::normalSurface ( ) const**

Returns the normal surface corresponding to this surface, as described in:  
Computing normal vector Bezier patches, Przemyslaw Kiciak, Computer Aided Design 18 (2001), 699–710.

**Returns**

pointer to a newly created [SplineSurface](#) which is the normal surface. User assumes ownership of this object, and is responsible for destroying it.

**29.485.5.92 int Go::SplineSurface::numberOfPatches\_u ( ) const**

Returns the number of knot intervals in u knot vector.

**Returns**

the number of knot intervals in the knotvector for the first parameter

**29.485.5.93 int Go::SplineSurface::numberOfPatches\_v ( ) const**

Returns the number of knot intervals in v knot vector.

**Returns**

the number of knot intervals in the knotvector for the second parameter

**29.485.5.94 int Go::SplineSurface::numCoefs\_u ( ) const [inline]**

Query the number of control points along the first parameter direction

**Returns**

the number of control points along the first parameter direction

Definition at line 664 of file SplineSurface.h.

**29.485.5.95 int Go::SplineSurface::numCoefs\_v ( ) const [inline]**

Query the number of control points along the second parameter direction

**Returns**

the number of control points along the second parameter direction

Definition at line 669 of file SplineSurface.h.

29.485.5.96 `int Go::SplineSurface::numElem ( ) const [inline]`

Query the number of elements in the [SplineSurface](#).

Definition at line 673 of file `SplineSurface.h`.

29.485.5.97 `int Go::SplineSurface::numElem ( int parDir ) const [inline]`

Query the number of elements in one parameter direction of the [SplineSurface](#)

Definition at line 680 of file `SplineSurface.h`.

29.485.5.98 `virtual bool Go::SplineSurface::onBoundary ( double u, double v, double eps = 1.0e-4 ) const [virtual]`

Check if a parameter pair lies at the boundary of this surface.

Implements [Go::ParamSurface](#).

29.485.5.99 `int Go::SplineSurface::order_u ( ) const [inline]`

Query the order of the [BsplineBasis](#) for the first parameter

Returns

the order of the [BsplineBasis](#) for the first parameter

Definition at line 688 of file `SplineSurface.h`.

29.485.5.100 `int Go::SplineSurface::order_v ( ) const [inline]`

Query the order of the [BsplineBasis](#) for the second parameter

Returns

the order of the [BsplineBasis](#) for the second parameter

Definition at line 693 of file `SplineSurface.h`.

29.485.5.101 `virtual CurveLoop Go::SplineSurface::outerBoundaryLoop ( double degenerate_epsilon = DEFAULT_SPACE_EPSILON ) const [virtual]`

Returns the anticlockwise, outer boundary loop of the surface.

## Parameters

<i>degenerate_epsilon</i>	edges whose length is smaller than this value are ignored.
---------------------------	------------------------------------------------------------

## Returns

a [CurveLoop](#) describing the anticlockwise, outer boundary loop of the surface. A negative *degenerate\_epsilon* indicates that all curves, also the degenerate ones are wanted

Implements [Go::ParamSurface](#).

29.485.5.102 `virtual const RectDomain& Go::SplineSurface::parameterDomain ( ) const` [virtual]

Return the parameter domain of the surface. This may be a simple rectangular domain ([RectDomain](#)) or any other subclass of [Domain](#) (such as [GoCurveBoundedDomain](#), found in the `sisl_dependent` module).

## Returns

a [Domain](#) object describing the parametric domain of the surface

Implements [Go::ParamSurface](#).

29.485.5.103 `virtual void Go::SplineSurface::point ( Point & pt, double upar, double vpar ) const` [virtual]

Evaluates the surface's position for a given parameter pair.

## Parameters

<i>pt</i>	the result of the evaluation is written here
<i>upar</i>	the first parameter
<i>vpar</i>	the second parameter

Implements [Go::ParamSurface](#).

29.485.5.104 `virtual void Go::SplineSurface::point ( std::vector< Point > & pts, double upar, double vpar, int derivs, bool u_from_right = true, bool v_from_right = true, double resolution = 1.0e-12 ) const` [virtual]

Evaluates the surface's position and a certain number of derivatives for a given parameter pair.

## Parameters

<i>pts</i>	the vector containing the evaluated values. Its size must be set by the user prior to calling this function, and should be equal to $(derivs+1) * (derivs+2) / 2$ . Upon completion of the function, its first entry is the surface's position at the given parameter pair. Then, if 'derivs' > 0, the two next entries will be the surface tangents along the first and second parameter direction. The next three entries are the second- and cross derivatives, in the order (du2, dudv, dv2), and similar for even higher derivatives.
------------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

## Parameters

<i>upar</i>	the first parameter
<i>vpar</i>	the second parameter
<i>derivs</i>	number of requested derivatives
<i>u_from_right</i>	specify whether derivatives along the first parameter are to be calculated from the right ('true', default) or from the left ('false')
<i>v_from_right</i>	specify whether derivatives along the second parameter are to be calculated from the right ('true', default) or from the left ('false')
<i>resolution</i>	tolerance used when determining whether parameters are located at special values of the parameter domain (in particular; knot values in case of spline objects).

Implements [Go::ParamSurface](#).

**29.485.5.105** void `Go::SplineSurface::pointsGrid ( int m1, int m2, int derivs, const double * basisvals1, const double * basisvals2, const int * knotint1, const int * knotint2, double * result, double * normals = 0 ) const`

Evaluate points and possibly normals in a regular grid. The B-spline basis functions must be pre evaluated. This is a convenience funtion lifted to the user interface to allow for better performance if several spline surfaces with the same spline spaces are to be evaluated in the same points. Must be used with care.

**29.485.5.106** void `Go::SplineSurface::raiseOrder ( int raise_u, int raise_v )`

Raise the order of the spline surface as indicated by parameters.

## Parameters

<i>raise_u</i>	the order of the <a href="#">BsplineBasis</a> associated with the first parameter will be raised this many times.
<i>raise_v</i>	the order of the <a href="#">BsplineBasis</a> associated with the second parameter will be raised this many times.

**29.485.5.107** bool `Go::SplineSurface::rational ( ) const` `[inline]`

Query whether the surface is rational

## Returns

'true' if the surface is rational, 'false' otherwise

Definition at line 698 of file `SplineSurface.h`.

**29.485.5.108** std::vector<double>::iterator `Go::SplineSurface::rcoefs_begin ( )` `[inline]`

Get an iterator to the start of the internal array of *rational* control points.

## Returns

an (nonconst) iterator ro the start of the internal array of rational control points.

Definition at line 733 of file `SplineSurface.h`.

29.485.5.109 `std::vector<double>::const_iterator Go::SplineSurface::rcoefs_begin ( ) const` `[inline]`

Get a const iterator to the start of the internal array of *rational* control points.

#### Returns

a const iterator to the start of the internal array of rational control points

Definition at line 747 of file SplineSurface.h.

29.485.5.110 `std::vector<double>::iterator Go::SplineSurface::rcoefs_end ( )` `[inline]`

Get an iterator to the one-past-end position of the internal array of *rational* control points.

#### Returns

an (nonconst) iterator to the start of the internal array of rational control points.

Definition at line 740 of file SplineSurface.h.

29.485.5.111 `std::vector<double>::const_iterator Go::SplineSurface::rcoefs_end ( ) const` `[inline]`

Get a const iterator to the one-past-end position of the internal array of *rational* control points.

#### Returns

a const iterator to the one-past-end position of the internal array of rational control points.

Definition at line 754 of file SplineSurface.h.

29.485.5.112 `virtual void Go::SplineSurface::read ( std::istream & is )` `[virtual]`

read object from stream

#### Parameters

<i>is</i>	stream from which object is read
-----------	----------------------------------

Implements [Go::Streamable](#).

29.485.5.113 `void Go::SplineSurface::removeKnot_u ( double tpar )`

Remove a knot from the knotvector of the first parameter.



## Parameters

<i>tpar</i>	the parameter value of the knot to be removed
-------------	-----------------------------------------------

29.485.5.114 void Go::SplineSurface::removeKnot\_v ( double *tpar* )

Remove a knot from the knotvector of the second parameter.

## Parameters

<i>tpar</i>	the parameter value of the knot to be removed.
-------------	------------------------------------------------

29.485.5.115 bool Go::SplineSurface::replaceBoundaryCurve ( int *bd\_nmb*, shared\_ptr< SplineCurve > *bd\_crv*, bool *unify = true* )

Replace one boundary curve of this surface Update spline spaces to enable replacement NB! Requires both the surface and the new boundary curve to be either rational or non-rational *bd\_nmb* = 0: *umin* = 1: *umax* = 2: *vmin* = 3: *vmax* Return value: true if a replacement is performed

29.485.5.116 void Go::SplineSurface::replaceCoefficient ( int *ix*, Point *coef* )

Replace one specified coefficient (local enumeration)

29.485.5.117 void Go::SplineSurface::representAsRational ( )

Ensure that the current surface is represented as a rational surface.

29.485.5.118 virtual void Go::SplineSurface::reverseParameterDirection ( bool *direction\_is\_u* ) [virtual]

Reverses the direction of the basis in input direction.

## Parameters

<i>direction_is_u</i>	if 'true', the first parameter direction will be reversed, otherwise, the second parameter direction will be reversed
-----------------------	-----------------------------------------------------------------------------------------------------------------------

Implements [Go::ParamSurface](#).

29.485.5.119 double Go::SplineSurface::setAvBdWeight ( double *wgt*, int *pardir*, bool *at\_start* )

Set the average weight at one boundary to a given value (if rational) *pardir* = 0 : 1. parameter direction *pardir* = 1 : 2. parameter direction output : Variance between existing weights

29.485.5.120 `void Go::SplineSurface::setElementarySurface ( shared_ptr< ElementarySurface > elsurf )`

Set shared pointer to the [ElementarySurface](#) that is represented by `this`.

NOTE: The current [SplineSurface](#) (i.e. `this`), must be created from the argument [ElementarySurface](#) by [ElementarySurface::createSplineSurface\(\)](#), otherwise undefined behaviour may occur. One may check to see if this is the case by using [checkElementarySurface\(\)](#).

29.485.5.121 `virtual void Go::SplineSurface::setParameterDomain ( double u1, double u2, double v1, double v2 )`  
[virtual]

set the parameter domain to a given rectangle

#### Parameters

<i>u1</i>	new min. value of first parameter span
<i>u2</i>	new max. value of first parameter span
<i>v1</i>	new min. value of second parameter span
<i>v2</i>	new max. value of second parameter span

Reimplemented from [Go::ParamSurface](#).

29.485.5.122 `double Go::SplineSurface::startparam ( int idx ) const` [inline]

Get the start parameter value for the parameter direction `idx`.

Definition at line 371 of file `SplineSurface.h`.

29.485.5.123 `virtual double Go::SplineSurface::startparam_u ( ) const` [virtual]

Get the start value for the u-parameter

#### Returns

the start value for the u-parameter

29.485.5.124 `virtual double Go::SplineSurface::startparam_v ( ) const` [virtual]

Get the start value for the v-parameter

#### Returns

the start value for the v-parameter

29.485.5.125 `SplineSurface* Go::SplineSurface::subSurface ( double from_upar, double from_vpar, double to_upar, double to_vpar, double fuzzy = DEFAULT_PARAMETER_EPSILON ) const`

Get a [SplineSurface](#) which represent a part of 'this' [SplineSurface](#)

## Parameters

<i>from_upar</i>	start value for u-parameter in the sub-surface to be generated
<i>from_vpar</i>	start value for v-parameter in the sub-surface to be generated
<i>to_upar</i>	end value for u-parameter in the sub-surface to be generated
<i>to_vpar</i>	end value for v-parameter in the sub-surface to be generated
<i>fuzzy</i>	tolerance used to determine whether given parameter values are located directly <i>knot</i> values.

29.485.5.126 `virtual std::vector<shared_ptr<ParamSurface> > Go::SplineSurface::subSurfaces ( double from_upar, double from_vpar, double to_upar, double to_vpar, double fuzzy = DEFAULT_PARAMETER_EPSILON ) const [virtual]`

Get the surface(s) obtained by cropping the parameter domain of this surface between given values for the first and second parameter. In general, for surfaces with no interior holes, the result will be *one* surface; however, for surfaces with interior holes, the result might be *several disjoint* surfaces.

## Parameters

<i>from_upar</i>	lower value for the first parameter in the subdomain
<i>from_vpar</i>	lower value for the second parameter in the subdomain
<i>to_upar</i>	upper value for the first parameter in the subdomain
<i>to_vpar</i>	upper value for the second parameter in the subdomain
<i>fuzzy</i>	tolerance used when determining intersection with interior boundaries

## Returns

a vector contained shared pointers to the obtained, newly constructed sub-surfaces.

Implements [Go::ParamSurface](#).

29.485.5.127 `void Go::SplineSurface::swap ( SplineSurface & other )`

quick swap of two [SplineSurface](#) objects with each other

29.485.5.128 `virtual void Go::SplineSurface::swapParameterDirection ( ) [virtual]`

Swaps the two parameter directions.

Implements [Go::ParamSurface](#).

29.485.5.129 `virtual DirectionCone Go::SplineSurface::tangentCone ( bool pardir_is_u ) const [virtual]`

Creates a [DirectionCone](#) covering all tangents to this surface along a given parameter direction.

## Parameters

<code>pardir_is↔ _u</code>	if 'true', then the <a href="#">DirectionCone</a> will be defined on basis of the surface's tangents along the first parameter direction. Otherwise the second parameter direction will be used.
--------------------------------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

## Returns

a [DirectionCone](#) (not necessarily the smallest) containing all tangents to this surface along the specified parameter direction.

Implements [Go::ParamSurface](#).

```
29.485.5.130 virtual void Go::SplineSurface::turnOrientation () [virtual]
```

Turns the direction of the normal of the surface.

Implements [Go::ParamSurface](#).

```
29.485.5.131 virtual void Go::SplineSurface::write (std::ostream & os) const [virtual]
```

write object to stream

## Parameters

<code>os</code>	stream to which object is written
-----------------	-----------------------------------

Implements [Go::Streamable](#).

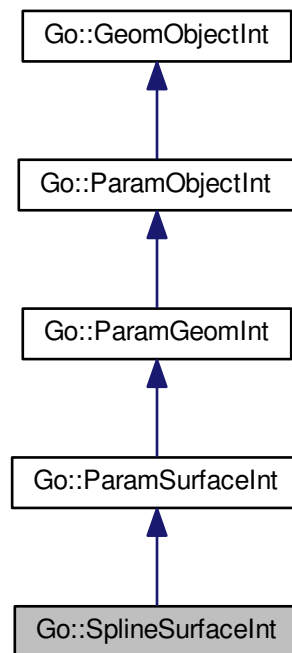
The documentation for this class was generated from the following file:

- [gtools-core/include/GoTools/geometry/SplineSurface.h](#)

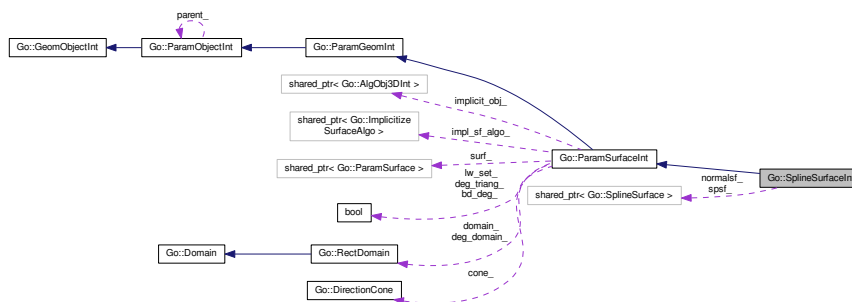
## 29.486 Go::SplineSurfaceInt Class Reference

```
#include <SplineSurfaceInt.h>
```

Inheritance diagram for Go::SplineSurfaceInt:



Collaboration diagram for Go::SplineSurfaceInt:



## Public Member Functions

- [SplineSurfaceInt](#) (shared\_ptr< [ParamSurface](#) > surf)  
*Constructor.*
- [SplineSurfaceInt](#) (shared\_ptr< [ParamSurface](#) > surf, [ParamGeomInt](#) \*parent)
- virtual [~SplineSurfaceInt](#) ()  
*Destructor.*
- virtual shared\_ptr< [ParamSurfaceInt](#) > [makeIntObject](#) (shared\_ptr< [ParamSurface](#) > surf)

- virtual `shared_ptr< ParamCurveInt > makeIntCurve` (`shared_ptr< ParamCurve > crv`, `ParamGeomInt *parent`)
- virtual `bool hasInnerKnots` (`int pdir`) `const`
- virtual `bool hasCriticalValsOrKnots` (`int pdir`) `const`
- virtual `std::vector< double > getInnerKnotVals` (`int pdir`, `bool sort=false`) `const`
- virtual `std::vector< double > getCriticalValsAndKnots` (`int pdir`) `const`
- virtual `int getMeshSize` (`int dir`)
- virtual `double paramFromMesh` (`int dir`, `int idx`)
- virtual `std::vector< double >::iterator getMesh` ()
  - Return the geometric sample mesh for the parametric function.*
- virtual `DirectionCone directionCone` () `const`
- virtual `DirectionCone reducedDirectionCone` (`bool reduce_at_bd[4]`, `double epsge`) `const`
  - Make exact direction cone. Only applicable for Bezier case.*
- virtual `void getBoundaryObjects` (`std::vector< shared_ptr< BoundaryGeomInt > > &bd_objs`)
- virtual `int checkPeriodicity` (`int pdir`) `const`
- virtual `bool isSimple` ()
- virtual `bool isSpline` ()
- virtual `bool isIsoParametric` (`ParamCurveInt *curve`, `int dir`, `double par`, `double ptol`, `double tol`)
- virtual `double getOptimizedConeAngle` (`Point &axis1`, `Point &axis2`)
- virtual `void knotIntervalFuzzy` (`double &u`, `double &v`, `double utol`, `double vtol`) `const`
- virtual `double nextSegmentVal` (`int dir`, `double par`, `bool forward`, `double tol`) `const`
- virtual `RotatedBox getRotatedBox` (`std::vector< Point > &axis`) `const`
- virtual `void splitAtG0` (`double angtol`, `std::vector< shared_ptr< ParamSurfaceInt > > &subG1`)
- virtual `shared_ptr< ParamSurfaceInt > getNormalSurface` () `const`
- virtual `bool canImplicitize` ()
- virtual `bool implicitize` (`double tol`)
- virtual `bool getImplicit` (`double tol`, `double &tol2`, `AlgObj3DInt &alg_obj_3d_int`)
- `SplineCurve * constParamCurve` (`double parameter`, `bool pdir_is_u`) `const`
- `shared_ptr< SplineSurface > getSplineSurface` ()
- virtual `shared_ptr< const SplineSurface > splineSurface` () `const`
- void `setImplicitDeg` ()
  - Choose a degree for the implicit representation.*

## Protected Attributes

- `shared_ptr< SplineSurface > spsf_`
- `shared_ptr< SplineSurface > normalsf_`

### 29.486.1 Detailed Description

This class represents the "intersection object" of a spline surface.

Definition at line 57 of file `SplineSurfaceInt.h`.

### 29.486.2 Constructor & Destructor Documentation

**29.486.2.1** `Go::SplineSurfaceInt::SplineSurfaceInt ( shared_ptr< ParamSurface > surf ) [explicit]`

Constructor.

## Parameters

<i>surf</i>	the parametric curve defining the intersection object.
-------------	--------------------------------------------------------

29.486.2.2 `Go::SplineSurfaceInt::SplineSurfaceInt ( shared_ptr< ParamSurface > surf, ParamGeomInt * parent ) [explicit]`

Constructor.

## Parameters

<i>surf</i>	the parametric curve defining the intersection object.
<i>parent</i>	the parent object to this object.

29.486.2.3 `virtual Go::SplineSurfaceInt::~~SplineSurfaceInt ( ) [inline],[virtual]`

Destructor.

Definition at line 73 of file SplineSurfaceInt.h.

### 29.486.3 Member Function Documentation

29.486.3.1 `virtual bool Go::SplineSurfaceInt::canImplicitize ( ) [virtual]`

Check whether the objects fulfills the requirements to implicitize.

## Returns

Return true if we may implicitize the object.

Reimplemented from [Go::ParamSurfaceInt](#).

29.486.3.2 `virtual int Go::SplineSurfaceInt::checkPeriodicity ( int paddir ) const [virtual]`

Check if the object is periodic in the specified direction. Analyze periodicity of surface based on number of repeating knots and control points. The return value is -1 if the surface ends are disjoint, otherwise k if cv is  $C^k$  continuous. These are sufficient but not necessary conditions for periodicity, so it is possible that a higher degree of periodicity exists. Should not be called on this layer, should be overruled by inherited class.

## Parameters

<i>paddir</i>	the parameter direction in question.
---------------	--------------------------------------

**Returns**

-1 if the curve ends are disjoint, or k if the surface is proven to be  $C^k$  continuous.

Reimplemented from [Go::ParamSurfaceInt](#).

### 29.486.3.3 `SplineCurve*` `Go::SplineSurfaceInt::constParamCurve ( double parameter, bool padir_is_u ) const`

Generate and return the specified isocurve.

**Parameters**

<i>parameter</i>	value of the fixed parameter.
<i>padir_is_u</i>	'true' if the first parameter is the running parameter, 'false' otherwise.

**Returns**

Pointer to the created isocurve.

### 29.486.3.4 `virtual DirectionCone` `Go::SplineSurfaceInt::directionCone ( ) const` `[virtual]`

A cone which contains all normals of the object.

**Returns**

A cone which contains all normals of the object.

Reimplemented from [Go::ParamSurfaceInt](#).

### 29.486.3.5 `virtual void` `Go::SplineSurfaceInt::getBoundaryObjects ( std::vector< shared_ptr< BoundaryGeomInt > > & bd_objs )` `[virtual]`

Return the boundary objects of this object.

**Parameters**

<i>bd_objs</i>	the boundary objects of this object.
----------------	--------------------------------------

Reimplemented from [Go::ParamSurfaceInt](#).

### 29.486.3.6 `virtual std::vector<double>` `Go::SplineSurfaceInt::getCriticalValsAndKnots ( int padir ) const` `[virtual]`

Return the critical parameter values and inner knots for the object.



## Parameters

<i>parDir</i>	the parameter direction in question. Indexing starts at 0.
---------------	------------------------------------------------------------

Reimplemented from [Go::ParamSurfaceInt](#).

**29.486.3.7** `virtual bool Go::SplineSurfaceInt::getImplicit ( double tol, double & tol2, AlgObj3DInt & alg_obj_3d_int )`  
[virtual]

Get the implicit representation of the object. Garbage is returned if we are not able to implicitize.

## Parameters

<i>tol</i>	geometric tolerance for the implicitization procedure.
<i>tol2</i>	geometric estimate for the accuracy of the implicitized object. Not yet in use!!!
<i>alg_obj_3d_int</i>	the algebraic object containing the implicitized surface.

## Returns

True if the implicitization was a success.

Reimplemented from [Go::ParamSurfaceInt](#).

**29.486.3.8** `virtual std::vector<double> Go::SplineSurfaceInt::getInnerKnotVals ( int parDir, bool sort = false ) const`  
[virtual]

Return the inner knot values in the specified direction.

## Parameters

<i>parDir</i>	the parameter direction in question. Indexing starts at 0.
<i>sort</i>	the returned values may be sorted by the function.

Reimplemented from [Go::ParamSurfaceInt](#).

**29.486.3.9** `virtual std::vector<double>::iterator Go::SplineSurfaceInt::getMesh ( )` [virtual]

Return the geometric sample mesh for the parametric function.

Reimplemented from [Go::ParamSurfaceInt](#).

**29.486.3.10** `virtual int Go::SplineSurfaceInt::getMeshSize ( int dir )` [virtual]

Return the size of the geometric sample mesh in the specified direction.

## Parameters

<i>dir</i>	the parameter direction in question. Indexing starts at 0.
------------	------------------------------------------------------------

Reimplemented from [Go::ParamSurfaceInt](#).

29.486.3.11 `virtual shared_ptr<ParamSurfaceInt> Go::SplineSurfaceInt::getNormalSurface ( ) const [virtual]`

Returns the normal surface corresponding to this surface.

## Returns

Pointer to the [SplineSurface](#) defining the normal surface.

Reimplemented from [Go::ParamSurfaceInt](#).

29.486.3.12 `virtual double Go::SplineSurfaceInt::getOptimizedConeAngle ( Point & axis1, Point & axis2 ) [virtual]`

We try to treat problems which will never result in a simple case by shrinking the domain slightly, resulting in smaller cones. This is useful for scenarios where the normals are parallel in a boundary point.

## Parameters

<i>axis1</i>	first vector defining a projection plane
<i>axis2</i>	second vector defining a projection plane

## Returns

The optimized cone angle

Reimplemented from [Go::ParamSurfaceInt](#).

29.486.3.13 `virtual RotatedBox Go::SplineSurfaceInt::getRotatedBox ( std::vector< Point > & axis ) const [virtual]`

Create a box containing the geometric sample mesh in the input coordinate system.

## Parameters

<i>axis</i>	the axis defining the coordinate system. The size of vector axis is 2, the last point is given as the cross product axis[0]axis[1].
-------------	-------------------------------------------------------------------------------------------------------------------------------------

## Returns

The rotated box

Reimplemented from [Go::ParamSurfaceInt](#).

29.486.3.14 `shared_ptr<SplineSurface> Go::SplineSurfaceInt::getSplineSurface ( ) [inline]`

Return pointer to the spline surface defining this object.

#### Returns

The spline surface defining this object.

Definition at line 257 of file SplineSurfaceInt.h.

29.486.3.15 `virtual bool Go::SplineSurfaceInt::hasCriticalValsOrKnots ( int pardir ) const [virtual]`

Return true if the object has any critical parameter values or inner knots in the specified parameter direction.

#### Parameters

<i>pardir</i>	the parameter direction in question. Indexing starts at 0.
---------------	------------------------------------------------------------

Reimplemented from [Go::ParamSurfaceInt](#).

29.486.3.16 `virtual bool Go::SplineSurfaceInt::hasInnerKnots ( int pardir ) const [virtual]`

Return true if the object has any inner knots in the specified parameter direction.

#### Parameters

<i>pardir</i>	the parameter direction in question. Indexing starts at 0.
---------------	------------------------------------------------------------

Reimplemented from [Go::ParamSurfaceInt](#).

29.486.3.17 `virtual bool Go::SplineSurfaceInt::implicitize ( double tol ) [virtual]`

Implicitize the object.

#### Parameters

<i>tol</i>	the geoemtric tolerance for the implicitization.
------------	--------------------------------------------------

#### Returns

True if the implicitization was a success.

Reimplemented from [Go::ParamSurfaceInt](#).

29.486.3.18 `virtual bool Go::SplineSurfaceInt::isIsoParametric ( ParamCurveInt * curve, int dir, double par, double ptol, double tol ) [virtual]`

Check if a curve is an iso parametric curve in the current surface. NB! Only valid for splines

Reimplemented from [Go::ParamSurfaceInt](#).

29.486.3.19 `virtual bool Go::SplineSurfaceInt::isSimple ( ) [virtual]`

Estimates if the current surface is simple enough for a singularity iteration. Checks the span of the normal cone and the size of the surface

#### Returns

True if the surface is characterized as simple.

Reimplemented from [Go::ParamSurfaceInt](#).

29.486.3.20 `virtual bool Go::SplineSurfaceInt::isSpline ( ) [virtual]`

Verify whether the object is a spline.

#### Returns

True if the object is a spline.

Reimplemented from [Go::ParamSurfaceInt](#).

29.486.3.21 `virtual void Go::SplineSurfaceInt::knotIntervalFuzzy ( double & u, double & v, double utol, double vtol ) const [virtual]`

Find the knot intervals for which u and v lie inside, moving the value u or v if they lie close to a knot.

#### Parameters

<i>u</i>	the <i>u</i> parameter value
<i>v</i>	the <i>v</i> parameter value
<i>utol</i>	the parametric tolerance deciding if the input parameter u should be moved.
<i>vtol</i>	the parametric tolerance deciding if the input parameter v should be moved.

Reimplemented from [Go::ParamSurfaceInt](#).

29.486.3.22 `virtual shared_ptr<ParamCurveInt> Go::SplineSurfaceInt::makeIntCurve ( shared_ptr< ParamCurve > crv, ParamGeomInt * parent ) [virtual]`

Return an intersection object for the input curve, using parameter parent as parent.

## Parameters

<i>crv</i>	the parametric curve defining the intersection object.
<i>parent</i>	the parent to the created intersection object.

Reimplemented from [Go::ParamSurfaceInt](#).

29.486.3.23 `virtual shared_ptr<ParamSurfaceInt> Go::SplineSurfaceInt::makeIntObject ( shared_ptr< ParamSurface > surf ) [virtual]`

Return an intersection object for the input surface, using this object as parent.

## Parameters

<i>surf</i>	the parametric surface defining the intersection object.
-------------	----------------------------------------------------------

Reimplemented from [Go::ParamSurfaceInt](#).

29.486.3.24 `virtual double Go::SplineSurfaceInt::nextSegmentVal ( int dir, double par, bool forward, double tol ) const [virtual]`

Return the value of the knot next to the input parameter par.

## Parameters

<i>dir</i>	the parameter direction in question. Indexing starts at 0.
<i>par</i>	the parameter value
<i>forward</i>	if true we return the closest knot to the right, otherwise the closest knot to the left.
<i>tol</i>	the tolerance to determine if <i>par</i> is already located on the start of the next segment.

## Returns

The knot closest to the input parameter.

Reimplemented from [Go::ParamSurfaceInt](#).

29.486.3.25 `virtual double Go::SplineSurfaceInt::paramFromMesh ( int dir, int idx ) [virtual]`

Return the corresponding mesh parameter.

## Parameters

<i>dir</i>	the parameter direction in question. Indexing starts at 0.
<i>idx</i>	the mesh idx in the specified direction. Indexing starts at 0.

Reimplemented from [Go::ParamSurfaceInt](#).

```
29.486.3.26 virtual DirectionCone Go::SplineSurfaceInt::reducedDirectionCone (bool reduce_at_bd[4], double epsge)
const [virtual]
```

Make exact direction cone. Only applicable for Bezier case.

Reimplemented from [Go::ParamGeomInt](#).

```
29.486.3.27 void Go::SplineSurfaceInt::setImplicitDeg ()
```

Choose a degree for the implicit representation.

```
29.486.3.28 virtual shared_ptr<const SplineSurface> Go::SplineSurfaceInt::splineSurface () const [inline],
[virtual]
```

Return pointer to the spline surface defining this object.

#### Returns

The spline surface defining this object.

Definition at line 262 of file SplineSurfaceInt.h.

```
29.486.3.29 virtual void Go::SplineSurfaceInt::splitAtG0 (double angtol, std::vector< shared_ptr< ParamSurfaceInt > >
& subG1) [virtual]
```

Requests from selfintersection computation. Split at G1 discontinuities.

#### Parameters

<i>angtol</i>	angular tolerance defining G1 discontinuity
<i>subG1</i>	vector of subdivided patches that are G1

Reimplemented from [Go::ParamSurfaceInt](#).

## 29.486.4 Member Data Documentation

```
29.486.4.1 shared_ptr<SplineSurface> Go::SplineSurfaceInt::normalsf_ [mutable],[protected]
```

Definition at line 272 of file SplineSurfaceInt.h.

29.486.4.2 `shared_ptr<SplineSurface> Go::SplineSurfaceInt::spsf_` [protected]

Definition at line 270 of file `SplineSurfaceInt.h`.

The documentation for this class was generated from the following file:

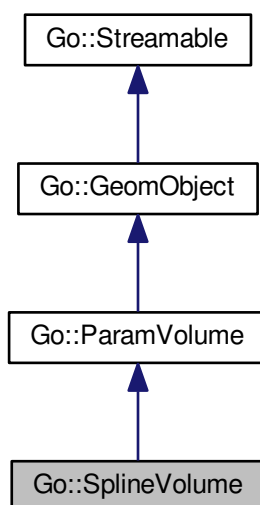
- `intersections/include/GoTools/intersections/SplineSurfaceInt.h`

## 29.487 Go::SplineVolume Class Reference

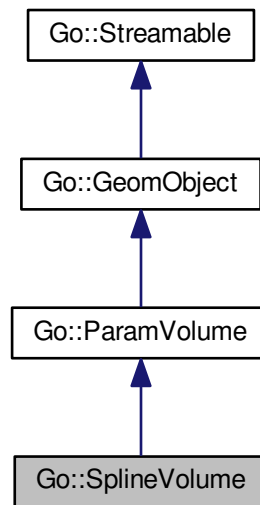
`SplineVolume` provides methods for storing, reading and manipulating rational and non-rational B-spline volumes.

```
#include <SplineVolume.h>
```

Inheritance diagram for `Go::SplineVolume`:



Collaboration diagram for Go::SplineVolume:



## Public Types

- typedef `std::vector< double >` `Dvector`
- typedef `std::vector< Dvector >` `Dmatrix`

## Public Member Functions

- `SplineVolume` ()
- `template<typename RandomIterator1 , typename RandomIterator2 , typename RandomIterator3 , typename RandomIterator4 > SplineVolume` (int number1, int number2, int number3, int order1, int order2, int order3, RandomIterator1 knot1start, RandomIterator2 knot2start, RandomIterator3 knot3start, RandomIterator4 coefsstart, int `dim`, bool `rational=false`)
- `template<typename RandomIterator > SplineVolume` (const `BsplineBasis` &basis\_u, const `BsplineBasis` &basis\_v, const `BsplineBasis` &basis\_w, RandomIterator coefsstart, int `dim`, bool `rational=false`)
- virtual `~SplineVolume` ()  
*Virtual destructor, enables safe inheritance.*
- virtual void `read` (std::istream &is)
- virtual void `write` (std::ostream &os) const
- virtual `BoundingBox boundingBox` () const  
*Return the object's bounding box.*
- virtual int `dimension` () const  
*Return the dimension of the space in which the object lies (3 is expected).*
- void `swap` (`SplineVolume` &other)  
*quick swap of two SplineVolume objects with each other*
- virtual `ClassType instanceType` () const  
*Return the class type identifier of a given, derived instance of GeomObject.*



- virtual [SplineVolume](#) \* [clone](#) () [const](#)
- virtual [const](#) [Array](#)< [double](#), 6 > [parameterSpan](#) () [const](#)
- virtual void [point](#) ([Point](#) &pt, [double](#) upar, [double](#) vpar, [double](#) wpar) [const](#)
- virtual void [point](#) (std::vector< [Point](#) > &pts, [double](#) upar, [double](#) vpar, [double](#) wpar, int derivs, [bool](#) u\_from←\_right=true, [bool](#) v\_from\_right=true, [bool](#) w\_from\_right=true, [double](#) resolution=1.0e-12) [const](#)
- virtual [double](#) [startparam](#) (int i) [const](#)
- virtual [double](#) [endparam](#) (int i) [const](#)
- virtual [DirectionCone](#) [tangentCone](#) (int pdir) [const](#)
- [SplineVolume](#) \* [derivVolume](#) (int ider1, int ider2, int ider3) [const](#)
- [SplineVolume](#) \* [subVolume](#) ([double](#) from\_upar, [double](#) from\_vpar, [double](#) from\_wpar, [double](#) to\_upar, [double](#) to\_vpar, [double](#) to\_wpar, [double](#) fuzzy=DEFAULT\_PARAMETER\_EPSILON) [const](#)
- std::vector< [shared\\_ptr](#)< [SplineVolume](#) > > [split](#) (std::vector< [double](#) > &param, int pdir, [double](#) fuzzy=DEFAULT\_PARAMETER\_EPSILON) [const](#)
- virtual void [closestPoint](#) ([const](#) [Point](#) &pt, [double](#) &clo\_u, [double](#) &clo\_v, [double](#) &clo\_w, [Point](#) &clo\_pt, [double](#) &clo\_dist, [double](#) epsilon, [double](#) \*seed=0) [const](#)
- int [closestCorner](#) ([const](#) [Point](#) &pt, [double](#) &upar, [double](#) &vpar, [double](#) &wpar, [Point](#) &corner, [double](#) &dist) [const](#)
- void [appendVolume](#) ([SplineVolume](#) \*vol, int join\_dir, int cont, [bool](#) repara=true)
- virtual void [reverseParameterDirection](#) (int pdir)
- virtual void [swapParameterDirection](#) (int pdir1, int pdir2)
- [const](#) [BsplineBasis](#) & [basis](#) (int pdir) [const](#)
- int [numCoefs](#) (int pdir) [const](#)
- int [order](#) (int pdir) [const](#)
- int [numElem](#) () [const](#)  
*Query the number of elements in the [SplineVolume](#).*
- int [numElem](#) (int pdir) [const](#)  
*Query the number of elements in one parameter direction of.*
- [bool](#) [rational](#) () [const](#)
- std::vector< [double](#) >::iterator [coefs\\_begin](#) ()
- std::vector< [double](#) >::iterator [coefs\\_end](#) ()
- std::vector< [double](#) >::const\_iterator [coefs\\_begin](#) () [const](#)
- std::vector< [double](#) >::const\_iterator [coefs\\_end](#) () [const](#)
- std::vector< [double](#) >::iterator [rcoefs\\_begin](#) ()
- std::vector< [double](#) >::iterator [rcoefs\\_end](#) ()
- std::vector< [double](#) >::const\_iterator [rcoefs\\_begin](#) () [const](#)
- std::vector< [double](#) >::const\_iterator [rcoefs\\_end](#) () [const](#)
- std::vector< [double](#) >::iterator [ctrl\\_begin](#) ()
- std::vector< [double](#) >::iterator [ctrl\\_end](#) ()
- std::vector< [double](#) >::const\_iterator [ctrl\\_begin](#) () [const](#)
- std::vector< [double](#) >::const\_iterator [ctrl\\_end](#) () [const](#)
- void [replaceCoefficient](#) (int ix, [Point](#) coef)  
*Replace one specified coefficient (local enumeration)*
- void [getWeights](#) (std::vector< [double](#) > &weights) [const](#)  
*has all weights equal to one*
- void [setParameterDomain](#) ([double](#) u1, [double](#) u2, [double](#) v1, [double](#) v2, [double](#) w1, [double](#) w2)
- void [insertKnot](#) (int pdir, [double](#) apar)
- void [insertKnot](#) (int pdir, [const](#) std::vector< [double](#) > &new\_knots)
- void [removeKnot](#) (int pdir, [double](#) tpar)
- void [makeBernsteinKnots](#) (int pdir)
- int [numberOfPatches](#) (int pdir) [const](#)
- [double](#) [knotSpan](#) (int pdir, int iknot) [const](#)
- void [raiseOrder](#) (int raise\_u, int raise\_v, int raise\_w)
- virtual [SplineSurface](#) \* [constParamSurface](#) ([double](#) parameter, int pdir) [const](#)
- virtual std::vector< [shared\\_ptr](#)< [ParamSurface](#) > > [getAllBoundarySurfaces](#) () [const](#)

*Fetch all boundary surfaces corresponding to the volume.*

- `std::vector< shared_ptr< SplineSurface > > getBoundarySurfaces (bool do_clear=false) const`
- `shared_ptr< SplineSurface > getBoundarySurface (int idx, bool do_clear=false) const`

*Fetch one boundary surface.*

- `void getElementBdPar (int elem_ix, double elem_par[]) const`
- `std::vector< shared_ptr< SplineSurface > > getElementBdSfs (int elem_ix, double elem_par[]) const`
- `void gridEvaluator (int num_u, int num_v, int num_w, std::vector< double > &points, std::vector< double > &der_u, std::vector< double > &der_v, std::vector< double > &der_w, std::vector< double > &param_u, std::vector< double > &param_v, std::vector< double > &param_w, bool evaluate_from_right=true) const`
- `void gridEvaluator (const std::vector< double > &param_u, const std::vector< double > &param_v, const std::vector< double > &param_w, std::vector< double > &points, std::vector< double > &der_u, std::vector< double > &der_v, std::vector< double > &der_w, bool evaluate_from_right=true) const`
- `void gridEvaluator (int num_u, int num_v, int num_w, std::vector< double > &points, std::vector< double > &param_u, std::vector< double > &param_v, std::vector< double > &param_w) const`
- `void gridEvaluator (const std::vector< double > &param_u, const std::vector< double > &param_v, const std::vector< double > &param_w, std::vector< double > &points) const`
- `void computeBasis (double param[], std::vector< double > &basisValues, std::vector< double > &basisDerivs_u, std::vector< double > &basisDerivs_v, std::vector< double > &basisDerivs_w, bool evaluate_from_right=true) const`
- `void computeBasis (const std::vector< double >::const_iterator &bas_vals_u, const std::vector< double >::const_iterator &bas_vals_v, const std::vector< double >::const_iterator &bas_vals_w, int left_u, int left_v, int left_w, std::vector< double > &basisValues, std::vector< double > &basisDerivs_u, std::vector< double > &basisDerivs_v, std::vector< double > &basisDerivs_w) const`
- `void computeBasisGrid (const Dvector &param_u, const Dvector &param_v, const Dvector &param_w, Dmatrix &basisValues) const`
- `void computeBasis (double param_u, double param_v, double param_w, BasisPts &result) const`
- `void computeBasis (double param_u, double param_v, double param_w, BasisDerivs &result, bool evaluate_from_right=true) const`
- `void computeBasis (double param_u, double param_v, double param_w, BasisDerivs2 &result, bool evaluate_from_right=true) const`
- `void computeBasisGrid (const Dvector &param_u, const Dvector &param_v, const Dvector &param_w, std::vector< BasisPts > &result) const`
- `void computeBasisGrid (const Dvector &param_u, const Dvector &param_v, const Dvector &param_w, Dmatrix &basisValues, Dmatrix &basisDerivs_u, Dmatrix &basisDerivs_v, Dmatrix &basisDerivs_w, bool evaluate_from_right=true) const`
- `void computeBasisGrid (const Dvector &param_u, const Dvector &param_v, const Dvector &param_w, std::vector< BasisDerivs > &result, bool evaluate_from_right=true) const`
- `void computeBasisGrid (const Dvector &param_u, const Dvector &param_v, const Dvector &param_w, std::vector< BasisDerivs2 > &result, bool evaluate_from_right=true) const`
- `virtual double nextSegmentVal (int dir, double par, bool forward, double tol) const`
- `int volumePeriodicity (int pdir, double epsilon) const`
- `void checkDegeneracy (double tol, int is_degenerate[]) const`
- `bool isDegenerate (int which_sf, int &type, bool &b, bool &r, bool &t, bool &l, double tol) const`
- `virtual bool isSpline () const`

*Check if the volume is of type spline.*

- `virtual SplineVolume * asSplineVolume ()`

*Return the spline surface represented by this surface.*

- `virtual void translate (const Point &vec)`

*Translate.*

- `void scale (double fac)`

*Scale the coefficients of the volume with a given factor.*

- `void deform (const std::vector< double > &vec, int vdim=0)`
- `void add (const SplineVolume *other, double tol=1.0e-10)`
- `bool isLeftHanded ()`

- void [pointsGrid](#) (int numu, int numv, int numw, std::vector< double >::iterator basisvals\_u, std::vector< double >::iterator basisvals\_v, std::vector< double >::iterator basisvals\_w, std::vector< int >::iterator knotinter\_u, std::vector< int >::iterator knotinter\_v, std::vector< int >::iterator knotinter\_w, int derivs, std::vector< double > &points) const

## Static Public Member Functions

- static [ClassType classType](#) ()

### 29.487.1 Detailed Description

[SplineVolume](#) provides methods for storing, reading and manipulating rational and non-rational B-spline volumes.

Non-rational B-spline volumes represented on the form

$$\sum_{i=1}^{n_1} \sum_{j=1}^{n_2} \sum_{k=1}^{n_3} P_{i,j,k} B_{i,o_1}(u) B_{j,o_2}(v) B_{k,o_3}(w),$$

where the B-spline coefficients are stored in a vector called coefs. The coefs are stored as

$$P_{0,0,0}, P_{1,0,0}, \dots, P_{n_1,n_2,n_3},$$

where  $P_{i,j,k}$  is represented as dim doubles.

NURBS volumes are represented on the form

$$\frac{\sum_{i=1}^{n_1} \sum_{j=1}^{n_2} \sum_{k=1}^{n_3} h_{i,j,k} P_{i,j,k} B_{i,o_1}(u) B_{j,o_2}(v) B_{k,o_3}(w)}{\sum_{i=1}^{n_1} \sum_{j=1}^{n_2} \sum_{k=1}^{n_3} h_{i,j,k} B_{i,o_1}(u) B_{j,o_2}(v) B_{k,o_3}(w)}$$

where  $B_{i,o}$  is the i'th non-rational B-spline of order o. The Projected coefficients are stored in the coefs vector, i.e.

$$h_{0,0,0} \cdot P_{0,0,0}, h_{1,0,0} \cdot P_{1,0,0}, \dots, h_{n_1,n_2,n_3} \cdot P_{n_1,n_2,n_3}.$$

In addition the coefficients for the volume in projective space are kept, i.e

$$P_{0,0,0}, h_{0,0,0}, P_{1,0,0}, h_{1,0,0}, \dots, P_{n_1,n_2,n_3}, h_{n_1,n_2,n_3}.$$

With this representation volume is in the convex hull of the coefficients stored in coefs both for rational and non rational volumes.

Definition at line 201 of file SplineVolume.h.

### 29.487.2 Member Typedef Documentation

#### 29.487.2.1 typedef std::vector<Dvector> Go::SplineVolume::Dmatrix

Definition at line 833 of file SplineVolume.h.

#### 29.487.2.2 typedef std::vector<double> Go::SplineVolume::Dvector

Evaluate positions of all basis values in a specified grid. For non-rationals this is an interface to [BsplineBasis::computeBasisValues](#) where the basis values in each parameter direction are multiplied to compute  $B_i(u) * B_j(v) * B_k(w)$ , for rationals the routine evaluates the rational basis functions, i.e. the basis functions are divided by the denominator of the volume in the given parameter direction

## Parameters

<i>param_u</i>	this vector holds all the numerical values for the first parameter where evaluation will take place
<i>param_v</i>	this vector holds all the numerical values for the second parameter where evaluation will take place
<i>param_w</i>	this vector holds all the numerical values for the third parameter where evaluation will take place
<i>basisValues</i>	the value of all basis functions. The first matrix dimension will correspond to all evaluation parameters, i.e. the size is <code>param_u.size()*param_v*size()*param_w.size()</code> . The second dimension correspond to all coefficients, i.e. <code>nmb_coef_u*nmb_coef_v*nmb_coef_w</code> . Most entries in the matrix will be zero

Definition at line 832 of file `SplineVolume.h`.

### 29.487.3 Constructor & Destructor Documentation

#### 29.487.3.1 `Go::SplineVolume::SplineVolume ( )` `[inline]`

Creates an uninitialized [SplineVolume](#), which can only be assigned to or read(...) into.

Definition at line 206 of file `SplineVolume.h`.

#### 29.487.3.2 `template<typename RandomIterator1 , typename RandomIterator2 , typename RandomIterator3 , typename RandomIterator4 > Go::SplineVolume::SplineVolume ( int number1, int number2, int number3, int order1, int order2, int order3, RandomIterator1 knot1start, RandomIterator2 knot2start, RandomIterator3 knot3start, RandomIterator4 coefsstart, int dim, bool rational = false )` `[inline]`

Create a [SplineVolume](#) by explicitly providing all spline-related information.

## Parameters

<i>number1</i>	number of control points along the u-parameter
<i>number2</i>	number of control points along the v-parameter
<i>number3</i>	number of control points along the w-parameter
<i>order1</i>	<a href="#">BsplineBasis</a> order along the u-parameter
<i>order2</i>	<a href="#">BsplineBasis</a> order along the v-parameter
<i>order3</i>	<a href="#">BsplineBasis</a> order along the w-parameter
<i>knot1start</i>	pointer to the array describing the knotvector for the u-parameter
<i>knot2start</i>	pointer to the array describing the knotvector for the v-parameter
<i>knot3start</i>	pointer to the array describing the knotvector for the w-parameter
<i>coefsstart</i>	pointer to the array where the control points are consecutively stored. The storage order is such that control points along the u-parameter have the shortest stride (stored right after each other), then comes the v-parameter and finally the w-parameter. If the volume is rational, pay attention to the comments below.
<i>dim</i>	dimension of the space in which the volume lies (usually 3).
<i>rational</i>	Specify whether the volume is rational or not. If the volume is rational, coefficients must be in the following format: <code>wP1 wP2 .... wPdim w</code> . I.e. a $(dim+1)$ -dimensional form. (This is the same form that is used within SISL).

Definition at line 245 of file `SplineVolume.h`.

```
29.487.3.3 template<typename RandomIterator > Go::SplineVolume::SplineVolume (const BsplineBasis & basis_u,
const BsplineBasis & basis_v, const BsplineBasis & basis_w, RandomIterator coefsstart, int dim, bool
rational = false) [inline]
```

Create a [SplineVolume](#) by explicitly providing all spline-related information.

#### Parameters

<i>basis_u</i>	the <a href="#">BsplineBasis</a> to be used along the u-direction
<i>basis_v</i>	the <a href="#">BsplineBasis</a> to be used along the v-direction
<i>basis_w</i>	the <a href="#">BsplineBasis</a> to be used along the v-direction
<i>coefsstart</i>	pointer to the array where the control points are consecutively stored. The storage order is such that control points along the u-parameter have the shortest stride (stored right after each other), then comes the v-parameter and finally the w-parameter. If the volume is rational, pay attention to the comments below.
<i>dim</i>	dimension of the space in which the volume lies (usually 3).
<i>rational</i>	Specify whether the volume is rational or not. If the volume is rational, coefficients must be in the following format: wP1 wP2 .... wPdim w. I.e. a (dim+1)-dimensional form. (This is the same form that is used within SISL).

Definition at line 297 of file SplineVolume.h.

```
29.487.3.4 virtual Go::SplineVolume::~~SplineVolume () [virtual]
```

Virtual destructor, enables safe inheritance.

## 29.487.4 Member Function Documentation

```
29.487.4.1 void Go::SplineVolume::add (const SplineVolume * other, double tol = 1.0e-10)
```

Add coefficients from another volume. Weights are not summed for rational cases Nothing is done and exception is raised if

- Spline spaces are different in any paramter direction (order or knot vectors are not identical)
- The geometry spaces of the volumes have different dimension
- One volume is rational while the other is not
- The weights are not equal within a given tolerance (only if volumes are rational)

#### Parameters

<i>other</i>	The other spline volume with coefficients to be added into this volume
<i>tol</i>	tolerance used to test if weights are considered equal in rational case

29.487.4.2 void Go::SplineVolume::appendVolume ( SplineVolume \* vol, int join\_dir, int cont, bool repara = true )

Appends a volume to the end of 'this' volume along a specified parameter direction.

#### Parameters

<i>vol</i>	the volume to append to 'this' volume
<i>join_dir</i>	the parameter direction along which to extend 'this' volume by appending the 'vol' volume. If it is equal to 0, we will extend the volume along the u-parameter; if it equals 1, we will extend the volume along the v-parameter; if it equals 2, we will extend the volume along the w-parameter.
<i>cont</i>	continuity at joint, legal values are 0 and 1 (i.e. $C^0$ or $C^1$ -continuity).
<i>repara</i>	The parametrization of the concerned parameter in the other volume will <i>always</i> be shifted so that it starts where the parametrization of 'this' volume ends. However, if 'repara' is set to 'true', it will also be <i>scaled</i> as a function of position of control points close to the transition.

29.487.4.3 virtual SplineVolume\* Go::SplineVolume::asSplineVolume ( ) [inline],[virtual]

Return the spline surface represented by this surface.

Reimplemented from [Go::ParamVolume](#).

Definition at line 957 of file SplineVolume.h.

29.487.4.4 const BsplineBasis& Go::SplineVolume::basis ( int pdir ) const [inline]

get one of the BsplineBasises of the volume

#### Parameters

<i>pdir</i>	specify whether to return the <a href="#">BsplineBasis</a> for the first parameter (0), for the second parameter (1) or the third (2) parameter.
-------------	--------------------------------------------------------------------------------------------------------------------------------------------------

#### Returns

const reference to the requested [BsplineBasis](#).

Definition at line 471 of file SplineVolume.h.

29.487.4.5 virtual BoundingBox Go::SplineVolume::boundingBox ( ) const [virtual]

Return the object's bounding box.

Implements [Go::GeomObject](#).

29.487.4.6 void Go::SplineVolume::checkDegeneracy ( double tol, int is\_degenerate[] ) const

Check degeneracy

## Parameters

<i>tol</i>	Tolerance used in test
<i>is_degenerate</i>	0 - Not degenerate, 1 - surface degenerate in specified boundary, 2 - surface degenerate to line, 3 - surface degenerate to point

29.487.4.7 `static ClassType Go::SplineVolume::classType ( ) [inline],[static]`

Definition at line 349 of file SplineVolume.h.

29.487.4.8 `virtual SplineVolume* Go::SplineVolume::clone ( ) const [inline],[virtual]`

make a clone of this volume and return a pointer to it (user is responsible for clearing up memory afterwards).

## Returns

pointer to cloned object

Implements [Go::ParamVolume](#).

Definition at line 352 of file SplineVolume.h.

29.487.4.9 `int Go::SplineVolume::closestCorner ( const Point & pt, double & upar, double & vpar, double & wpar, Point & corner, double & dist ) const`

Returns the corner closest to a given point together with the associated enumeration of the corner coefficient. In degenerate cases, the enumeration will reflect an arbitrary corner

29.487.4.10 `virtual void Go::SplineVolume::closestPoint ( const Point & pt, double & clo_u, double & clo_v, double & clo_w, Point & clo_pt, double & clo_dist, double epsilon, double * seed = 0 ) const [virtual]`

Iterates to the closest point to pt on the volume.

## Parameters

<i>pt</i>	the point to find the closest point to
<i>clo_u</i>	u parameter of the closest point
<i>clo_v</i>	v parameter of the closest point
<i>clo_w</i>	w parameter of the closest point
<i>clo_pt</i>	the geometric position of the closest point
<i>clo_dist</i>	the distance between pt and clo_pt
<i>epsilon</i>	parameter tolerance (will in any case not be higher than sqrt(machine_precision) x magnitude of solution)
<i>seed</i>	pointer to parameter values where iteration starts.

Implements [Go::ParamVolume](#).

29.487.4.11 `std::vector<double>::iterator Go::SplineVolume::coefs_begin ( ) [inline]`

Get an iterator to the start of the internal array of non-rational control points.

#### Returns

an (nonconst) iterator to the start of the internal array of non-rational control points

Definition at line 521 of file SplineVolume.h.

29.487.4.12 `std::vector<double>::const_iterator Go::SplineVolume::coefs_begin ( ) const [inline]`

Get a const iterator to the start of the internal array of non-rational control points.

#### Returns

a const iterator to the start of the internal array of non-rational control points.

Definition at line 535 of file SplineVolume.h.

29.487.4.13 `std::vector<double>::iterator Go::SplineVolume::coefs_end ( ) [inline]`

Get an iterator to the one-past-end position of the internal array of non-rational control points

#### Returns

an (nonconst) iterator to the one-past-end position of the internal array of non-rational control points

Definition at line 528 of file SplineVolume.h.

29.487.4.14 `std::vector<double>::const_iterator Go::SplineVolume::coefs_end ( ) const [inline]`

Get a const iterator to the one-past-end position of the internal array of non-rational control points.

#### Returns

a const iterator to the one-past-end position of the internal array of non-rational control points.

Definition at line 542 of file SplineVolume.h.

29.487.4.15 `void Go::SplineVolume::computeBasis ( double param[], std::vector< double > & basisValues, std::vector< double > & basisDerivs_u, std::vector< double > & basisDerivs_v, std::vector< double > & basisDerivs_w, bool evaluate_from_right = true ) const`

Evaluate positions and first derivatives of all basis values in a given parameter tripple For non-rationals this is an interface to [BsplineBasis::computeBasisValues](#) where the basis values in each parameter direction are multiplied to compute  $B_i(u)*B_j(v)*B_k(w)$ , for rationals the routine evaluates the rational basis functions, i.e. the basis functions are divided by the denominator of the volume in the given parameter direction



## Parameters

<i>param</i>	the parameter tripple in which to compute
<i>basisValues</i>	the value of all basis functions, size equal to (degree_u+1)*(degree_v+1)*(degree_w+1)
<i>basisDerivs_u</i>	the derivative of all basis functions, same size as previous
<i>basisDerivs_v</i>	the derivative of all basis functions, same size as previous
<i>basisDerivs_w</i>	the derivative of all basis functions, same size as previous
<i>evaluate_from_right</i>	specifies directional derivatives, true=right, false=left

- 29.487.4.16 void Go::SplineVolume::computeBasis ( const std::vector< double >::const\_iterator & *bas\_vals\_u*, const std::vector< double >::const\_iterator & *bas\_vals\_v*, const std::vector< double >::const\_iterator & *bas\_vals\_w*, int *left\_u*, int *left\_v*, int *left\_w*, std::vector< double > & *basisValues*, std::vector< double > & *basisDerivs\_u*, std::vector< double > & *basisDerivs\_v*, std::vector< double > & *basisDerivs\_w* ) const
- 29.487.4.17 void Go::SplineVolume::computeBasis ( double *param\_u*, double *param\_v*, double *param\_w*, BasisPts & *result* ) const
- 29.487.4.18 void Go::SplineVolume::computeBasis ( double *param\_u*, double *param\_v*, double *param\_w*, BasisDerivs & *result*, bool *evaluate\_from\_right* = true ) const
- 29.487.4.19 void Go::SplineVolume::computeBasis ( double *param\_u*, double *param\_v*, double *param\_w*, BasisDerivs2 & *result*, bool *evaluate\_from\_right* = true ) const
- 29.487.4.20 void Go::SplineVolume::computeBasisGrid ( const Dvector & *param\_u*, const Dvector & *param\_v*, const Dvector & *param\_w*, Dmatrix & *basisValues* ) const
- 29.487.4.21 void Go::SplineVolume::computeBasisGrid ( const Dvector & *param\_u*, const Dvector & *param\_v*, const Dvector & *param\_w*, std::vector< BasisPts > & *result* ) const
- 29.487.4.22 void Go::SplineVolume::computeBasisGrid ( const Dvector & *param\_u*, const Dvector & *param\_v*, const Dvector & *param\_w*, Dmatrix & *basisValues*, Dmatrix & *basisDerivs\_u*, Dmatrix & *basisDerivs\_v*, Dmatrix & *basisDerivs\_w*, bool *evaluate\_from\_right* = true ) const

Evaluate positions and first derivatives of all basis values in a specified grid. For non-rationals this is an interface to [BsplineBasis::computeBasisValues](#) where the basis values in each parameter direction are multiplied to compute  $B_i(u)*B_j(v)*B_k(w)$ , for rationals the routine evaluates the rational basis functions, i.e. the basis functions are divided by the denominator of the volume in the given parameter direction

## Parameters

<i>param_u</i>	this vector holds all the numerical values for the first parameter where evaluation will take place
<i>param_v</i>	this vector holds all the numerical values for the second parameter where evaluation will take place
<i>param_w</i>	this vector holds all the numerical values for the third parameter where evaluation will take place
<i>basisValues</i>	the value of all basis functions. The first matrix dimension will correspond to all evaluation parameters, i.e. the size is <i>param_u.size()*param_v.size()*param_w.size()</i> . The second dimension correspond to all coefficients, i.e. <i>nmb_coef_u*nmb_coef_v*nmb_coef_w</i> . Most entries in the matrix will be zero
<i>basisDerivs_u</i>	the derivative of all basis functions, size as above
<i>basisDerivs_v</i>	the derivative of all basis functions
<i>basisDerivs_w</i>	the derivative of all basis functions
<i>evaluate_from_right</i>	specifies directional derivatives, true=right, false=left

29.487.4.23 `void Go::SplineVolume::computeBasisGrid ( const Dvector & param_u, const Dvector & param_v, const Dvector & param_w, std::vector< BasisDerivs > & result, bool evaluate_from_right = true ) const`

29.487.4.24 `void Go::SplineVolume::computeBasisGrid ( const Dvector & param_u, const Dvector & param_v, const Dvector & param_w, std::vector< BasisDerivs2 > & result, bool evaluate_from_right = true ) const`

29.487.4.25 `virtual SplineSurface* Go::SplineVolume::constParamSurface ( double parameter, int pdir ) const`  
[virtual]

Generate and return a [SplineSurface](#) that represents a constant parameter surface on the volume

#### Parameters

<i>parameter</i>	value of the fixed parameter
<i>pdir</i>	0 if the surface is constant in the u-parameter, 1 if the surface is constant in the v-parameter, 2 if the surface is constant in the w-parameter.

#### Returns

pointer to a newly constructed [SplineSurface](#) representing the specified constant parameter surface. It is the user's responsibility to delete it when it is no longer needed.

Reimplemented from [Go::ParamVolume](#).

29.487.4.26 `std::vector<double>::iterator Go::SplineVolume::ctrl_begin ( )` [inline]

Get an iterator to the start of the internal array of active control points.

#### Returns

an (nonconst) iterator to the start of the internal array of rational or non-rational control points

Definition at line 577 of file SplineVolume.h.

29.487.4.27 `std::vector<double>::const_iterator Go::SplineVolume::ctrl_begin ( ) const` [inline]

Get a const iterator to the start of the internal array of active control points.

#### Returns

a const iterator to the start of the internal array of rational or non-rational control points.

Definition at line 591 of file SplineVolume.h.

29.487.4.28 `std::vector<double>::iterator Go::SplineVolume::ctrl_end ( ) [inline]`

Get an iterator to the one-past-end position of the internal array of active control points

#### Returns

an (nonconst) iterator to the one-past-end position of the internal array of rational or non-rational control points

Definition at line 584 of file SplineVolume.h.

29.487.4.29 `std::vector<double>::const_iterator Go::SplineVolume::ctrl_end ( ) const [inline]`

Get a const iterator to the one-past-end position of the internal array of active control points.

#### Returns

a const iterator to the one-past-end position of the internal array of rational or non-rational control points.

Definition at line 598 of file SplineVolume.h.

29.487.4.30 `void Go::SplineVolume::deform ( const std::vector< double > & vec, int vdim = 0 )`

Adds the given deformation vector to the coefficients. Similar as translate, but with separate vector for each point

29.487.4.31 `SplineVolume* Go::SplineVolume::derivVolume ( int ider1, int ider2, int ider3 ) const`

Get the derivative volume. Expresses the i,j,k-th derivative of a spline volume as a spline volume.

#### Parameters

<i>ider1</i>	what derivative to use along the u parameter
<i>ider2</i>	what derivative to use along the v parameter
<i>ider3</i>	what derivative to use along the w parameter

#### Returns

pointer to a newly constructed [SplineVolume](#) which is the derivative volume. User assumes ownership of this object, and is responsible for destroying it.

29.487.4.32 `virtual int Go::SplineVolume::dimension ( ) const [virtual]`

Return the dimension of the space in which the object lies (3 is expected).

Implements [Go::GeomObject](#).

29.487.4.33 `virtual double Go::SplineVolume::endparam ( int i ) const` [virtual]

Get the end value for the specified parameter direction.

## Parameters

<i>i</i>	the parameter direction
----------	-------------------------

## Returns

the end value for the parameter direction given by the parameter *pardir*

29.487.4.34 `virtual std::vector<shared_ptr<ParamSurface>> Go::SplineVolume::getAllBoundarySurfaces ( ) const`  
[virtual]

Fetch all boundary surfaces corresponding to the volume.

Implements [Go::ParamVolume](#).

29.487.4.35 `shared_ptr<SplineSurface> Go::SplineVolume::getBoundarySurface ( int idx, bool do_clear = false ) const`  
[inline]

Fetch one boundary surface.

Definition at line 682 of file `SplineVolume.h`.

29.487.4.36 `std::vector<shared_ptr<SplineSurface>> Go::SplineVolume::getBoundarySurfaces ( bool do_clear = false ) const`

Fetch all 6 boundary surfaces corresponding to the volume. Sequence: *u\_min*, *u\_max*, *v\_min*, *v\_max*, *w\_min*, *w\_max*

29.487.4.37 `void Go::SplineVolume::getElementBdPar ( int elem_ix, double elem_par[] ) const`

Fetch parameter boundaries of a specified element *elem\_ix* - parameter values of element boundaries, sequence *u\_min*, *u\_max*, *v\_min*, *v\_max*, *w\_min*, *w\_max*

29.487.4.38 `std::vector<shared_ptr<SplineSurface>> Go::SplineVolume::getElementBdSfs ( int elem_ix, double elem_par[] ) const`

Fetch all boundary surfaces of a specified element *elem\_ix* - parameter values of element boundaries, sequence *u\_min*, *u\_max*, *v\_min*, *v\_max*, *w\_min*, *w\_max*

29.487.4.39 `void Go::SplineVolume::getWeights ( std::vector< double > & weights ) const`

has all weights equal to one

Return all weights corresponding to this volume, a non-rational volume

29.487.4.40 `void Go::SplineVolume::gridEvaluator ( int num_u, int num_v, int num_w, std::vector< double > & points, std::vector< double > & der_u, std::vector< double > & der_v, std::vector< double > & der_w, std::vector< double > & param_u, std::vector< double > & param_v, std::vector< double > & param_w, bool evaluate_from_right = true ) const`

Evaluate points and derivatives on an entire grid, taking computational advantage over calculating all these values simultaneously rather than one-by-one.

## Parameters

<i>num_u</i>	number of values to evaluate along first parameter direction
<i>num_v</i>	number of values to evaluate along second parameter direction
<i>num_w</i>	number of values to evaluate along third parameter direction
<i>points</i>	upon function return, this vector holds all the evaluated points
<i>der_u</i>	upon function return, this vector holds all first derivatives in the first parameter direction
<i>der_v</i>	upon function return, this vector holds all first derivatives in the second parameter direction
<i>der_w</i>	upon function return, this vector holds all first derivatives in the third parameter direction
<i>param_u</i>	upon function return, this vector holds all the numerical values for the first parameter where evaluation has taken place
<i>param_v</i>	upon function return, this vector holds all the numerical values for the second parameter where evaluation has taken place.
<i>param_w</i>	upon function return, this vector holds all the numerical values for the third parameter where evaluation has taken place.
<i>evaluate_from_right</i>	specifies directional derivatives, true=right, false=left same behaviour for all parameter directions

```
29.487.4.41 void Go::SplineVolume::gridEvaluator (const std::vector< double > & param_u, const std::vector< double > & param_v, const std::vector< double > & param_w, std::vector< double > & points, std::vector< double > & der_u, std::vector< double > & der_v, std::vector< double > & der_w, bool evaluate_from_right = true) const
```

Evaluate points and derivatives on an entire grid, taking computational advantage over calculating all these values simultaneously rather than one-by-one.

## Parameters

<i>param_u</i>	this vector holds all the numerical values for the first parameter where evaluation will take place
<i>param_v</i>	this vector holds all the numerical values for the second parameter where evaluation will take place
<i>param_w</i>	this vector holds all the numerical values for the third parameter where evaluation will take place
<i>points</i>	upon function return, this vector holds all the evaluated points
<i>der_u</i>	upon function return, this vector holds all first derivatives in the first parameter direction
<i>der_v</i>	upon function return, this vector holds all first derivatives in the second parameter direction
<i>der_w</i>	upon function return, this vector holds all first derivatives in the third parameter direction
<i>evaluate_from_right</i>	specifies directional derivatives, true=right, false=left same behaviour for all parameter directions

```
29.487.4.42 void Go::SplineVolume::gridEvaluator (int num_u, int num_v, int num_w, std::vector< double > & points, std::vector< double > & param_u, std::vector< double > & param_v, std::vector< double > & param_w) const
```

Evaluate points on an entire grid, taking computational advantage over calculating all these values simultaneously rather than one-by-one.

## Parameters

<i>num_u</i>	number of values to evaluate along first parameter direction
<i>num_v</i>	number of values to evaluate along second parameter direction
<i>num_w</i>	number of values to evaluate along third parameter direction
<i>points</i>	upon function return, this vector holds all the evaluated points
<i>param_u</i>	upon function return, this vector holds all the numerical values for the first parameter where evaluation has taken place
<i>param_v</i>	upon function return, this vector holds all the numerical values for the second parameter where evaluation has taken place.
<i>param_w</i>	upon function return, this vector holds all the numerical values for the third parameter where evaluation has taken place.

29.487.4.43 `void Go::SplineVolume::gridEvaluator ( const std::vector< double > & param_u, const std::vector< double > & param_v, const std::vector< double > & param_w, std::vector< double > & points ) const`

Evaluate points on an entire grid, taking computational advantage over calculating all these values simultaneously rather than one-by-one.

## Parameters

<i>param_u</i>	this vector holds all the numerical values for the first parameter where evaluation will take place
<i>param_v</i>	this vector holds all the numerical values for the second parameter where evaluation will take place
<i>param_w</i>	this vector holds all the numerical values for the third parameter where evaluation will take place
<i>points</i>	upon function return, this vector holds all the evaluated points

29.487.4.44 `void Go::SplineVolume::insertKnot ( int pdir, double apar )`

Insert a new knot in the knotvector of the given parameter direction

## Parameters

<i>pdir</i>	parameter direction in which to insert the knots (u=0, v=1, w=2)
<i>apar</i>	the parameter value at which a new knot will be inserted

29.487.4.45 `void Go::SplineVolume::insertKnot ( int pdir, const std::vector< double > & new_knots )`

Insert new knots in the knotvector of the given parameter direction

## Parameters

<i>pdir</i>	parameter direction in which to insert the knots (u=0, v=1, w=2)
<i>new_knots</i>	a vector containing the parameter values of the new knots to insert.

29.487.4.46 **virtual ClassType** Go::SplineVolume::instanceType ( ) const [virtual]

Return the class type identifier of a given, derived instance of [GeomObject](#).

Implements [Go::GeomObject](#).

29.487.4.47 **bool** Go::SplineVolume::isDegenerate ( int *which\_sf*, int & *type*, bool & *b*, bool & *r*, bool & *t*, bool & *l*, double *tol* ) const

Specify degeneracy

#### Parameters

<i>which_sf</i>	0=u_min, 1=u_max, 2=v_min, 3=v_max, 4=w_min, 5=w_max
<i>type</i>	0 - Not degenerate, 1 - surface degenerate in specified boundary, 2 - surface degenerate to line, 3 - surface degenerate to point
<i>b</i>	true if surface is degenerate along bottom curve (v=min for surface in parameterized)
<i>r</i>	true if surface is degenerate along right curve
<i>t</i>	true if surface is degenerate along top curve
<i>l</i>	true if surface is degenerate along left curve
<i>tol</i>	Tolerance used in test

29.487.4.48 **bool** Go::SplineVolume::isLeftHanded ( )

Test if the volume is lefthanded. Returns false if volume does not lie in 3-dimensional space. The test is done by taking the XXX of the derivatives at the centre point. Assumes volume is not self intersecting, and has linearly independent derivatives at the centre.

29.487.4.49 **virtual bool** Go::SplineVolume::isSpline ( ) const [inline],[virtual]

Check if the volume is of type spline.

Reimplemented from [Go::ParamVolume](#).

Definition at line 951 of file SplineVolume.h.

29.487.4.50 **double** Go::SplineVolume::knotSpan ( int *pardir*, int *iknot* ) const

Returns the size of the knot interval (knot[*iknot*],knot[*iknot*+1]) in the specified direction. An index outside the legal range will result in a zero knot span

29.487.4.51 **void** Go::SplineVolume::makeBernsteinKnots ( int *pardir* )

Inserts knots in the specified knot vector, such that all knots have multiplicity order



## Parameters

<i>pardir</i>	the parameter direction (0, 1, or 2)
---------------	--------------------------------------

29.487.4.52 `virtual double Go::SplineVolume::nextSegmentVal ( int dir, double par, bool forward, double tol ) const`  
`[virtual]`

Determine the parameter value of the start of the 'next segment' from a parameter value, along a given parameter direction. A 'segment' is here defined as a parameter interval in which there will be no discontinuities in derivatives or other artifacts. For spline objects, a segment will typically be the interval between two consecutive, non-coincident knots.

## Parameters

<i>dir</i>	the parameter direction in which we search for the next segment (0, 1 or 2)
<i>par</i>	the parameter value starting from which we search for the start value of the next segment
<i>forward</i>	define whether we shall move forward ('true') or backwards when searching along this parameter
<i>tol</i>	tolerance used for determining whether the 'par' is already located <i>on</i> the next segment value

## Returns

the value of the start value of the next segment (or the end of the previous segment, if we are moving backwards...)

Implements [Go::ParamVolume](#).

29.487.4.53 `int Go::SplineVolume::numberOfPatches ( int pardir ) const`

Returns the number of knot intervals in the specified knot vector.

## Parameters

<i>pardir</i>	parameter direction
---------------	---------------------

## Returns

the number of knot intervals in the knotvector for the specified parameter direction

29.487.4.54 `int Go::SplineVolume::numCoefs ( int pardir ) const` `[inline]`

Query the number of control points along the specified parameter direction

## Returns

the number of control points along the specified parameter direction

Definition at line 480 of file SplineVolume.h.

29.487.4.55 `int Go::SplineVolume::numElem ( ) const [inline]`

Query the number of elements in the [SplineVolume](#).

Definition at line 497 of file `SplineVolume.h`.

29.487.4.56 `int Go::SplineVolume::numElem ( int pardir ) const [inline]`

Query the number of elements in one parameter direction of.

`pardir = 0`: u-direction, `pardir = 1`: vdirection, `pardir = 2`: wdirection

Definition at line 505 of file `SplineVolume.h`.

29.487.4.57 `int Go::SplineVolume::order ( int pardir ) const [inline]`

Query the order of the [BsplineBasis](#) for the specified parameter

#### Returns

the order of the [BsplineBasis](#) for the specified parameter

Definition at line 489 of file `SplineVolume.h`.

29.487.4.58 `virtual const Array<double,6> Go::SplineVolume::parameterSpan ( ) const [virtual]`

Return the parameter domain of the volume. This is an array containing the start and end parameter values. The spline's parameter  $i$  has its start value at the array position  $(2i)$  and its end parameter value at the array position  $(2i+1)$

#### Returns

An array describing the parametric domain of the volume

Implements [Go::ParamVolume](#).

29.487.4.59 `virtual void Go::SplineVolume::point ( Point & pt, double upar, double vpar, double wpar ) const [virtual]`

Evaluates the volume's position for a given parameter triple.

#### Parameters

<i>pt</i>	the result of the evaluation is written here
<i>upar</i>	the first parameter
<i>vpar</i>	the second parameter
<i>wpar</i>	the third parameter

Implements [Go::ParamVolume](#).

```
29.487.4.60 virtual void Go::SplineVolume::point (std::vector< Point > & pts, double upar, double vpar, double wpar,
int derivs, bool u_from_right = true, bool v_from_right = true, bool w_from_right = true, double resolution =
1.0e-12) const [virtual]
```

Evaluates the volume's position and a certain number of derivatives for a given parameter triple.

#### Parameters

<i>pts</i>	the vector containing the evaluated values. Its size must be set by the user prior to calling this function. Upon completion of the function, its first entry is the volume's position at the given parameter pair. Then, if 'derivs' > 0, the two next entries will be the volume tangents along the first, second and third parameter direction. The next three entries are the second- and cross derivatives, in the order (du2, dudv, dudw, dv2, dvdw, dw2), and similar for even higher derivatives.
<i>upar</i>	the first parameter
<i>vpar</i>	the second parameter
<i>wpar</i>	the third parameter
<i>derivs</i>	number of requested derivatives
<i>u_from_right</i>	specify whether derivatives along the first parameter are to be calculated from the right ('true', default) or from the left ('false')
<i>v_from_right</i>	specify whether derivatives along the second parameter are to be calculated from the right ('true', default) or from the left ('false')
<i>w_from_right</i>	specify whether derivatives along the third parameter are to be calculated from the right ('true', default) or from the left ('false')
<i>resolution</i>	tolerance used when determining whether parameters are located at special values of the parameter domain (in particular; knot values in case of spline objects).

Implements [Go::ParamVolume](#).

```
29.487.4.61 void Go::SplineVolume::pointsGrid (int numu, int numv, int numw, std::vector< double >::iterator basisvals_u,
std::vector< double >::iterator basisvals_v, std::vector< double >::iterator basisvals_w, std::vector< int
>::iterator knotinter_u, std::vector< int >::iterator knotinter_v, std::vector< int >::iterator knotinter_w, int
derivs, std::vector< double > & points) const
```

```
29.487.4.62 void Go::SplineVolume::raiseOrder (int raise_u, int raise_v, int raise_w)
```

Raise the order of the spline volume as indicated by parameters.

#### Parameters

<i>raise<sub>u</sub></i>	the order of the <a href="#">BsplineBasis</a> associated with the first parameter will be raised this many times.
<i>raise<sub>v</sub></i>	the order of the <a href="#">BsplineBasis</a> associated with the second parameter will be raised this many times.
<i>raise<sub>w</sub></i>	the order of the <a href="#">BsplineBasis</a> associated with the third parameter will be raised this many times.

**29.487.4.63** `bool Go::SplineVolume::rational ( ) const [inline]`

Query whether the volume is rational

**Returns**

'true' if the volume is rational, 'false' otherwise

Definition at line 514 of file SplineVolume.h.

**29.487.4.64** `std::vector<double>::iterator Go::SplineVolume::rcoefs_begin ( ) [inline]`

Get an iterator to the start of the internal array of *rational* control points.

**Returns**

an (nonconst) iterator to the start of the internal array of rational control points.

Definition at line 549 of file SplineVolume.h.

**29.487.4.65** `std::vector<double>::const_iterator Go::SplineVolume::rcoefs_begin ( ) const [inline]`

Get a const iterator to the start of the internal array of *rational* control points.

**Returns**

a const iterator to the start of the internal array of rational control points

Definition at line 563 of file SplineVolume.h.

**29.487.4.66** `std::vector<double>::iterator Go::SplineVolume::rcoefs_end ( ) [inline]`

Get an iterator to the one-past-end position of the internal array of *rational* control points.

**Returns**

an (nonconst) iterator to the start of the internal array of rational control points.

Definition at line 556 of file SplineVolume.h.

**29.487.4.67** `std::vector<double>::const_iterator Go::SplineVolume::rcoefs_end ( ) const [inline]`

Get a const iterator to the one-past-end position of the internal array of *rational* control points.

**Returns**

a const iterator to the one-past-end position of the internal array of rational control points.

Definition at line 570 of file SplineVolume.h.

**29.487.4.68** `virtual void Go::SplineVolume::read ( std::istream & is ) [virtual]`

read object from stream

## Parameters

<i>is</i>	stream from which object is read
-----------	----------------------------------

Implements [Go::Streamable](#).

29.487.4.69 void `Go::SplineVolume::removeKnot ( int pardir, double tpar )`

Remove a knot from the knotvector of the given parameter direction.

## Parameters

<i>pardir</i>	the parameter direction (0, 1 or 2)
<i>tpar</i>	the parameter value of the knot to be removed

29.487.4.70 void `Go::SplineVolume::replaceCoefficient ( int ix, Point coef )`

Replace one specified coefficient (local enumeration)

29.487.4.71 virtual void `Go::SplineVolume::reverseParameterDirection ( int pardir )` [virtual]

Reverses the direction of the basis in input direction.

## Parameters

<i>pardir</i>	which parameter direction to reverse
---------------	--------------------------------------

Implements [Go::ParamVolume](#).

29.487.4.72 void `Go::SplineVolume::scale ( double fac )`

Scale the coefficients of the volume with a given factor.

29.487.4.73 void `Go::SplineVolume::setParameterDomain ( double u1, double u2, double v1, double v2, double w1, double w2 )`

Set the parameter domain to a given box

## Parameters

<i>u1</i>	new min. value of first parameter span
<i>u2</i>	new max. value of first parameter span
<i>v1</i>	new min. value of second parameter span
<i>v2</i>	new max. value of second parameter span
<i>w1</i>	new min. value of third parameter span
<i>w2</i>	new max. value of third parameter span

29.487.4.74 `std::vector<shared_ptr<SplineVolume> > Go::SplineVolume::split ( std::vector< double > & param, int pdir, double fuzzy = DEFAULT_PARAMETER_EPSILON ) const`

29.487.4.75 `virtual double Go::SplineVolume::startparam ( int i ) const` [virtual]

Get the start value for the specified parameter direction.

#### Parameters

<i>i</i>	the parameter direction
----------	-------------------------

#### Returns

the start value for the parameter direction given by the parameter pdir

29.487.4.76 `SplineVolume* Go::SplineVolume::subVolume ( double from_upar, double from_vpar, double from_wpar, double to_upar, double to_vpar, double to_wpar, double fuzzy = DEFAULT_PARAMETER_EPSILON ) const`

Get a [SplineVolume](#) which represents a part of 'this' [SplineVolume](#)

#### Parameters

<i>from_upar</i>	start value for u-parameter in the sub-volume to be generated
<i>from_vpar</i>	start value for v-parameter in the sub-volume to be generated
<i>from_wpar</i>	start value for w-parameter in the sub-volume to be generated
<i>to_upar</i>	end value for u-parameter in the sub-volume to be generated
<i>to_vpar</i>	end value for v-parameter in the sub-volume to be generated
<i>to_wpar</i>	end value for w-parameter in the sub-volume to be generated
<i>fuzzy</i>	tolerance used to determine whether given parameter values are located directly <i>knot</i> values.

29.487.4.77 `void Go::SplineVolume::swap ( SplineVolume & other )`

quick swap of two [SplineVolume](#) objects with each other

29.487.4.78 `virtual void Go::SplineVolume::swapParameterDirection ( int pdir1, int pdir2 )` [virtual]

Swaps two parameter directions

#### Parameters

<i>pdir1</i>	One of the parameter directions (0, 1 or 2) to be swapped
<i>pdir2</i>	The other parameter direction (0, 1 or 2) to be swapped

Implements [Go::ParamVolume](#).

29.487.4.79 `virtual DirectionCone Go::SplineVolume::tangentCone ( int pardir ) const` `[virtual]`

Creates a [DirectionCone](#) covering all tangents to this volume along a given parameter direction.

#### Parameters

<i>pardir</i>	if 1, the <a href="#">DirectionCone</a> will be defined on basis of the volume's tangents along the first parameter direction. If 2, the second parameter direction will be used. If 3, the third parameter direction will be used.
---------------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

#### Returns

a [DirectionCone](#) (not necessarily the smallest) containing all tangents to this volume along the specified parameter direction.

Implements [Go::ParamVolume](#).

29.487.4.80 `virtual void Go::SplineVolume::translate ( const Point & vec )` `[virtual]`

Translate.

Implements [Go::ParamVolume](#).

29.487.4.81 `int Go::SplineVolume::volumePeriodicity ( int pardir, double epsilon ) const`

Check if a volume is open, closed or parametric in a given parameter direction

#### Parameters

<i>pardir</i>	The parameter direction in which to check periodicity
<i>epsilon</i>	Tolerance used in computation \ return 0 - Open, i.e. multiple knots in end parameters and not closed 1 - Closed, multiple knots in end parameters > 1 - Periodic knot vector with position and (result-1) derivative equal across the seam

29.487.4.82 `virtual void Go::SplineVolume::write ( std::ostream & os ) const` `[virtual]`

write object to stream

#### Parameters

<i>os</i>	stream to which object is written
-----------	-----------------------------------

Implements [Go::Streamable](#).

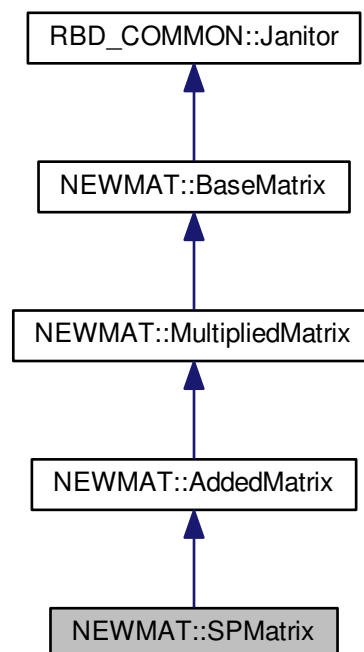
The documentation for this class was generated from the following file:

- [trivariate/include/GoTools/trivariate/SplineVolume.h](#)

## 29.488 NEWMAT::SPMatrix Class Reference

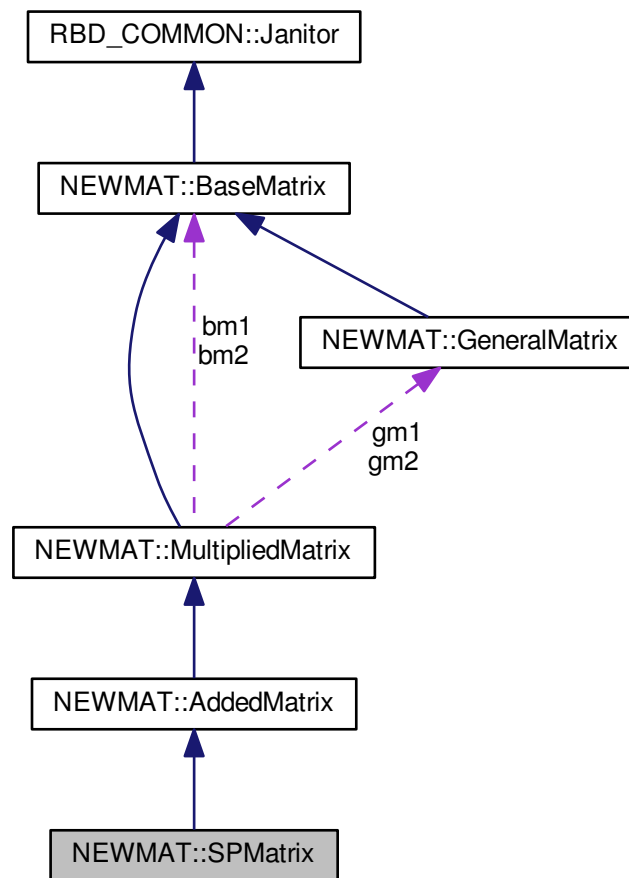
```
#include <newmat.h>
```

Inheritance diagram for NEWMAT::SPMatrix:





Collaboration diagram for NEWMAT::SPMatrix:



### Public Member Functions

- `~SPMatrix ()`
- `GeneralMatrix * Evaluate (MatrixType mt=MatrixTypeUnSp)`
- `MatrixBandWidth BandWidth () const`

### Protected Member Functions

- `SPMatrix (const BaseMatrix *bm1x, const BaseMatrix *bm2x)`

### Friends

- class `BaseMatrix`
- class `GeneralMatrix`
- class `GenericMatrix`
- `SPMatrix SP (const BaseMatrix &, const BaseMatrix &)`

## Additional Inherited Members

### 29.488.1 Detailed Description

Definition at line 1210 of file newmat.h.

### 29.488.2 Constructor & Destructor Documentation

29.488.2.1 **NEWMAT::SPMatrix::SPMatrix** ( **const BaseMatrix \* bm1x**, **const BaseMatrix \* bm2x** ) [*inline*],  
[*protected*]

Definition at line 1213 of file newmat.h.

29.488.2.2 **NEWMAT::SPMatrix::~~SPMatrix** ( ) [*inline*]

Definition at line 1220 of file newmat.h.

### 29.488.3 Member Function Documentation

29.488.3.1 **MatrixBandWidth** **SPMatrix::BandWidth** ( ) **const** [*virtual*]

Reimplemented from [NEWMAT::AddedMatrix](#).

Definition at line 421 of file newmat4.cpp.

29.488.3.2 **GeneralMatrix \* SPMatrix::Evaluate** ( **MatrixType mt = MatrixTypeUnSp** ) [*virtual*]

Reimplemented from [NEWMAT::AddedMatrix](#).

Definition at line 681 of file newmat7.cpp.

### 29.488.4 Friends And Related Function Documentation

29.488.4.1 **friend class BaseMatrix** [*friend*]

Definition at line 1216 of file newmat.h.

29.488.4.2 **friend class GeneralMatrix** [*friend*]

Definition at line 1217 of file newmat.h.

29.488.4.3 friend class **GenericMatrix** [friend]

Definition at line 1218 of file newmat.h.

29.488.4.4 **SPMatrix** SP ( const **BaseMatrix** & , const **BaseMatrix** & ) [friend]

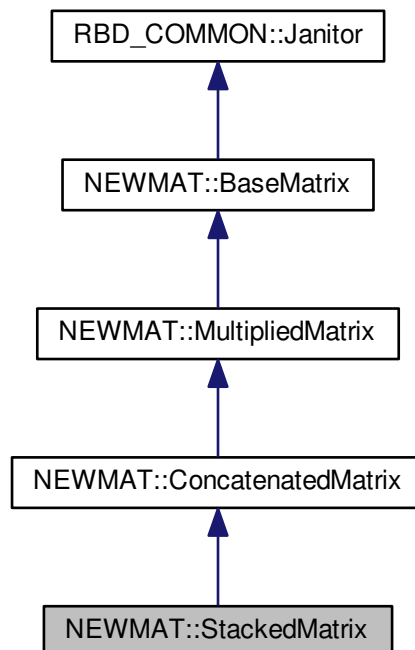
The documentation for this class was generated from the following files:

- [newmat/include/newmat.h](#)
- [newmat/src/newmat4.cpp](#)
- [newmat/src/newmat7.cpp](#)

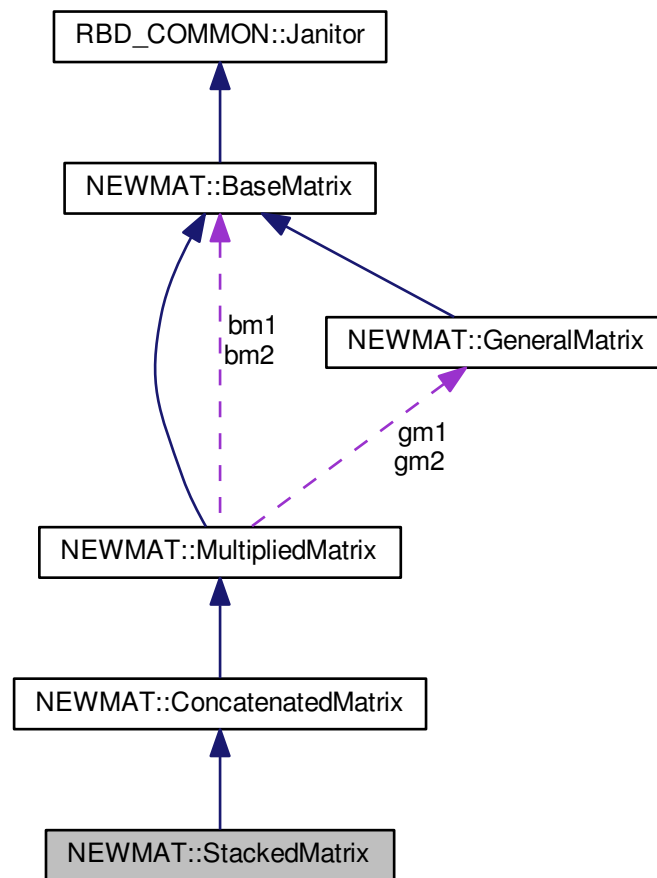
## 29.489 NEWMAT::StackedMatrix Class Reference

```
#include <newmat.h>
```

Inheritance diagram for NEWMAT::StackedMatrix:



Collaboration diagram for NEWMAT::StackedMatrix:



### Public Member Functions

- `~StackedMatrix ()`
- `GeneralMatrix * Evaluate (MatrixType mt=MatrixTypeUnSp)`

### Protected Member Functions

- `StackedMatrix (const BaseMatrix *bm1x, const BaseMatrix *bm2x)`

### Friends

- class `BaseMatrix`
- class `GeneralMatrix`
- class `GenericMatrix`

## Additional Inherited Members

### 29.489.1 Detailed Description

Definition at line 1270 of file newmat.h.

### 29.489.2 Constructor & Destructor Documentation

29.489.2.1 **NEWMAT::StackedMatrix::StackedMatrix** ( **const BaseMatrix \* bm1x**, **const BaseMatrix \* bm2x** )  
[inline], [protected]

Definition at line 1273 of file newmat.h.

29.489.2.2 **NEWMAT::StackedMatrix::~StackedMatrix** ( ) [inline]

Definition at line 1280 of file newmat.h.

### 29.489.3 Member Function Documentation

29.489.3.1 **GeneralMatrix \* StackedMatrix::Evaluate** ( **MatrixType mt = MatrixTypeUnSp** ) [virtual]

Reimplemented from [NEWMAT::ConcatenatedMatrix](#).

Definition at line 835 of file newmat7.cpp.

### 29.489.4 Friends And Related Function Documentation

29.489.4.1 **friend class BaseMatrix** [friend]

Definition at line 1276 of file newmat.h.

29.489.4.2 **friend class GeneralMatrix** [friend]

Definition at line 1277 of file newmat.h.

29.489.4.3 **friend class GenericMatrix** [friend]

Definition at line 1278 of file newmat.h.

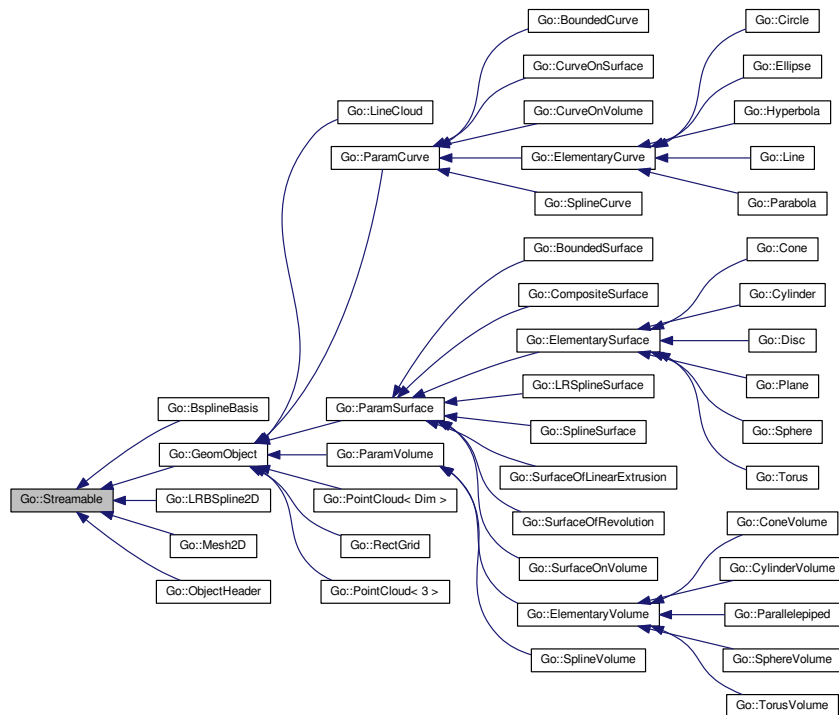
The documentation for this class was generated from the following files:

- [newmat/include/newmat.h](#)
- [newmat/src/newmat7.cpp](#)

## 29.490 Go::Streamable Class Reference

```
#include <Streamable.h>
```

Inheritance diagram for Go::Streamable:



### Classes

- class [EofException](#)

### Public Member Functions

- virtual [~Streamable](#) ()
- virtual void [read](#) (std::istream &is)=0
- virtual void [write](#) (std::ostream &os) [const](#) =0

### 29.490.1 Detailed Description

Base class for streamable objects, i.e., objects which can be read from and written to a stream.

Definition at line 56 of file Streamable.h.

## 29.490.2 Constructor & Destructor Documentation

29.490.2.1 virtual Go::Streamable::~Streamable ( ) [virtual]

## 29.490.3 Member Function Documentation

29.490.3.1 virtual void Go::Streamable::read ( std::istream & *is* ) [pure virtual]

read object from stream

## Parameters

<i>is</i>	stream from which object is read
-----------	----------------------------------

Implemented in [Go::SplineVolume](#), [Go::SplineSurface](#), [Go::LRSplineSurface](#), [Go::PointCloud< Dim >](#), [Go::PointCloud< 3 >](#), [Go::SplineCurve](#), [Go::BoundedSurface](#), [Go::CurveOnSurface](#), [Go::BsplineBasis](#), [Go::LRBSpline2D](#), [Go::SurfaceOnVolume](#), [Go::Disc](#), [Go::CurveOnVolume](#), [Go::ObjectHeader](#), [Go::Torus](#), [Go::RectGrid](#), [Go::SurfaceOfRevolution](#), [Go::Mesh2D](#), [Go::Plane](#), [Go::Cone](#), [Go::Cylinder](#), [Go::LineCloud](#), [Go::BoundedCurve](#), [Go::Sphere](#), [Go::TorusVolume](#), [Go::ConeVolume](#), [Go::CylinderVolume](#), [Go::SphereVolume](#), [Go::Line](#), [Go::Parallelepiped](#), [Go::Circle](#), [Go::Hyperbola](#), [Go::Parabola](#), [Go::Ellipse](#), [Go::SurfaceOfLinearExtrusion](#), and [Go::CompositeSurface](#).

**29.490.3.2** `virtual void Go::Streamable::write ( std::ostream & os ) const` [pure virtual]

write object to stream

## Parameters

<i>os</i>	stream to which object is written
-----------	-----------------------------------

Implemented in [Go::SplineVolume](#), [Go::SplineSurface](#), [Go::PointCloud< Dim >](#), [Go::PointCloud< 3 >](#), [Go::LRSplineSurface](#), [Go::SplineCurve](#), [Go::BoundedSurface](#), [Go::CurveOnSurface](#), [Go::BsplineBasis](#), [Go::SurfaceOnVolume](#), [Go::LRBSpline2D](#), [Go::Disc](#), [Go::CurveOnVolume](#), [Go::ObjectHeader](#), [Go::Torus](#), [Go::RectGrid](#), [Go::SurfaceOfRevolution](#), [Go::Mesh2D](#), [Go::Plane](#), [Go::Cone](#), [Go::Cylinder](#), [Go::LineCloud](#), [Go::BoundedCurve](#), [Go::Sphere](#), [Go::TorusVolume](#), [Go::ConeVolume](#), [Go::CylinderVolume](#), [Go::SphereVolume](#), [Go::Line](#), [Go::Parallelepiped](#), [Go::Circle](#), [Go::Hyperbola](#), [Go::Parabola](#), [Go::CompositeSurface](#), [Go::Ellipse](#), and [Go::SurfaceOfLinearExtrusion](#).

The documentation for this class was generated from the following file:

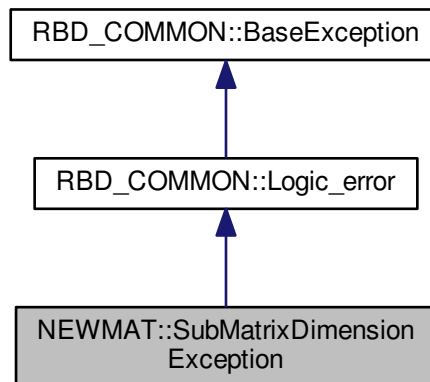
- [gotools-core/include/GoTools/geometry/Streamable.h](#)

## 29.491 NEWMAT::SubMatrixDimensionException Class Reference

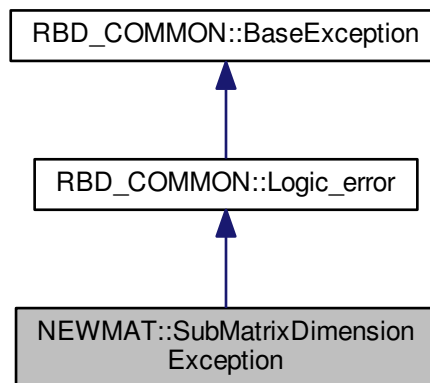
```
#include <newmat.h>
```



Inheritance diagram for NEWMAT::SubMatrixDimensionException:



Collaboration diagram for NEWMAT::SubMatrixDimensionException:



## Public Member Functions

- [SubMatrixDimensionException](#) ()

## Static Public Attributes

- static unsigned long [Select](#)

## Additional Inherited Members

### 29.491.1 Detailed Description

Definition at line 1662 of file newmat.h.

### 29.491.2 Constructor & Destructor Documentation

#### 29.491.2.1 SubMatrixDimensionException::SubMatrixDimensionException ( )

Definition at line 152 of file newmatex.cpp.

### 29.491.3 Member Data Documentation

#### 29.491.3.1 unsigned long SubMatrixDimensionException::Select [static]

Definition at line 1665 of file newmat.h.

The documentation for this class was generated from the following files:

- newmat/include/newmat.h
- newmat/src/newmatex.cpp

## 29.492 Go::boxStructuring::SubSurfaceBoundingBox Class Reference

```
#include <ClosestPointUtils.h>
```

### Public Member Functions

- [SubSurfaceBoundingBox](#) ([shared\\_ptr](#)< [SurfaceData](#) > [surface\\_data](#), int [pos\\_u](#), int [pos\\_v](#), [BoundingBox](#) [box](#), [shared\\_ptr](#)< [RectDomain](#) > [par\\_domain](#))
- void [setInside](#) ([bool](#) [inside](#))
  - Set whether we know for sure this segment is entirely inside the parameter domain.*
- [bool](#) [inside](#) () [const](#)
  - Tell if the entire segment is known to be inside the parameter domain.*
- [bool](#) [inside](#) ([double](#) [par\\_u](#), [double](#) [par\\_v](#)) [const](#)
- int [pos\\_u](#) () [const](#)
  - Get the segment position in the first direction in segment mesh of the surface.*
- int [pos\\_v](#) () [const](#)
  - Get the segment position in the second direction in segment mesh of the surface.*
- [BoundingBox](#) [box](#) () [const](#)
  - Get the geometry space bounding box of the image of the segment.*
- [shared\\_ptr](#)< [SurfaceData](#) > [surface\\_data](#) () [const](#)
  - Get the structure data of the surface.*
- [shared\\_ptr](#)< [RectDomain](#) > [par\\_domain](#) () [const](#)
  - Get the domain of the segment in the surface parameter domain.*
- void [add\\_polygon\\_corners](#) ([double](#) [par\\_u](#), [double](#) [par\\_v](#))
- [bool](#) [has\\_polygon](#) () [const](#)
  - Tell if any of the generated polygon inside the parameter domain hits this segment.*
- int [size\\_polygon](#) () [const](#)
  - Get the number of polygon points.*
- void [remove\\_polygon](#) ()
  - Remove the polygon information.*

### 29.492.1 Detailed Description

Class for the preprocessed data on a specific segment on the tensor mesh sub structure on the parameter space of a surface in the surface model

Definition at line 165 of file ClosestPointUtils.h.

### 29.492.2 Constructor & Destructor Documentation

**29.492.2.1** `Go::boxStructuring::SubSurfaceBoundingBox::SubSurfaceBoundingBox ( shared_ptr< SurfaceData > surface_data, int pos_u, int pos_v, BoundingBox box, shared_ptr< RectDomain > par_domain )`  
`[inline]`

Constructor

- `surface_data` is the surface data of the surface for this segment
- `pos_u` and `pos_v` are the positions of this segment in the segment tensor mesh of the surface
- `box` is the bounding box in the geometry space of the surface
- `par_domain` holds the parameter domain of the segment (the bounding box in the parameter space)

Definition at line 174 of file ClosestPointUtils.h.

### 29.492.3 Member Function Documentation

**29.492.3.1** `void Go::boxStructuring::SubSurfaceBoundingBox::add_polygon_corners ( double par_u, double par_v )`  
`[inline]`

Add a point from the generated polygon inside the parameter domain of the surface. The point is either inside or on the boundary of the segment.

Definition at line 296 of file ClosestPointUtils.h.

**29.492.3.2** `BoundingBox Go::boxStructuring::SubSurfaceBoundingBox::box ( ) const` `[inline]`

Get the geometry space bounding box of the image of the segment.

Definition at line 277 of file ClosestPointUtils.h.

**29.492.3.3** `bool Go::boxStructuring::SubSurfaceBoundingBox::has_polygon ( ) const` `[inline]`

Tell if any of the generated polygon inside the parameter domain hits this segment.

Definition at line 303 of file ClosestPointUtils.h.

29.492.3.4 `bool Go::boxStructuring::SubSurfaceBoundingBox::inside ( ) const` `[inline]`

Tell if the entire segment is known to be inside the parameter domain.

Definition at line 186 of file `ClosestPointUtils.h`.

29.492.3.5 `bool Go::boxStructuring::SubSurfaceBoundingBox::inside ( double par_u, double par_v ) const` `[inline]`

Tell if a specific point is guaranteed to be inside the parameter domain, by testing against an inside polygon of the entire parameter domain. Should always return false if the point is outside Should in most cases return true if the point is inside

Definition at line 195 of file `ClosestPointUtils.h`.

29.492.3.6 `shared_ptr<RectDomain> Go::boxStructuring::SubSurfaceBoundingBox::par_domain ( ) const` `[inline]`

Get the domain of the segment in the surface parameter domain.

Definition at line 289 of file `ClosestPointUtils.h`.

29.492.3.7 `int Go::boxStructuring::SubSurfaceBoundingBox::pos_u ( ) const` `[inline]`

Get the segment position in the first direction in segment mesh of the surface.

Definition at line 265 of file `ClosestPointUtils.h`.

29.492.3.8 `int Go::boxStructuring::SubSurfaceBoundingBox::pos_v ( ) const` `[inline]`

Get the segment position in the second direction in segment mesh of the surface.

Definition at line 271 of file `ClosestPointUtils.h`.

29.492.3.9 `void Go::boxStructuring::SubSurfaceBoundingBox::remove_polygon ( )` `[inline]`

Remove the polygon information.

Definition at line 315 of file `ClosestPointUtils.h`.

29.492.3.10 `void Go::boxStructuring::SubSurfaceBoundingBox::setInside ( bool inside )` `[inline]`

Set whether we know for sure this segment is entirely inside the parameter domain.

Definition at line 180 of file `ClosestPointUtils.h`.

```
29.492.3.11 int Go::boxStructuring::SubSurfaceBoundingBox::size_polygon () const [inline]
```

Get the number of polygon points.

Definition at line 309 of file ClosestPointUtils.h.

```
29.492.3.12 shared_ptr<SurfaceData> Go::boxStructuring::SubSurfaceBoundingBox::surface_data () const [inline]
```

Get the structure data of the surface.

Definition at line 283 of file ClosestPointUtils.h.

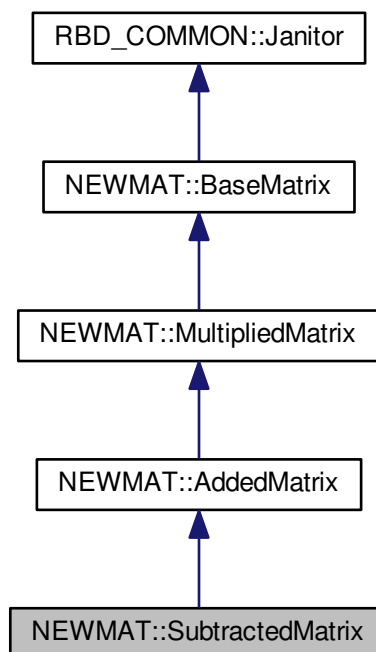
The documentation for this class was generated from the following file:

- [gotools-core/include/GoTools/Utils/ClosestPointUtils.h](#)

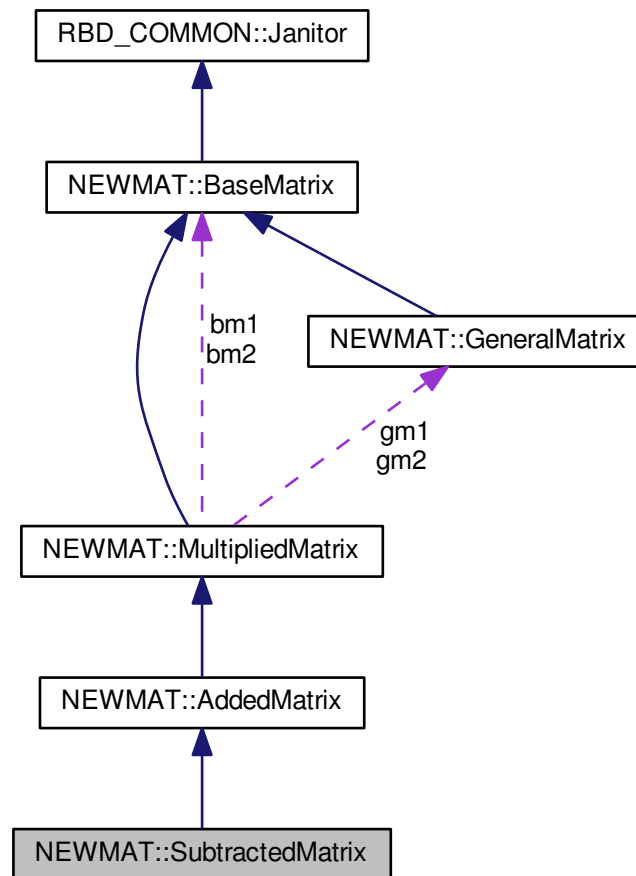
## 29.493 NEWMAT::SubtractedMatrix Class Reference

```
#include <newmat.h>
```

Inheritance diagram for NEWMAT::SubtractedMatrix:



Collaboration diagram for NEWMAT::SubtractedMatrix:



### Public Member Functions

- `~SubtractedMatrix ()`
- `GeneralMatrix * Evaluate (MatrixType mt=MatrixTypeUnSp)`

### Friends

- class `BaseMatrix`
- class `GeneralMatrix`
- class `GenericMatrix`

### Additional Inherited Members

#### 29.493.1 Detailed Description

Definition at line 1298 of file newmat.h.

## 29.493.2 Constructor & Destructor Documentation

29.493.2.1 `NEWMAT::SubtractedMatrix::~~SubtractedMatrix ( )` `[inline]`

Definition at line 1306 of file `newmat.h`.

## 29.493.3 Member Function Documentation

29.493.3.1 `GeneralMatrix * SubtractedMatrix::Evaluate ( MatrixType mt = MatrixTypeUnSp )` `[virtual]`

Reimplemented from [NEWMAT::AddedMatrix](#).

Definition at line 594 of file `newmat7.cpp`.

## 29.493.4 Friends And Related Function Documentation

29.493.4.1 `friend class BaseMatrix` `[friend]`

Definition at line 1302 of file `newmat.h`.

29.493.4.2 `friend class GeneralMatrix` `[friend]`

Definition at line 1303 of file `newmat.h`.

29.493.4.3 `friend class GenericMatrix` `[friend]`

Definition at line 1304 of file `newmat.h`.

The documentation for this class was generated from the following files:

- `newmat/include/newmat.h`
- `newmat/src/newmat7.cpp`

## 29.494 Go::LRSplineUtils::support\_compare Struct Reference

```
#include <LRSplineUtils.h>
```

### Public Member Functions

- `bool operator() (const LRBSpline2D *b1, const LRBSpline2D *b2) const`

### 29.494.1 Detailed Description

Definition at line 158 of file LRSplineUtils.h.

### 29.494.2 Member Function Documentation

29.494.2.1 `bool Go::LRSplineUtils::support_compare::operator() ( const LRBSpline2D * b1, const LRBSpline2D * b2 ) const [inline]`

Definition at line 161 of file LRSplineUtils.h.

The documentation for this struct was generated from the following file:

- [Irsplines2D/include/GoTools/Irsplines2D/LRSplineUtils.h](#)

## 29.495 Go::SurfaceAssembly Class Reference

```
#include <SurfaceAssembly.h>
```

### Public Member Functions

- [SurfaceAssembly](#) (shared\_ptr< [ParamSurfaceInt](#) > surf, std::vector< std::pair< [double](#), int > > u\_div, std::vector< std::pair< [double](#), int > > v\_div, std::vector< [RectDomain](#) > sing\_domain, [double](#) rel\_par\_res)  
*Constructor.*
- [~SurfaceAssembly](#) ()  
*Destructor.*
- void [resetSubIndex](#) (int idx=0)
- void [resetAssemblyIndex](#) ()
- int [getNmbSubSurface](#) ()
- bool [getNextSubSurface](#) (shared\_ptr< [ParamSurfaceInt](#) > &sub\_sf, int &idx, int &sing\_idx)
- bool [getNextAssembly](#) (shared\_ptr< [ParamSurfaceInt](#) > &assembly, int &idx, bool &potential\_sing)
- bool [subSfNeighbour](#) (int idx1, int idx2)
- bool [doTouch](#) (int idx1, int idx2)
- bool [touchAtSingularity](#) (int idx1, int idx2)
- bool [isInPrevAssembly](#) (int idx1, int idx2)
- bool [isInFirstAssembly](#) (shared\_ptr< [ParamSurfaceInt](#) > sub\_srf)
- int [getSubSurfaceIndex](#) (shared\_ptr< [ParamSurfaceInt](#) > sub\_srf, bool &at\_end)
- std::vector< std::pair< [double](#), int > > [getUdiv](#) ()
- std::vector< std::pair< [double](#), int > > [getVdiv](#) ()

### 29.495.1 Detailed Description

Collection of sub surfaces used in computation of surface self intersections

Definition at line 50 of file SurfaceAssembly.h.



## 29.495.2 Constructor & Destructor Documentation

29.495.2.1 `Go::SurfaceAssembly::SurfaceAssembly ( shared_ptr< ParamSurfaceInt > surf, std::vector< std::pair< double, int > > u_div, std::vector< std::pair< double, int > > v_div, std::vector< RectDomain > sing_domain, double rel_par_res )`

Constructor.

29.495.2.2 `Go::SurfaceAssembly::~~SurfaceAssembly ( )`

Destructor.

## 29.495.3 Member Function Documentation

29.495.3.1 `bool Go::SurfaceAssembly::doTouch ( int idx1, int idx2 )`

29.495.3.2 `bool Go::SurfaceAssembly::getNextAssembly ( shared_ptr< ParamSurfaceInt > & assembly, int & idx, bool & potential_sing )`

29.495.3.3 `bool Go::SurfaceAssembly::getNextSubSurface ( shared_ptr< ParamSurfaceInt > & sub_sf, int & idx, int & sing_idx )`

29.495.3.4 `int Go::SurfaceAssembly::getNmbSubSurface ( )`

29.495.3.5 `int Go::SurfaceAssembly::getSubSurfaceIndex ( shared_ptr< ParamSurfaceInt > sub_srf, bool & at_end )`

29.495.3.6 `std::vector<std::pair<double,int> > Go::SurfaceAssembly::getUdiv ( ) [inline]`

Definition at line 95 of file SurfaceAssembly.h.

29.495.3.7 `std::vector<std::pair<double,int> > Go::SurfaceAssembly::getVdiv ( ) [inline]`

Definition at line 100 of file SurfaceAssembly.h.

29.495.3.8 `bool Go::SurfaceAssembly::isInFirstAssembly ( shared_ptr< ParamSurfaceInt > sub_srf )`

29.495.3.9 `bool Go::SurfaceAssembly::isInPrevAssembly ( int idx1, int idx2 )`

29.495.3.10 `void Go::SurfaceAssembly::resetAssemblyIndex ( ) [inline]`

Definition at line 67 of file SurfaceAssembly.h.

29.495.3.11 `void Go::SurfaceAssembly::resetSubIndex ( int idx = 0 ) [inline]`

Definition at line 64 of file SurfaceAssembly.h.

### 29.495.3.12 bool Go::SurfaceAssembly::subSfNeighbour ( int *idx1*, int *idx2* )

This function makes a check on whether two sub surfaces are neighbours, but if the surfaces meet at a singularity they are NOT classified as neighbours

### 29.495.3.13 bool Go::SurfaceAssembly::touchAtSingularity ( int *idx1*, int *idx2* )

The documentation for this class was generated from the following file:

- intersections/include/GoTools/intersections/[SurfaceAssembly.h](#)

## 29.496 Go::boxStructuring::SurfaceData Class Reference

```
#include <ClosestPointUtils.h>
```

### Public Member Functions

- [SurfaceData](#) (shared\_ptr< [ParamSurface](#) > *surface*)  
*Constructor.*
- void [setSegments](#) (int *segs\_u*, int *segs\_v*)  
*Set the number of segments in the mesh structure in both parameter directions.*
- int [segs\\_u](#) () const  
*Get the number of segments in first parameter direction.*
- int [segs\\_v](#) () const  
*Get the number of segments in second parameter direction.*
- void [setIndex](#) (int *index*)  
*Set the index of this surface in the [BoundingBoxStructure](#) instance.*
- int [index](#) () const  
*Get the index of this surface in the [BoundingBoxStructure](#) instance.*
- void [setSurfaceCopies](#) (int *nmb\_copies*)  
*Set number of surface copies.*
- shared\_ptr< [ParamSurface](#) > [surface](#) (int *idx*) const  
*Get a specific copy of the surface.*
- void [add\\_inside\\_point](#) ([Point](#) *pt*)  
*Add internal surface point.*
- std::vector< [Point](#) > [inside\\_points](#) () const  
*Get the internal surface points.*

### 29.496.1 Detailed Description

Class for preprocessed data on a specific surface in the model The surface parameter domain is split into a tensor mesh of sub segments. The splitting is neither required to be regular nor follow knot vector lines (when the underlying surface is a spline surface), but the current code creating the mesh structure chooses regular splittings for elementary surfaces and internal knot vector values for spline surfaces. For trimmed surfaces ([BoundedSurface](#) instances), the structure is created on the underlying surface, thus a segment might be either entirely inside the parameter domain, interely outside the parameter domain, or split by the boundary

Definition at line 70 of file [ClosestPointUtils.h](#).

## 29.496.2 Constructor & Destructor Documentation

29.496.2.1 `Go::boxStructuring::SurfaceData::SurfaceData ( shared_ptr< ParamSurface > surface )` `[inline]`

Constructor.

Definition at line 76 of file `ClosestPointUtils.h`.

## 29.496.3 Member Function Documentation

29.496.3.1 `void Go::boxStructuring::SurfaceData::add_inside_point ( Point pt )` `[inline]`

Add internal surface point.

Definition at line 130 of file `ClosestPointUtils.h`.

29.496.3.2 `int Go::boxStructuring::SurfaceData::index ( ) const` `[inline]`

Get the index of this surface in the [BoundingBoxStructure](#) instance.

Definition at line 107 of file `ClosestPointUtils.h`.

29.496.3.3 `std::vector<Point> Go::boxStructuring::SurfaceData::inside_points ( ) const` `[inline]`

Get the internal surface points.

Definition at line 136 of file `ClosestPointUtils.h`.

29.496.3.4 `int Go::boxStructuring::SurfaceData::segs_u ( ) const` `[inline]`

Get the number of segments in first parameter direction.

Definition at line 89 of file `ClosestPointUtils.h`.

29.496.3.5 `int Go::boxStructuring::SurfaceData::segs_v ( ) const` `[inline]`

Get the number of segments in second parameter direction.

Definition at line 95 of file `ClosestPointUtils.h`.

29.496.3.6 `void Go::boxStructuring::SurfaceData::setIndex ( int index )` `[inline]`

Set the index of this surface in the [BoundingBoxStructure](#) instance.

Definition at line 101 of file `ClosestPointUtils.h`.

29.496.3.7 `void Go::boxStructuring::SurfaceData::setSegments ( int segs_u, int segs_v ) [inline]`

Set the number of segments in the mesh structure in both parameter directions.

Definition at line 82 of file `ClosestPointUtils.h`.

29.496.3.8 `void Go::boxStructuring::SurfaceData::setSurfaceCopies ( int nmb_copies ) [inline]`

Set number of surface copies.

Definition at line 113 of file `ClosestPointUtils.h`.

29.496.3.9 `shared_ptr<ParamSurface> Go::boxStructuring::SurfaceData::surface ( int idx ) const [inline]`

Get a specific copy of the surface.

Definition at line 124 of file `ClosestPointUtils.h`.

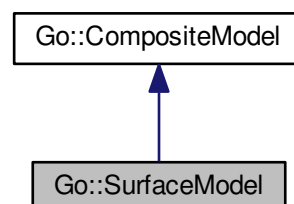
The documentation for this class was generated from the following file:

- [gotools-core/include/GoTools/Utils/ClosestPointUtils.h](#)

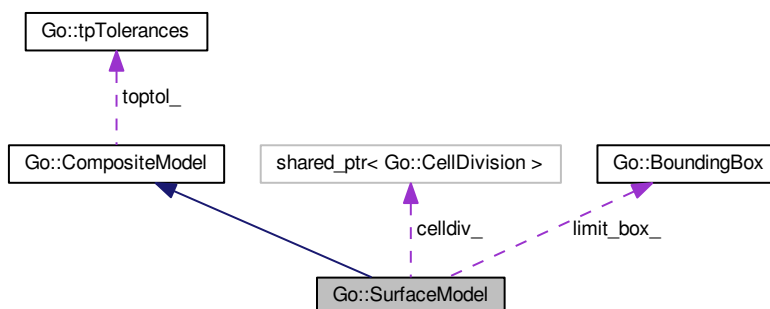
## 29.497 Go::SurfaceModel Class Reference

```
#include <SurfaceModel.h>
```

Inheritance diagram for `Go::SurfaceModel`:



Collaboration diagram for Go::SurfaceModel:



## Public Member Functions

- `SurfaceModel` (`double` approxtol, `double` gap, `double` neighbour, `double` kink, `double` bend, `std::vector`< `shared_ptr`< `ftSurface` > > &faces, `bool` adjacency\_set=false)
- `SurfaceModel` (`std::vector`< `shared_ptr`< `ftSurface` > > &faces, `double` space\_epsilon, `double` kink=0.01, `bool` adjacency\_set=false)
- `SurfaceModel` (`double` approxtol, `double` gap, `double` neighbour, `double` kink, `double` bend, `std::vector`< `shared_ptr`< `ParamSurface` > > &surfaces)
- `SurfaceModel` (`const SurfaceModel` &sm)
- `virtual ~SurfaceModel` ()

*Destructor.*

- `void setTolerances` (`double` approxtol, `double` gap, `double` kink)
- `void setTolerances` (`double` approxtol, `double` gap, `double` neighbour, `double` kink, `double` bend)
- `virtual SurfaceModel *` `asSurfaceModel` ()
- `virtual SurfaceModel *` `clone` () `const`
- `virtual int` `nmbEntities` () `const`
- `shared_ptr`< `ftSurface` > `getFace` (`int` idx) `const`
- `std::vector`< `shared_ptr`< `ftSurface` > > `allFaces` () `const`
- `shared_ptr`< `ParamSurface` > `getSurface` (`int` idx) `const`
- `shared_ptr`< `SplineSurface` > `getSplineSurface` (`int` idx) `const`
- `int` `getIndex` (`shared_ptr`< `ftSurface` > face) `const`
- `int` `getIndex` (`ftSurface` \*face) `const`
- `int` `getIndex` (`ParamSurface` \*surf) `const`
- `shared_ptr`< `ftSurface` > `fetchAsSharedPtr` (`ftFaceBase` \*face) `const`
- `void` `swapFaces` (`int` idx1, `int` idx2)
- `void` `initializeCelldiv` ()

*Creates the CellDivision object.*

- `const ftCell` & `getCell` (`int` i) `const`
- `void` `limitVolume` (`double` xmin, `double` xmax, `double` ymin, `double` ymax, `double` zmin, `double` zmax)

*Set limiting volume to mark area of interest.*

- `bool` `pointWithinLimits` (`const ftPoint` &point)
- `bool` `pointWithinLimits` (`const Point` &point)
- `virtual void` `evaluate` (`int` idx, `double` par[], `Point` &pnt) `const`
- `virtual void` `evaluate` (`int` idx, `double` par[], `int` nder, `std::vector`< `Point` > &der) `const`
- `virtual void` `closestPoint` (`Point` &pnt, `Point` &clo\_pnt, `int` &idx, `double` clo\_par[], `double` &dist)
- `ftPoint` `closestPoint` (`const Point` &point)

- `ftPoint` `closestPoint` (`const ftPoint` &point)
- virtual void `extremalPoint` (`Point` &dir, `Point` &ext\_pnt, int &idx, `double` ext\_par[ ])
- virtual `BoundingBox` `boundingBox` ()
- virtual `BoundingBox` `boundingBox` (int idx) `const`
- virtual `bool` `isDegenerate` (int idx) `const`
- `bool` `isDegenerate` (int idx, `bool` &b, `bool` &t, `bool` &l, `bool` &r) `const`
- virtual `shared_ptr`< `IntResultsModel` > `intersect_plane` (`const ftPlane` &plane)
- virtual `shared_ptr`< `IntResultsModel` > `intersect` (`const ftLine` &line)
- void `intersect` (`const ftLine` &line, `ftCurve` &int\_curves, `std::vector`< `ftPoint` > &int\_points)
- `bool` `hit` (`const Point` &point, `const Point` &dir, `ftPoint` &result)
- `ftCurve` `intersect` (`const ftPlane` &plane)
- void `booleanIntersect` (`const ftPlane` &plane)
- `shared_ptr`< `SurfaceModel` > `trimWithPlane` (`const ftPlane` &plane)
- `std::vector`< `ftPoint` > `intersect` (`const ftLine` &line, `std::vector`< `bool` > &represent\_segment)
- `std::vector`< `std::pair`< `ftPoint`, `double` > > `intersect` (`shared_ptr`< `SplineCurve` > crv, `std::vector`< `bool` > &represent\_segment)
- `std::vector`< `shared_ptr`< `SurfaceModel` > > `splitSurfaceModels` (`shared_ptr`< `SurfaceModel` > &model2)
- void `splitSurfaceModel` (`std::vector`< `shared_ptr`< `ftSurface` > > &faces, `Body` \*model2, `std::vector`< `std::vector`< `shared_ptr`< `ParamSurface` > > > &result)
- `bool` `doIntersect` (`shared_ptr`< `SplineSurface` > sf)
  - Check if a spline surface intersects the current surface model.*
- virtual `double` `curvature` (int idx, `double` \*par) `const`
- virtual void `turn` (int idx)
- virtual void `turn` ()
- void `append` (`shared_ptr`< `ftSurface` > face, `bool` set\_twin=true, `bool` adjacency\_set=false, `bool` remove\_twins=false, int idx=-1)
- void `append` (`std::vector`< `shared_ptr`< `ftSurface` > > faces, `bool` adjacency\_set=false, `bool` set\_twin=true)
- void `append` (`shared_ptr`< `SurfaceModel` > anotherModel)
- `bool` `removeFace` (`shared_ptr`< `ftSurface` > face)
- void `updateFaceTopology` (`shared_ptr`< `ftSurface` > face)
- virtual void `tessellate` (`std::vector`< `shared_ptr`< `GeneralMesh` > > &meshes) `const`
- void `tessellate` (int uv\_res, `std::vector`< `shared_ptr`< `GeneralMesh` > > &meshes) `const`
- virtual void `tessellate` (int resolution[ ], `std::vector`< `shared_ptr`< `GeneralMesh` > > &meshes) `const`
- virtual void `tessellate` (`double` density, `std::vector`< `shared_ptr`< `GeneralMesh` > > &meshes) `const`
- void `tessellate` (`const` `std::vector`< `shared_ptr`< `ftFaceBase` > > &faces, int uv\_res, `std::vector`< `shared_ptr`< `GeneralMesh` > > &meshes) `const`
- void `tessellate` (`const` `std::vector`< `shared_ptr`< `ftFaceBase` > > &faces, int resolution[ ], `std::vector`< `shared_ptr`< `GeneralMesh` > > &meshes) `const`
- void `tessellate` (`const` `std::vector`< `shared_ptr`< `ftFaceBase` > > &faces, `double` density, `std::vector`< `shared_ptr`< `GeneralMesh` > > &meshes) `const`
- virtual void `tessellatedCtrPolygon` (`std::vector`< `shared_ptr`< `LineCloud` > > &ctr\_pol) `const`
- void `tessellatedCtrPolygon` (`const` `std::vector`< `shared_ptr`< `ftFaceBase` > > &faces, `std::vector`< `shared_ptr`< `LineCloud` > > &ctr\_pol) `const`
- void `fetchSamplePoints` (`double` density, `std::vector`< `SamplePointData` > &sample\_points) `const`
- void `setBoundaryCurves` ()
  - Set the elements in boundary\_curves\_ (based on top\_table\_).*
- `ftMessage` `buildTopology` (int first\_idx=0, `bool` set\_twin\_face\_info=true)
- `ftMessage` `setTopology` ()
- void `setVertexIdentity` ()
  - Add information about faces at the boundary meeting only in vertices.*
- void `setTwinFaceInfo` ()
  - Add information about twin faces.*
- `bool` `isClosed` () `const`
- int `nmbBoundaries` () `const`

- [ftCurve](#) [getBoundary](#) (int whichbound)
- [ftCurve](#) [getGaps](#) ()
- void [getGaps](#) (std::vector< [ftEdge](#) \* > &gaps)
- [ftCurve](#) [getKinks](#) ()
- void [getKinks](#) (std::vector< [ftEdge](#) \* > &kinks)
- [ftCurve](#) [getG1Disconts](#) ()
- void [getCorners](#) (std::vector< [ftEdge](#) \* > &corners)
- [ftSurface](#) \* [getSurface2](#) (int index) **const**
- std::vector< shared\_ptr< [SurfaceModel](#) > > [getConnectedModels](#) () **const**
- void [getInconsistentFacePairs](#) (std::vector< std::pair< [ftFaceBase](#) \*, [ftFaceBase](#) \* > > &faces)
- void [getOverlappingEdges](#) (double tol, std::vector< std::pair< shared\_ptr< [ftEdgeBase](#) >, shared\_ptr< [ftEdgeBase](#) > > &edges)
- shared\_ptr< [ftPointSet](#) > [triangulate](#) (double density) **const**
- void [getOverlappingFaces](#) (double tol, std::vector< std::pair< [ftSurface](#) \*, [ftSurface](#) \* > > &faces)
- void [getAllVertices](#) (std::vector< shared\_ptr< [Vertex](#) > > &vertices) **const**
- void [getBoundaryVertices](#) (std::vector< shared\_ptr< [Vertex](#) > > &vertices) **const**
- std::vector< shared\_ptr< [ftEdge](#) > > [getBoundaryEdges](#) () **const**
- std::vector< shared\_ptr< [ftEdge](#) > > [getBoundaryEdges](#) (int boundary\_idx) **const**
- std::vector< shared\_ptr< [ftEdge](#) > > [getUniqueInnerEdges](#) () **const**
- [Body](#) \* [getBody](#) ()
  - Return body (if any)*
- double [getApproximationTol](#) () **const**
- bool [simplifyTrimLoops](#) (double &max\_dist)
- bool [isAxisRotational](#) ([Point](#) &centre, [Point](#) &axis, [Point](#) &vec, double &angle, double &min\_ang)
- bool [isLinearSwept](#) ([Point](#) &pnt, [Point](#) &axis, double &len)
- std::vector< shared\_ptr< [ftSurface](#) > > [facesInPlane](#) ([Point](#) &pnt, [Point](#) &axis)
  - Fetch all planar faces lying in a specified plane.*
- bool [allSplines](#) () **const**
- bool [isCornerToCorner](#) () **const**
- void [makeCornerToCorner](#) ()
- void [makeCommonSplineSpaces](#) ()
  - Ensure that the blocks in the model has got common spline spaces.*
- void [enforceCoLinearCoefs](#) ()
- void [regularizeTwin](#) ([ftSurface](#) \*face, std::vector< shared\_ptr< [ftSurface](#) > > &twinset)
- shared\_ptr< [ftSurface](#) > [mergeFaces](#) ([ftSurface](#) \*face1, int paddir1, double parval1, bool atstart1, [ftSurface](#) \*face2, int paddir2, double parval2, bool atstart2, std::pair< [Point](#), [Point](#) > co\_par1, std::pair< [Point](#), [Point](#) > co\_par2, std::vector< [Point](#) > &seam\_joints)
- shared\_ptr< [ftSurface](#) > [mergeSeamFaces](#) ([ftSurface](#) \*face1, [ftSurface](#) \*face2, int paddir, std::vector< [Point](#) > &seam\_joints)
- shared\_ptr< [ftSurface](#) > [mergeSeamCrvFaces](#) ([ftSurface](#) \*face1, [ftSurface](#) \*face2, std::vector< [Point](#) > &seam\_joints)
- void [replaceRegularSurfaces](#) ()
- shared\_ptr< [ftSurface](#) > [replaceRegularSurface](#) ([ftSurface](#) \*face, bool only\_corner=false)
- void [simplifyShell](#) ()
- shared\_ptr< [SplineSurface](#) > [approxFaceSet](#) (double &error, int degree=3)
- bool [checkShellTopology](#) ()
  - Debug. Check topology.*

## Protected Member Functions

- [SurfaceModel](#) (double approxtol, double gap, double neighbour, double kink, double bend)
- void [addSegment](#) ([ftCurve](#) &cv, [ftEdgeBase](#) \*edge, [ftCurveType](#) ty)

## Protected Attributes

- [double approxtol\\_](#)
- [double tol2d\\_](#)
- [std::vector< shared\\_ptr< ftFaceBase > > faces\\_](#)
- [std::vector< std::vector< shared\\_ptr< Loop > > > boundary\\_curves\\_](#)
- [shared\\_ptr< CellDivision > celldiv\\_](#)
- [std::vector< bool > face\\_checked\\_](#)
- [int highest\\_face\\_checked\\_](#)
- [BoundingBox limit\\_box\\_](#)
- [std::vector< std::pair< ftFaceBase \\*, ftFaceBase \\* > > inconsistent\\_orientation\\_](#)

### 29.497.1 Detailed Description

A surface set or shell including topological information

Definition at line 81 of file SurfaceModel.h.

### 29.497.2 Constructor & Destructor Documentation

**29.497.2.1** `Go::SurfaceModel::SurfaceModel ( double approxtol, double gap, double neighbour, double kink, double bend, std::vector< shared_ptr< ftSurface > > & faces, bool adjacency_set = false )`

Constructor taking a vector of faces

#### Parameters

<i>approxtol</i>	Approximation error tolerance.
<i>gap</i>	If the distance between two points are less than 'gap' they are viewed as identical.
<i>neighbour</i>	Maximum distance between surfaces viewed as adjacent.
<i>kink</i>	If two adjacent surfaces meet with an angle less than 'kink', they are seen as G1 continous. (angles in radians)
<i>bend</i>	If two surfaces meet along a common boundary and corresponding surface normals form an angle which is larger than 'bend', there is an intentional sharp edge between the surfaces.(angles in radians)
<i>faces</i>	A vector of faces.
<i>adjacency_set</i>	True if the application knows that twin information between edges is set.

**29.497.2.2** `Go::SurfaceModel::SurfaceModel ( std::vector< shared_ptr< ftSurface > > & faces, double space_epsilon, double kink = 0.01, bool adjacency_set = false )`

Constructor taking a vector of faces.

#### Parameters

<i>faces</i>	A vector of faces.
<i>space_epsilon</i>	
<i>kink</i>	
<i>adjacency_set</i>	



29.497.2.3 `Go::SurfaceModel::SurfaceModel ( double approxol, double gap, double neighbour, double kink, double bend, std::vector< shared_ptr< ParamSurface > > & surfaces )`

Constructor taking a vector of parametric surfaces

#### Parameters

<i>approxol</i>	Approximation error tolerance. Not used.
<i>gap</i>	If the distance between two points is less than 'gap' they are viewed as identical.
<i>neighbour</i>	Maximum distance between surfaces or curves viewed as adjacent.
<i>kink</i>	If two adjacent surfaces meet with an angle less than 'kink', they are seen as G1 continuous. (angles in radians)
<i>bend</i>	If two surfaces meet along a common boundary and corresponding surface normals form an angle which is larger than 'bend', there is an intentional sharp edge between the surfaces.(angles in radians)
<i>surfaces</i>	A vector of surfaces.

29.497.2.4 `Go::SurfaceModel::SurfaceModel ( double approxol, double gap, double neighbour, double kink, double bend )` `[protected]`

29.497.2.5 `Go::SurfaceModel::SurfaceModel ( const SurfaceModel & sm )`

Constructor Create shallow copy of another [SurfaceModel](#)

#### Parameters

<i>sm</i>	The <a href="#">SurfaceModel</a> to be copied
-----------	-----------------------------------------------

29.497.2.6 `virtual Go::SurfaceModel::~~SurfaceModel ( )` `[virtual]`

Destructor.

### 29.497.3 Member Function Documentation

29.497.3.1 `void Go::SurfaceModel::addSegment ( ftCurve & cv, ftEdgeBase * edge, ftCurveType ty )` `[protected]`

29.497.3.2 `std::vector< shared_ptr< ftSurface > > Go::SurfaceModel::allFaces ( ) const`

Return all faces

#### Returns

Vector of pointer to all faces

29.497.3.3 `bool Go::SurfaceModel::allSplines ( ) const`

Check if all entities are NURBS

Returns

Whether all entities are NURBS

29.497.3.4 `void Go::SurfaceModel::append ( shared_ptr< ftSurface > face, bool set_twin = true, bool adjacency_set = false, bool remove_twins = false, int idx = -1 )`

Append a new face to the surface model. The face is included in the topological structure

Parameters

<i>face</i>	The new face
<i>set_twin</i>	If true, set twin face info.

29.497.3.5 `void Go::SurfaceModel::append ( std::vector< shared_ptr< ftSurface > > faces, bool adjacency_set = false, bool set_twin = true )`

Append a vector of faces to the surface model. The faces are included in the topological structure

Parameters

<i>faces</i>	Vector of pointers to the new faces
--------------	-------------------------------------

29.497.3.6 `void Go::SurfaceModel::append ( shared_ptr< SurfaceModel > anotherModel )`

Append all faces from another surface model. The faces are included in the topological structure

Parameters

<i>anotherModel</i>	Pointer to the other surface model
---------------------	------------------------------------

29.497.3.7 `shared_ptr< SplineSurface > Go::SurfaceModel::approxFaceSet ( double & error, int degree = 3 )`

Approximate surface sets with 4 boundaries with a spline surface. If the set has more than 4 corners, no surface will be produced.

29.497.3.8 `virtual SurfaceModel* Go::SurfaceModel::asSurfaceModel ( ) [inline], [virtual]`

Return surface model pointer

**Returns**

Pointer to this [SurfaceModel](#)

Reimplemented from [Go::CompositeModel](#).

Definition at line 176 of file SurfaceModel.h.

**29.497.3.9 void Go::SurfaceModel::booleanIntersect ( const ftPlane & plane )**

Intersect the model with a plane and trim this model with respect to the plane, the part of the model at the positive side of the plane is removed.

**Parameters**

<i>plane</i>	The plane.
--------------	------------

**29.497.3.10 virtual BoundingBox Go::SurfaceModel::boundingBox ( ) [virtual]**

Bounding box of the entire surface model

**Returns**

Bounding box

Implements [Go::CompositeModel](#).

**29.497.3.11 virtual BoundingBox Go::SurfaceModel::boundingBox ( int idx ) const [virtual]**

Bounding box corresponding to one surface

**Parameters**

<i>idx</i>	Index of surface
------------	------------------

**Returns**

Bounding box of surface with index *idx*.

Implements [Go::CompositeModel](#).

**29.497.3.12 ftMessage Go::SurfaceModel::buildTopology ( int first\_idx = 0, bool set\_twin\_face\_info = true )**

Construct the topology information regarding the input geometry.

**Returns**

Messages

29.497.3.13 **bool** Go::SurfaceModel::checkShellTopology ( )

Debug. Check topology.

29.497.3.14 **virtual SurfaceModel\*** Go::SurfaceModel::clone ( ) **const** [inline],[virtual]

Make a copy of the current model

Returns

Pointer to a copy of this [SurfaceModel](#)

Reimplemented from [Go::CompositeModel](#).

Definition at line 183 of file SurfaceModel.h.

29.497.3.15 **virtual void** Go::SurfaceModel::closestPoint ( **Point & pnt**, **Point & clo\_pnt**, **int & idx**, **double clo\_par[]**, **double & dist** ) [virtual]

Closest point between a given point and this surface model Returns one point

Parameters

<i>pnt</i>	Input point
<i>clo_pnt</i>	Found closest point
<i>idx</i>	Index of surface where the closest point is found
<i>clo_par[]</i>	Parameter value corresponding to the closest point
<i>dist</i>	Distance between input point and found closest point

Implements [Go::CompositeModel](#).

29.497.3.16 **ftPoint** Go::SurfaceModel::closestPoint ( **const Point & point** )

Closest point between a given point and this surface model

Parameters

<i>point</i>	Input point
--------------	-------------

Returns

Closest point

29.497.3.17 **ftPoint** Go::SurfaceModel::closestPoint ( **const ftPoint & point** ) [inline]

Closest point between a given point and this surface model

## Parameters

<i>point</i>	Input point
--------------	-------------

## Returns

Closest point

Definition at line 304 of file SurfaceModel.h.

**29.497.3.18** `virtual double Go::SurfaceModel::curvature ( int idx, double * par ) const` `[inline], [virtual]`

[Curvature](#) of an entity (Curve or Surface), Only Curve is implemented yet.

## Parameters

<i>idx</i>	Index of entity
<i>par</i>	Parameter value at which to compute curvature

## Returns

The curvature.

Implements [Go::CompositeModel](#).

Definition at line 454 of file SurfaceModel.h.

**29.497.3.19** `bool Go::SurfaceModel::doIntersect ( shared_ptr< SplineSurface > sf )`

Check if a spline surface intersects the current surface model.

**29.497.3.20** `void Go::SurfaceModel::enforceCoLinearCoefs ( )`

Modify adjacent spline surfaces from having almost co-linear coefficients to exact co-linearity

**29.497.3.21** `virtual void Go::SurfaceModel::evaluate ( int idx, double par[], Point & pnt ) const` `[virtual]`

Evaluate position

## Parameters

<i>idx</i>	Index of surface
<i>par</i> []	Parameter value
<i>pnt</i>	Result

Implements [Go::CompositeModel](#).

29.497.3.22 `virtual void Go::SurfaceModel::evaluate ( int idx, double par[], int nder, std::vector< Point > & der ) const`  
`[virtual]`

Evaluate position and a number of derivatives The sequence is position, first derivative in first parameter direction, first derivative in second parameter direction, second derivative in first parameter direction, mixed second derivative etc.

#### Parameters

<i>idx</i>	Index
<i>par</i>	Parameter value
<i>nder</i>	Number of derivatives to compute, 0=only position
<i>der</i>	Result

Implements [Go::CompositeModel](#).

29.497.3.23 `virtual void Go::SurfaceModel::extremalPoint ( Point & dir, Point & ext_pnt, int & idx, double ext_par[] )`  
`[virtual]`

Extremal point(s) in a given direction Note that the found extremal point may be less accurate for trimmed surfaces

#### Parameters

<i>dir</i>	Direction
<i>ext_pnt</i>	Found extremal point
<i>idx</i>	Index of surface where the extremal point is found
<i>ext_par</i> []	Parameter value of extremal point

Implements [Go::CompositeModel](#).

29.497.3.24 `std::vector<shared_ptr<ftSurface>> Go::SurfaceModel::facesInPlane ( Point & pnt, Point & axis )`

Fetch all planar faces lying in a specified plane.

29.497.3.25 `shared_ptr<ftSurface> Go::SurfaceModel::fetchAsSharedPtr ( ftFaceBase * face ) const`

Return a specified face as a shared pointer

#### Parameters

<i>face</i>	Pointer to face
-------------	-----------------

**Returns**

Shared pointer to face

**29.497.3.26** void Go::SurfaceModel::fetchSamplePoints ( double *density*, std::vector< SamplePointData > & *sample\_points* ) const

Fetch sample points from all faces including surface normal and curvature information. The distribution is computed from the input parameter density

**29.497.3.27** void Go::SurfaceModel::getAllVertices ( std::vector< shared\_ptr< Vertex > > & *vertices* ) const

Return all vertices associated with this surface model

**Return values**

<i>vertices</i>	Vector of pointers to all vertices.
-----------------	-------------------------------------

**29.497.3.28** double Go::SurfaceModel::getApproximationTol ( ) const [inline]

Return approximation tolerance.

**Returns**

Approximation tolerance.

Definition at line 744 of file SurfaceModel.h.

**29.497.3.29** Body\* Go::SurfaceModel::getBody ( )

Return body (if any)

**29.497.3.30** ftCurve Go::SurfaceModel::getBoundary ( int *whichbound* )

Return a given boundary (may be a hole).

**Parameters**

<i>whichbound</i>	refers to element number in boundary_curves_.
-------------------	-----------------------------------------------

**Returns**

boundary curve

29.497.3.31 `std::vector<shared_ptr<ftEdge> > Go::SurfaceModel::getBoundaryEdges ( ) const`

Fetch all edges at all the boundaries of this model

Return values

<i>edges</i>	Vector of pointers to edges at the boundaries.
--------------	------------------------------------------------

29.497.3.32 `std::vector<shared_ptr<ftEdge> > Go::SurfaceModel::getBoundaryEdges ( int boundary_idx ) const`

Fetch all edges at one of the boundaries of this model

Parameters

<i>boundary_idx</i>	Index of one boundary
---------------------	-----------------------

Return values

<i>edges</i>	Vector of pointers to the edges at one boundary.
--------------	--------------------------------------------------

29.497.3.33 `void Go::SurfaceModel::getBoundaryVertices ( std::vector< shared_ptr< Vertex > > & vertices ) const`

Fetch vertices at the boundaries

Return values

<i>vertices</i>	Vector of pointers to vertices at the boundaries.
-----------------	---------------------------------------------------

29.497.3.34 `const ftCell& Go::SurfaceModel::getCell ( int i ) const`

Return a cell in the cell division

Parameters

<i>i</i>	Index of cell
----------	---------------

Returns

The cell

29.497.3.35 `std::vector<shared_ptr<SurfaceModel> > Go::SurfaceModel::getConnectedModels ( ) const`

Return all compact face sets



29.497.3.36 `void Go::SurfaceModel::getCorners ( std::vector< ftEdge * > & corners )`

Information about kinks on an alternative format

29.497.3.37 `shared_ptr<ftSurface> Go::SurfaceModel::getFace ( int idx ) const`

Return one face

Parameters

<i>idx</i>	Index of face
------------	---------------

Returns

Pointer to face with index *idx*

29.497.3.38 `ftCurve Go::SurfaceModel::getG1Disconts ( )`

Return information about all G1 discontinuities.

29.497.3.39 `ftCurve Go::SurfaceModel::getGaps ( )`

Return information about all gaps.

29.497.3.40 `void Go::SurfaceModel::getGaps ( std::vector< ftEdge * > & gaps )`

Information about gaps on an alternative format

29.497.3.41 `void Go::SurfaceModel::getInconsistentFacePairs ( std::vector< std::pair< ftFaceBase *, ftFaceBase * > > & faces ) [inline]`

Return pointers to pairs of faces that have a problem with the consistency of the face orientation

Definition at line 692 of file SurfaceModel.h.

29.497.3.42 `int Go::SurfaceModel::getIndex ( shared_ptr< ftSurface > face ) const`

Given a face in the surface model, return the index of this face

Parameters

<i>face</i>	Shared pointer to face
-------------	------------------------

**Returns**

Index to face

29.497.3.43 `int Go::SurfaceModel::getIndex ( ftSurface * face ) const`

Given a face in the surface model, return the index of this face

**Parameters**

<i>face</i>	Pointer to face
-------------	-----------------

**Returns**

Index to face

29.497.3.44 `int Go::SurfaceModel::getIndex ( ParamSurface * surf ) const`

Given a surface in the surface model, return the index of this face

**Parameters**

<i>surf</i>	Pointer to surface
-------------	--------------------

**Returns**

Index to face

29.497.3.45 `ftCurve Go::SurfaceModel::getKinks ( )`

Return information about all kinks.

29.497.3.46 `void Go::SurfaceModel::getKinks ( std::vector< ftEdge * > & kinks )`

Information about kinks on an alternative format

29.497.3.47 `void Go::SurfaceModel::getOverlappingEdges ( double tol, std::vector< std::pair< shared_ptr< ftEdgeBase >, shared_ptr< ftEdgeBase > > & edges )`

Return pointers to pairs of overlapping edges.

**Parameters**

<i>tol</i>	Overlap tolerance
------------	-------------------

## Return values

<i>edges</i>	Vector of pairs of overlapping edges
--------------	--------------------------------------

29.497.3.48 `void Go::SurfaceModel::getOverlappingFaces ( double tol, std::vector< std::pair< ftSurface *, ftSurface * > > & faces )`

Return pointers to pairs of overlapping faces.

## Parameters

<i>tol</i>	Overlap tolerance
------------	-------------------

## Return values

<i>faces</i>	Vector of pairs of overlapping faces
--------------	--------------------------------------

29.497.3.49 `shared_ptr<SplineSurface> Go::SurfaceModel::getSplineSurface ( int idx ) const`

Return one surface as [SplineSurface](#) if possible

## Parameters

<i>idx</i>	Index of surface
------------	------------------

## Returns

Pointer to [SplineSurface](#)

29.497.3.50 `shared_ptr<ParamSurface> Go::SurfaceModel::getSurface ( int idx ) const`

Return one surface

## Parameters

<i>idx</i>	Index of surface
------------	------------------

## Returns

Pointer to [ParamSurface](#)

29.497.3.51 `ftSurface* Go::SurfaceModel::getSurface2 ( int index ) const`

Return single surface

29.497.3.52 `std::vector<shared_ptr<ftEdge> > Go::SurfaceModel::getUniqueInnerEdges ( ) const`

Fetch all interval unique inner edges in this model, i.e. a [ftEdge](#) for each boundary edge with a twin, only one of the edges in the pair is returned.

Return values

<i>Vector</i>	of pointers unique inner edges
---------------	--------------------------------

29.497.3.53 `bool Go::SurfaceModel::hit ( const Point & point, const Point & dir, ftPoint & result )`

Test if a line with direction 'dir' through the point 'point' hits this surface model. If it hits, return 'true' and the intersection point closest to 'point'.

Parameters

<i>point</i>	<a href="#">Point</a> on the line.
<i>dir</i>	<a href="#">Line</a> direction.

Return values

<i>result</i>	Closest intersection point.
---------------	-----------------------------

Returns

Whether the line hits or not.

29.497.3.54 `void Go::SurfaceModel::initializeCelldiv ( )`

Creates the [CellDivision](#) object.

29.497.3.55 `virtual shared_ptr<IntResultsModel> Go::SurfaceModel::intersect ( const ftLine & line ) [virtual]`

Intersection with a line. Expected output is points, probably one point. Curves can occur in special configurations

Parameters

<i>line</i>	The line.
-------------	-----------

Returns

Pointer to an [IntResultsModel](#).

Implements [Go::CompositeModel](#).

29.497.3.56 `void Go::SurfaceModel::intersect ( const ftLine & line, ftCurve & int_curves, std::vector< ftPoint > & int_points )`

Intersection with a line. Expected output is points, probably one point. Curves can occur in special configurations

#### Parameters

<i>line</i>	Consist of one point and one direction represented by <a href="#">Point</a> .
-------------	-------------------------------------------------------------------------------

#### Return values

<i>int_curves</i>	Intersection curves, one curve may cross several of the surfaces in the model, but each curve is connected and simple.
<i>int_points</i>	Found intersection points.

29.497.3.57 `ftCurve Go::SurfaceModel::intersect ( const ftPlane & plane )`

Intersect the surface model with a plane.

#### Parameters

<i>plane</i>	The plane.
--------------	------------

#### Returns

Intersection curve.

29.497.3.58 `std::vector<ftPoint> Go::SurfaceModel::intersect ( const ftLine & line, std::vector< bool > & represent_segment )`

Intersect the model with a line.

#### Parameters

<i>line</i>	The intersecting line
-------------	-----------------------

#### Return values

<i>represent_segment</i>	???
--------------------------	-----

#### Returns

Vector of intersection points

```
29.497.3.59 std::vector<std::pair<ftPoint, double> > Go::SurfaceModel::intersect (shared_ptr< SplineCurve > crv,
std::vector< bool > & represent_segment)
```

Intersect the model with a [SplineCurve](#).

#### Parameters

<i>crv</i>	The intersecting <a href="#">SplineCurve</a> .
------------	------------------------------------------------

#### Return values

<i>represent_segment</i>	???
--------------------------	-----

#### Returns

Vector of pairs of intersection points and their curve parameter value

```
29.497.3.60 virtual shared_ptr<IntResultsModel> Go::SurfaceModel::intersect_plane (const ftPlane & plane)
[virtual]
```

Intersection with a plane.

#### Parameters

<i>plane</i>	The plane.
--------------	------------

#### Returns

Pointer to an [IntResultsModel](#).

Implements [Go::CompositeModel](#).

```
29.497.3.61 bool Go::SurfaceModel::isAxisRotational (Point & centre, Point & axis, Point & vec, double & angle,
double & min_ang)
```

Check if a surface model represents a rotational object and extract the eventual rotational axis and angle

```
29.497.3.62 bool Go::SurfaceModel::isClosed () const
```

Check if the [SurfaceModel](#) is a closed shell

#### Returns

*true* if closed, *false* if open

29.497.3.63 **bool** Go::SurfaceModel::isCornerToCorner ( ) const

Check if the model has got a corner-to-corner configuration

#### Returns

Whether the model has got a corner-to-corner configuration

29.497.3.64 **virtual bool** Go::SurfaceModel::isDegenerate ( int *idx* ) const [virtual]

Whether one particular surface is degenerated such that one boundary degenerates to a point

#### Parameters

<i>idx</i>	Index of surface
------------	------------------

#### Returns

Whether the surface is degenerated

Implements [Go::CompositeModel](#).

29.497.3.65 **bool** Go::SurfaceModel::isDegenerate ( int *idx*, **bool** & *b*, **bool** & *t*, **bool** & *l*, **bool** & *r* ) const

Whether one particular surface is degenerated More specific information, relevant only for surfaces

#### Parameters

<i>idx</i>	Index of surface
------------	------------------

#### Return values

<i>b</i>	The bottom curve degenerates to a point
<i>t</i>	The top curve degenerates to a point
<i>l</i>	The left curve degenerates to a point
<i>r</i>	The right curve degenerates to a point

#### Returns

Whether the surface is degenerated

29.497.3.66 **bool** Go::SurfaceModel::isLinearSwept ( **Point** & *pnt*, **Point** & *axis*, **double** & *len* )

Check if a surface model represents a simple linearly swept object that can be generated by sweeping a set of planar surfaces along a vector parallel to the plane normal. Return plane information and distance of sweep

29.497.3.67 `void Go::SurfaceModel::limitVolume ( double xmin, double xmax, double ymin, double ymax, double zmin, double zmax )`

Set limiting volume to mark area of interest.

29.497.3.68 `void Go::SurfaceModel::makeCommonSplineSpaces ( )`

Ensure that the blocks in the model has got common spline spaces.

29.497.3.69 `void Go::SurfaceModel::makeCornerToCorner ( )`

Ensure that the blocks in the model meet in a corner-to-corner configuration. NB! This function must not be applied to closed surface sets.

29.497.3.70 `shared_ptr<ftSurface> Go::SurfaceModel::mergeFaces ( ftSurface * face1, int paddir1, double parval1, bool atstart1, ftSurface * face2, int paddir2, double parval2, bool atstart2, std::pair< Point, Point > co_par1, std::pair< Point, Point > co_par2, std::vector< Point > & seam_joints )`

Merge two faces

Returns

Pointer to resulting face

29.497.3.71 `shared_ptr<ftSurface> Go::SurfaceModel::mergeSeamCrvFaces ( ftSurface * face1, ftSurface * face2, std::vector< Point > & seam_joints )`

Merge two faces

Returns

Pointer to resulting face

29.497.3.72 `shared_ptr<ftSurface> Go::SurfaceModel::mergeSeamFaces ( ftSurface * face1, ftSurface * face2, int paddir, std::vector< Point > & seam_joints )`

Merge two faces

Returns

Pointer to resulting face

29.497.3.73 `int Go::SurfaceModel::nmbBoundaries ( ) const`

Return the number of boundaries of a surface set (including holes).



29.497.3.74 `virtual int Go::SurfaceModel::nmbEntities ( ) const [virtual]`

Number of simple entities

#### Returns

Number of simple entities

Implements [Go::CompositeModel](#).

29.497.3.75 `bool Go::SurfaceModel::pointWithinLimits ( const ftPoint & point )`

Check if a point is within the volume of interest.

#### Parameters

<i>point</i>	The point to check.
--------------	---------------------

#### Returns

Whether the point is within the limits.

29.497.3.76 `bool Go::SurfaceModel::pointWithinLimits ( const Point & point )`

Check if a point is within the volume of interest.

#### Parameters

<i>point</i>	The point to check.
--------------	---------------------

#### Returns

Whether the point is within the limits.

29.497.3.77 `void Go::SurfaceModel::regularizeTwin ( ftSurface * face, std::vector< shared_ptr< ftSurface > > & twinset )`

Regularize face to mimic the division of a twin surface

#### Parameters

<i>face</i>	
-------------	--

#### Return values

<i>twinset</i>	
----------------	--

29.497.3.78 **bool** Go::SurfaceModel::removeFace ( shared\_ptr< ftSurface > *face* )

Remove one face from the face set

Parameters

<i>face</i>	Pointer to the face to be removed
-------------	-----------------------------------

Returns

Whether the face was removed

29.497.3.79 **shared\_ptr<ftSurface>** Go::SurfaceModel::replaceRegularSurface ( ftSurface \* *face*, **bool** *only\_corner* = **false** )

29.497.3.80 **void** Go::SurfaceModel::replaceRegularSurfaces ( )

Approximate regular trimmed surfaces with spline surfaces and replace

29.497.3.81 **void** Go::SurfaceModel::setBoundaryCurves ( )

Set the elements in *boundary\_curves\_* (based on *top\_table\_*).

29.497.3.82 **void** Go::SurfaceModel::setTolerances ( **double** *aproxtol*, **double** *gap*, **double** *kink* )

Set or reset topology tolerances

Parameters

<i>aproxtol</i>	Approximation error tolerance.
<i>gap</i>	If the distance between two points is less than 'gap' they are viewed as identical.
<i>kink</i>	If two adjacent surfaces meet with an angle less than 'kink', they are seen as G1 continuous. (angles in radians)

29.497.3.83 **void** Go::SurfaceModel::setTolerances ( **double** *aproxtol*, **double** *gap*, **double** *neighbour*, **double** *kink*, **double** *bend* )

Set or reset topology tolerances

Parameters

<i>aproxtol</i>	Approximation error tolerance. Not used.
<i>gap</i>	If the distance between two points is less than 'gap' they are viewed as identical.
<i>neighbour</i>	Maximum distance between surfaces or curves viewed as adjacent.
<i>kink</i>	If two adjacent surfaces meet with an angle less than 'kink', they are seen as G1 continuous. (angles in radians)

## Parameters

<i>bend</i>	If two surfaces meet along a common boundary and corresponding surface normals form an angle which is larger than 'bend', there is an intentional sharp edge between the surfaces.(angles in radians)
-------------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

29.497.3.84 **ftMessage** Go::SurfaceModel::setTopology ( )

Fetch the topology information from twin information in the input geometry.

## Returns

Messages

29.497.3.85 **void** Go::SurfaceModel::setTwinFaceInfo ( )

Add information about twin faces.

29.497.3.86 **void** Go::SurfaceModel::setVertexIdentity ( )

Add information about faces at the boundary meeting only in vertices.

29.497.3.87 **void** Go::SurfaceModel::simplifyShell ( )

Simplify current shell by merging surfaces with a smooth connection both with regard to the surfaces themselves and the associated boundary curves. The common boundary must be isoparametric in both surfaces

29.497.3.88 **bool** Go::SurfaceModel::simplifyTrimLoops ( **double** & *max\_dist* )

Simplify fragmented trimming loops

## Return values

<i>max_dist</i>	Maximum local distortion around the transitions. ???
-----------------	------------------------------------------------------

## Returns

Whether any faces were modified.

29.497.3.89 **void** Go::SurfaceModel::splitSurfaceModel ( **std::vector**< **shared\_ptr**< **ftSurface** > > & *faces*, **Body** \* *model2*, **std::vector**< **std::vector**< **shared\_ptr**< **ParamSurface** > > > & *result* )

Split a surface models according to a set of faces belonging to a specified surface model

29.497.3.90 `std::vector<shared_ptr<SurfaceModel> > Go::SurfaceModel::splitSurfaceModels ( shared_ptr<SurfaceModel> & model2 )`

Split two surface models according to intersections between them.

#### Parameters

<code>model2</code>	The other model.
---------------------	------------------

#### Returns

Vector of new surface models.

29.497.3.91 `void Go::SurfaceModel::swapFaces ( int idx1, int idx2 )`

Switches the ordering of the faces internal in this construction. To be used if the order in which the faces are handled has a significance

29.497.3.92 `virtual void Go::SurfaceModel::tessellate ( std::vector< shared_ptr< GeneralMesh > > & meshes ) const [virtual]`

Tessellate surface model Tessellate all surfaces with respect to a default resolution

#### Return values

<code>meshes</code>	Tesselated model
---------------------	------------------

Implements [Go::CompositeModel](#).

29.497.3.93 `void Go::SurfaceModel::tessellate ( int uv_res, std::vector< shared_ptr< GeneralMesh > > & meshes ) const`

Tessellate all surfaces with respect to a given total resolution. The resolution in each parameter direction is set from the method

#### Parameters

<code>uv_res</code>	Tesselation resolution
---------------------	------------------------

#### Return values

<code>meshes</code>	Tesselated model
---------------------	------------------

29.497.3.94 `virtual void Go::SurfaceModel::tessellate ( int resolution[], std::vector< shared_ptr< GeneralMesh > > & meshes ) const [virtual]`

Tessellate all surfaces with respect to given resolutions in each parameter direction.

## Parameters

<i>resolution</i> []	Tessellation resolution
----------------------	-------------------------

## Return values

<i>meshes</i>	Tesselated model
---------------	------------------

Implements [Go::CompositeModel](#).

29.497.3.95 `virtual void Go::SurfaceModel::tessellate ( double density, std::vector< shared_ptr< GeneralMesh > > & meshes ) const` [virtual]

Tessellate all surfaces with respect to a given tessellation density

## Parameters

<i>density</i>	Tessellation density
----------------	----------------------

## Return values

<i>meshes</i>	Tesselated model
---------------	------------------

Implements [Go::CompositeModel](#).

29.497.3.96 `void Go::SurfaceModel::tessellate ( const std::vector< shared_ptr< ftFaceBase > > & faces, int uv_res, std::vector< shared_ptr< GeneralMesh > > & meshes ) const`

Tessellate specified surfaces with respect to a given total resolution. The resolution in each parameter direction is set from the method

## Parameters

<i>faces</i>	Specified surfaces
<i>uv_res</i>	Tessellation resolution

## Return values

<i>meshes</i>	Tesselated surfaces
---------------	---------------------

29.497.3.97 `void Go::SurfaceModel::tessellate ( const std::vector< shared_ptr< ftFaceBase > > & faces, int resolution[], std::vector< shared_ptr< GeneralMesh > > & meshes ) const`

Tessellate specified surfaces with respect to given resolutions in each parameter direction.

## Parameters

<i>faces</i>	Specified surfaces
<i>resolution[]</i>	Tessellation resolutions

## Return values

<i>meshes</i>	Tesselated model
---------------	------------------

29.497.3.98 void Go::SurfaceModel::tessellate ( const std::vector< shared\_ptr< ftFaceBase > > & *faces*, double *density*, std::vector< shared\_ptr< GeneralMesh > > & *meshes* ) const

Tessellate specified surfaces with respect to a given tessellation density

## Parameters

<i>faces</i>	Specified surfaces
<i>density</i>	Tessellation density

## Return values

<i>meshes</i>	Tesselated model
---------------	------------------

29.497.3.99 virtual void Go::SurfaceModel::tesselatedCtrPolygon ( std::vector< shared\_ptr< LineCloud > > & *ctr\_pol* ) const [virtual]

Return a tessellation of the control polygon of all surfaces

## Return values

<i>ctr_pol</i>	Tessellation of the control polygon of all surfaces.
----------------	------------------------------------------------------

Implements [Go::CompositeModel](#).

29.497.3.100 void Go::SurfaceModel::tesselatedCtrPolygon ( const std::vector< shared\_ptr< ftFaceBase > > & *faces*, std::vector< shared\_ptr< LineCloud > > & *ctr\_pol* ) const

Return a tessellation of the control polygon of specified surfaces

## Parameters

<i>faces</i>	Specified surfaces
--------------	--------------------

## Return values

<i>ctr_pol</i>	Tessellation of the control polygon of the specified surfaces.
----------------	----------------------------------------------------------------

29.497.3.101 `shared_ptr<ftPointSet> Go::SurfaceModel::triangulate ( double density ) const`

Triangulate the complete surface set with a prescribed point density. NB! The density may be adjusted if the number of points become too high and it is used as an indicator, not an absolute measure.

29.497.3.102 `shared_ptr<SurfaceModel> Go::SurfaceModel::trimWithPlane ( const ftPlane & plane )`

Intersect the model with a plane and return the surface model trimmed with respect to this plane, the part of the model at the positive side of the plane is removed.

## Parameters

<i>plane</i>	The plane.
--------------	------------

## Returns

Pointer to trimmed surface model

29.497.3.103 `virtual void Go::SurfaceModel::turn ( int idx ) [virtual]`

Turn parameter directions of one surface Will turn the orientation of the surface normal An update in the topology structures is required.

## Parameters

<i>idx</i>	Index of surface
------------	------------------

Implements [Go::CompositeModel](#).

29.497.3.104 `virtual void Go::SurfaceModel::turn ( ) [virtual]`

Turn parameter directions of all surfaces An update in the topology structures is required.

Implements [Go::CompositeModel](#).

29.497.3.105 `void Go::SurfaceModel::updateFaceTopology ( shared_ptr< ftSurface > face )`

Update neighbourhood information related to face

## Parameters

<i>face</i>	Pointer to the face
-------------	---------------------

#### 29.497.4 Member Data Documentation

29.497.4.1 **double** `Go::SurfaceModel::approx_tol_` `[protected]`

Definition at line 837 of file `SurfaceModel.h`.

29.497.4.2 **std::vector**<**std::vector**<**shared\_ptr**<**Loop**>>> `Go::SurfaceModel::boundary_curves_` `[protected]`

Definition at line 851 of file `SurfaceModel.h`.

29.497.4.3 **shared\_ptr**<**CellDivision**> `Go::SurfaceModel::celldiv_` `[protected]`

Definition at line 853 of file `SurfaceModel.h`.

29.497.4.4 **std::vector**<**bool**> `Go::SurfaceModel::face_checked_` `[mutable]`, `[protected]`

Definition at line 854 of file `SurfaceModel.h`.

29.497.4.5 **std::vector**<**shared\_ptr**<**ftFaceBase**>> `Go::SurfaceModel::faces_` `[protected]`

Definition at line 847 of file `SurfaceModel.h`.

29.497.4.6 **int** `Go::SurfaceModel::highest_face_checked_` `[mutable]`, `[protected]`

Definition at line 855 of file `SurfaceModel.h`.

29.497.4.7 **std::vector**<**std::pair**<**ftFaceBase\***, **ftFaceBase\***>> `Go::SurfaceModel::inconsistent_orientation_` `[protected]`

Definition at line 859 of file `SurfaceModel.h`.

29.497.4.8 **BoundingBox** `Go::SurfaceModel::limit_box_` `[protected]`

Definition at line 857 of file `SurfaceModel.h`.



29.497.4.9 `double Go::SurfaceModel::tol2d_` [protected]

Definition at line 838 of file SurfaceModel.h.

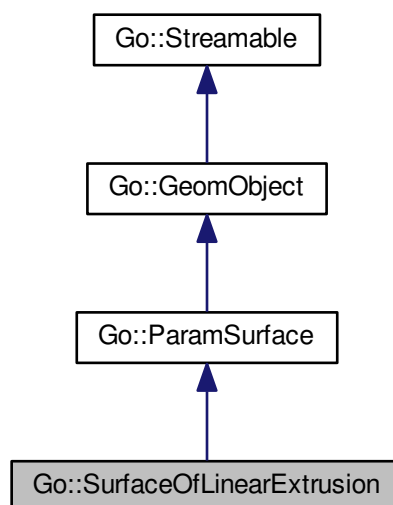
The documentation for this class was generated from the following file:

- [compositemodel/include/GoTools/compositemodel/SurfaceModel.h](#)

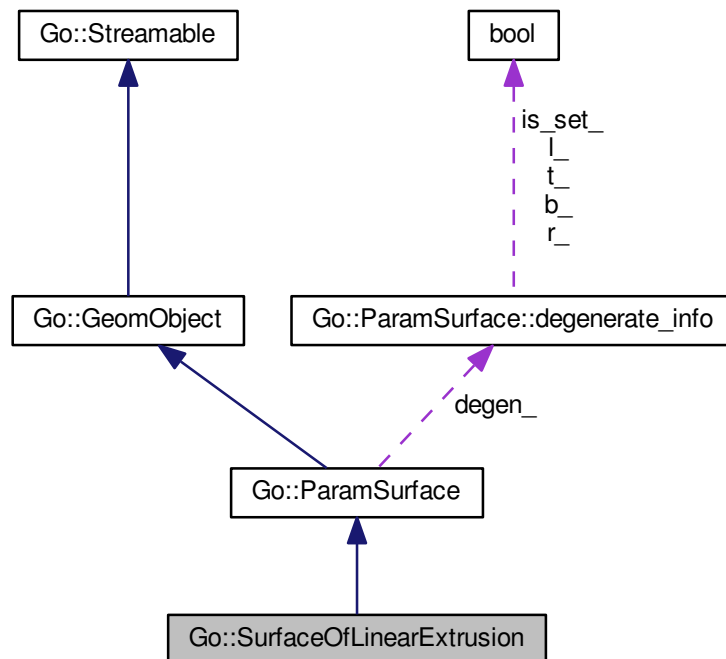
## 29.498 Go::SurfaceOfLinearExtrusion Class Reference

```
#include <SurfaceOfLinearExtrusion.h>
```

Inheritance diagram for Go::SurfaceOfLinearExtrusion:



Collaboration diagram for Go::SurfaceOfLinearExtrusion:



## Public Member Functions

- [SurfaceOfLinearExtrusion](#) ()
- [SurfaceOfLinearExtrusion](#) (shared\_ptr< [SplineCurve](#) > curve, Point axis\_dir, bool isSwapped=false)
- virtual [~SurfaceOfLinearExtrusion](#) ()  
*Virtual destructor - ensures safe inheritance.*
- virtual void [read](#) (std::istream &is)
- virtual void [write](#) (std::ostream &os) const
- virtual int [dimension](#) () const  
*Return the dimension of the space in which the object lies (usually 2 or 3)*
- virtual [ClassType instanceType](#) () const  
*Return the class type identifier of a given, derived instance of [GeomObject](#).*
- virtual [BoundingBox boundingBox](#) () const  
*Return the object's bounding box.*
- virtual [SurfaceOfLinearExtrusion \\* clone](#) () const
- const [RectDomain & parameterDomain](#) () const
- virtual [RectDomain containingDomain](#) () const
- virtual bool [inDomain](#) (double u, double v, double eps=1.0e-4) const  
*Check if a parameter pair lies inside the domain of this surface.*
- virtual int [inDomain2](#) (double u, double v, double eps=1.0e-4) const
- virtual bool [onBoundary](#) (double u, double v, double eps=1.0e-4) const  
*Check if a parameter pair lies at the boundary of this surface.*
- virtual [Point closestInDomain](#) (double u, double v) const

- [CurveLoop](#) [outerBoundaryLoop](#) ([double](#) degenerate\_epsilon=DEFAULT\_SPACE\_EPSILON) [const](#)
- [std::vector](#)< [CurveLoop](#) > [allBoundaryLoops](#) ([double](#) degenerate\_epsilon=DEFAULT\_SPACE\_EPSILON) [const](#)
- [DirectionCone](#) [normalCone](#) () [const](#)
- [DirectionCone](#) [tangentCone](#) ([bool](#) pdir\_is\_u) [const](#)
- [void](#) [point](#) ([Point](#) &pt, [double](#) upar, [double](#) vpar) [const](#)
- [void](#) [point](#) ([std::vector](#)< [Point](#) > &pts, [double](#) upar, [double](#) vpar, [int](#) derivs, [bool](#) u\_from\_right=true, [bool](#) v\_↔\_from\_right=true, [double](#) resolution=1.0e-12) [const](#)
- [void](#) [normal](#) ([Point](#) &n, [double](#) upar, [double](#) vpar) [const](#)
- [std::vector](#)< [shared\\_ptr](#)< [ParamCurve](#) > > [constParamCurves](#) ([double](#) parameter, [bool](#) pdir\_is\_u) [const](#)
- [bool](#) [isBounded](#) () [const](#)
- [std::vector](#)< [shared\\_ptr](#)< [ParamSurface](#) > > [subSurfaces](#) ([double](#) from\_upar, [double](#) from\_vpar, [double](#) to\_upar, [double](#) to\_vpar, [double](#) fuzzy=DEFAULT\_PARAMETER\_EPSILON) [const](#)
- [double](#) [nextSegmentVal](#) ([int](#) dir, [double](#) par, [bool](#) forward, [double](#) tol) [const](#)
- [void](#) [closestPoint](#) ([const](#) [Point](#) &pt, [double](#) &clo\_u, [double](#) &clo\_v, [Point](#) &clo\_pt, [double](#) &clo\_dist, [double](#) epsilon, [const](#) [RectDomain](#) \*domain\_of\_interest=NULL, [double](#) \*seed=0) [const](#)
- [void](#) [closestBoundaryPoint](#) ([const](#) [Point](#) &pt, [double](#) &clo\_u, [double](#) &clo\_v, [Point](#) &clo\_pt, [double](#) &clo\_dist, [double](#) epsilon, [const](#) [RectDomain](#) \*rd=NULL, [double](#) \*seed=0) [const](#)
- [void](#) [getBoundaryInfo](#) ([Point](#) &pt1, [Point](#) &pt2, [double](#) epsilon, [SplineCurve](#) \*&cv, [SplineCurve](#) \*&crosscv, [double](#) knot\_tol=1e-05) [const](#)
- [void](#) [turnOrientation](#) ()
  - Turns the direction of the normal of the surface.*
- [void](#) [reverseParameterDirection](#) ([bool](#) direction\_is\_u)
- [void](#) [swapParameterDirection](#) ()
  - Swaps the two parameter directions.*
- [virtual](#) [double](#) [area](#) ([double](#) tol) [const](#)
- [virtual](#) [void](#) [getDegenerateCorners](#) ([std::vector](#)< [Point](#) > &deg\_corners, [double](#) tol) [const](#)
- [virtual](#) [void](#) [getCornerPoints](#) ([std::vector](#)< [std::pair](#)< [Point](#), [Point](#) > > &corners) [const](#)
  - Return surface corners, geometric and parametric points in that sequence.*
- [Point](#) [getAxisDir](#) () [const](#)
  - Direction of axis of extrusion.*
- [shared\\_ptr](#)< [SplineCurve](#) > [getCurve](#) () [const](#)
  - Generating curve, i.e. the curve that are rotated around the axis.*
- [void](#) [setParameterBounds](#) ([double](#) from\_upar, [double](#) from\_vpar, [double](#) to\_upar, [double](#) to\_vpar)
  - Limit the surface by limiting the parameter domain.*
- [SurfaceOfLinearExtrusion](#) \* [subSurface](#) ([double](#) from\_upar, [double](#) from\_vpar, [double](#) to\_upar, [double](#) to\_↔\_vpar, [double](#) fuzzy=DEFAULT\_PARAMETER\_EPSILON) [const](#)
  - Pick part of surface.*
- [SplineSurface](#) \* [geometrySurface](#) () [const](#)
  - Return spline representation of the surface of revolution.*
- [SplineSurface](#) \* [createSplineSurface](#) () [const](#)
  - Create a [SplineSurface](#) representation of the surface of revolution.*
- [virtual](#) [bool](#) [isSwapped](#) () [const](#)

## Static Public Member Functions

- [static](#) [ClassType](#) [classType](#) ()

## Additional Inherited Members

### 29.498.1 Detailed Description

Definition at line 55 of file [SurfaceOfLinearExtrusion.h](#).

## 29.498.2 Constructor & Destructor Documentation

### 29.498.2.1 `Go::SurfaceOfLinearExtrusion::SurfaceOfLinearExtrusion ( ) [inline]`

Default constructor. Constructs an uninitialized [SurfaceOfLinearExtrusion](#) which can only be assigned to or read into.

Definition at line 61 of file `SurfaceOfLinearExtrusion.h`.

### 29.498.2.2 `Go::SurfaceOfLinearExtrusion::SurfaceOfLinearExtrusion ( shared_ptr< SplineCurve > curve, Point axis_dir, bool isSwapped = false )`

Constructor. Input is the location and normalized direction of the axis, and the [SplineCurve](#) that is swept out by the revolution.

### 29.498.2.3 `virtual Go::SurfaceOfLinearExtrusion::~~SurfaceOfLinearExtrusion ( ) [virtual]`

Virtual destructor - ensures safe inheritance.

## 29.498.3 Member Function Documentation

### 29.498.3.1 `std::vector<CurveLoop> Go::SurfaceOfLinearExtrusion::allBoundaryLoops ( double degenerate_epsilon = DEFAULT_SPACE_EPSILON ) const [virtual]`

Returns the anticlockwise outer boundary loop of the surface, together with clockwise loops of any interior boundaries, such that the surface always is 'to the left of' the loops.

#### Parameters

<i>degenerate_epsilon</i>	edges whose length is smaller than this value are ignored.
---------------------------	------------------------------------------------------------

#### Returns

a vector containing `CurveLoops`. The first of these describe the outer boundary of the surface (clockwise), whereas the others describe boundaries of interior holes (clockwise).

Implements [Go::ParamSurface](#).

### 29.498.3.2 `virtual double Go::SurfaceOfLinearExtrusion::area ( double tol ) const [virtual]`

Compute the total area of this surface up to some tolerance

#### Parameters

<i>tol</i>	the relative tolerance when approximating the area, i.e. stop iteration when error becomes smaller than $tol/(\text{surface area})$
------------	-------------------------------------------------------------------------------------------------------------------------------------

**Returns**

the area calculated

Implements [Go::ParamSurface](#).

**29.498.3.3** virtual **BoundingBox** Go::SurfaceOfLinearExtrusion::boundingBox ( ) const [virtual]

Return the object's bounding box.

Implements [Go::GeomObject](#).

**29.498.3.4** static **ClassType** Go::SurfaceOfLinearExtrusion::classType ( ) [static]

**29.498.3.5** virtual **SurfaceOfLinearExtrusion\*** Go::SurfaceOfLinearExtrusion::clone ( ) const [virtual]

make a clone of this surface and return a pointer to it (user is responsible for clearing up memory afterwards).

**Returns**

pointer to cloned object

Implements [Go::ParamSurface](#).

**29.498.3.6** void Go::SurfaceOfLinearExtrusion::closestBoundaryPoint ( const **Point** & *pt*, double & *clo\_u*, double & *clo\_v*, **Point** & *clo\_pt*, double & *clo\_dist*, double *epsilon*, const **RectDomain** \* *rd* = NULL, double \* *seed* = 0 ) const [virtual]

Iterates to the closest point to *pt* on the boundary of the surface.

**See also**

[closestPoint\(\)](#)

Implements [Go::ParamSurface](#).

**29.498.3.7** virtual **Point** Go::SurfaceOfLinearExtrusion::closestInDomain ( double *u*, double *v* ) const [virtual]

Return the parameter value in the domain of this surface closest to the parameter pair (u,v).

Implements [Go::ParamSurface](#).

**29.498.3.8** void Go::SurfaceOfLinearExtrusion::closestPoint ( const **Point** & *pt*, double & *clo\_u*, double & *clo\_v*, **Point** & *clo\_pt*, double & *clo\_dist*, double *epsilon*, const **RectDomain** \* *domain\_of\_interest* = NULL, double \* *seed* = 0 ) const [virtual]

Iterates to the closest point to *pt* on the surface.

## Parameters

<i>pt</i>	the point to find the closest point to
<i>clo_u</i>	u parameter of the closest point
<i>clo_v</i>	v parameter of the closest point
<i>clo_pt</i>	the geometric position of the closest point
<i>clo_dist</i>	the distance between pt and clo_pt
<i>epsilon</i>	parameter tolerance (will in any case not be higher than sqrt(machine_precision) x magnitude of solution)
<i>domain_of_interest</i>	pointer to parameter domain in which to search for closest point. If a NULL pointer is used, the entire surface is searched.
<i>seed</i>	pointer to parameter values where iteration starts.

Reimplemented from [Go::ParamSurface](#).

```
29.498.3.9 std::vector<shared_ptr<ParamCurve>> Go::SurfaceOfLinearExtrusion::constParamCurves (double
parameter, bool pdir_is_u) const [virtual]
```

Get the curve(s) obtained by intersecting the surface with one of its constant parameter curves. For surfaces without holes, this will be the parameter curve itself; for surfaces with interior holes this may be a collection of several, disjoint curves.

## Parameters

<i>parameter</i>	parameter value for the constant parameter (either u or v)
<i>pdir_is_u</i>	specify whether the <i>moving</i> parameter (as opposed to the <i>constant</i> parameter) is the first ('true') or the second ('false') one.

## Returns

a vector containing shared pointers to the obtained, newly constructed constant-parameter curves.

Implements [Go::ParamSurface](#).

```
29.498.3.10 virtual RectDomain Go::SurfaceOfLinearExtrusion::containingDomain () const [virtual]
```

Get a rectangular parameter domain that is guaranteed to contain the surface's [parameterDomain\(\)](#). It may be the same. There is no guarantee that this is the smallest domain containing the actual domain.

## Returns

a [RectDomain](#) that is guaranteed to include the surface's total parameter domain.

Implements [Go::ParamSurface](#).

```
29.498.3.11 SplineSurface* Go::SurfaceOfLinearExtrusion::createSplineSurface () const
```

Create a [SplineSurface](#) representation of the surface of revolution.

29.498.3.12 `virtual int Go::SurfaceOfLinearExtrusion::dimension ( ) const` `[virtual]`

Return the dimension of the space in which the object lies (usually 2 or 3)

Implements [Go::GeomObject](#).

29.498.3.13 `SplineSurface* Go::SurfaceOfLinearExtrusion::geometrySurface ( ) const`

Return spline representation of the surface of revolution.

29.498.3.14 `Point Go::SurfaceOfLinearExtrusion::getAxisDir ( ) const` `[inline]`

Direction of axis of extrusion.

Definition at line 195 of file `SurfaceOfLinearExtrusion.h`.

29.498.3.15 `void Go::SurfaceOfLinearExtrusion::getBoundaryInfo ( Point & pt1, Point & pt2, double epsilon, SplineCurve *& cv, SplineCurve *& crosscv, double knot_tol=1e-05 ) const` `[virtual]`

Get the boundary curve segment between two points on the boundary, as well as the cross-tangent curve. If the given points are not positioned on the same boundary (within a certain tolerance), no curves will be created.

#### Parameters

<i>pt1</i>	the first point on the boundary, given by the user
<i>pt2</i>	the second point on the boundary, given by the user
<i>epsilon</i>	the tolerance used when determining whether the given points are lying on a boundary, and if they do, whether they both lie on the <i>same</i> boundary.
<i>cv</i>	upon return, this will point to a newly created curve representing the boundary curve between 'pt1' and 'pt2'. The user assumes ownership of this object and is responsible for its deletion. No curve is created if the given points are not found to lie on the same boundary.
<i>crosscv</i>	upon return, this will point to a newly created curve representing the cross-boundary curve between 'pt1' and 'pt2'. The user assumes ownership of this object and is responsible for its deletion. The direction is outwards from the surface. No curve is created if the given points are not found to lie on the same boundary.
<i>knot_tol</i>	tolerance used when working with the knot-vector, to specify how close a parameter value must be to a knot in order to be considered 'on' the knot.

Implements [Go::ParamSurface](#).

29.498.3.16 `virtual void Go::SurfaceOfLinearExtrusion::getCornerPoints ( std::vector< std::pair< Point, Point > > & corners ) const` `[virtual]`

Return surface corners, geometric and parametric points in that sequence.

Implements [Go::ParamSurface](#).

29.498.3.17 `shared_ptr<SplineCurve> Go::SurfaceOfLinearExtrusion::getCurve ( ) const [inline]`

Generating curve, i.e. the curve that are rotated around the axis.

Definition at line 199 of file `SurfaceOfLinearExtrusion.h`.

29.498.3.18 `virtual void Go::SurfaceOfLinearExtrusion::getDegenerateCorners ( std::vector< Point > & deg_corners, double tol ) const [virtual]`

Check for parallel and anti parallel partial derivatives in surface corners

Implements [Go::ParamSurface](#).

29.498.3.19 `virtual bool Go::SurfaceOfLinearExtrusion::inDomain ( double u, double v, double eps = 1.0e-4 ) const [virtual]`

Check if a parameter pair lies inside the domain of this surface.

Implements [Go::ParamSurface](#).

29.498.3.20 `virtual int Go::SurfaceOfLinearExtrusion::inDomain2 ( double u, double v, double eps = 1.0e-4 ) const [virtual]`

Check if a parameter pair lies inside the domain of this surface return value = 0: outside = 1: internal = 2: at the boundary

Implements [Go::ParamSurface](#).

29.498.3.21 `virtual ClassType Go::SurfaceOfLinearExtrusion::instanceType ( ) const [virtual]`

Return the class type identifier of a given, derived instance of [GeomObject](#).

Implements [Go::GeomObject](#).

29.498.3.22 `bool Go::SurfaceOfLinearExtrusion::isBounded ( ) const`

29.498.3.23 `virtual bool Go::SurfaceOfLinearExtrusion::isSwapped ( ) const [virtual]`

29.498.3.24 `double Go::SurfaceOfLinearExtrusion::nextSegmentVal ( int dir, double par, bool forward, double tol ) const [virtual]`

Determine the parameter value of the start of the 'next segment' from a parameter value, along a given parameter direction. A 'segment' is here defined as a parameter interval in which there will be no discontinuities in derivatives or other artifacts. For spline objects, a segment will typically be the interval between two consecutive, non-coincident knots.



## Parameters

<i>dir</i>	the parameter direction in which we search for the next segment (0 or 1)
<i>par</i>	the parameter value starting from which we search for the start value of the next segment
<i>forward</i>	define whether we shall move forward ('true') or backwards when searching along this parameter
<i>tol</i>	tolerance used for determining whether the 'par' is already located <i>on</i> the next segment value

## Returns

the value of the start value of the next segment (or the end of the previous segment, if we are moving backwards...)

Implements [Go::ParamSurface](#).

29.498.3.25 `void Go::SurfaceOfLinearExtrusion::normal ( Point & n, double upar, double vpar ) const` [virtual]

Evaluates the surface normal for a given parameter pair

## Parameters

<i>n</i>	the computed normal will be written to this variable
<i>upar</i>	the first parameter
<i>vpar</i>	the second parameter

Implements [Go::ParamSurface](#).

29.498.3.26 `DirectionCone Go::SurfaceOfLinearExtrusion::normalCone ( ) const` [virtual]

Creates a [DirectionCone](#) covering all normals to this surface.

## Returns

a [DirectionCone](#) (not necessarily the smallest) containing all normals to this surface.

Implements [Go::ParamSurface](#).

29.498.3.27 `virtual bool Go::SurfaceOfLinearExtrusion::onBoundary ( double u, double v, double eps = 1.0e-4 ) const` [virtual]

Check if a parameter pair lies at the boundary of this surface.

Implements [Go::ParamSurface](#).

29.498.3.28 `CurveLoop Go::SurfaceOfLinearExtrusion::outerBoundaryLoop ( double degenerate_epsilon = DEFAULT_SPACE_EPSILON ) const` [virtual]

Returns the anticlockwise, outer boundary loop of the surface.

## Parameters

<i>degenerate_epsilon</i>	edges whose length is smaller than this value are ignored.
---------------------------	------------------------------------------------------------

## Returns

a [CurveLoop](#) describing the anticlockwise, outer boundary loop of the surface. A negative *degenerate\_epsilon* indicates that all curves, also the degenerate ones are wanted

Implements [Go::ParamSurface](#).

29.498.3.29 **const RectDomain& Go::SurfaceOfLinearExtrusion::parameterDomain ( ) const** [virtual]

Return the parameter domain of the surface. This may be a simple rectangular domain ([RectDomain](#)) or any other subclass of [Domain](#) (such as [GoCurveBoundedDomain](#), found in the `sisl_dependent` module).

## Returns

a [Domain](#) object describing the parametric domain of the surface

Implements [Go::ParamSurface](#).

29.498.3.30 **void Go::SurfaceOfLinearExtrusion::point ( Point & pt, double upar, double vpar ) const** [virtual]

Evaluates the surface's position for a given parameter pair.

## Parameters

<i>pt</i>	the result of the evaluation is written here
<i>upar</i>	the first parameter
<i>vpar</i>	the second parameter

Implements [Go::ParamSurface](#).

29.498.3.31 **void Go::SurfaceOfLinearExtrusion::point ( std::vector< Point > & pts, double upar, double vpar, int derivs, bool u\_from\_right = true, bool v\_from\_right = true, double resolution = 1.0e-12 ) const** [virtual]

Evaluates the surface's position and a certain number of derivatives for a given parameter pair.

## Parameters

<i>pts</i>	the vector containing the evaluated <i>pts</i> values. Its size must be set by the user prior to calling this function, and should be equal to $(derivs+1) * (derivs+2) / 2$ . Upon completion of the function, its first entry is the surface's position at the given parameter pair. Then, if <i>'derivs' &gt; 0</i> , the two next entries will be the surface tangents along the first and second parameter direction. The next three entries are the second- and cross derivatives, in the order $(du_2, dudv, dv_2)$ , and similar for even higher derivatives.
------------	-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

## Parameters

<i>upar</i>	the first parameter
<i>vpar</i>	the second parameter
<i>derivs</i>	number of requested derivatives
<i>u_from_right</i>	specify whether derivatives along the first parameter are to be calculated from the right ('true', default) or from the left ('false')
<i>v_from_right</i>	specify whether derivatives along the second parameter are to be calculated from the right ('true', default) or from the left ('false')
<i>resolution</i>	tolerance used when determining whether parameters are located at special values of the parameter domain (in particular; knot values in case of spline objects).

Implements [Go::ParamSurface](#).

29.498.3.32 `virtual void Go::SurfaceOfLinearExtrusion::read ( std::istream & is ) [virtual]`

read object from stream

## Parameters

<i>is</i>	stream from which object is read
-----------	----------------------------------

Implements [Go::Streamable](#).

29.498.3.33 `void Go::SurfaceOfLinearExtrusion::reverseParameterDirection ( bool direction_is_u ) [virtual]`

Reverses the direction of the basis in input direction.

## Parameters

<i>direction_is_u</i>	if 'true', the first parameter direction will be reversed, otherwise, the second parameter direction will be reversed
-----------------------	-----------------------------------------------------------------------------------------------------------------------

Implements [Go::ParamSurface](#).

29.498.3.34 `void Go::SurfaceOfLinearExtrusion::setParameterBounds ( double from_upar, double from_vpar, double to_upar, double to_vpar )`

Limit the surface by limiting the parameter domain.

29.498.3.35 `SurfaceOfLinearExtrusion* Go::SurfaceOfLinearExtrusion::subSurface ( double from_upar, double from_vpar, double to_upar, double to_vpar, double fuzzy = DEFAULT_PARAMETER_EPSILON ) const`

Pick part of surface.

```
29.498.3.36 std::vector<shared_ptr<ParamSurface> > Go::SurfaceOfLinearExtrusion::subSurfaces (double from_upar,
double from_vpar, double to_upar, double to_vpar, double fuzzy = DEFAULT_PARAMETER_EPSILON
) const [virtual]
```

Get the surface(s) obtained by cropping the parameter domain of this surface between given values for the first and second parameter. In general, for surfaces with no interior holes, the result will be *one* surface; however, for surfaces with interior holes, the result might be *several disjoint* surfaces.

#### Parameters

<i>from_upar</i>	lower value for the first parameter in the subdomain
<i>from_vpar</i>	lower value for the second parameter in the subdomain
<i>to_upar</i>	upper value for the first parameter in the subdomain
<i>to_vpar</i>	upper value for the second parameter in the subdomain
<i>fuzzy</i>	tolerance used when determining intersection with interior boundaries

#### Returns

a vector contained shared pointers to the obtained, newly constructed sub-surfaces.

Implements [Go::ParamSurface](#).

```
29.498.3.37 void Go::SurfaceOfLinearExtrusion::swapParameterDirection () [virtual]
```

Swaps the two parameter directions.

Implements [Go::ParamSurface](#).

```
29.498.3.38 DirectionCone Go::SurfaceOfLinearExtrusion::tangentCone (bool paddir_is_u) const [virtual]
```

Creates a [DirectionCone](#) covering all tangents to this surface along a given parameter direction.

#### Parameters

<i>paddir_is_u</i>	if 'true', then the <a href="#">DirectionCone</a> will be defined on basis of the surface's tangents along the first parameter direction. Otherwise the second parameter direction will be used.
--------------------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

#### Returns

a [DirectionCone](#) (not necessarily the smallest) containing all tangents to this surface along the specified parameter direction.

Implements [Go::ParamSurface](#).

```
29.498.3.39 void Go::SurfaceOfLinearExtrusion::turnOrientation () [virtual]
```

Turns the direction of the normal of the surface.

Implements [Go::ParamSurface](#).

29.498.3.40 `virtual void Go::SurfaceOfLinearExtrusion::write ( std::ostream & os ) const` [virtual]

write object to stream

#### Parameters

<code>os</code>	stream to which object is written
-----------------	-----------------------------------

Implements [Go::Streamable](#).

The documentation for this class was generated from the following file:

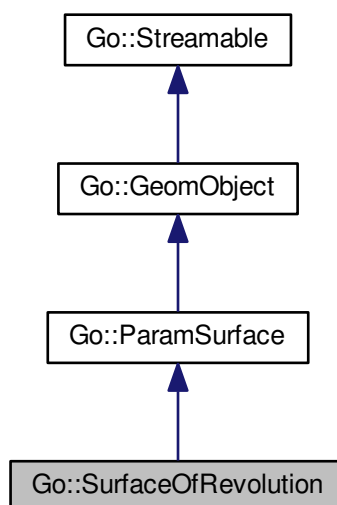
- `gotools-core/include/GoTools/geometry/SurfaceOfLinearExtrusion.h`

## 29.499 Go::SurfaceOfRevolution Class Reference

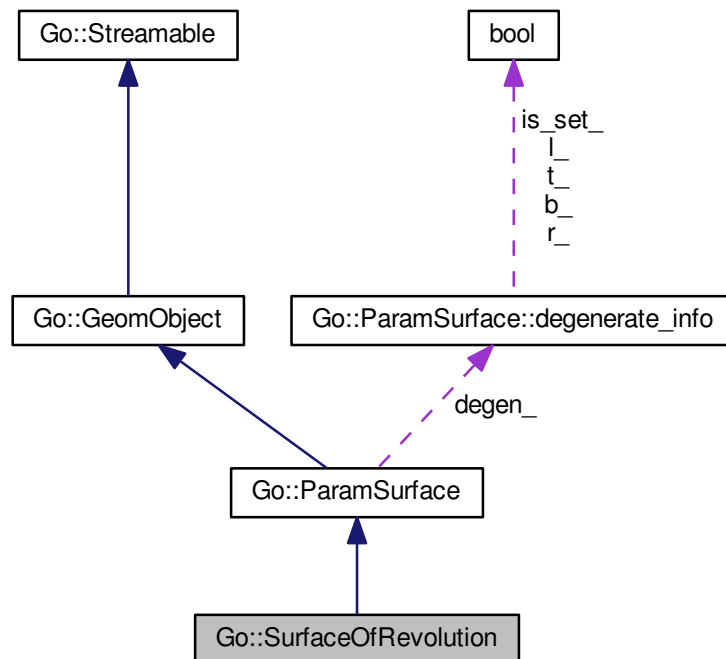
Class that represents a surface of revolution. A [SurfaceOfRevolution](#) is swept out by a [SplineCurve](#) that is rotated around an axis with a complete revolution, and is thereby a parametric surface.

```
#include <SurfaceOfRevolution.h>
```

Inheritance diagram for `Go::SurfaceOfRevolution`:



Collaboration diagram for Go::SurfaceOfRevolution:



## Public Member Functions

- [SurfaceOfRevolution](#) ()
- [SurfaceOfRevolution](#) ([Point](#) location, [Point](#) axis\_dir, [shared\\_ptr](#)< [SplineCurve](#) > curve, [bool](#) is↔  
Swapped=false)
- virtual [~SurfaceOfRevolution](#) ()  
*Virtual destructor - ensures safe inheritance.*
- virtual void [read](#) ([std::istream](#) &is)
- virtual void [write](#) ([std::ostream](#) &os) **const**
- virtual int [dimension](#) () **const**  
*Return the dimension of the space in which the object lies (usually 2 or 3)*
- virtual [ClassType](#) [instanceType](#) () **const**  
*Return the class type identifier of a given, derived instance of [GeomObject](#).*
- virtual [BoundingBox](#) [boundingBox](#) () **const**  
*Return the object's bounding box.*
- virtual [SurfaceOfRevolution](#) \* [clone](#) () **const**
- **const** [RectDomain](#) & [parameterDomain](#) () **const**
- virtual [RectDomain](#) [containingDomain](#) () **const**
- virtual [bool](#) [inDomain](#) ([double](#) u, [double](#) v, [double](#) eps=1.0e-4) **const**  
*Check if a parameter pair lies inside the domain of this surface.*
- virtual int [inDomain2](#) ([double](#) u, [double](#) v, [double](#) eps=1.0e-4) **const**
- virtual [bool](#) [onBoundary](#) ([double](#) u, [double](#) v, [double](#) eps=1.0e-4) **const**  
*Check if a parameter pair lies at the boundary of this surface.*

- virtual [Point](#) [closestInDomain](#) (double u, double v) const
- [CurveLoop](#) [outerBoundaryLoop](#) (double degenerate\_epsilon=DEFAULT\_SPACE\_EPSILON) const
- std::vector< [CurveLoop](#) > [allBoundaryLoops](#) (double degenerate\_epsilon=DEFAULT\_SPACE\_EPSILON) const
- [DirectionCone](#) [normalCone](#) () const
- [DirectionCone](#) [tangentCone](#) (bool paddir\_is\_u) const
- void [point](#) ([Point](#) &pt, double upar, double vpar) const
- void [point](#) (std::vector< [Point](#) > &pts, double upar, double vpar, int derivs, bool u\_from\_right=true, bool v\_from\_right=true, double resolution=1.0e-12) const
- void [normal](#) ([Point](#) &n, double upar, double vpar) const
- std::vector< shared\_ptr< [ParamCurve](#) > > [constParamCurves](#) (double parameter, bool paddir\_is\_u) const
- std::vector< shared\_ptr< [ParamSurface](#) > > [subSurfaces](#) (double from\_upar, double from\_vpar, double to\_upar, double to\_vpar, double fuzzy=DEFAULT\_PARAMETER\_EPSILON) const
- double [nextSegmentVal](#) (int dir, double par, bool forward, double tol) const
- void [closestPoint](#) (const [Point](#) &pt, double &clo\_u, double &clo\_v, [Point](#) &clo\_pt, double &clo\_dist, double epsilon, const [RectDomain](#) \*domain\_of\_interest=NULL, double \*seed=0) const
- void [closestBoundaryPoint](#) (const [Point](#) &pt, double &clo\_u, double &clo\_v, [Point](#) &clo\_pt, double &clo\_dist, double epsilon, const [RectDomain](#) \*rd=NULL, double \*seed=0) const
- void [getBoundaryInfo](#) ([Point](#) &pt1, [Point](#) &pt2, double epsilon, [SplineCurve](#) \*&cv, [SplineCurve](#) \*&crosscv, double knot\_tol=1e-05) const
- void [turnOrientation](#) ()
  - Turns the direction of the normal of the surface.*
- void [reverseParameterDirection](#) (bool direction\_is\_u)
- void [swapParameterDirection](#) ()
  - Swaps the two parameter directions.*
- virtual double [area](#) (double tol) const
- virtual void [getDegenerateCorners](#) (std::vector< [Point](#) > &deg\_corners, double tol) const
- virtual void [getCornerPoints](#) (std::vector< std::pair< [Point](#), [Point](#) > > &corners) const
  - Return surface corners, geometric and parametric points in that sequence.*
- [Point](#) [getLocation](#) () const
  - Point on axis of revolution.*
- [Point](#) [getAxisDir](#) () const
  - Direction of axis of revolution.*
- shared\_ptr< [SplineCurve](#) > [getCurve](#) () const
  - Generating curve, i.e. the curve that are rotated around the axis.*
- void [setParameterBounds](#) (double from\_upar, double from\_vpar, double to\_upar, double to\_vpar)
  - Limit the surface by limiting the parameter domain.*
- [SurfaceOfRevolution](#) \* [subSurface](#) (double from\_upar, double from\_vpar, double to\_upar, double to\_vpar, double fuzzy=DEFAULT\_PARAMETER\_EPSILON) const
  - Pick part of surface.*
- [SplineSurface](#) \* [geometrySurface](#) () const
  - Return spline representation of the surface of revolution.*
- [SplineSurface](#) \* [createSplineSurface](#) () const
  - Create a [SplineSurface](#) representation of the surface of revolution.*
- shared\_ptr< [Circle](#) > [getCircle](#) (double vpar) const
- virtual bool [isSwapped](#) () const

## Static Public Member Functions

- static [ClassType](#) [classType](#) ()

## Additional Inherited Members

### 29.499.1 Detailed Description

Class that represents a surface of revolution. A [SurfaceOfRevolution](#) is swept out by a [SplineCurve](#) that is rotated around an axis with a complete revolution, and is thereby a parametric surface.

The parametrization of the surface is given in terms of a location  $\mathbf{C}$ , an axis line  $\mathbf{V}$ , and the spline curve  $\lambda(v)$  with parameter  $v$ :

$$\sigma(u, v) = C + (\lambda(v) - C) \cos u + ((\lambda(v) - C) \cdot V)V(1 - \cos u) + V \times (\lambda(v) - C) \sin u$$

The parameter  $u$  is bounded by:  $0 \leq u \leq 2\pi$ . The axis  $\mathbf{V}$  is normalized.

The curve  $\lambda$  must be such that it doesn't lead to a self-intersecting surface.

Definition at line 77 of file `SurfaceOfRevolution.h`.

### 29.499.2 Constructor & Destructor Documentation

#### 29.499.2.1 `Go::SurfaceOfRevolution::SurfaceOfRevolution ( ) [inline]`

Default constructor. Constructs an uninitialized [SurfaceOfRevolution](#) which can only be assigned to or read into.

Definition at line 83 of file `SurfaceOfRevolution.h`.

#### 29.499.2.2 `Go::SurfaceOfRevolution::SurfaceOfRevolution ( Point location, Point axis_dir, shared_ptr< SplineCurve > curve, bool isSwapped = false )`

Constructor. Input is the location and normalized direction of the axis, and the [SplineCurve](#) that is swept out by the revolution.

#### 29.499.2.3 `virtual Go::SurfaceOfRevolution::~~SurfaceOfRevolution ( ) [virtual]`

Virtual destructor - ensures safe inheritance.

### 29.499.3 Member Function Documentation

#### 29.499.3.1 `std::vector<CurveLoop> Go::SurfaceOfRevolution::allBoundaryLoops ( double degenerate_epsilon = DEFAULT_SPACE_EPSILON ) const [virtual]`

Returns the anticlockwise outer boundary loop of the surface, together with clockwise loops of any interior boundaries, such that the surface always is 'to the left of' the loops.

Parameters

<code>degenerate_epsilon</code>	edges whose length is smaller than this value are ignored.
---------------------------------	------------------------------------------------------------



**Returns**

a vector containing CurveLoops. The first of these describe the outer boundary of the surface (clockwise), whereas the others describe boundaries of interior holes (clockwise).

Implements [Go::ParamSurface](#).

**29.499.3.2** `virtual double Go::SurfaceOfRevolution::area ( double tol ) const [virtual]`

Compute the total area of this surface up to some tolerance

**Parameters**

<i>tol</i>	the relative tolerance when approximating the area, i.e. stop iteration when error becomes smaller than $tol/(surface\ area)$
------------	-------------------------------------------------------------------------------------------------------------------------------

**Returns**

the area calculated

Implements [Go::ParamSurface](#).

**29.499.3.3** `virtual BoundingBox Go::SurfaceOfRevolution::boundingBox ( ) const [virtual]`

Return the object's bounding box.

Implements [Go::GeomObject](#).

**29.499.3.4** `static ClassType Go::SurfaceOfRevolution::classType ( ) [static]`

**29.499.3.5** `virtual SurfaceOfRevolution* Go::SurfaceOfRevolution::clone ( ) const [virtual]`

make a clone of this surface and return a pointer to it (user is responsible for clearing up memory afterwards).

**Returns**

pointer to cloned object

Implements [Go::ParamSurface](#).

**29.499.3.6** `void Go::SurfaceOfRevolution::closestBoundaryPoint ( const Point & pt, double & clo_u, double & clo_v, Point & clo_pt, double & clo_dist, double epsilon, const RectDomain * rd = NULL, double * seed = 0 ) const [virtual]`

Iterates to the closest point to *pt* on the boundary of the surface.

**See also**

[closestPoint\(\)](#)

Implements [Go::ParamSurface](#).

29.499.3.7 `virtual Point Go::SurfaceOfRevolution::closestInDomain ( double u, double v ) const` [virtual]

Return the parameter value in the domain of this surface closest to the parameter pair (u,v).

Implements [Go::ParamSurface](#).

29.499.3.8 `void Go::SurfaceOfRevolution::closestPoint ( const Point & pt, double & clo_u, double & clo_v, Point & clo_pt, double & clo_dist, double epsilon, const RectDomain * domain_of_interest = NULL, double * seed = 0 ) const` [virtual]

Iterates to the closest point to pt on the surface.

#### Parameters

<i>pt</i>	the point to find the closest point to
<i>clo_u</i>	u parameter of the closest point
<i>clo_v</i>	v parameter of the closest point
<i>clo_pt</i>	the geometric position of the closest point
<i>clo_dist</i>	the distance between pt and clo_pt
<i>epsilon</i>	parameter tolerance (will in any case not be higher than sqrt(machine_precision) x magnitude of solution)
<i>domain_of_interest</i>	pointer to parameter domain in which to search for closest point. If a NULL pointer is used, the entire surface is searched.
<i>seed</i>	pointer to parameter values where iteration starts.

Reimplemented from [Go::ParamSurface](#).

29.499.3.9 `std::vector<shared_ptr<ParamCurve>> Go::SurfaceOfRevolution::constParamCurves ( double parameter, bool pdir_is_u ) const` [virtual]

Get the curve(s) obtained by intersecting the surface with one of its constant parameter curves. For surfaces without holes, this will be the parameter curve itself; for surfaces with interior holes this may be a collection of several, disjoint curves.

#### Parameters

<i>parameter</i>	parameter value for the constant parameter (either u or v)
<i>pdir_is_u</i>	specify whether the <i>moving</i> parameter (as opposed to the <i>constant</i> parameter) is the first ('true') or the second ('false') one.

#### Returns

a vector containing shared pointers to the obtained, newly constructed constant-parameter curves.

Implements [Go::ParamSurface](#).

29.499.3.10 `virtual RectDomain Go::SurfaceOfRevolution::containingDomain ( ) const` [virtual]

Get a rectangular parameter domain that is guaranteed to contain the surface's [parameterDomain\(\)](#). It may be the same. There is no guarantee that this is the smallest domain containing the actual domain.

## Returns

a [RectDomain](#) that is guaranteed to include the surface's total parameter domain.

Implements [Go::ParamSurface](#).

29.499.3.11 [SplineSurface\\*](#) [Go::SurfaceOfRevolution::createSplineSurface \( \) const](#)

Create a [SplineSurface](#) representation of the surface of revolution.

29.499.3.12 `virtual int` [Go::SurfaceOfRevolution::dimension \( \) const](#) `[virtual]`

Return the dimension of the space in which the object lies (usually 2 or 3)

Implements [Go::GeomObject](#).

29.499.3.13 [SplineSurface\\*](#) [Go::SurfaceOfRevolution::geometrySurface \( \) const](#)

Return spline representation of the surface of revolution.

29.499.3.14 `Point` [Go::SurfaceOfRevolution::getAxisDir \( \) const](#) `[inline]`

Direction of axis of revolution.

Definition at line 219 of file `SurfaceOfRevolution.h`.

29.499.3.15 `void` [Go::SurfaceOfRevolution::getBoundaryInfo \( `Point & pt1`, `Point & pt2`, `double epsilon`, `SplineCurve \*& cv`, `SplineCurve \*& crosscv`, `double knot\_tol = 1e-05` \) const](#) `[virtual]`

Get the boundary curve segment between two points on the boundary, as well as the cross-tangent curve. If the given points are not positioned on the same boundary (within a certain tolerance), no curves will be created.

## Parameters

<i>pt1</i>	the first point on the boundary, given by the user
<i>pt2</i>	the second point on the boundary, given by the user
<i>epsilon</i>	the tolerance used when determining whether the given points are lying on a boundary, and if they do, whether they both lie on the <i>same</i> boundary.
<i>cv</i>	upon return, this will point to a newly created curve representing the boundary curve between 'pt1' and 'pt2'. The user assumes ownership of this object and is responsible for its deletion. No curve is created if the given points are not found to lie on the same boundary.
<i>crosscv</i>	upon return, this will point to a newly created curve representing the cross-boundary curve between 'pt1' and 'pt2'. The user assumes ownership of this object and is responsible for its deletion. The direction is outwards from the surface. No curve is created if the given points are not found to lie on the same boundary.
<i>knot_tol</i>	tolerance used when working with the knot-vector, to specify how close a parameter value must be to a knot in order to be considered 'on' the knot.

Implements [Go::ParamSurface](#).

29.499.3.16 `shared_ptr<Circle> Go::SurfaceOfRevolution::getCircle ( double vpar ) const`

Get the circle for a given v parameter.

#### Parameters

<i>vpar</i>	v parameter
-------------	-------------

#### Returns

A circle for the corresponding v parameter. If the v parameter is bounded, only a segment of a full circle is returned.

29.499.3.17 `virtual void Go::SurfaceOfRevolution::getCornerPoints ( std::vector< std::pair< Point, Point > > & corners ) const [virtual]`

Return surface corners, geometric and parametric points in that sequence.

Implements [Go::ParamSurface](#).

29.499.3.18 `shared_ptr<SplineCurve> Go::SurfaceOfRevolution::getCurve ( ) const [inline]`

Generating curve, i.e. the curve that are rotated around the axis.

Definition at line 223 of file SurfaceOfRevolution.h.

29.499.3.19 `virtual void Go::SurfaceOfRevolution::getDegenerateCorners ( std::vector< Point > & deg_corners, double tol ) const [virtual]`

Check for parallel and anti parallel partial derivatives in surface corners

Implements [Go::ParamSurface](#).

29.499.3.20 `Point Go::SurfaceOfRevolution::getLocation ( ) const [inline]`

[Point](#) on axis of revolution.

Definition at line 215 of file SurfaceOfRevolution.h.

29.499.3.21 `virtual bool Go::SurfaceOfRevolution::inDomain ( double u, double v, double eps = 1.0e-4 ) const [virtual]`

Check if a parameter pair lies inside the domain of this surface.

Implements [Go::ParamSurface](#).

29.499.3.22 `virtual int Go::SurfaceOfRevolution::inDomain2 ( double u, double v, double eps = 1.0e-4 ) const` [virtual]

Check if a parameter pair lies inside the domain of this surface return value = 0: outside = 1: internal = 2: at the boundary

Implements [Go::ParamSurface](#).

29.499.3.23 `virtual ClassType Go::SurfaceOfRevolution::instanceType ( ) const` [virtual]

Return the class type identifier of a given, derived instance of [GeomObject](#).

Implements [Go::GeomObject](#).

29.499.3.24 `virtual bool Go::SurfaceOfRevolution::isSwapped ( ) const` [virtual]

29.499.3.25 `double Go::SurfaceOfRevolution::nextSegmentVal ( int dir, double par, bool forward, double tol ) const` [virtual]

Determine the parameter value of the start of the 'next segment' from a parameter value, along a given parameter direction. A 'segment' is here defined as a parameter interval in which there will be no discontinuities in derivatives or other artifacts. For spline objects, a segment will typically be the interval between two consecutive, non-coincident knots.

#### Parameters

<i>dir</i>	the parameter direction in which we search for the next segment (0 or 1)
<i>par</i>	the parameter value starting from which we search for the start value of the next segment
<i>forward</i>	define whether we shall move forward ('true') or backwards when searching along this parameter
<i>tol</i>	tolerance used for determining whether the 'par' is already located <i>on</i> the next segment value

#### Returns

the value of the start value of the next segment (or the end of the previous segment, if we are moving backwards...)

Implements [Go::ParamSurface](#).

29.499.3.26 `void Go::SurfaceOfRevolution::normal ( Point & n, double upar, double vpar ) const` [virtual]

Evaluates the surface normal for a given parameter pair

#### Parameters

<i>n</i>	the computed normal will be written to this variable
<i>upar</i>	the first parameter
<i>vpar</i>	the second parameter

Implements [Go::ParamSurface](#).

29.499.3.27 **DirectionCone** `Go::SurfaceOfRevolution::normalCone ( ) const` [virtual]

Creates a [DirectionCone](#) covering all normals to this surface.

Returns

a [DirectionCone](#) (not necessarily the smallest) containing all normals to this surface.

Implements [Go::ParamSurface](#).

29.499.3.28 **virtual bool** `Go::SurfaceOfRevolution::onBoundary ( double u, double v, double eps = 1.0e-4 ) const` [virtual]

Check if a parameter pair lies at the boundary of this surface.

Implements [Go::ParamSurface](#).

29.499.3.29 **CurveLoop** `Go::SurfaceOfRevolution::outerBoundaryLoop ( double degenerate_epsilon = DEFAULT_SPACE_EPSILON ) const` [virtual]

Returns the anticlockwise, outer boundary loop of the surface.

Parameters

<i>degenerate_epsilon</i>	edges whose length is smaller than this value are ignored.
---------------------------	------------------------------------------------------------

Returns

a [CurveLoop](#) describing the anticlockwise, outer boundary loop of the surface. A negative `degenerate_epsilon` indicates that all curves, also the degenerate ones are wanted

Implements [Go::ParamSurface](#).

29.499.3.30 **const RectDomain&** `Go::SurfaceOfRevolution::parameterDomain ( ) const` [virtual]

Return the parameter domain of the surface. This may be a simple rectangular domain ([RectDomain](#)) or any other subclass of [Domain](#) (such as [GoCurveBoundedDomain](#), found in the `sisl_dependent` module).

Returns

a [Domain](#) object describing the parametric domain of the surface

Implements [Go::ParamSurface](#).

29.499.3.31 **void** `Go::SurfaceOfRevolution::point ( Point & pt, double upar, double vpar ) const` [virtual]

Evaluates the surface's position for a given parameter pair.

## Parameters

<i>pt</i>	the result of the evaluation is written here
<i>upar</i>	the first parameter
<i>vpar</i>	the second parameter

Implements [Go::ParamSurface](#).

```
29.499.332 void Go::SurfaceOfRevolution::point (std::vector< Point > & pts, double upar, double vpar, int derivs, bool
u_from_right = true, bool v_from_right = true, double resolution = 1.0e-12) const [virtual]
```

Evaluates the surface's position and a certain number of derivatives for a given parameter pair.

## Parameters

<i>pts</i>	the vector containing the evaluated values. Its size must be set by the user prior to calling this function, and should be equal to $(derivs+1) * (derivs+2) / 2$ . Upon completion of the function, its first entry is the surface's position at the given parameter pair. Then, if 'derivs' > 0, the two next entries will be the surface tangents along the first and second parameter direction. The next three entries are the second- and cross derivatives, in the order (du2, dudv, dv2), and similar for even higher derivatives.
<i>upar</i>	the first parameter
<i>vpar</i>	the second parameter
<i>derivs</i>	number of requested derivatives
<i>u_from_right</i>	specify whether derivatives along the first parameter are to be calculated from the right ('true', default) or from the left ('false')
<i>v_from_right</i>	specify whether derivatives along the second parameter are to be calculated from the right ('true', default) or from the left ('false')
<i>resolution</i>	tolerance used when determining whether parameters are located at special values of the parameter domain (in particular; knot values in case of spline objects).

Implements [Go::ParamSurface](#).

```
29.499.333 virtual void Go::SurfaceOfRevolution::read (std::istream & is) [virtual]
```

read object from stream

## Parameters

<i>is</i>	stream from which object is read
-----------	----------------------------------

Implements [Go::Streamable](#).

```
29.499.334 void Go::SurfaceOfRevolution::reverseParameterDirection (bool direction_is_u) [virtual]
```

Reverses the direction of the basis in input direction.

## Parameters

<i>direction_is_u</i>	if 'true', the first parameter direction will be reversed, otherwise, the second parameter direction will be reversed
-----------------------	-----------------------------------------------------------------------------------------------------------------------

Implements [Go::ParamSurface](#).

```
29.499.3.35 void Go::SurfaceOfRevolution::setParameterBounds (double from_upar, double from_vpar, double to_upar,
double to_vpar)
```

Limit the surface by limiting the parameter domain.

```
29.499.3.36 SurfaceOfRevolution* Go::SurfaceOfRevolution::subSurface (double from_upar, double from_vpar,
double to_upar, double to_vpar, double fuzzy = DEFAULT_PARAMETER_EPSILON) const
```

Pick part of surface.

```
29.499.3.37 std::vector<shared_ptr<ParamSurface>> Go::SurfaceOfRevolution::subSurfaces (double from_upar,
double from_vpar, double to_upar, double to_vpar, double fuzzy = DEFAULT_PARAMETER_EPSILON
) const [virtual]
```

Get the surface(s) obtained by cropping the parameter domain of this surface between given values for the first and second parameter. In general, for surfaces with no interior holes, the result will be *one* surface; however, for surfaces with interior holes, the result might be *several disjoint* surfaces.

## Parameters

<i>from_upar</i>	lower value for the first parameter in the subdomain
<i>from_vpar</i>	lower value for the second parameter in the subdomain
<i>to_upar</i>	upper value for the first parameter in the subdomain
<i>to_vpar</i>	upper value for the second parameter in the subdomain
<i>fuzzy</i>	tolerance used when determining intersection with interior boundaries

## Returns

a vector contained shared pointers to the obtained, newly constructed sub-surfaces.

Implements [Go::ParamSurface](#).

```
29.499.3.38 void Go::SurfaceOfRevolution::swapParameterDirection () [virtual]
```

Swaps the two parameter directions.

Implements [Go::ParamSurface](#).

```
29.499.3.39 DirectionCone Go::SurfaceOfRevolution::tangentCone (bool pdir_is_u) const [virtual]
```

Creates a [DirectionCone](#) covering all tangents to this surface along a given parameter direction.



## Parameters

<code>pardir_is↔ _u</code>	if 'true', then the <a href="#">DirectionCone</a> will be defined on basis of the surface's tangents along the first parameter direction. Otherwise the second parameter direction will be used.
--------------------------------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

## Returns

a [DirectionCone](#) (not necessarily the smallest) containing all tangents to this surface along the specified parameter direction.

Implements [Go::ParamSurface](#).

29.499.3.40 `void Go::SurfaceOfRevolution::turnOrientation ( ) [virtual]`

Turns the direction of the normal of the surface.

Implements [Go::ParamSurface](#).

29.499.3.41 `virtual void Go::SurfaceOfRevolution::write ( std::ostream & os ) const [virtual]`

write object to stream

## Parameters

<code>os</code>	stream to which object is written
-----------------	-----------------------------------

Implements [Go::Streamable](#).

The documentation for this class was generated from the following file:

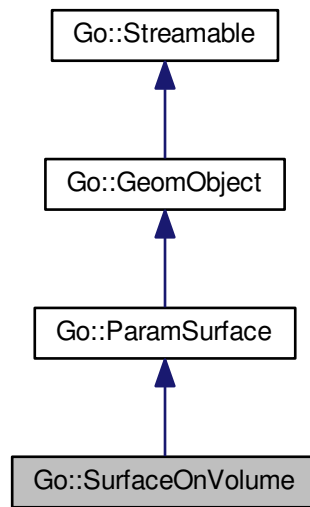
- [gotools-core/include/GoTools/geometry/SurfaceOfRevolution.h](#)

## 29.500 Go::SurfaceOnVolume Class Reference

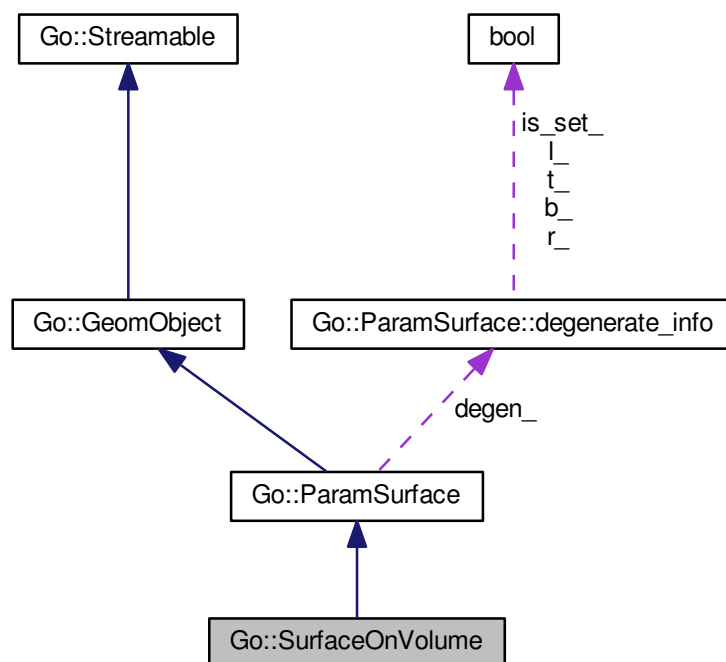
A surface living on a parametric volume. It either has got information about the surface in geometry space and in the parameter domain of the volume or both. The surface may have information on whether it is a constant parameter or boundary surface on the volume.

```
#include <SurfaceOnVolume.h>
```

Inheritance diagram for Go::SurfaceOnVolume:



Collaboration diagram for Go::SurfaceOnVolume:



## Public Member Functions

- [SurfaceOnVolume](#) ()  
*Empty constructor.*
- [SurfaceOnVolume](#) (shared\_ptr< [ParamVolume](#) > vol, shared\_ptr< [ParamSurface](#) > parsurf, shared\_ptr< [ParamSurface](#) > spacesurf, bool preferparameter)
- [SurfaceOnVolume](#) (shared\_ptr< [ParamVolume](#) > vol, shared\_ptr< [ParamSurface](#) > spacesurf, int constdir, double constpar, int boundary, bool swapped, int orientation=0)
- [SurfaceOnVolume](#) (shared\_ptr< [ParamVolume](#) > vol, int constdir, double constpar, int boundary)
- [SurfaceOnVolume](#) (shared\_ptr< [ParamVolume](#) > vol, shared\_ptr< [ParamSurface](#) > spacesurf, shared\_ptr< [ParamSurface](#) > parsurf, bool prefer\_parameter, int constdir, double constpar, int boundary, bool swapped)
- [SurfaceOnVolume](#) (const [SurfaceOnVolume](#) &other)  
*Assignment constructor.*
- virtual ~[SurfaceOnVolume](#) ()  
*Destructor.*
- virtual void [read](#) (std::istream &is)
- virtual void [write](#) (std::ostream &os) const
- virtual [BoundingBox](#) boundingBox () const  
*Axis align box surrounding this object.*
- virtual int [dimension](#) () const  
*Dimension of geometry space.*
- virtual [ClassType](#) instanceType () const  
*Return the class type identifier of type [SurfaceOnVolume](#).*
- virtual [SurfaceOnVolume](#) \* [clone](#) () const
- virtual [SplineSurface](#) \* [asSplineSurface](#) ()  
*Return the spline surface represented by this surface, if any.*
- virtual const [Domain](#) & [parameterDomain](#) () const
- virtual [RectDomain](#) containingDomain () const
- virtual bool [inDomain](#) (double u, double v, double eps=1.0e-4) const  
*Check if a parameter pair lies inside the domain of this surface.*
- virtual int [inDomain2](#) (double u, double v, double eps=1.0e-4) const
- virtual bool [onBoundary](#) (double u, double v, double eps=1.0e-4) const  
*Check if a parameter pair lies at the boundary of this surface.*
- virtual [Point](#) [closestInDomain](#) (double u, double v) const
- virtual [CurveLoop](#) [outerBoundaryLoop](#) (double degenerate\_epsilon=DEFAULT\_SPACE\_EPSILON) const
- virtual std::vector< [CurveLoop](#) > [allBoundaryLoops](#) (double degenerate\_epsilon=DEFAULT\_SPACE\_EPSILON) const
- virtual [DirectionCone](#) [normalCone](#) () const
- virtual [DirectionCone](#) [tangentCone](#) (bool paddir\_is\_u) const
- virtual [CompositeBox](#) [compositeBox](#) () const
- virtual void [point](#) ([Point](#) &pt, double upar, double vpar) const
- virtual void [point](#) (std::vector< [Point](#) > &pts, double upar, double vpar, int derivs, bool u\_from\_right=true, bool v\_from\_right=true, double resolution=1.0e-12) const
- virtual void [normal](#) ([Point](#) &n, double upar, double vpar) const
- virtual std::vector< shared\_ptr< [ParamCurve](#) > > [constParamCurves](#) (double parameter, bool paddir\_is\_u) const
- virtual std::vector< shared\_ptr< [ParamSurface](#) > > [subSurfaces](#) (double from\_upar, double from\_vpar, double to\_upar, double to\_vpar, double fuzzy=DEFAULT\_PARAMETER\_EPSILON) const
- virtual double [nextSegmentVal](#) (int dir, double par, bool forward, double tol) const
- virtual void [closestPoint](#) (const [Point](#) &pt, double &clo\_u, double &clo\_v, [Point](#) &clo\_pt, double &clo\_dist, double epsilon, const [RectDomain](#) \*domain\_of\_interest=NULL, double \*seed=0) const
- virtual void [closestBoundaryPoint](#) (const [Point](#) &pt, double &clo\_u, double &clo\_v, [Point](#) &clo\_pt, double &clo\_dist, double epsilon, const [RectDomain](#) \*rd=NULL, double \*seed=0) const

- virtual void [getBoundaryInfo](#) ([Point](#) &pt1, [Point](#) &pt2, [double](#) epsilon, [SplineCurve](#) \*&cv, [SplineCurve](#) \*&crosscv, [double](#) knot\_tol=1e-05) const
- virtual void [turnOrientation](#) ()
  - Turns the direction of the normal of the surface.*
- virtual void [reverseParameterDirection](#) ([bool](#) direction\_is\_u)
- virtual void [swapParameterDirection](#) ()
  - Swaps the two parameter directions.*
- virtual [double](#) [area](#) ([double](#) tol) const
- virtual [bool](#) [isDegenerate](#) ([bool](#) &b, [bool](#) &r, [bool](#) &t, [bool](#) &l, [double](#) tolerance) const
- virtual void [getDegenerateCorners](#) (std::vector< [Point](#) > &deg\_corners, [double](#) tol) const
  - Check for parallell and anti parallell partial derivatives in surface corners.*
- virtual void [getCornerPoints](#) (std::vector< std::pair< [Point](#), [Point](#) > > &corners) const
- virtual [bool](#) [isAlsoTrimmed](#) ([double](#) tol) const
  - Check if the current surface is trimmed along constant parameter curves.*
- int [whichBoundary](#) ([double](#) tol, int &orientation, [bool](#) &swap) const
- [Point](#) [volumeParameter](#) ([double](#) u\_par, [double](#) v\_par) const
  - Volume parameter corresponding to surface parameter.*
- shared\_ptr< [const ParamVolume](#) > [getVolume](#) () const
  - Get volume.*
- shared\_ptr< [ParamVolume](#) > [getVolume](#) ()
  - Get volume.*
- void [setVolume](#) (shared\_ptr< [ParamVolume](#) > volume)
  - Set/replace volume. NB! Use care.*
- shared\_ptr< [const ParamSurface](#) > [parameterSurface](#) () const
  - Get parameter surface.*
- shared\_ptr< [ParamSurface](#) > [parameterSurface](#) ()
  - Get parameter surface.*
- shared\_ptr< [const ParamSurface](#) > [spaceSurface](#) () const
  - Get space surface.*
- shared\_ptr< [ParamSurface](#) > [spaceSurface](#) ()
  - Get space surface.*
- void [setSpaceSurface](#) (shared\_ptr< [ParamSurface](#) > spacesurf)
  - For internal use. Careful!*
- [bool](#) [parPref](#) () const
- [double](#) [getConstVal](#) () const
  - Fetch the constant parameter value of the volume associated with this surface.*
- int [getConstDir](#) () const
  - Fetch the constant parameter direction of the volume, if any, associated.*
- void [unsetParamSurf](#) ()
  - Unset parameter surface information.*
- virtual [bool](#) [isLinear](#) ([Point](#) &dir1, [Point](#) &dir2, [double](#) tol)
  - Check if the surface is linear in one or both parameter directions.*
- virtual int [ElementOnBoundary](#) (int elem\_ix, [double](#) eps)
- virtual int [ElementBoundaryStatus](#) (int elem\_ix, [double](#) eps)

## Static Public Member Functions

- static [ClassType](#) [classType](#) ()
  - Return the class type identifier of type [SurfaceOnVolumeBoundedSurface](#).*

## Additional Inherited Members

### 29.500.1 Detailed Description

A surface living on a parametric volume. It either has got information about the surface in geometry space and in the parameter domain of the volume or both. The surface may have information on whether it is a constant parameter or boundary surface on the volume.

Definition at line 54 of file SurfaceOnVolume.h.

### 29.500.2 Constructor & Destructor Documentation

#### 29.500.2.1 Go::SurfaceOnVolume::SurfaceOnVolume ( )

Empty constructor.

#### 29.500.2.2 Go::SurfaceOnVolume::SurfaceOnVolume ( shared\_ptr< ParamVolume > vol, shared\_ptr< ParamSurface > parsurf, shared\_ptr< ParamSurface > spacesurf, bool preferparameter )

Constructor given associated volume, the surface in the parameter plane of this volume and the corresponding surface in geometry space. One surface representation may be a dummy. In that case the parameter indicating which surface representation is the master, must be set accordingly

##### Parameters

<i>vol</i>	associated volume
<i>parsurf</i>	surface in parameter domain of the volume
<i>spacesurf</i>	surface in geometry space
<i>preferparameter</i>	true if the parameter surface is the master

#### 29.500.2.3 Go::SurfaceOnVolume::SurfaceOnVolume ( shared\_ptr< ParamVolume > vol, shared\_ptr< ParamSurface > spacesurf, int constdir, double constpar, int boundary, bool swapped, int orientation = 0 )

Constructor given the volume, the surface in geometry space and constant parameter information related to the surface

##### Parameters

<i>vol</i>	associated volume
<i>spacesurf</i>	surface in geometry space
<i>constdir</i>	0 = not set, 1 = u-parameter constant, 2 = v-parameter constant, 3 = w-parameter constant
<i>constpar</i>	value of constant parameter
<i>boundary</i>	index: -1=no, 0=umin, 1=umax, 2=vmin, 3=vmax, 4=wmin, 5=wmax
<i>swapped</i>	orientation of surface related to underlying volume

29.500.2.4 `Go::SurfaceOnVolume::SurfaceOnVolume ( shared_ptr< ParamVolume > vol, int constdir, double constpar, int boundary )`

Constructor given volume and constant parameter information. Must only be used if the constant parameter information is set

29.500.2.5 `Go::SurfaceOnVolume::SurfaceOnVolume ( shared_ptr< ParamVolume > vol, shared_ptr< ParamSurface > spacesurf, shared_ptr< ParamSurface > parsurf, bool prefer_parameter, int constdir, double constpar, int boundary, bool swapped )`

Constructor to be used if all information is known

#### Parameters

<i>vol</i>	associated volume
<i>parsurf</i>	surface in parameter domain of the volume
<i>spacesurf</i>	surface in geometry space
<i>constdir</i>	0 = not set, 1 = u-parameter constant, 2 = v-parameter constant, 3 = w-parameter constant
<i>constpar</i>	value of constant parameter
<i>boundary</i>	index: -1=no, 0=umin, 1=umax, 2=vmin, 3=vmax, 4=wmin, 5=wmax
<i>swapped</i>	orientation of surface related to underlying volume

29.500.2.6 `Go::SurfaceOnVolume::SurfaceOnVolume ( const SurfaceOnVolume & other )`

Assignment constructor.

29.500.2.7 `virtual Go::SurfaceOnVolume::~~SurfaceOnVolume ( ) [virtual]`

Destructor.

### 29.500.3 Member Function Documentation

29.500.3.1 `virtual std::vector<CurveLoop> Go::SurfaceOnVolume::allBoundaryLoops ( double degenerate_epsilon = DEFAULT_SPACE_EPSILON ) const [virtual]`

Returns the anticlockwise outer boundary loop of the surface, together with clockwise loops of any interior boundaries, such that the surface always is 'to the left of' the loops.

#### Parameters

<i>degenerate_epsilon</i>	edges whose length is smaller than this value are ignored.
---------------------------	------------------------------------------------------------

#### Returns

a vector containing CurveLoops. The first of these describe the outer boundary of the surface (clockwise), whereas the others describe boundaries of interior holes (clockwise).

Implements [Go::ParamSurface](#).

**29.500.3.2** `virtual double Go::SurfaceOnVolume::area ( double tol ) const [virtual]`

Compute the total area of this surface up to some tolerance

#### Parameters

<i>tol</i>	the relative tolerance when approximating the area, i.e. stop iteration when error becomes smaller than $tol/(surface\ area)$
------------	-------------------------------------------------------------------------------------------------------------------------------

#### Returns

the area calculated

Implements [Go::ParamSurface](#).

**29.500.3.3** `virtual SplineSurface* Go::SurfaceOnVolume::asSplineSurface ( ) [inline],[virtual]`

Return the spline surface represented by this surface, if any.

Reimplemented from [Go::ParamSurface](#).

Definition at line 150 of file SurfaceOnVolume.h.

**29.500.3.4** `virtual BoundingBox Go::SurfaceOnVolume::boundingBox ( ) const [virtual]`

Axis align box surrounding this object.

Implements [Go::GeomObject](#).

**29.500.3.5** `static ClassType Go::SurfaceOnVolume::classType ( ) [inline],[static]`

Return the class type identifier of type SurfaceOnVolumeBoundedSurface.

Definition at line 138 of file SurfaceOnVolume.h.

**29.500.3.6** `virtual SurfaceOnVolume* Go::SurfaceOnVolume::clone ( ) const [inline],[virtual]`

make a clone of this surface and return a pointer to it (user is responsible for clearing up memory afterwards).

#### Returns

pointer to cloned object

Implements [Go::ParamSurface](#).

Definition at line 144 of file SurfaceOnVolume.h.

```
29.500.3.7 virtual void Go::SurfaceOnVolume::closestBoundaryPoint (const Point & pt, double & clo_u, double & clo_v,
 Point & clo_pt, double & clo_dist, double epsilon, const RectDomain * rd = NULL, double * seed = 0)
 const [virtual]
```

Iterates to the closest point to pt on the boundary of the surface.

See also

[closestPoint\(\)](#)

Implements [Go::ParamSurface](#).

```
29.500.3.8 virtual Point Go::SurfaceOnVolume::closestInDomain (double u, double v) const [virtual]
```

Return the closest parameter pair in the domain of this surface, given an initial parameter pair

Implements [Go::ParamSurface](#).

```
29.500.3.9 virtual void Go::SurfaceOnVolume::closestPoint (const Point & pt, double & clo_u, double & clo_v, Point &
 clo_pt, double & clo_dist, double epsilon, const RectDomain * domain_of_interest = NULL, double *
 seed = 0) const [virtual]
```

Iterates to the closest point to pt on the surface.

Parameters

<i>pt</i>	the point to find the closest point to
<i>clo_u</i>	u parameter of the closest point
<i>clo_v</i>	v parameter of the closest point
<i>clo_pt</i>	the geometric position of the closest point
<i>clo_dist</i>	the distance between pt and clo_pt
<i>epsilon</i>	parameter tolerance (will in any case not be higher than sqrt(machine_precision) x magnitude of solution)
<i>domain_of_interest</i>	pointer to parameter domain in which to search for closest point. If a NULL pointer is used, the entire surface is searched.
<i>seed</i>	pointer to parameter values where iteration starts.

Reimplemented from [Go::ParamSurface](#).

```
29.500.3.10 virtual CompositeBox Go::SurfaceOnVolume::compositeBox () const [virtual]
```

Creates a composite box enclosing the surface. The composite box consists of an inner and an edge box. The inner box is supposed to be made from the interior of the surface, while the edge box is made from the boundary curves. The default implementation simply makes both boxes identical to the regular bounding box.

Returns

the [CompositeBox](#) of the surface, as specified above

Reimplemented from [Go::ParamSurface](#).



29.500.3.11 `virtual std::vector<shared_ptr<ParamCurve> > Go::SurfaceOnVolume::constParamCurves ( double parameter, bool pdir_is_u ) const [virtual]`

Get the curve(s) obtained by intersecting the surface with one of its constant parameter curves. For surfaces without holes, this will be the parameter curve itself; for surfaces with interior holes this may be a collection of several, disjoint curves.

#### Parameters

<i>parameter</i>	parameter value for the constant parameter (either u or v)
<i>pdir_is_u</i>	specify whether the <i>moving</i> parameter (as opposed to the <i>constant</i> parameter) is the first ('true') or the second ('false') one.

#### Returns

a vector containing shared pointers to the obtained, newly constructed constant-parameter curves.

Implements [Go::ParamSurface](#).

29.500.3.12 `virtual RectDomain Go::SurfaceOnVolume::containingDomain ( ) const [virtual]`

Get a rectangular parameter domain that is guaranteed to contain the surface's [parameterDomain\(\)](#). It may be the same. There is no guarantee that this is the smallest domain containing the actual domain.

#### Returns

a [RectDomain](#) that is guaranteed to include the surface's total parameter domain.

Implements [Go::ParamSurface](#).

29.500.3.13 `virtual int Go::SurfaceOnVolume::dimension ( ) const [virtual]`

Dimension of geometry space.

Implements [Go::GeomObject](#).

29.500.3.14 `virtual int Go::SurfaceOnVolume::ElementBoundaryStatus ( int elem_ix, double eps ) [inline], [virtual]`

Check if a polynomial element (for spline surfaces) intersects the (trimming) boundaries of this [ftSurface](#), is inside or outside

#### Parameters

<i>elem_ix</i>	Element index counted according to distinct knot values. Sequence of coordinates: x runs fastest, then y
<i>eps</i>	Intersection tolerance

**Returns**

-1: Not a spline surface or element index out of range 0: Outside trimmed volume 1: On boundary (intersection with boundary found) 2: Internal to trimmed surfaces Note that a touch with the boundaries of the underlying surface is not considered a boundary intersection while touching a trimming curve is seen as an intersection

Reimplemented from [Go::ParamSurface](#).

Definition at line 547 of file SurfaceOnVolume.h.

```
29.500.3.15 virtual int Go::SurfaceOnVolume::ElementOnBoundary (int elem_ix, double eps) [inline],
 [virtual]
```

Check if a polynomial element (for spline surfaces) intersects the (trimming) boundaries of this surface

**Parameters**

<i>elem_ix</i>	Element index counted according to distinct knot values. Sequence of coordinates: x runs fastest, then y
<i>eps</i>	Intersection tolerance

**Returns**

-1: Not a spline surface or element index out of range 0: Not on boundary or touching a boundary curve 1: On boundary (intersection with boundary found) Note that a touch with the boundaries of the underlying surfaces is not considered a boundary intersection while touching a trimming curve is seen as an intersection

Reimplemented from [Go::ParamSurface](#).

Definition at line 527 of file SurfaceOnVolume.h.

```
29.500.3.16 virtual void Go::SurfaceOnVolume::getBoundaryInfo (Point & pt1, Point & pt2, double epsilon, SplineCurve
 *& cv, SplineCurve *& crosscv, double knot_tol = 1e-05) const [virtual]
```

Get the boundary curve segment between two points on the boundary, as well as the cross-tangent curve. If the given points are not positioned on the same boundary (within a certain tolerance), no curves will be created.

**Parameters**

<i>pt1</i>	the first point on the boundary, given by the user
<i>pt2</i>	the second point on the boundary, given by the user
<i>epsilon</i>	the tolerance used when determining whether the given points are lying on a boundary, and if they do, whether they both lie on the <i>same</i> boundary.
<i>cv</i>	upon return, this will point to a newly created curve representing the boundary curve between 'pt1' and 'pt2'. The user assumes ownership of this object and is responsible for its deletion. No curve is created if the given points are not found to lie on the same boundary.
<i>crosscv</i>	upon return, this will point to a newly created curve representing the cross-boundary curve between 'pt1' and 'pt2'. The user assumes ownership of this object and is responsible for its deletion. The direction is outwards from the surface. No curve is created if the given points are not found to lie on the same boundary.
<i>knot_tol</i>	tolerance used when working with the knot-vector, to specify how close a parameter value must be to a knot in order to be considered 'on' the knot.

Implements [Go::ParamSurface](#).

29.500.3.17 `int Go::SurfaceOnVolume::getConstDir ( ) const [inline]`

Fetch the constant parameter direction of the volume, if any, associated.

Parameters

<i>return</i>	value: 0 = not set, 1 = u-parameter constant, 2 = v-parameter constant, 3 = w-parameter constant
---------------	--------------------------------------------------------------------------------------------------

Definition at line 505 of file SurfaceOnVolume.h.

29.500.3.18 `double Go::SurfaceOnVolume::getConstVal ( ) const [inline]`

Fetch the constant parameter value of the volume associated with this surface.

Definition at line 495 of file SurfaceOnVolume.h.

29.500.3.19 `virtual void Go::SurfaceOnVolume::getCornerPoints ( std::vector< std::pair< Point, Point > > & corners ) const [virtual]`

Return surface corners, i.e joints between trimming curves, geometric and parametric points in that sequence

Implements [Go::ParamSurface](#).

29.500.3.20 `virtual void Go::SurfaceOnVolume::getDegenerateCorners ( std::vector< Point > & deg_corners, double tol ) const [virtual]`

Check for paralell and anti paralell partial derivatives in surface corners.

Implements [Go::ParamSurface](#).

29.500.3.21 `shared_ptr<const ParamVolume> Go::SurfaceOnVolume::getVolume ( ) const [inline]`

Get volume.

Definition at line 441 of file SurfaceOnVolume.h.

29.500.3.22 `shared_ptr<ParamVolume> Go::SurfaceOnVolume::getVolume ( ) [inline]`

Get volume.

Definition at line 447 of file SurfaceOnVolume.h.

29.500.3.23 `virtual bool Go::SurfaceOnVolume::inDomain ( double u, double v, double eps = 1.0e-4 ) const`  
`[virtual]`

Check if a parameter pair lies inside the domain of this surface.

Implements [Go::ParamSurface](#).

29.500.3.24 `virtual int Go::SurfaceOnVolume::inDomain2 ( double u, double v, double eps = 1.0e-4 ) const`  
`[virtual]`

Check if a parameter pair lies inside the domain of this surface return value = 0: outside = 1: internal = 2: at the boundary

Implements [Go::ParamSurface](#).

29.500.3.25 `virtual ClassType Go::SurfaceOnVolume::instanceType ( ) const` `[virtual]`

Return the class type identifier of type [SurfaceOnVolume](#).

Implements [Go::GeomObject](#).

29.500.3.26 `virtual bool Go::SurfaceOnVolume::isDegenerate ( bool & b, bool & r, bool & t, bool & l, double tolerance )`  
`const [virtual]`

The order of the edge indicators (bottom, right, top, left) matches the `edge_number` of `edgeCurve()`. Query whether any of the four boundary curves are degenerate (zero length) within a certain tolerance. In the below, we refer to 'u' as the first parameter and 'v' as the second.

#### Parameters

<i>b</i>	'true' upon return of function if the boundary ( $v = v_{\min}$ ) is degenerate
<i>r</i>	'true' upon return of function if the boundary ( $v = v_{\max}$ ) is degenerate
<i>t</i>	'true' upon return of function if the boundary ( $u = u_{\min}$ ) is degenerate
<i>l</i>	'true' upon return of function if the boundary ( $u = u_{\max}$ ) is degenerate
<i>tolerance</i>	boundaries are considered degenerate if their length is shorter than this value, given by the user

#### Returns

'true' if at least one boundary curve was found to be degenerate, 'false' otherwise.

Reimplemented from [Go::ParamSurface](#).

29.500.3.27 `virtual bool Go::SurfaceOnVolume::isIsoTrimmed ( double tol ) const` `[virtual]`

Check if the current surface is trimmed along constant parameter curves.

Reimplemented from [Go::ParamSurface](#).

29.500.3.28 `virtual bool Go::SurfaceOnVolume::isLinear ( Point & dir1, Point & dir2, double tol ) [virtual]`

Check if the surface is linear in one or both parameter directions.

Reimplemented from [Go::ParamSurface](#).

29.500.3.29 `virtual double Go::SurfaceOnVolume::nextSegmentVal ( int dir, double par, bool forward, double tol ) const [virtual]`

Determine the parameter value of the start of the 'next segment' from a parameter value, along a given parameter direction. A 'segment' is here defined as a parameter interval in which there will be no discontinuities in derivatives or other artifacts. For spline objects, a segment will typically be the interval between two consecutive, non-coincident knots.

#### Parameters

<i>dir</i>	the parameter direction in which we search for the next segment (0 or 1)
<i>par</i>	the parameter value starting from which we search for the start value of the next segment
<i>forward</i>	define whether we shall move forward ('true') or backwards when searching along this parameter
<i>tol</i>	tolerance used for determining whether the 'par' is already located <i>on</i> the next segment value

#### Returns

the value of the start value of the next segment (or the end of the previous segment, if we are moving backwards...)

Implements [Go::ParamSurface](#).

29.500.3.30 `virtual void Go::SurfaceOnVolume::normal ( Point & n, double upar, double vpar ) const [virtual]`

Evaluates the surface normal for a given parameter pair

#### Parameters

<i>n</i>	the computed normal will be written to this variable
<i>upar</i>	the first parameter
<i>vpar</i>	the second parameter

Implements [Go::ParamSurface](#).

29.500.3.31 `virtual DirectionCone Go::SurfaceOnVolume::normalCone ( ) const [virtual]`

Creates a [DirectionCone](#) covering all normals to this surface.

#### Returns

a [DirectionCone](#) (not necessarily the smallest) containing all normals to this surface.

Implements [Go::ParamSurface](#).

29.500.3.32 `virtual bool Go::SurfaceOnVolume::onBoundary ( double u, double v, double eps = 1.0e-4 ) const` [virtual]

Check if a parameter pair lies at the boundary of this surface.

Implements [Go::ParamSurface](#).

29.500.3.33 `virtual CurveLoop Go::SurfaceOnVolume::outerBoundaryLoop ( double degenerate_epsilon = DEFAULT_SPACE_EPSILON ) const` [virtual]

Returns the anticlockwise, outer boundary loop of the surface.

Parameters

<i>degenerate_epsilon</i>	edges whose length is smaller than this value are ignored.
---------------------------	------------------------------------------------------------

Returns

a [CurveLoop](#) describing the anticlockwise, outer boundary loop of the surface.

Implements [Go::ParamSurface](#).

29.500.3.34 `virtual const Domain& Go::SurfaceOnVolume::parameterDomain ( ) const` [virtual]

Return the parameter domain of the surface. This may be a simple rectangular domain ([RectDomain](#)) or any other subclass of [Domain](#) (such as [GoCurveBoundedDomain](#), found in the `sisl_dependent` module).

Returns

a [Domain](#) object describing the parametric domain of the surface

Implements [Go::ParamSurface](#).

29.500.3.35 `shared_ptr<const ParamSurface> Go::SurfaceOnVolume::parameterSurface ( ) const` [inline]

Get parameter surface.

Definition at line 459 of file `SurfaceOnVolume.h`.

29.500.3.36 `shared_ptr<ParamSurface> Go::SurfaceOnVolume::parameterSurface ( )` [inline]

Get parameter surface.

Definition at line 465 of file `SurfaceOnVolume.h`.

29.500.3.37 `bool Go::SurfaceOnVolume::parPref ( ) const` [inline]

Query whether the parameter surface or the space surface is preferred for computation in this object.

Definition at line 491 of file `SurfaceOnVolume.h`.

29.500.3.38 `virtual void Go::SurfaceOnVolume::point ( Point & pt, double upar, double vpar ) const` [virtual]

Evaluates the surface's position for a given parameter pair.

## Parameters

<i>pt</i>	the result of the evaluation is written here
<i>upar</i>	the first parameter
<i>vpar</i>	the second parameter

Implements [Go::ParamSurface](#).

**29.500.3.39** `virtual void Go::SurfaceOnVolume::point ( std::vector< Point > & pts, double upar, double vpar, int derivs, bool u_from_right = true, bool v_from_right = true, double resolution = 1.0e-12 ) const [virtual]`

Evaluates the surface's position and a certain number of derivatives for a given parameter pair.

## Parameters

<i>pts</i>	the vector containing the evaluated values. Its size must be set by the user prior to calling this function, and should be equal to $(derivs+1) * (derivs+2) / 2$ . Upon completion of the function, its first entry is the surface's position at the given parameter pair. Then, if 'derivs' > 0, the two next entries will be the surface tangents along the first and second parameter direction. The next three entries are the second- and cross derivatives, in the order (du2, dudv, dv2), and similar for even higher derivatives.
<i>upar</i>	the first parameter
<i>vpar</i>	the second parameter
<i>derivs</i>	number of requested derivatives
<i>u_from_right</i>	specify whether derivatives along the first parameter are to be calculated from the right ('true', default) or from the left ('false')
<i>v_from_right</i>	specify whether derivatives along the second parameter are to be calculated from the right ('true', default) or from the left ('false')
<i>resolution</i>	tolerance used when determining whether parameters are located at special values of the parameter domain (in particular; knot values in case of spline objects).

Implements [Go::ParamSurface](#).

**29.500.3.40** `virtual void Go::SurfaceOnVolume::read ( std::istream & is ) [virtual]`

Read surface on volume information from file Not implemented

Implements [Go::Streamable](#).

**29.500.3.41** `virtual void Go::SurfaceOnVolume::reverseParameterDirection ( bool direction_is_u ) [virtual]`

Reverses the direction of the basis in input direction.

## Parameters

<i>direction_is_u</i>	if 'true', the first parameter direction will be reversed, otherwise, the second parameter direction will be reversed
-----------------------	-----------------------------------------------------------------------------------------------------------------------

Implements [Go::ParamSurface](#).

29.500.3.42 `void Go::SurfaceOnVolume::setSpaceSurface ( shared_ptr< ParamSurface > spacesurf ) [inline]`

For internal use. Careful!

Definition at line 484 of file SurfaceOnVolume.h.

29.500.3.43 `void Go::SurfaceOnVolume::setVolume ( shared_ptr< ParamVolume > volume ) [inline]`

Set/replace volume. NB! Use care.

Definition at line 453 of file SurfaceOnVolume.h.

29.500.3.44 `shared_ptr<const ParamSurface> Go::SurfaceOnVolume::spaceSurface ( ) const [inline]`

Get space surface.

Definition at line 471 of file SurfaceOnVolume.h.

29.500.3.45 `shared_ptr<ParamSurface> Go::SurfaceOnVolume::spaceSurface ( ) [inline]`

Get space surface.

Definition at line 477 of file SurfaceOnVolume.h.

29.500.3.46 `virtual std::vector<shared_ptr<ParamSurface> > Go::SurfaceOnVolume::subSurfaces ( double from_upar, double from_vpar, double to_upar, double to_vpar, double fuzzy = DEFAULT_PARAMETER_EPSILON ) const [virtual]`

Get the surface(s) obtained by cropping the parameter domain of this surface between given values for the first and second parameter. In general, for surfaces with no interior holes, the result will be *one* surface; however, for surfaces with interior holes, the result might be *several disjoint* surfaces.

#### Parameters

<i>from_upar</i>	lower value for the first parameter in the subdomain
<i>from_vpar</i>	lower value for the second parameter in the subdomain
<i>to_upar</i>	upper value for the first parameter in the subdomain
<i>to_vpar</i>	upper value for the second parameter in the subdomain
<i>fuzzy</i>	tolerance used when determining intersection with interior boundaries

#### Returns

a vector contained shared pointers to the obtained, newly constructed sub-surfaces.

Implements [Go::ParamSurface](#).



29.500.3.47 `virtual void Go::SurfaceOnVolume::swapParameterDirection ( ) [virtual]`

Swaps the two parameter directions.

Implements [Go::ParamSurface](#).

29.500.3.48 `virtual DirectionCone Go::SurfaceOnVolume::tangentCone ( bool pardir_is_u ) const [virtual]`

Creates a [DirectionCone](#) covering all tangents to this surface along a given parameter direction.

Parameters

<code><i>pardir_is_u</i></code>	if 'true', then the <a href="#">DirectionCone</a> will be defined on basis of the surface's tangents along the first parameter direction. Otherwise the second parameter direction will be used.
---------------------------------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Returns

a [DirectionCone](#) (not necessarily the smallest) containing all tangents to this surface along the specified parameter direction.

Implements [Go::ParamSurface](#).

29.500.3.49 `virtual void Go::SurfaceOnVolume::turnOrientation ( ) [virtual]`

Turns the direction of the normal of the surface.

Implements [Go::ParamSurface](#).

29.500.3.50 `void Go::SurfaceOnVolume::unsetParamSurf ( )`

Unset parameter surface information.

29.500.3.51 `Point Go::SurfaceOnVolume::volumeParameter ( double u_par, double v_par ) const`

Volume parameter corresponding to surface parameter.

29.500.3.52 `int Go::SurfaceOnVolume::whichBoundary ( double tol, int & orientation, bool & swap ) const`

Info on relation to corresponding volume Return value: -1=none, 0=umin, 1=umax, 2=vmin, 3=vmax, 4=wmin, 5=wmax orientation = -1 and return value >= 0: the orientation of the surface compared to the volume boundary is not known or the two surfaces are coincident but not identical

```
29.500.3.53 virtual void Go::SurfaceOnVolume::write (std::ostream & os) const [virtual]
```

Write surface on volume information to file for visualization purposes If the geometry space surface exists, only that surface is written

Implements [Go::Streamable](#).

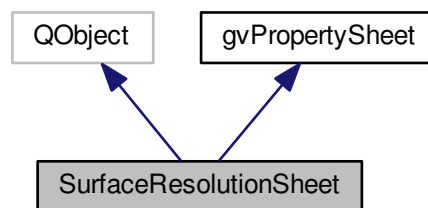
The documentation for this class was generated from the following file:

- [trivariate/include/GoTools/trivariate/SurfaceOnVolume.h](#)

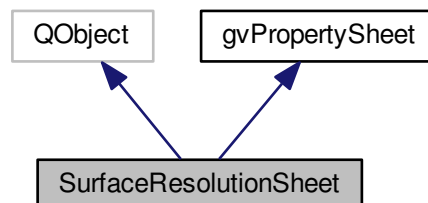
## 29.501 SurfaceResolutionSheet Class Reference

```
#include <SurfaceResolutionSheet.h>
```

Inheritance diagram for SurfaceResolutionSheet:



Collaboration diagram for SurfaceResolutionSheet:



### Public Slots

- void [ok](#) ()

## Signals

- void [return\\_value](#) (int, int)

## Public Member Functions

- [SurfaceResolutionSheet](#) (int ures=20, int vres=20)
- virtual void [createSheet](#) (QWidget \*parent, [gvObserver](#) \*obs)

### 29.501.1 Detailed Description

Documentation ... etc

Definition at line 58 of file SurfaceResolutionSheet.h.

### 29.501.2 Constructor & Destructor Documentation

29.501.2.1 [SurfaceResolutionSheet::SurfaceResolutionSheet](#) ( int *ures* = 20, int *vres* = 20 ) `[inline]`

Definition at line 64 of file SurfaceResolutionSheet.h.

### 29.501.3 Member Function Documentation

29.501.3.1 virtual void [SurfaceResolutionSheet::createSheet](#) ( [QWidget](#) \* *parent*, [gvObserver](#) \* *obs* ) `[virtual]`

Implements [gvPropertySheet](#).

29.501.3.2 void [SurfaceResolutionSheet::ok](#) ( ) `[slot]`

29.501.3.3 void [SurfaceResolutionSheet::return\\_value](#) ( int , int ) `[signal]`

The documentation for this class was generated from the following file:

- [viewlib/include/GoTools/viewlib/SurfaceResolutionSheet.h](#)

## 29.502 Go::SweepSurfaceCreator Class Reference

Functionality to create a spline surface by linear or rotational sweep.

```
#include <SweepSurfaceCreator.h>
```

## Public Member Functions

- virtual `~SweepSurfaceCreator ()`

## Static Public Member Functions

- static `SplineSurface * linearSweptSurface (const SplineCurve &curv1, const SplineCurve &curv2, const Point &pt)`
- static `SplineSurface * rotationalSweptSurface (const SplineCurve &curve, double angle, const Point &pt, const Point &axis)`

### 29.502.1 Detailed Description

Functionality to create a spline surface by linear or rotational sweep.

Definition at line 51 of file SweepSurfaceCreator.h.

### 29.502.2 Constructor & Destructor Documentation

29.502.2.1 `virtual Go::SweepSurfaceCreator::~~SweepSurfaceCreator ( ) [inline],[virtual]`

Definition at line 57 of file SweepSurfaceCreator.h.

### 29.502.3 Member Function Documentation

29.502.3.1 `static SplineSurface* Go::SweepSurfaceCreator::linearSweptSurface ( const SplineCurve & curv1, const SplineCurve & curv2, const Point & pt ) [static]`

Create a linearly swept surface as a tensor product of two curves. If  $P$  is the point to be swept, and the curves have B-spline coefficients  $C_i$  and  $D_j$ , then the surface has B-spline coefficients  $C_i + D_j - P$ . If the point lies on the first curve, the first curve will be swept along the second curve. If the point lies on the second curve, the second curve will be swept along the first curve.

#### Parameters

<i>curv1</i>	the first curve
<i>curv2</i>	the second curve
<i>pt</i>	the point to be swept

#### Returns

a pointer to the swept surface

29.502.3.2 `static SplineSurface* Go::SweepSurfaceCreator::rotationalSweptSurface ( const SplineCurve & curve, double angle, const Point & pt, const Point & axis ) [static]`

Create a surface as the rotation of a curve around an axis

## Parameters

<i>curve</i>	the curve
<i>angle</i>	the rotation angle (in radians). If less than $2\pi$ , it is truncated down to $2\pi$ . If more than $-2\pi$ , it is truncated up to $-2\pi$ .
<i>pt</i>	a point on the rotation axis
<i>axis</i>	a vector describing the direction of the rotation axis

## Returns

a pointer to the rotated surface

The documentation for this class was generated from the following file:

- [gotools-core/include/GoTools/geometry/SweepSurfaceCreator.h](#)

## 29.503 Go::SweepVolumeCreator Class Reference

Class with static methods for volume creation by sweep methods.

```
#include <SweepVolumeCreator.h>
```

### Public Member Functions

- virtual [~SweepVolumeCreator](#) ()

### Static Public Member Functions

- static [SplineVolume](#) \* [linearSweptVolume](#) (const [SplineSurface](#) &surface, const [SplineCurve](#) &curve, const [Point](#) &pt)
- static [SplineVolume](#) \* [rotationalSweptVolume](#) (const [SplineSurface](#) &surface, double angle, const [Point](#) &pt, const [Point](#) &axis)

#### 29.503.1 Detailed Description

Class with static methods for volume creation by sweep methods.

Definition at line 53 of file [SweepVolumeCreator.h](#).

#### 29.503.2 Constructor & Destructor Documentation

29.503.2.1 virtual [Go::SweepVolumeCreator::~~SweepVolumeCreator](#) ( ) [[inline](#)], [[virtual](#)]

Definition at line 59 of file [SweepVolumeCreator.h](#).

#### 29.503.3 Member Function Documentation

29.503.3.1 static [SplineVolume](#)\* [Go::SweepVolumeCreator::linearSweptVolume](#) ( const [SplineSurface](#) & surface, const [SplineCurve](#) & curve, const [Point](#) & pt ) [[static](#)]

Create a linearly swept volume as a tensor product of a surface and a curve. If  $P$  is the point to be swept, the surface has B-spline coefficients  $S_{i,j}$  and the curve has B-spline coefficients  $C_k$  then the volume has B-spline coefficients  $S_{i,j} + C_k - P$ . If the point lies on the curve, the curve will be swept along the surface. If the point lies on the surface, the surface will be swept along the curve.

## Parameters

<i>surface</i>	the surface
<i>curve</i>	the curve
<i>pt</i>	the point to be swept

## Returns

a pointer to the swept volume

29.503.3.2 `static SplineVolume* Go::SweepVolumeCreator::rotationalSweptVolume ( const SplineSurface & surface, double angle, const Point & pt, const Point & axis ) [static]`

Create a volume as the rotation of a surface around an axis

## Parameters

<i>surface</i>	the surface
<i>angle</i>	the rotation angle (in radians). If more than $2\pi$ , it is truncated down to $2\pi$ . If less than $-2\pi$ , it is truncated up to $-2\pi$ .
<i>pt</i>	a point on the rotation axis
<i>axis</i>	a vector describing the direction of the rotation axis

## Returns

a pointer to the rotated volume

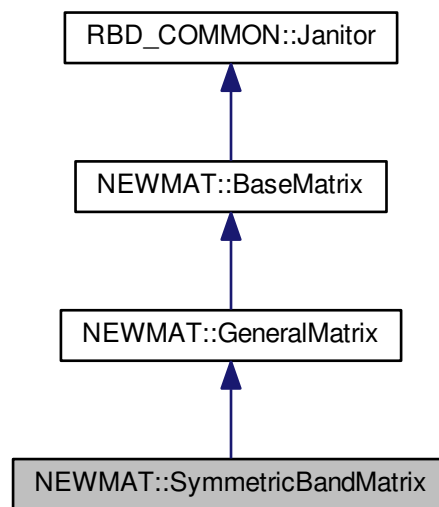
The documentation for this class was generated from the following file:

- `trivariate/include/GoTools/trivariate/SweepVolumeCreator.h`

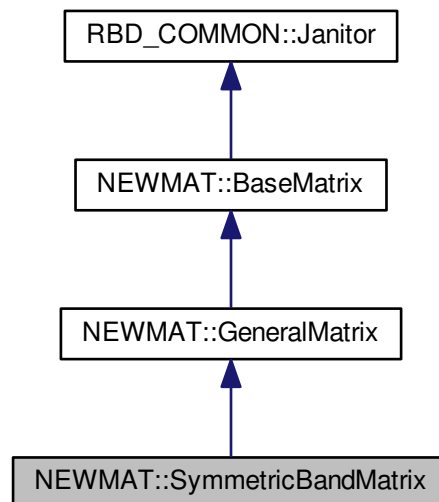
## 29.504 NEWMAT::SymmetricBandMatrix Class Reference

```
#include <newmat.h>
```

Inheritance diagram for NEWMAT::SymmetricBandMatrix:



Collaboration diagram for NEWMAT::SymmetricBandMatrix:



## Public Member Functions

- [SymmetricBandMatrix \(\)](#)

- [~SymmetricBandMatrix \(\)](#)
- [SymmetricBandMatrix \(int n, int lb\)](#)
- [SymmetricBandMatrix \(const BaseMatrix &\)](#)
- [void operator= \(const BaseMatrix &\)](#)
- [void operator= \(Real f\)](#)
- [void operator= \(const SymmetricBandMatrix &m\)](#)
- [Real & operator\(\) \(int, int\)](#)
- [Real & element \(int, int\)](#)
- [Real operator\(\) \(int, int\) const](#)
- [Real element \(int, int\) const](#)
- [MatrixType Type \(\) const](#)
- [SymmetricBandMatrix \(const SymmetricBandMatrix &gm\)](#)
- [GeneralMatrix \\* MakeSolver \(\)](#)
- [Real SumSquare \(\) const](#)
- [Real SumAbsoluteValue \(\) const](#)
- [Real Sum \(\) const](#)
- [Real MaximumAbsoluteValue \(\) const](#)
- [Real MinimumAbsoluteValue \(\) const](#)
- [Real Maximum \(\) const](#)
- [Real Minimum \(\) const](#)
- [Real Trace \(\) const](#)
- [LogAndSign LogDeterminant \(\) const](#)
- [void GetRow \(MatrixRowCol &\)](#)
- [void GetCol \(MatrixRowCol &\)](#)
- [void GetCol \(MatrixColX &\)](#)
- [void RestoreCol \(MatrixRowCol &\)](#)
- [void RestoreCol \(MatrixColX &\)](#)
- [GeneralMatrix \\* Transpose \(TransposedMatrix \\*, MatrixType\)](#)
- [void ReSize \(int, int\)](#)
- [void ReSize \(const GeneralMatrix &A\)](#)
- [bool SameStorageType \(const GeneralMatrix &A\) const](#)
- [void ReSizeForAdd \(const GeneralMatrix &A, const GeneralMatrix &B\)](#)
- [void ReSizeForSP \(const GeneralMatrix &A, const GeneralMatrix &B\)](#)
- [MatrixBandWidth BandWidth \(\) const](#)
- [void SetParameters \(const GeneralMatrix \\*\)](#)

## Public Attributes

- int [lower](#)

## Additional Inherited Members

### 29.504.1 Detailed Description

Definition at line 1017 of file newmat.h.

### 29.504.2 Constructor & Destructor Documentation

#### 29.504.2.1 NEWMAT::SymmetricBandMatrix::SymmetricBandMatrix ( ) `[inline]`

Definition at line 1024 of file newmat.h.



29.504.2.2 NEWMAT::SymmetricBandMatrix::~~SymmetricBandMatrix ( ) [inline]

Definition at line 1025 of file newmat.h.

29.504.2.3 NEWMAT::SymmetricBandMatrix::SymmetricBandMatrix ( int *n*, int *lb* ) [inline]

Definition at line 1026 of file newmat.h.

29.504.2.4 NEWMAT::SymmetricBandMatrix::SymmetricBandMatrix ( const BaseMatrix & )

29.504.2.5 NEWMAT::SymmetricBandMatrix::SymmetricBandMatrix ( const SymmetricBandMatrix & *gm* ) [inline]

Definition at line 1041 of file newmat.h.

### 29.504.3 Member Function Documentation

29.504.3.1 MatrixBandWidth SymmetricBandMatrix::BandWidth ( ) const [virtual]

Reimplemented from [NEWMAT::BaseMatrix](#).

Definition at line 529 of file bandmat.cpp.

29.504.3.2 Real& NEWMAT::SymmetricBandMatrix::element ( int, int )

29.504.3.3 Real NEWMAT::SymmetricBandMatrix::element ( int, int ) const

29.504.3.4 void NEWMAT::SymmetricBandMatrix::GetCol ( MatrixRowCol & ) [virtual]

Implements [NEWMAT::GeneralMatrix](#).

29.504.3.5 void NEWMAT::SymmetricBandMatrix::GetCol ( MatrixColX & ) [virtual]

Implements [NEWMAT::GeneralMatrix](#).

29.504.3.6 void SymmetricBandMatrix::GetRow ( MatrixRowCol & *mrc* ) [virtual]

Implements [NEWMAT::GeneralMatrix](#).

Definition at line 648 of file newmat3.cpp.

29.504.3.7 **LogAndSign** SymmetricBandMatrix::LogDeterminant ( ) const [virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 439 of file bandmat.cpp.

29.504.3.8 **GeneralMatrix \* SymmetricBandMatrix::MakeSolver** ( ) [virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 421 of file bandmat.cpp.

29.504.3.9 **Real** NEWMAT::SymmetricBandMatrix::Maximum ( ) const [inline],[virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 1050 of file newmat.h.

29.504.3.10 **Real** NEWMAT::SymmetricBandMatrix::MaximumAbsoluteValue ( ) const [inline],[virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 1046 of file newmat.h.

29.504.3.11 **Real** NEWMAT::SymmetricBandMatrix::Minimum ( ) const [inline],[virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 1051 of file newmat.h.

29.504.3.12 **Real** NEWMAT::SymmetricBandMatrix::MinimumAbsoluteValue ( ) const [inline],[virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 1048 of file newmat.h.

29.504.3.13 **Real&** NEWMAT::SymmetricBandMatrix::operator()( int , int )

29.504.3.14 **Real** NEWMAT::SymmetricBandMatrix::operator()( int , int ) const

29.504.3.15 **void** NEWMAT::SymmetricBandMatrix::operator=( const BaseMatrix & )

29.504.3.16 **void** NEWMAT::SymmetricBandMatrix::operator=( Real f ) [inline]

Definition at line 1029 of file newmat.h.

29.504.3.17 void NEWMAT::SymmetricBandMatrix::operator= ( const SymmetricBandMatrix & m ) [inline]

Definition at line 1030 of file newmat.h.

29.504.3.18 void NEWMAT::SymmetricBandMatrix::ReSize ( int , int )

29.504.3.19 void SymmetricBandMatrix::ReSize ( const GeneralMatrix & A ) [virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 457 of file bandmat.cpp.

29.504.3.20 void SymmetricBandMatrix::ReSizeForAdd ( const GeneralMatrix & A, const GeneralMatrix & B )  
[virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 482 of file bandmat.cpp.

29.504.3.21 void SymmetricBandMatrix::ReSizeForSP ( const GeneralMatrix & A, const GeneralMatrix & B )  
[virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 494 of file bandmat.cpp.

29.504.3.22 void NEWMAT::SymmetricBandMatrix::RestoreCol ( MatrixRowCol & ) [inline],[virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 1057 of file newmat.h.

29.504.3.23 void NEWMAT::SymmetricBandMatrix::RestoreCol ( MatrixColX & ) [virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

29.504.3.24 bool SymmetricBandMatrix::SameStorageType ( const GeneralMatrix & A ) const [virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 475 of file bandmat.cpp.

29.504.3.25 `void SymmetricBandMatrix::SetParameters ( const GeneralMatrix * gmx )` [virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 445 of file `bandmat.cpp`.

29.504.3.26 `Real SymmetricBandMatrix::Sum ( ) const` [virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 555 of file `bandmat.cpp`.

29.504.3.27 `Real SymmetricBandMatrix::SumAbsoluteValue ( ) const` [virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 545 of file `bandmat.cpp`.

29.504.3.28 `Real SymmetricBandMatrix::SumSquare ( ) const` [virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 535 of file `bandmat.cpp`.

29.504.3.29 `Real SymmetricBandMatrix::Trace ( ) const` [virtual]

Reimplemented from [NEWMAT::BaseMatrix](#).

Definition at line 590 of file `newmat8.cpp`.

29.504.3.30 `GeneralMatrix * SymmetricBandMatrix::Transpose ( TransposedMatrix *, MatrixType mt )`  
[virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 436 of file `bandmat.cpp`.

29.504.3.31 `MatrixType SymmetricBandMatrix::Type ( ) const` [virtual]

Implements [NEWMAT::GeneralMatrix](#).

Definition at line 396 of file `newmat4.cpp`.

## 29.504.4 Member Data Documentation

### 29.504.4.1 int NEWMAT::SymmetricBandMatrix::lower

Definition at line 1023 of file newmat.h.

The documentation for this class was generated from the following files:

- [newmat/include/newmat.h](#)
- [newmat/src/bandmat.cpp](#)
- [newmat/src/newmat3.cpp](#)
- [newmat/src/newmat4.cpp](#)
- [newmat/src/newmat8.cpp](#)

## 29.505 NEWMAT::SymmetricEigenAnalysis Class Reference

```
#include <newmatap.h>
```

### Public Member Functions

- [SymmetricEigenAnalysis](#) (const [SymmetricMatrix](#) &)

### 29.505.1 Detailed Description

Definition at line 63 of file newmatap.h.

### 29.505.2 Constructor & Destructor Documentation

#### 29.505.2.1 NEWMAT::SymmetricEigenAnalysis::SymmetricEigenAnalysis ( const [SymmetricMatrix](#) & )

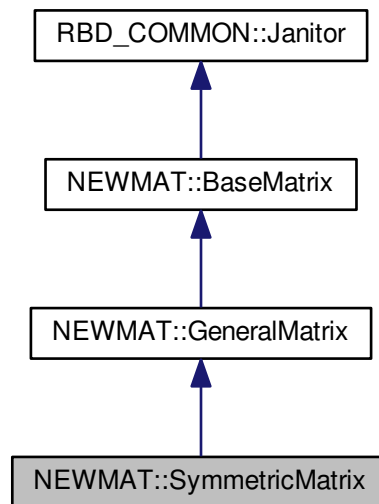
The documentation for this class was generated from the following file:

- [newmat/include/newmatap.h](#)

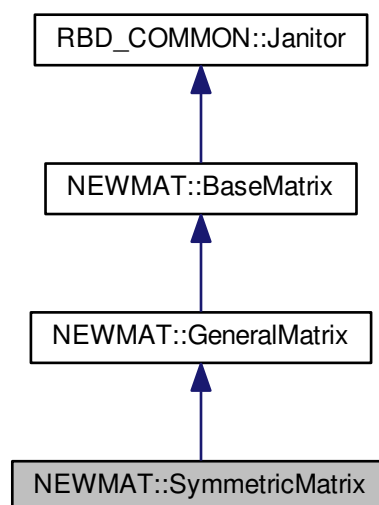
## 29.506 NEWMAT::SymmetricMatrix Class Reference

```
#include <newmat.h>
```

Inheritance diagram for NEWMAT::SymmetricMatrix:



Collaboration diagram for NEWMAT::SymmetricMatrix:



## Public Member Functions

- [SymmetricMatrix](#) ()
- [~SymmetricMatrix](#) ()
- [SymmetricMatrix](#) (ArrayLengthSpecifier)
- [SymmetricMatrix](#) (const BaseMatrix &)
- void [operator=](#) (const BaseMatrix &)
- void [operator=](#) (Real f)
- void [operator=](#) (const SymmetricMatrix &m)
- [Real & operator\(\)](#) (int, int)
- [Real & element](#) (int, int)
- [Real operator\(\)](#) (int, int) const
- [Real element](#) (int, int) const
- [MatrixType Type](#) () const
- [SymmetricMatrix](#) (const SymmetricMatrix &gm)
- [Real SumSquare](#) () const
- [Real SumAbsoluteValue](#) () const
- [Real Sum](#) () const
- [Real Trace](#) () const
- void [GetRow](#) (MatrixRowCol &)
- void [GetCol](#) (MatrixRowCol &)
- void [GetCol](#) (MatrixColX &)
- void [RestoreCol](#) (MatrixRowCol &)
- void [RestoreCol](#) (MatrixColX &)
- [GeneralMatrix \\* Transpose](#) (TransposedMatrix \*, MatrixType)
- void [ReSize](#) (int)
- void [ReSize](#) (const GeneralMatrix &A)

## Additional Inherited Members

### 29.506.1 Detailed Description

Definition at line 634 of file newmat.h.

### 29.506.2 Constructor & Destructor Documentation

**29.506.2.1** [NEWMAT::SymmetricMatrix::SymmetricMatrix](#) ( ) [inline]

Definition at line 638 of file newmat.h.

**29.506.2.2** [NEWMAT::SymmetricMatrix::~~SymmetricMatrix](#) ( ) [inline]

Definition at line 639 of file newmat.h.

**29.506.2.3** [NEWMAT::SymmetricMatrix::SymmetricMatrix](#) ( **ArrayLengthSpecifier** )

**29.506.2.4** [NEWMAT::SymmetricMatrix::SymmetricMatrix](#) ( **const BaseMatrix &** )

**29.506.2.5** [NEWMAT::SymmetricMatrix::SymmetricMatrix](#) ( **const SymmetricMatrix & gm** ) [inline]

Definition at line 654 of file newmat.h.

### 29.506.3 Member Function Documentation

29.506.3.1 `Real& NEWMAT::SymmetricMatrix::element ( int , int )`

29.506.3.2 `Real NEWMAT::SymmetricMatrix::element ( int , int ) const`

29.506.3.3 `void NEWMAT::SymmetricMatrix::GetCol ( MatrixRowCol & ) [virtual]`

Implements [NEWMAT::GeneralMatrix](#).

29.506.3.4 `void NEWMAT::SymmetricMatrix::GetCol ( MatrixColIX & ) [virtual]`

Implements [NEWMAT::GeneralMatrix](#).

29.506.3.5 `void SymmetricMatrix::GetRow ( MatrixRowCol & mrc ) [virtual]`

Implements [NEWMAT::GeneralMatrix](#).

Definition at line 397 of file newmat3.cpp.

29.506.3.6 `Real& NEWMAT::SymmetricMatrix::operator() ( int , int )`

29.506.3.7 `Real NEWMAT::SymmetricMatrix::operator() ( int , int ) const`

29.506.3.8 `void NEWMAT::SymmetricMatrix::operator= ( const BaseMatrix & )`

29.506.3.9 `void NEWMAT::SymmetricMatrix::operator= ( Real f ) [inline]`

Definition at line 643 of file newmat.h.

29.506.3.10 `void NEWMAT::SymmetricMatrix::operator= ( const SymmetricMatrix & m ) [inline]`

Definition at line 644 of file newmat.h.

29.506.3.11 `void NEWMAT::SymmetricMatrix::ReSize ( int )`

29.506.3.12 `void SymmetricMatrix::ReSize ( const GeneralMatrix & A ) [virtual]`

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 279 of file newmat4.cpp.



29.506.3.13 void NEWMAT::SymmetricMatrix::RestoreCol ( MatrixRowCol & ) [inline],[virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 662 of file newmat.h.

29.506.3.14 void NEWMAT::SymmetricMatrix::RestoreCol ( MatrixColX & ) [virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

29.506.3.15 Real SymmetricMatrix::Sum ( ) const [virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 411 of file newmat8.cpp.

29.506.3.16 Real SymmetricMatrix::SumAbsoluteValue ( ) const [virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 395 of file newmat8.cpp.

29.506.3.17 Real SymmetricMatrix::SumSquare ( ) const [virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 382 of file newmat8.cpp.

29.506.3.18 Real SymmetricMatrix::Trace ( ) const [virtual]

Reimplemented from [NEWMAT::BaseMatrix](#).

Definition at line 554 of file newmat8.cpp.

29.506.3.19 GeneralMatrix \* SymmetricMatrix::Transpose ( TransposedMatrix \*, MatrixType mt ) [virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 53 of file newmat5.cpp.

29.506.3.20 **MatrixType** SymmetricMatrix::Type ( ) const [virtual]

Implements [NEWMAT::GeneralMatrix](#).

Definition at line 386 of file newmat4.cpp.

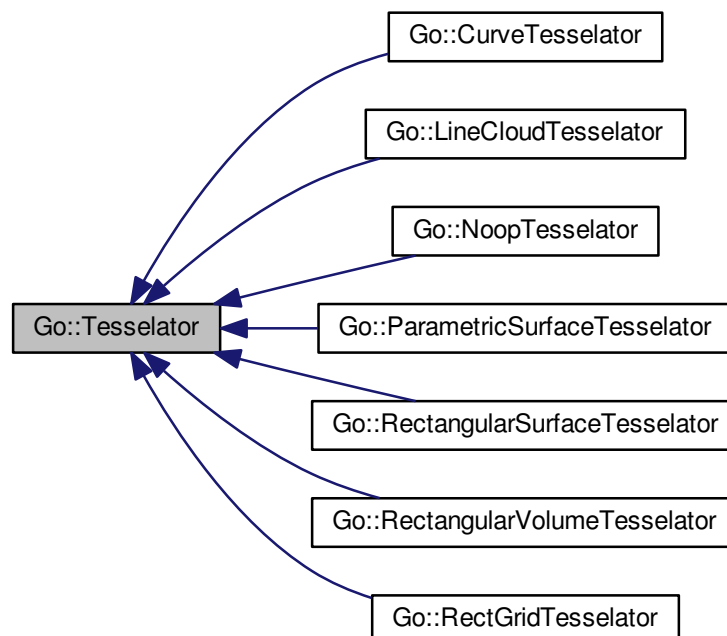
The documentation for this class was generated from the following files:

- [newmat/include/newmat.h](#)
- [newmat/src/newmat3.cpp](#)
- [newmat/src/newmat4.cpp](#)
- [newmat/src/newmat5.cpp](#)
- [newmat/src/newmat8.cpp](#)

## 29.507 Go::Tesselator Class Reference

```
#include <Tesselator.h>
```

Inheritance diagram for Go::Tesselator:



### Public Member Functions

- virtual [~Tesselator](#) ()  
*Destructor.*
- virtual void [tesselate](#) ()=0  
*Perform tessellation.*

### 29.507.1 Detailed Description

[Tesselator](#): super class for tesselators.

Definition at line 53 of file Tesselator.h.

### 29.507.2 Constructor & Destructor Documentation

#### 29.507.2.1 virtual Go::Tesselator::~Tesselator ( ) [virtual]

Destructor.

### 29.507.3 Member Function Documentation

#### 29.507.3.1 virtual void Go::Tesselator::tessellate ( ) [pure virtual]

Perform tessellation.

Implemented in [Go::RectangularVolumeTesselator](#), [Go::RectangularSurfaceTesselator](#), [Go::ParametricSurfaceTesselator](#), [Go::CurveTesselator](#), [Go::RectGridTesselator](#), [Go::LineCloudTesselator](#), and [Go::NoopTesselator](#).

The documentation for this class was generated from the following file:

- [gotools-core/include/GoTools/tesselator/Tesselator.h](#)

## 29.508 TestClass Class Reference

### Public Member Functions

- [TestClass](#) ( )
- [ReturnMatrix Sum](#) ( )

### 29.508.1 Detailed Description

Definition at line 19 of file tmtb.cpp.

### 29.508.2 Constructor & Destructor Documentation

#### 29.508.2.1 TestClass::TestClass ( )

Definition at line 28 of file tmtb.cpp.

### 29.508.3 Member Function Documentation

#### 29.508.3.1 ReturnMatrix TestClass::Sum ( )

Definition at line 39 of file tmtb.cpp.

The documentation for this class was generated from the following file:

- [newmat/app/tmtb.cpp](#)

## 29.509 Go::TestInDomain Class Reference

IntersectionPoolUtils. predicate for STL function.

```
#include <IntersectionPoolUtils.h>
```

### Public Types

- typedef [const shared\\_ptr< IntersectionLink > argument\\_type](#)
- typedef [bool result\\_type](#)

### Public Member Functions

- [TestInDomain](#) ([const ParamObjectInt \\*obj1](#), [const ParamObjectInt \\*obj2](#), [shared\\_ptr< IntersectionPoint > ref\\_point](#))
- [bool operator\(\)](#) ([const shared\\_ptr< IntersectionLink > &l](#)) [const](#)

### 29.509.1 Detailed Description

IntersectionPoolUtils. predicate for STL function.

Definition at line 168 of file IntersectionPoolUtils.h.

### 29.509.2 Member Typedef Documentation

#### 29.509.2.1 typedef [const shared\\_ptr<IntersectionLink>](#) [Go::TestInDomain::argument\\_type](#)

Definition at line 213 of file IntersectionPoolUtils.h.

#### 29.509.2.2 typedef [bool](#) [Go::TestInDomain::result\\_type](#)

Definition at line 214 of file IntersectionPoolUtils.h.

### 29.509.3 Constructor & Destructor Documentation

29.509.3.1 `Go::TestInDomain::TestInDomain ( const ParamObjectInt * obj1, const ParamObjectInt * obj2, shared_ptr< IntersectionPoint > ref_point ) [inline]`

Definition at line 172 of file IntersectionPoolUtils.h.

### 29.509.4 Member Function Documentation

29.509.4.1 `bool Go::TestInDomain::operator() ( const shared_ptr< IntersectionLink > & / ) const [inline]`

Definition at line 194 of file IntersectionPoolUtils.h.

The documentation for this class was generated from the following file:

- [intersections/include/GoTools/intersections/IntersectionPoolUtils.h](#)

## 29.510 time\_lapse Class Reference

```
#include <tmt.h>
```

### Public Member Functions

- [time\\_lapse \(\)](#)
- [~time\\_lapse \(\)](#)

### 29.510.1 Detailed Description

Definition at line 11 of file tmt.h.

### 29.510.2 Constructor & Destructor Documentation

29.510.2.1 `time_lapse::time_lapse ( )`

29.510.2.2 `time_lapse::~~time_lapse ( )`

The documentation for this class was generated from the following file:

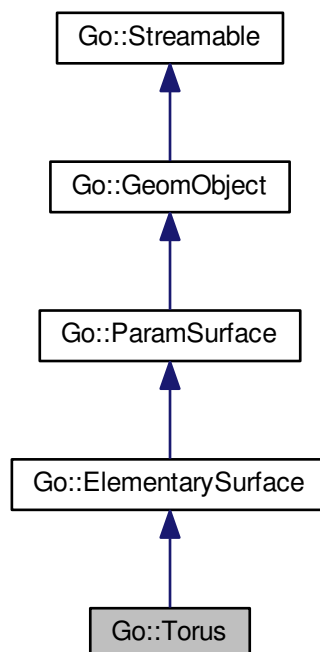
- [newmat/include/tmt.h](#)

## 29.511 Go::Torus Class Reference

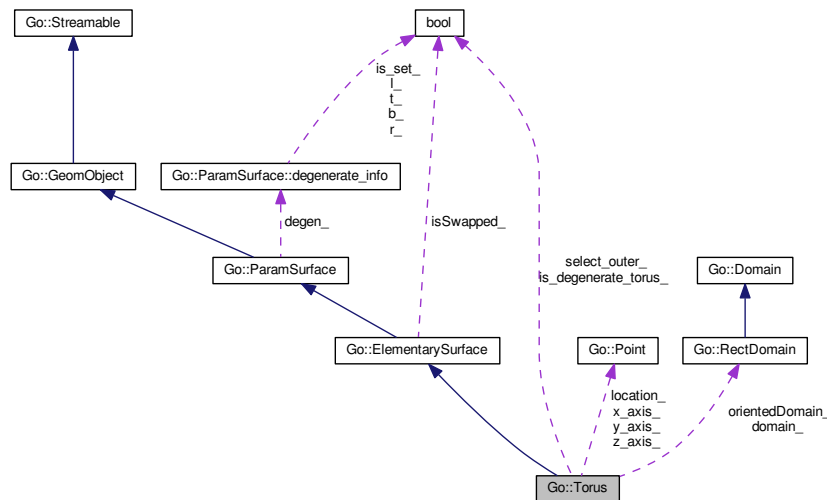
Class that represents a torus. It is a subclass of [ElementarySurface](#), and thus has a parametrization. A torus may be degenerate. Then the minor radius is greater than the major radius, and it is in principle selfintersecting.

```
#include <Torus.h>
```

Inheritance diagram for Go::Torus:



Collaboration diagram for Go::Torus:



## Public Member Functions

- [Torus](#) ()
- [Torus](#) (double major\_radius, double minor\_radius, [Point](#) location, [Point](#) z\_axis, [Point](#) x\_axis, bool select\_outer=↔ true, bool isSwapped=false)
- virtual [~Torus](#) ()
  - Virtual destructor - ensures safe inheritance.*
- virtual void [read](#) (std::istream &is)
- virtual void [write](#) (std::ostream &os) const
- virtual int [dimension](#) () const
  - Return the dimension of the space in which the object lies (usually 2 or 3)*
- virtual [ClassType](#) [instanceType](#) () const
  - Return the class type identifier of a given, derived instance of [GeomObject](#).*
- virtual [BoundingBox](#) [boundingBox](#) () const
  - Return the object's bounding box.*
- virtual [Torus](#) \* [clone](#) () const
- const [RectDomain](#) & [parameterDomain](#) () const
- std::vector< [CurveLoop](#) > [allBoundaryLoops](#) (double degenerate\_epsilon=DEFAULT\_SPACE\_EPSILON) const
- [DirectionCone](#) [normalCone](#) () const
- [DirectionCone](#) [tangentCone](#) (bool paddir\_is\_u) const
- void [point](#) ([Point](#) &pt, double upar, double vpar) const
- void [point](#) (std::vector< [Point](#) > &pts, double upar, double vpar, int derivs, bool u\_from\_right=true, bool v\_from\_right=↔ true, double resolution=1.0e-12) const
- void [normal](#) ([Point](#) &n, double upar, double vpar) const
- std::vector< shared\_ptr< [ParamCurve](#) > > [constParamCurves](#) (double parameter, bool paddir\_is\_u) const
- std::vector< shared\_ptr< [ParamSurface](#) > > [subSurfaces](#) (double from\_upar, double from\_vpar, double to\_upar, double to\_vpar, double fuzzy=DEFAULT\_PARAMETER\_EPSILON) const
- double [nextSegmentVal](#) (int dir, double par, bool forward, double tol) const
- void [closestPoint](#) (const [Point](#) &pt, double &clo\_u, double &clo\_v, [Point](#) &clo\_pt, double &clo\_dist, double epsilon, const [RectDomain](#) \*domain\_of\_interest=NULL, double \*seed=0) const

- void `closestBoundaryPoint` (`const Point` &pt, `double` &clo\_u, `double` &clo\_v, `Point` &clo\_pt, `double` &clo\_dist, `double` epsilon, `const RectDomain` \*rd=NULL, `double` \*seed=0) `const`
- void `getBoundaryInfo` (`Point` &pt1, `Point` &pt2, `double` epsilon, `SplineCurve` \*&cv, `SplineCurve` \*&crosscv, `double` knot\_tol=1e-05) `const`
- `bool` `isDegenerate` (`bool` &b, `bool` &r, `bool` &t, `bool` &l, `double` tolerance) `const`
- `bool` `isBounded` () `const`
- `bool` `isClosed` (`bool` &closed\_dir\_u, `bool` &closed\_dir\_v) `const`  
*Check if the surface is closed.*
- virtual void `getDegenerateCorners` (`std::vector`< `Point` > &deg\_corners, `double` tol) `const`  
*Check for parallell and anti parallell partial derivatives in surface corners.*
- `double` `getMajorRadius` () `const`  
*Fetch major radius.*
- `double` `getMinorRadius` () `const`  
*Fetch major radius.*
- `Point` `getLocation` () `const`  
*Fetch centre.*
- void `getCoordinateAxes` (`Point` &x\_axis, `Point` &y\_axis, `Point` &z\_axis) `const`  
*Local coordinate axes.*
- `bool` `getSelectOuter` () `const`
- void `setSelectOuter` (`bool` select\_outer)
- `double` `getPhi` () `const`
- `bool` `isDegenerateTorus` () `const`
- virtual void `setParameterBounds` (`double` from\_upar, `double` from\_vpar, `double` to\_upar, `double` to\_vpar)  
*Limit the surface by limiting the parameter domain.*
- `Torus` \* `subSurface` (`double` from\_upar, `double` from\_vpar, `double` to\_upar, `double` to\_vpar, `double` fuzzy=DEFAULT\_PARAMETER\_EPSILON) `const`  
*Return the part of the torus surface limited by the parameter bounds.*
- virtual `SplineSurface` \* `geometrySurface` () `const`  
*Create a `SplineSurface` representation of the `Torus`.*
- virtual `SplineSurface` \* `createSplineSurface` () `const`  
*Create a `SplineSurface` representation of the `Torus`.*
- `shared_ptr`< `Circle` > `getMajorCircle` (`double` vpar) `const`
- `shared_ptr`< `Circle` > `getMinorCircle` (`double` upar) `const`

## Static Public Member Functions

- static `ClassType` `classType` ()

## Protected Member Functions

- void `setCoordinateAxes` ()
- void `setDegenerateInfo` ()
- void `setDefaultDomain` ()



## Protected Attributes

- [double major\\_radius\\_](#)
- [double minor\\_radius\\_](#)
- [Point location\\_](#)
- [Point z\\_axis\\_](#)
- [Point x\\_axis\\_](#)
- [Point y\\_axis\\_](#)
- [bool is\\_degenerate\\_torus\\_](#)
- [bool select\\_outer\\_](#)
- [double phi\\_](#)
- [RectDomain domain\\_](#)
- [RectDomain orientedDomain\\_](#)

### 29.511.1 Detailed Description

Class that represents a torus. It is a subclass of [ElementarySurface](#), and thus has a parametrization. A torus may be degenerate. Then the minor radius is greater than the major radius, and it is in principle selfintersecting.

A nondegenerate [Torus](#) has a natural parametrization in terms of the two angles  $u$  and  $v$ :  $\mathbf{p}(u, v) = \mathbf{C} + (R + r \cos v)((\cos u) \mathbf{x} + (\sin u) \mathbf{y}) + (r \sin v) \mathbf{z}$ , where  $\mathbf{C}$  is a position vector,  $R$  is the major radius,  $r$  is the minor radius, and  $\mathbf{x}$ ,  $\mathbf{y}$  and  $\mathbf{z}$  are the (local) axes. The parametrization is bounded by:  $0 \leq u, v \leq 2\pi$ . The dimension is 3.

If the [Torus](#) is degenerate, there is an additional boolean parameter, `select_outer`, which selects the inner or outer (non-selfintersecting) part of the surface. This restricts the parametrization by:  $-\phi \leq v \leq \phi$ , if `select_outer` is true, and  $\phi \leq v \leq 2\pi - \phi$ , if `select_outer` is false. `select_outer` affects the direction of the normal, which always points outwards.

Definition at line 78 of file `Torus.h`.

### 29.511.2 Constructor & Destructor Documentation

#### 29.511.2.1 `Go::Torus::Torus( )` `[inline]`

Default constructor. Constructs an uninitialized [Torus](#) which can only be assigned to or read into.

Definition at line 83 of file `Torus.h`.

#### 29.511.2.2 `Go::Torus::Torus( double major_radius, double minor_radius, Point location, Point z_axis, Point x_axis, bool select_outer = true, bool isSwapped = false )`

Constructor. Input is the major and minor radii, the location, the direction of the z-axis and the (possibly approximate) x-axis. The local coordinate axes are normalized even if `z_axis` and/or `x_axis` are not unit vectors. If degenerate, one may also provide the `select_outer` flag.

#### 29.511.2.3 `virtual Go::Torus::~~Torus( )` `[virtual]`

Virtual destructor - ensures safe inheritance.

### 29.511.3 Member Function Documentation

#### 29.511.3.1 `std::vector<CurveLoop> Go::Torus::allBoundaryLoops( double degenerate_epsilon = DEFAULT_SPACE_EPSILON ) const` `[virtual]`

Returns the anticlockwise outer boundary loop of the surface, together with clockwise loops of any interior boundaries, such that the surface always is 'to the left of' the loops.

## Parameters

<i>degenerate_epsilon</i>	edges whose length is smaller than this value are ignored.
---------------------------	------------------------------------------------------------

## Returns

a vector containing CurveLoops. The first of these describe the outer boundary of the surface (clockwise), whereas the others describe boundaries of interior holes (clockwise).

Implements [Go::ParamSurface](#).

29.511.3.2 **virtual BoundingBox** `Go::Torus::boundingBox ( ) const` [virtual]

Return the object's bounding box.

Implements [Go::GeomObject](#).

29.511.3.3 **static ClassType** `Go::Torus::classType ( )` [inline],[static]

Definition at line 112 of file Torus.h.

29.511.3.4 **virtual Torus\*** `Go::Torus::clone ( ) const` [virtual]

make a clone of this surface and return a pointer to it (user is responsible for clearing up memory afterwards).

## Returns

pointer to cloned object

Implements [Go::ElementarySurface](#).

29.511.3.5 **void** `Go::Torus::closestBoundaryPoint ( const Point & pt, double & clo_u, double & clo_v, Point & clo_pt, double & clo_dist, double epsilon, const RectDomain * rd = NULL, double * seed = 0 ) const` [virtual]

Iterates to the closest point to pt on the boundary of the surface.

## See also

[closestPoint\(\)](#)

Implements [Go::ParamSurface](#).

29.511.3.6 **void** `Go::Torus::closestPoint ( const Point & pt, double & clo_u, double & clo_v, Point & clo_pt, double & clo_dist, double epsilon, const RectDomain * domain_of_interest = NULL, double * seed = 0 ) const` [virtual]

Iterates to the closest point to pt on the surface.

## Parameters

<i>pt</i>	the point to find the closest point to
<i>clo_u</i>	u parameter of the closest point
<i>clo_v</i>	v parameter of the closest point
<i>clo_pt</i>	the geometric position of the closest point
<i>clo_dist</i>	the distance between pt and clo_pt
<i>epsilon</i>	parameter tolerance (will in any case not be higher than sqrt(machine_precision) x magnitude of solution)
<i>domain_of_interest</i>	pointer to parameter domain in which to search for closest point. If a NULL pointer is used, the entire surface is searched.
<i>seed</i>	pointer to parameter values where iteration starts.

Reimplemented from [Go::ParamSurface](#).

```
29.511.3.7 std::vector<shared_ptr<ParamCurve>> > Go::Torus::constParamCurves (double parameter, bool
pardir_is_u) const [virtual]
```

Get the curve(s) obtained by intersecting the surface with one of its constant parameter curves. For surfaces without holes, this will be the parameter curve itself; for surfaces with interior holes this may be a collection of several, disjoint curves.

## Parameters

<i>parameter</i>	parameter value for the constant parameter (either u or v)
<i>pardir_is_u</i>	specify whether the <i>moving</i> parameter (as opposed to the <i>constant</i> parameter) is the first ('true') or the second ('false') one.

## Returns

a vector containing shared pointers to the obtained, newly constructed constant-parameter curves.

Implements [Go::ParamSurface](#).

```
29.511.3.8 virtual SplineSurface* Go::Torus::createSplineSurface () const [virtual]
```

Create a [SplineSurface](#) representation of the [Torus](#).

Implements [Go::ElementarySurface](#).

```
29.511.3.9 virtual int Go::Torus::dimension () const [virtual]
```

Return the dimension of the space in which the object lies (usually 2 or 3)

Implements [Go::GeomObject](#).

29.511.3.10 `virtual SplineSurface* Go::Torus::geometrySurface ( ) const [virtual]`

Create a [SplineSurface](#) representation of the [Torus](#).

Implements [Go::ElementarySurface](#).

29.511.3.11 `void Go::Torus::getBoundaryInfo ( Point & pt1, Point & pt2, double epsilon, SplineCurve *& cv, SplineCurve *& crosscv, double knot_tol = 1e-05 ) const [virtual]`

Get the boundary curve segment between two points on the boundary, as well as the cross-tangent curve. If the given points are not positioned on the same boundary (within a certain tolerance), no curves will be created.

#### Parameters

<i>pt1</i>	the first point on the boundary, given by the user
<i>pt2</i>	the second point on the boundary, given by the user
<i>epsilon</i>	the tolerance used when determining whether the given points are lying on a boundary, and if they do, whether they both lie on the <i>same</i> boundary.
<i>cv</i>	upon return, this will point to a newly created curve representing the boundary curve between 'pt1' and 'pt2'. The user assumes ownership of this object and is responsible for its deletion. No curve is created if the given points are not found to lie on the same boundary.
<i>crosscv</i>	upon return, this will point to a newly created curve representing the cross-boundary curve between 'pt1' and 'pt2'. The user assumes ownership of this object and is responsible for its deletion. The direction is outwards from the surface. No curve is created if the given points are not found to lie on the same boundary.
<i>knot_tol</i>	tolerance used when working with the knot-vector, to specify how close a parameter value must be to a knot in order to be considered 'on' the knot.

Implements [Go::ParamSurface](#).

29.511.3.12 `void Go::Torus::getCoordinateAxes ( Point & x_axis, Point & y_axis, Point & z_axis ) const [inline]`

Local coordinate axes.

Definition at line 209 of file [Torus.h](#).

29.511.3.13 `virtual void Go::Torus::getDegenerateCorners ( std::vector< Point > & deg_corners, double tol ) const [virtual]`

Check for parallel and anti parallel partial derivatives in surface corners.

Implements [Go::ParamSurface](#).

29.511.3.14 `Point Go::Torus::getLocation ( ) const [inline]`

Fetch centre.

Definition at line 205 of file [Torus.h](#).

29.511.3.15 `shared_ptr<Circle> Go::Torus::getMajorCircle ( double vpar ) const`

Get the major circle for a given v parameter.

## Parameters

<i>vpar</i>	v parameter
-------------	-------------

## Returns

A circle for the corresponding v parameter. If the v parameter is bounded, only a segment of a full circle is returned.

**29.511.3.16** `double Go::Torus::getMajorRadius ( ) const` `[inline]`

Fetch major radius.

Definition at line 197 of file Torus.h.

**29.511.3.17** `shared_ptr<Circle> Go::Torus::getMinorCircle ( double upar ) const`

Get the minor circle for a given u parameter.

## Parameters

<i>upar</i>	u parameter
-------------	-------------

## Returns

A circle for the corresponding u parameter. If the u parameter is bounded, only a segment of a full circle is returned.

**29.511.3.18** `double Go::Torus::getMinorRadius ( ) const` `[inline]`

Fetch major radius.

Definition at line 201 of file Torus.h.

**29.511.3.19** `double Go::Torus::getPhi ( ) const` `[inline]`

$\arccos(-\text{major\_radius}/\text{minor\_radius})$ . Used in degeneracy checks and to set the parameter domain of the surface

Definition at line 231 of file Torus.h.

**29.511.3.20** `bool Go::Torus::getSelectOuter ( ) const` `[inline]`

Fetch the `select_outer` flag. If the torus is not degenerate, this function has no effect. If the torus is degenerate, and the value of the flag is changed, this will change the parameter domain of the surface to the appropriate default.

Definition at line 220 of file Torus.h.

29.511.3.21 `virtual ClassType Go::Torus::instanceType ( ) const [virtual]`

Return the class type identifier of a given, derived instance of [GeomObject](#).

Implements [Go::GeomObject](#).

29.511.3.22 `bool Go::Torus::isBounded ( ) const [virtual]`

Query if parametrization is bounded. All four parameter bounds must be finite for this to be true.

Returns

*true* if bounded, *false* otherwise

Reimplemented from [Go::ElementarySurface](#).

29.511.3.23 `bool Go::Torus::isClosed ( bool & closed_dir_u, bool & closed_dir_v ) const [virtual]`

Check if the surface is closed.

Reimplemented from [Go::ElementarySurface](#).

29.511.3.24 `bool Go::Torus::isDegenerate ( bool & b, bool & r, bool & t, bool & l, double tolerance ) const [virtual]`

The order of the edge indicators (bottom, right, top, left) matches the `edge_number` of `edgeCurve()`. Query whether any of the four boundary curves are degenerate (zero length) within a certain tolerance. In the below, we refer to 'u' as the first parameter and 'v' as the second.

Parameters

<i>b</i>	'true' upon return of function if the boundary ( $v = v_{\min}$ ) is degenerate
<i>r</i>	'true' upon return of function if the boundary ( $v = v_{\max}$ ) is degenerate
<i>t</i>	'true' upon return of function if the boundary ( $u = u_{\min}$ ) is degenerate
<i>l</i>	'true' upon return of function if the boundary ( $u = u_{\max}$ ) is degenerate
<i>tolerance</i>	boundaries are considered degenerate if their length is shorter than this value, given by the user

Returns

'true' if at least one boundary curve was found to be degenerate, 'false' otherwise.

Reimplemented from [Go::ParamSurface](#).

29.511.3.25 `bool Go::Torus::isDegenerateTorus ( ) const [inline]`

Query if the [Torus](#) is degenerate. Note that a [Torus](#) may be degenerate, although the parameter domain might not represent a degenerate patch. Thus, `isDegenerateTorus()` might return `true` and `isDegerate()` might return `false` at the same time.

Definition at line 238 of file `Torus.h`.

29.511.3.26 `double Go::Torus::nextSegmentVal ( int dir, double par, bool forward, double tol ) const` [virtual]

Determine the parameter value of the start of the 'next segment' from a parameter value, along a given parameter direction. A 'segment' is here defined as a parameter interval in which there will be no discontinuities in derivatives or other artifacts. For spline objects, a segment will typically be the interval between two consecutive, non-coincident knots.

#### Parameters

<i>dir</i>	the parameter direction in which we search for the next segment (0 or 1)
<i>par</i>	the parameter value starting from which we search for the start value of the next segment
<i>forward</i>	define whether we shall move forward ('true') or backwards when searching along this parameter
<i>tol</i>	tolerance used for determining whether the 'par' is already located <i>on</i> the next segment value

#### Returns

the value of the start value of the next segment (or the end of the previous segment, if we are moving backwards...)

Implements [Go::ParamSurface](#).

29.511.3.27 `void Go::Torus::normal ( Point & n, double upar, double vpar ) const` [virtual]

The normal to the torus. The normal always points *outwards* from the surface, as seen from the circle of radius `major_radius_`. Thus, if the torus is degenerate and `select_outer_` is `false`, then the direction of the normal is opposite to  $\partial p / \partial u \times \partial p / \partial v$ .

Implements [Go::ParamSurface](#).

29.511.3.28 `DirectionCone Go::Torus::normalCone ( ) const` [virtual]

Creates a [DirectionCone](#) covering all normals to this surface.

#### Returns

a [DirectionCone](#) (not necessarily the smallest) containing all normals to this surface.

Implements [Go::ParamSurface](#).

29.511.3.29 `const RectDomain& Go::Torus::parameterDomain ( ) const` [virtual]

Return the parameter domain of the surface. This may be a simple rectangular domain ([RectDomain](#)) or any other subclass of [Domain](#) (such as [GoCurveBoundedDomain](#), found in the `sisl_dependent` module).

#### Returns

a [Domain](#) object describing the parametric domain of the surface

Implements [Go::ParamSurface](#).

29.511.3.30 `void Go::Torus::point ( Point & pt, double upar, double vpar ) const` [virtual]

Evaluates the surface's position for a given parameter pair.

## Parameters

<i>pt</i>	the result of the evaluation is written here
<i>upar</i>	the first parameter
<i>vpar</i>	the second parameter

Implements [Go::ParamSurface](#).

```
29.511.3.31 void Go::Torus::point (std::vector< Point > & pts, double upar, double vpar, int derivs, bool u_from_right = true, bool v_from_right = true, double resolution = 1.0e-12) const [virtual]
```

Evaluates the surface's position and a certain number of derivatives for a given parameter pair.

## Parameters

<i>pts</i>	the vector containing the evaluated values. Its size must be set by the user prior to calling this function, and should be equal to $(derivs+1) * (derivs+2) / 2$ . Upon completion of the function, its first entry is the surface's position at the given parameter pair. Then, if 'derivs' > 0, the two next entries will be the surface tangents along the first and second parameter direction. The next three entries are the second- and cross derivatives, in the order (du2, dudv, dv2), and similar for even higher derivatives.
<i>upar</i>	the first parameter
<i>vpar</i>	the second parameter
<i>derivs</i>	number of requested derivatives
<i>u_from_right</i>	specify whether derivatives along the first parameter are to be calculated from the right ('true', default) or from the left ('false')
<i>v_from_right</i>	specify whether derivatives along the second parameter are to be calculated from the right ('true', default) or from the left ('false')
<i>resolution</i>	tolerance used when determining whether parameters are located at special values of the parameter domain (in particular; knot values in case of spline objects).

Implements [Go::ParamSurface](#).

```
29.511.3.32 virtual void Go::Torus::read (std::istream & is) [virtual]
```

read object from stream

## Parameters

<i>is</i>	stream from which object is read
-----------	----------------------------------

Implements [Go::Streamable](#).

```
29.511.3.33 void Go::Torus::setCoordinateAxes () [protected]
```

```
29.511.3.34 void Go::Torus::setDefaultDomain () [protected]
```



29.511.3.35 void Go::Torus::setDegenerateInfo ( ) [protected]

29.511.3.36 virtual void Go::Torus::setParameterBounds ( double *from\_upar*, double *from\_vpar*, double *to\_upar*, double *to\_vpar* ) [virtual]

Limit the surface by limiting the parameter domain.

Implements [Go::ElementarySurface](#).

29.511.3.37 void Go::Torus::setSelectOuter ( bool *select\_outer* )

Set the *select\_outer* flag. If the torus is not degenerate, this function has no effect. If the torus is degenerate, and the value of the flag is changed, this will change the parameter domain of the surface to the appropriate default.

29.511.3.38 **Torus\*** Go::Torus::subSurface ( double *from\_upar*, double *from\_vpar*, double *to\_upar*, double *to\_vpar*, double *fuzzy* = DEFAULT\_PARAMETER\_EPSILON ) const

Return the part of the torus surface limited by the parameter bounds.

29.511.3.39 std::vector<shared\_ptr<ParamSurface>> Go::Torus::subSurfaces ( double *from\_upar*, double *from\_vpar*, double *to\_upar*, double *to\_vpar*, double *fuzzy* = DEFAULT\_PARAMETER\_EPSILON ) const [virtual]

Get the surface(s) obtained by cropping the parameter domain of this surface between given values for the first and second parameter. In general, for surfaces with no interior holes, the result will be *one* surface; however, for surfaces with interior holes, the result might be *several disjoint* surfaces.

#### Parameters

<i>from_upar</i>	lower value for the first parameter in the subdomain
<i>from_vpar</i>	lower value for the second parameter in the subdomain
<i>to_upar</i>	upper value for the first parameter in the subdomain
<i>to_vpar</i>	upper value for the second parameter in the subdomain
<i>fuzzy</i>	tolerance used when determining intersection with interior boundaries

#### Returns

a vector contained shared pointers to the obtained, newly constructed sub-surfaces.

Implements [Go::ParamSurface](#).

29.511.3.40 **DirectionCone** Go::Torus::tangentCone ( bool *pardir\_is\_u* ) const [virtual]

Creates a [DirectionCone](#) covering all tangents to this surface along a given parameter direction.

## Parameters

<code>pardir_is_↔ _u</code>	if 'true', then the <a href="#">DirectionCone</a> will be defined on basis of the surface's tangents along the first parameter direction. Otherwise the second parameter direction will be used.
---------------------------------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

## Returns

a [DirectionCone](#) (not necessarily the smallest) containing all tangents to this surface along the specified parameter direction.

Implements [Go::ParamSurface](#).

**29.511.3.41** `virtual void Go::Torus::write ( std::ostream & os ) const` [virtual]

write object to stream

## Parameters

<code>os</code>	stream to which object is written
-----------------	-----------------------------------

Implements [Go::Streamable](#).

**29.511.4 Member Data Documentation**

**29.511.4.1** `RectDomain Go::Torus::domain_` [protected]

Definition at line 282 of file Torus.h.

**29.511.4.2** `bool Go::Torus::is_degenerate_torus_` [protected]

Definition at line 278 of file Torus.h.

**29.511.4.3** `Point Go::Torus::location_` [protected]

Definition at line 274 of file Torus.h.

**29.511.4.4** `double Go::Torus::major_radius_` [protected]

Definition at line 272 of file Torus.h.

**29.511.4.5** `double Go::Torus::minor_radius_` [protected]

Definition at line 273 of file Torus.h.

29.511.4.6 **RectDomain** Go::Torus::orientedDomain\_ [mutable],[protected]

Definition at line 283 of file Torus.h.

29.511.4.7 **double** Go::Torus::phi\_ [protected]

Definition at line 280 of file Torus.h.

29.511.4.8 **bool** Go::Torus::select\_outer\_ [protected]

Definition at line 279 of file Torus.h.

29.511.4.9 **Point** Go::Torus::x\_axis\_ [protected]

Definition at line 276 of file Torus.h.

29.511.4.10 **Point** Go::Torus::y\_axis\_ [protected]

Definition at line 277 of file Torus.h.

29.511.4.11 **Point** Go::Torus::z\_axis\_ [protected]

Definition at line 275 of file Torus.h.

The documentation for this class was generated from the following file:

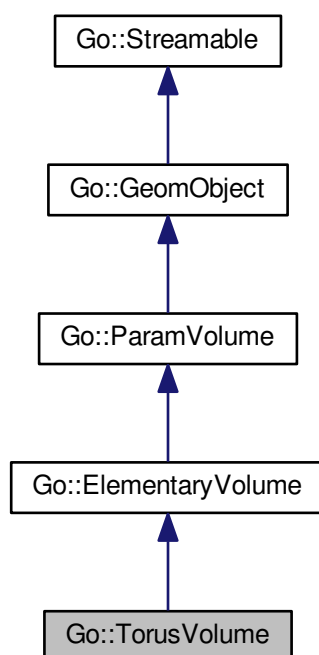
- [gotools-core/include/GoTools/geometry/Torus.h](#)

## 29.512 Go::TorusVolume Class Reference

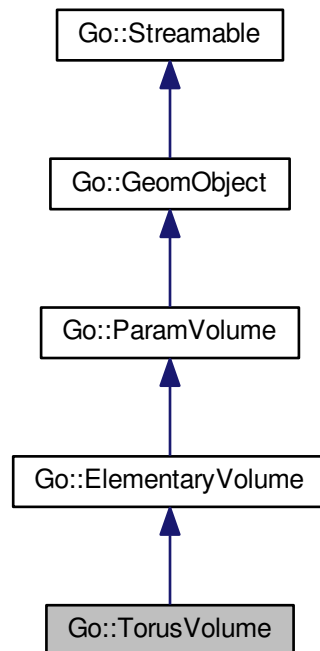
Class that represents a solid torus, maybe with empty interior like a circular pipe, and/or a section (not full revolution along the main axis). It is a subclass of [ElementaryVolume](#), and has a natural parametrization in terms of a radius  $u$  and two angles  $v$  and  $w$ :  $\mathbf{p}(u, v, w) = \mathbf{C} + (R + u \cos w)((\cos v) \mathbf{x} + (\sin v) \mathbf{y}) + (u \sin w) \mathbf{z}$ , where  $\mathbf{C}$  is a position vector,  $R$  is the major radius, and  $\mathbf{x}$ ,  $\mathbf{y}$  and  $\mathbf{z}$  are the (local) axes. The parametrization for a full solid torus is bounded by:  $0 \leq u \leq r$ ,  $0 \leq v, w \leq 2\pi$ , where  $r$  is the minor axis. A bended pipe will have a positive minimal limit for  $u$ , while a torus segment will have other limits for  $v$ . The dimension is 3.

```
#include <TorusVolume.h>
```

Inheritance diagram for Go::TorusVolume:



Collaboration diagram for Go::TorusVolume:



## Public Member Functions

- [TorusVolume](#) ()
- [TorusVolume](#) (double major\_radius, double minor\_radius, [Point](#) location, [Point](#) z\_axis, [Point](#) x\_axis)
- virtual [~TorusVolume](#) ()
  - Virtual destructor - ensures safe inheritance.*
- virtual void [read](#) (std::istream &is)
- virtual void [write](#) (std::ostream &os) const
- virtual int [dimension](#) () const
  - Return the dimension of the space in which the object lies (usually 2 or 3)*
- virtual [ClassType](#) [instanceType](#) () const
  - Return the class type identifier of a given, derived instance of [GeomObject](#).*
- virtual [BoundingBox](#) [boundingBox](#) () const
  - Return the object's bounding box.*
- virtual [TorusVolume](#) \* [clone](#) () const
- [DirectionCone](#) [tangentCone](#) (int paddir) const
- const [Array](#) < double, 6 > [parameterSpan](#) () const
- void [point](#) ([Point](#) &pt, double upar, double vpar, double wpar) const
- void [point](#) (std::vector< [Point](#) > &pts, double upar, double vpar, double wpar, int derivs, bool u\_from\_↔right=true, bool v\_from\_right=true, bool w\_from\_right=true, double resolution=1.0e-12) const
- double [nextSegmentVal](#) (int dir, double par, bool forward, double tol) const
- void [closestPoint](#) (const [Point](#) &pt, double &clo\_u, double &clo\_v, double &clo\_w, [Point](#) &clo\_pt, double &clo\_↔\_dist, double epsilon, double \*seed=0) const
- void [reverseParameterDirection](#) (int paddir)

- void [swapParameterDirection](#) (int paddir1, int paddir2)
- virtual std::vector< shared\_ptr< [ParamSurface](#) > > [getAllBoundarySurfaces](#) () const  
*Fetch all boundary surfaces corresponding to the volume.*
- virtual void [translate](#) (const [Point](#) &vec)  
*Translate.*
- [SplineVolume](#) \* [geometryVolume](#) () const  
*Make a NURBS representation of the object.*
- void [setParameters](#) (double from\_par, double to\_par, int paddir)  
*Restrict the size of the torus in one parameter direction.*
- void [useCentreDegen](#) ()
- void [useCornerDegen](#) ()

### Static Public Member Functions

- static [ClassType](#) [classType](#) ()

#### 29.512.1 Detailed Description

Class that represents a solid torus, maybe with empty interior like a circular pipe, and/or a section (not full revolution along the main axis). It is a subclass of [ElementaryVolume](#), and has a natural parametrization in terms of a radius  $u$  and two angles  $v$  and  $w$ :  $\mathbf{p}(u, v, w) = \mathbf{C} + (R + u \cos w)((\cos v) \mathbf{x} + (\sin v) \mathbf{y}) + (u \sin w) \mathbf{z}$ , where  $\mathbf{C}$  is a position vector,  $R$  is the major radius, and  $\mathbf{x}$ ,  $\mathbf{y}$  and  $\mathbf{z}$  are the (local) axes. The parametrization for a full solid torus is bounded by:  $0 \leq u \leq r$ ,  $0 \leq v, w \leq 2\pi$ , where  $r$  is the minor axis. A bended pipe will have a positive minimal limit for  $u$ , while a torus segment will have other limits for  $v$ . The dimension is 3.

Definition at line 68 of file [TorusVolume.h](#).

#### 29.512.2 Constructor & Destructor Documentation

##### 29.512.2.1 [Go::TorusVolume::TorusVolume](#) ( ) [inline]

Default constructor. Constructs an uninitialized [TorusVolume](#) which can only be assigned to or read into.

Definition at line 73 of file [TorusVolume.h](#).

##### 29.512.2.2 [Go::TorusVolume::TorusVolume](#) ( double *major\_radius*, double *minor\_radius*, [Point](#) *location*, [Point](#) *z\_axis*, [Point](#) *x\_axis* )

Constructor. Input is the major and minor radii, the location, the direction of the z-axis and the (possibly approximate) x-axis.

##### 29.512.2.3 virtual [Go::TorusVolume::~~TorusVolume](#) ( ) [virtual]

Virtual destructor - ensures safe inheritance.

### 29.512.3 Member Function Documentation

29.512.3.1 virtual **BoundingBox** Go::TorusVolume::boundingBox ( ) const [virtual]

Return the object's bounding box.

Implements [Go::GeomObject](#).

29.512.3.2 static **ClassType** Go::TorusVolume::classType ( ) [inline],[static]

Definition at line 99 of file TorusVolume.h.

29.512.3.3 virtual **TorusVolume\*** Go::TorusVolume::clone ( ) const [virtual]

make a clone of this volume and return a pointer to it (user is responsible for clearing up memory afterwards).

#### Returns

pointer to cloned object

Implements [Go::ElementaryVolume](#).

29.512.3.4 void Go::TorusVolume::closestPoint ( const **Point** & *pt*, double & *clo\_u*, double & *clo\_v*, double & *clo\_w*, **Point** & *clo\_pt*, double & *clo\_dist*, double *epsilon*, double \* *seed* = 0 ) const [virtual]

Iterates to the closest point to *pt* on the volume.

#### Parameters

<i>pt</i>	the point to find the closest point to
<i>clo_u</i>	u parameter of the closest point
<i>clo_v</i>	v parameter of the closest point
<i>clo_w</i>	w parameter of the closest point
<i>clo_pt</i>	the geometric position of the closest point
<i>clo_dist</i>	the distance between <i>pt</i> and <i>clo_pt</i>
<i>epsilon</i>	parameter tolerance (will in any case not be higher than $\sqrt{\text{machine\_precision}}$ x magnitude of solution)
<i>seed</i>	pointer to parameter values where iteration starts.

Implements [Go::ParamVolume](#).

29.512.3.5 virtual int Go::TorusVolume::dimension ( ) const [virtual]

Return the dimension of the space in which the object lies (usually 2 or 3)

Implements [Go::GeomObject](#).

29.512.3.6 **SplineVolume\*** `Go::TorusVolume::geometryVolume ( ) const` [virtual]

Make a NURBS representation of the object.

Implements [Go::ElementaryVolume](#).

29.512.3.7 `virtual std::vector<shared_ptr<ParamSurface> > Go::TorusVolume::getAllBoundarySurfaces ( ) const`  
[virtual]

Fetch all boundary surfaces corresponding to the volume.

Implements [Go::ParamVolume](#).

29.512.3.8 `virtual ClassType Go::TorusVolume::instanceType ( ) const` [virtual]

Return the class type identifier of a given, derived instance of [GeomObject](#).

Implements [Go::GeomObject](#).

29.512.3.9 `double Go::TorusVolume::nextSegmentVal ( int dir, double par, bool forward, double tol ) const`  
[virtual]

Determine the parameter value of the start of the 'next segment' from a parameter value, along a given parameter direction. A 'segment' is here defined as a parameter interval in which there will be no discontinuities in derivatives or other artifacts. For spline objects, a segment will typically be the interval between two consecutive, non-coincident knots.

#### Parameters

<i>dir</i>	the parameter direction in which we search for the next segment (0, 1 or 2)
<i>par</i>	the parameter value starting from which we search for the start value of the next segment
<i>forward</i>	define whether we shall move forward ('true') or backwards when searching along this parameter
<i>tol</i>	tolerance used for determining whether the 'par' is already located <i>on</i> the next segment value

#### Returns

the value of the start value of the next segment (or the end of the previous segment, if we are moving backwards...)

Implements [Go::ParamVolume](#).

29.512.3.10 `const Array<double,6> Go::TorusVolume::parameterSpan ( ) const` [virtual]

Return the parameter domain of the volume. This is an array containing the start and end parameter values. The spline's parameter *i* has its start value at the array position (2*i*) and its end parameter value at the array position (2*i*+1)

#### Returns

An array describing the parametric domain of the volume

Implements [Go::ParamVolume](#).



29.512.3.11 `void Go::TorusVolume::point ( Point & pt, double upar, double vpar, double wpar ) const` `[virtual]`

Evaluates the volume's position for a given parameter triple.

#### Parameters

<i>pt</i>	the result of the evaluation is written here
<i>upar</i>	the first parameter
<i>vpar</i>	the second parameter
<i>wpar</i>	the third parameter

Implements [Go::ParamVolume](#).

29.512.3.12 `void Go::TorusVolume::point ( std::vector< Point > & pts, double upar, double vpar, double wpar, int derivs, bool u_from_right = true, bool v_from_right = true, bool w_from_right = true, double resolution = 1.0e-12 ) const` `[virtual]`

Evaluates the volume's position and a certain number of derivatives for a given parameter triple.

#### Parameters

<i>pts</i>	the vector containing the evaluated values. Its size must be set by the user prior to calling this function. Upon completion of the function, its first entry is the volume's position at the given parameter pair. Then, if ' <i>derivs</i> ' > 0, the two next entries will be the volume tangents along the first, second and third parameter direction. The next three entries are the second- and cross derivatives, in the order ( <i>du</i> <sup>2</sup> , <i>dudv</i> , <i>dudw</i> , <i>dv</i> <sup>2</sup> , <i>dvdw</i> , <i>dw</i> <sup>2</sup> ), and similar for even higher derivatives.
<i>upar</i>	the first parameter
<i>vpar</i>	the second parameter
<i>wpar</i>	the third parameter
<i>derivs</i>	number of requested derivatives
<i>u_from_right</i>	specify whether derivatives along the first parameter are to be calculated from the right ('true', default) or from the left ('false')
<i>v_from_right</i>	specify whether derivatives along the second parameter are to be calculated from the right ('true', default) or from the left ('false')
<i>w_from_right</i>	specify whether derivatives along the third parameter are to be calculated from the right ('true', default) or from the left ('false')
<i>resolution</i>	tolerance used when determining whether parameters are located at special values of the parameter domain (in particular; knot values in case of spline objects).

Implements [Go::ParamVolume](#).

29.512.3.13 `virtual void Go::TorusVolume::read ( std::istream & is )` `[virtual]`

read object from stream

#### Parameters

<i>is</i>	stream from which object is read
-----------	----------------------------------

Implements [Go::Streamable](#).

29.512.3.14 void `Go::TorusVolume::reverseParameterDirection ( int pardir )` [virtual]

Reverses the direction of the basis in input direction.

#### Parameters

<i>pardir</i>	which parameter direction to reverse
---------------	--------------------------------------

Implements [Go::ParamVolume](#).

29.512.3.15 void `Go::TorusVolume::setParameters ( double from_par, double to_par, int pardir )`

Restrict the size of the torus in one parameter direction.

29.512.3.16 void `Go::TorusVolume::swapParameterDirection ( int pardir1, int pardir2 )` [virtual]

Swaps two parameter directions

#### Parameters

<i>pardir1</i>	One of the parameter directions (0, 1 or 2) to be swapped
<i>pardir2</i>	The other parameter direction (0, 1 or 2) to be swapped

Implements [Go::ParamVolume](#).

29.512.3.17 **DirectionCone** `Go::TorusVolume::tangentCone ( int pardir ) const` [virtual]

Creates a [DirectionCone](#) covering all tangents to this volume along a given parameter direction.

#### Parameters

<i>pardir</i>	if 1, the <a href="#">DirectionCone</a> will be defined on basis of the volume's tangents along the first parameter direction. If 2, the second parameter direction will be used. If 3, the third parameter direction will be used.
---------------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

#### Returns

a [DirectionCone](#) (not necessarily the smallest) containing all tangents to this volume along the specified parameter direction.

Implements [Go::ParamVolume](#).

29.512.3.18 `virtual void Go::TorusVolume::translate ( const Point & vec ) [virtual]`

Translate.

Implements [Go::ParamVolume](#).

29.512.3.19 `void Go::TorusVolume::useCentreDegen ( ) [inline]`

A NURBS representation will be a disc with degeneracy in the centre rotated around the centre

Definition at line 156 of file `TorusVolume.h`.

29.512.3.20 `void Go::TorusVolume::useCornerDegen ( ) [inline]`

A NURBS representation will be a disc with degenerate corners rotated around the centre

Definition at line 161 of file `TorusVolume.h`.

29.512.3.21 `virtual void Go::TorusVolume::write ( std::ostream & os ) const [virtual]`

write object to stream

Parameters

<code>os</code>	stream to which object is written
-----------------	-----------------------------------

Implements [Go::Streamable](#).

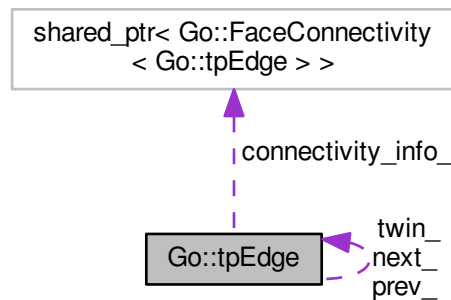
The documentation for this class was generated from the following file:

- `trivariate/include/GoTools/trivariate/TorusVolume.h`

## 29.513 Go::tpEdge Class Reference

```
#include <tpEdge.h>
```

Collaboration diagram for Go::tpEdge:



## Public Member Functions

- virtual `~tpEdge ()`  
*Empty destructor.*
- `tpEdge * next ()`  
*Next edge in the edge loop.*
- `tpEdge * twin ()`
- virtual `double tMin () const =0`  
*Start parameter value of edge.*
- virtual `double tMax () const =0`  
*End parameter value of edge.*
- virtual `tpFace * face ()=0`  
*The face corresponding to this edge.*
- virtual `BoundingBox boundingBox ()=0`  
*Coordinate box surrounding this edge.*
- virtual `void setEntryId (int id)=0`  
*Id of edge. Default set to -1.*
- virtual `tpEdge * split (double t)=0`
- virtual `Point point (double t) const =0`  
*Evaluate position.*
- virtual `Point normal (double t) const =0`  
*Evaluate normal of associated face.*
- virtual `void closestpoint (const Point &pt, double &clo_t, Point &clo_pt, double &clo_dist, double const *seed=0) const =0`  
*Closest point on this edge to a given point.*
- `tpEdge ()`  
*Default constructor.*
- `tpEdge * prev ()`  
*Previous edge in the edge loop.*
- virtual `int entryId ()=0`  
*Get id of edge.*
- virtual `void connectAfter (tpEdge *edge)`  
*Add the edge edge to the edge loop after this edge.*

- virtual void `closeLoop` (`tpEdge *last`)
- virtual void `disconnectThis` ()
  - Remove this edge from the edge loop it belongs to.*
- virtual void `connectTwin` (`tpEdge *twin`, `int &status`)
- virtual void `disconnectTwin` ()
- `bool onBoundary` ()
- void `adjacentEdges` (`bool at_start_of_edge`, `std::vector< tpEdge * > &adjacent`, `std::vector< bool > &at_start`)
- virtual `Point tangent` (`double t`) `const =0`
  - Evaluate tangent.*
- `tpJointType checkContinuity` (`tpEdge *nextedge`, `double neighbour`, `double gap`, `double bend`, `double kink`) `const`
- `bool hasConnectivityInfo` ()
- `shared_ptr< FaceConnectivity< tpEdge > > getConnectivityInfo` ()
- void `setConnectivityInfo` (`shared_ptr< FaceConnectivity< tpEdge > > info`)
  - corresponding to this edge and the adjacent face along this edge*
- void `resetConnectivityInfo` ()
  - Mark the connectivity information associated to this edge as outdated.*

## Protected Attributes

- `tpEdge * next_`
- `tpEdge * prev_`
- `tpEdge * twin_`
- `shared_ptr< FaceConnectivity< tpEdge > > connectivity_info_`

### 29.513.1 Detailed Description

`tpEdge` is an abstract interface for a half-edge topology structure. Suggested (and minimal) structure of template `edgeType` when using `FaceAdjacency`.

#### Author

Atgeirr F Rasmussen [atgeirr@sintef.no](mailto:atgeirr@sintef.no)

#### See also

`faceType`

Definition at line 63 of file `tpEdge.h`.

### 29.513.2 Constructor & Destructor Documentation

#### 29.513.2.1 virtual `Go::tpEdge::~tpEdge` ( ) [virtual]

Empty destructor.

29.513.2.2 `Go::tpEdge::tpEdge ( )`

Default constructor.

## 29.513.3 Member Function Documentation

29.513.3.1 `void Go::tpEdge::adjacentEdges ( bool at_start_of_edge, std::vector< tpEdge * > & adjacent, std::vector< bool > & at_start )`

Consider the 'node' or 'corner' implied by the startpoint (if *at\_start\_of\_edge* is true) or endpoint (if it is false). The vector *adjacent* is filled with all edges with that 'node' as a startpoint (indicated by a true at the corresponding place in the *at\_start* vector) or endpoint (false in *at\_start*). The edge originally asked for is not included.

29.513.3.2 `virtual BoundingBox Go::tpEdge::boundingBox ( ) [pure virtual]`

Coordinate box surrounding this edge.

29.513.3.3 `tpJointType Go::tpEdge::checkContinuity ( tpEdge * nextedge, double neighbour, double gap, double bend, double kink ) const`

Compute the continuity between this edge and *nextedge* given the necessary continuity parameters. 0 = g1 continuity, 1 = not quite g1, 2 = g0 continuity, 3 = gap, 4 = discontinuous, 5 = last segment

29.513.3.4 `virtual void Go::tpEdge::closeLoop ( tpEdge * last ) [virtual]`

Close edge loop by setting the appropriate pointers between this edge and last edge

29.513.3.5 `virtual void Go::tpEdge::closestpoint ( const Point & pt, double & clo_t, Point & clo_pt, double & clo_dist, double const * seed = 0 ) const [pure virtual]`

Closest point on this edge to a given point.

29.513.3.6 `virtual void Go::tpEdge::connectAfter ( tpEdge * edge ) [virtual]`

Add the edge *edge* to the edge loop after this edge.

29.513.3.7 `virtual void Go::tpEdge::connectTwin ( tpEdge * twin, int & status ) [virtual]`

Set adjacency between the face connected to this edge and the face connected to *twin* by setting twin pointers

29.513.3.8 `virtual void Go::tpEdge::disconnectThis ( ) [virtual]`

Remove this edge from the edge loop it belongs to.

29.513.3.9 `virtual void Go::tpEdge::disconnectTwin ( ) [virtual]`

Disconnect this edge from its twin, i.e. remove adjacency information between the face connected to this edge and the face connected to the twin edge

29.513.3.10 `virtual int Go::tpEdge::entryId ( ) [pure virtual]`

Get id of edge.

29.513.3.11 `virtual tpFace* Go::tpEdge::face ( ) [pure virtual]`

The face corresponding to this edge.

29.513.3.12 `shared_ptr<FaceConnectivity<tpEdge>> Go::tpEdge::getConnectivityInfo ( ) [inline]`

Fetch the continuity information regarding the joint between the face corresponding to this edge and the adjacent face along this edge

Definition at line 160 of file tpEdge.h.

29.513.3.13 `bool Go::tpEdge::hasConnectivityInfo ( ) [inline]`

Check if this edge has access to continuity information regarding the joint between the face corresponding to this edge and the adjacent face along this edge

Definition at line 153 of file tpEdge.h.

29.513.3.14 `tpEdge* Go::tpEdge::next ( )`

Next edge in the edge loop.

29.513.3.15 `virtual Point Go::tpEdge::normal ( double t ) const [pure virtual]`

Evaluate normal of associated face.

29.513.3.16 `bool Go::tpEdge::onBoundary ( ) [inline]`

Check if this edge lies on the boundary of the associated face set, i.e. it has no neighbour

Definition at line 129 of file tpEdge.h.

29.513.3.17 `virtual Point Go::tpEdge::point ( double t ) const [pure virtual]`

Evaluate position.

29.513.3.18 `tpEdge* Go::tpEdge::prev ( )`

Previous edge in the edge loop.

29.513.3.19 `void Go::tpEdge::resetConnectivityInfo ( ) [inline]`

Mark the connectivity information associated to this edge as outdated.

Definition at line 173 of file `tpEdge.h`.

29.513.3.20 `void Go::tpEdge::setConnectivityInfo ( shared_ptr< FaceConnectivity< tpEdge > > info ) [inline]`

corresponding to this edge and the adjacent face along this edge

Store the continuity information regarding the joint between the face

Definition at line 167 of file `tpEdge.h`.

29.513.3.21 `virtual void Go::tpEdge::setEntryId ( int id ) [pure virtual]`

Id of edge. Default set to -1.

29.513.3.22 `virtual tpEdge* Go::tpEdge::split ( double t ) [pure virtual]`

Split should be implemented so that the new edge (returned by the function), is fully connected. It should also be created by operator `new`, so callers should keep in mind that they are responsible for managing the lifetime of the new edge.

29.513.3.23 `virtual Point Go::tpEdge::tangent ( double t ) const [pure virtual]`

Evaluate tangent.

29.513.3.24 `virtual double Go::tpEdge::tMax ( ) const [pure virtual]`

End parameter value of edge.

29.513.3.25 `virtual double Go::tpEdge::tMin ( ) const [pure virtual]`

Start parameter value of edge.

29.513.3.26 `tpEdge* Go::tpEdge::twin ( )`

The corresponding edge on the neighbouring face. If no such neighbour exists, the function returns null.



### 29.513.4 Member Data Documentation

29.513.4.1 `shared_ptr<FaceConnectivity<tpEdge>>` `Go::tpEdge::connectivity_info_` [protected]

Definition at line 183 of file `tpEdge.h`.

29.513.4.2 `tpEdge*` `Go::tpEdge::next_` [protected]

Definition at line 180 of file `tpEdge.h`.

29.513.4.3 `tpEdge*` `Go::tpEdge::prev_` [protected]

Definition at line 181 of file `tpEdge.h`.

29.513.4.4 `tpEdge*` `Go::tpEdge::twin_` [protected]

Definition at line 182 of file `tpEdge.h`.

The documentation for this class was generated from the following file:

- [topology/include/GoTools/topology/tpEdge.h](#)

## 29.514 Go::tpFace Class Reference

```
#include <tpFace.h>
```

### Public Member Functions

- virtual `~tpFace ()`  
*Destructor.*
- virtual `std::vector< shared_ptr< tpEdge > > createInitialEdges (double degenerate_epsilon=DEFAULT_↔SPACE_EPSILON)=0`
- virtual `std::vector< shared_ptr< tpEdge > > startEdges ()=0`  
*Return pointers to first part of all bd cvs.*
- virtual `Point point (double u, double v) const =0`  
*Evaluate point on face.*
- virtual `Point normal (double u, double v) const =0`  
*Evaluate surface normal.*
- virtual `BoundingBox boundingBox ()=0`  
*The bounding box corresponding to this face.*
- virtual `int getId ()=0`  
*Return id, default id is -1.*
- virtual `void isolateFace ()`  
*Remove all adjacency information related to this face.*

### 29.514.1 Detailed Description

`tpFace` - Minimal structure of template `faceType` when using `FaceAdjacency`. The interface is limited to the functionality required to compute adjacency information in a set of faces and to remove this face from a face set, i.e. remove all topology pointers related to this face.

#### Author

Atgeirr F Rasmussen [atgeirr@sintef.no](mailto:atgeirr@sintef.no)

#### See also

`Class1`

Definition at line 67 of file `tpFace.h`.

### 29.514.2 Constructor & Destructor Documentation

29.514.2.1 `virtual Go::tpFace::~tpFace ( ) [virtual]`

Destructor.

### 29.514.3 Member Function Documentation

29.514.3.1 `virtual BoundingBox Go::tpFace::boundingBox ( ) [pure virtual]`

The bounding box corresponding to this face.

29.514.3.2 `virtual std::vector<shared_ptr<tpEdge>> Go::tpFace::createInitialEdges ( double degenerate_epsilon = DEFAULT_SPACE_EPSILON ) [pure virtual]`

Compute the edges associated to this face or fetch already existing edges

29.514.3.3 `virtual int Go::tpFace::getId ( ) [pure virtual]`

Return id, default id is -1.

29.514.3.4 `virtual void Go::tpFace::isolateFace ( ) [virtual]`

Remove all adjacency information related to this face.

29.514.3.5 `virtual Point Go::tpFace::normal ( double u, double v ) const [pure virtual]`

Evaluate surface normal.

29.514.3.6 virtual Point Go::tpFace::point ( double *u*, double *v* ) const [pure virtual]

Evaluate point on face.

29.514.3.7 virtual std::vector<shared\_ptr<tpEdge>> Go::tpFace::startEdges ( ) [pure virtual]

Return pointers to first part of all bd cvs.

The documentation for this class was generated from the following file:

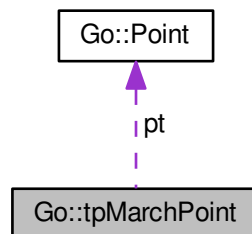
- [topology/include/GoTools/topology/tpFace.h](#)

## 29.515 Go::tpMarchPoint Class Reference

Helper class for marching.

```
#include <tpTopologyTable.h>
```

Collaboration diagram for Go::tpMarchPoint:



### Public Member Functions

- [tpMarchPoint](#) (const Go::Point &p, double pa, double pa2, double d, double ca, int s)
- [bool operator<](#) (const tpMarchPoint &other) const

### Public Attributes

- [Go::Point pt](#)
- [double par](#)
- [double par2](#)
- [double dist](#)
- [double cos\\_ang](#)
- [int status](#)

### 29.515.1 Detailed Description

Helper class for marching.

Definition at line 119 of file tpTopologyTable.h.

### 29.515.2 Constructor & Destructor Documentation

29.515.2.1 **Go::tpMarchPoint::tpMarchPoint** ( **const Go::Point & p**, **double pa**, **double pa2**, **double d**, **double ca**, **int s** ) **[inline]**

Definition at line 128 of file tpTopologyTable.h.

### 29.515.3 Member Function Documentation

29.515.3.1 **bool Go::tpMarchPoint::operator<** ( **const tpMarchPoint & other** ) **const** **[inline]**

Definition at line 132 of file tpTopologyTable.h.

### 29.515.4 Member Data Documentation

29.515.4.1 **double Go::tpMarchPoint::cos\_ang**

Definition at line 126 of file tpTopologyTable.h.

29.515.4.2 **double Go::tpMarchPoint::dist**

Definition at line 125 of file tpTopologyTable.h.

29.515.4.3 **double Go::tpMarchPoint::par**

Definition at line 123 of file tpTopologyTable.h.

29.515.4.4 **double Go::tpMarchPoint::par2**

Definition at line 124 of file tpTopologyTable.h.

29.515.4.5 **Go::Point Go::tpMarchPoint::pt**

Definition at line 122 of file tpTopologyTable.h.

29.515.4.6 int Go::tpMarchPoint::status

Definition at line 127 of file tpTopologyTable.h.

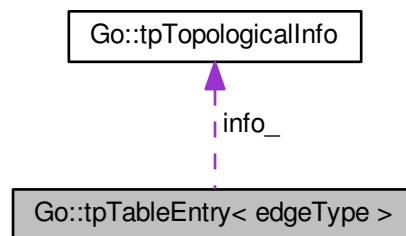
The documentation for this class was generated from the following file:

- [topology/include/GoTools/topology/tpTopologyTable.h](#)

## 29.516 Go::tpTableEntry< edgeType > Class Template Reference

```
#include <tpTopologyTable.h>
```

Collaboration diagram for Go::tpTableEntry< edgeType >:



### Public Member Functions

- [tpTableEntry](#) (edgeType \*e1, edgeType \*e2, [const tpTopologicalInfo &info](#))
- [const tpTopologicalInfo & Info](#) ()

### Public Attributes

- edgeType \* [e1\\_](#)
- edgeType \* [e2\\_](#)
- [tpTopologicalInfo info\\_](#)

### 29.516.1 Detailed Description

```
template<class edgeType>
class Go::tpTableEntry< edgeType >
```

[tpTableEntry](#) - Short description. Detailed description.

Definition at line 103 of file tpTopologyTable.h.

## 29.516.2 Constructor & Destructor Documentation

29.516.2.1 `template<class edgeType > Go::tpTableEntry< edgeType >::tpTableEntry ( edgeType * e1, edgeType * e2, const tpTopologicalInfo & info ) [inline]`

Definition at line 109 of file tpTopologyTable.h.

## 29.516.3 Member Function Documentation

29.516.3.1 `template<class edgeType > const tpTopologicalInfo& Go::tpTableEntry< edgeType >::Info ( ) [inline]`

Definition at line 114 of file tpTopologyTable.h.

## 29.516.4 Member Data Documentation

29.516.4.1 `template<class edgeType > edgeType* Go::tpTableEntry< edgeType >::e1_`

Definition at line 106 of file tpTopologyTable.h.

29.516.4.2 `template<class edgeType > edgeType* Go::tpTableEntry< edgeType >::e2_`

Definition at line 107 of file tpTopologyTable.h.

29.516.4.3 `template<class edgeType > tpTopologicalInfo Go::tpTableEntry< edgeType >::info_`

Definition at line 108 of file tpTopologyTable.h.

The documentation for this class was generated from the following file:

- [topology/include/GoTools/topology/tpTopologyTable.h](#)

## 29.517 Go::tpTolerances Struct Reference

```
#include <tpTolerances.h>
```

### Public Member Functions

- [tpTolerances \(double g, double n, double k, double b\)](#)
- [tpTolerances \(const tpTolerances &tol\)](#)

## Public Attributes

- [double gap](#)  
*Tolerance for when two faces are assumed to be C0 continous.*
- [double neighbour](#)  
*Tolerance for when two faces are assumed to be neighbours.*
- [double kink](#)  
*Tolerance for when two adjacent faces are assumed to be C1 continous.*
- [double bend](#)

### 29.517.1 Detailed Description

[tpTolerances](#) - Tolerances used in adjacency analysis for faces

Definition at line 48 of file tpTolerances.h.

### 29.517.2 Constructor & Destructor Documentation

29.517.2.1 `Go::tpTolerances::tpTolerances ( double g, double n, double k, double b )` `[inline]`

Definition at line 60 of file tpTolerances.h.

29.517.2.2 `Go::tpTolerances::tpTolerances ( const tpTolerances & tol )` `[inline]`

Definition at line 63 of file tpTolerances.h.

### 29.517.3 Member Data Documentation

29.517.3.1 `double Go::tpTolerances::bend`

Tolerance for when two adjacent faces are assumed to have an intentially smooth connection

Definition at line 59 of file tpTolerances.h.

29.517.3.2 `double Go::tpTolerances::gap`

Tolerance for when two faces are assumed to be C0 continous.

Definition at line 52 of file tpTolerances.h.

29.517.3.3 `double Go::tpTolerances::kink`

Tolerance for when two adjacent faces are assumed to be C1 continous.

Definition at line 56 of file tpTolerances.h.

#### 29.517.3.4 double Go::tpTolerances::neighbour

Tolerance for when two faces are assumed to be neighbours.

Definition at line 54 of file tpTolerances.h.

The documentation for this struct was generated from the following file:

- [topology/include/GoTools/topology/tpTolerances.h](#)

## 29.518 Go::tpTopologicalInfo Struct Reference

```
#include <tpTopologyTable.h>
```

### Public Member Functions

- int [BestStatus](#) () const  
*The highest continuity between the two faces.*
- int [WorstStatus](#) () const  
*The lowest continuity between the two faces.*

### Public Attributes

- std::vector< int > [status\\_](#)
- std::vector< std::pair< double, double > > [parameters\\_](#)  
*Parameter intervals limiting the areas of the found state of continuity.*

### 29.518.1 Detailed Description

A structure storing the connectivity information between two adjacent faces. Used with [tpTopologyTable](#)

Definition at line 64 of file tpTopologyTable.h.

### 29.518.2 Member Function Documentation

#### 29.518.2.1 int Go::tpTopologicalInfo::BestStatus ( ) const [inline]

The highest continuity between the two faces.

Definition at line 81 of file tpTopologyTable.h.

#### 29.518.2.2 int Go::tpTopologicalInfo::WorstStatus ( ) const [inline]

The lowest continuity between the two faces.

Definition at line 90 of file tpTopologyTable.h.



### 29.518.3 Member Data Documentation

#### 29.518.3.1 `std::vector< std::pair<double, double> >` `Go::tpTopologicalInfo::parameters_`

Parameter intervals limiting the areas of the found state of continuity.

Definition at line 78 of file `tpTopologyTable.h`.

#### 29.518.3.2 `std::vector<int>` `Go::tpTopologicalInfo::status_`

The status is: 0 : edges join smoothly.  $G^1$ . 1 : edges join, but normals are slightly discontinuous. A kink. 2 : edges join, but the normals are discontinuous.  $G^0$ . 3 : edges almost join. A gap. 4 : edges are totally discontinuous. The minimal `tpTopologicalInfo` has a one-element status vector and a two-element parameters vector

Definition at line 76 of file `tpTopologyTable.h`.

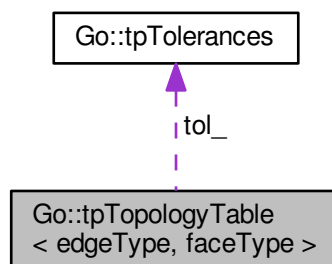
The documentation for this struct was generated from the following file:

- `topology/include/GoTools/topology/tpTopologyTable.h`

## 29.519 `Go::tpTopologyTable< edgeType, faceType >` Class Template Reference

```
#include <tpTopologyTable.h>
```

Collaboration diagram for `Go::tpTopologyTable< edgeType, faceType >`:



## Public Member Functions

- [tpTopologyTable](#) ([double](#) tol\_gap, [double](#) tol\_neighbour, [double](#) tol\_kink, [double](#) tol\_bend)
- [~tpTopologyTable](#) ()
- void [setTolerances](#) ([const](#) [tpTolerances](#) &tol)
  - Changes the tolerances used in constructing the table.*
- [tpTolerances](#) [getTolerances](#) ()
- void [prepareTable](#) ([const](#) [std::vector](#)< [shared\\_ptr](#)< [faceType](#) > > &faces)
- void [constructTable](#) ([const](#) [std::vector](#)< [shared\\_ptr](#)< [faceType](#) > > &faces, [bool](#) test\_orientation=false)
- void [addEntry](#) ([edgeType](#) \*e1, [edgeType](#) \*e2, [double](#) min1, [double](#) max1, [double](#) min2, [double](#) max2, [const](#) [tpTopologicalInfo](#) &info, [int](#) status)
- [std::vector](#)< [edgeType](#) \* > [connectTwins](#) ([edgeType](#) \*e1, [edgeType](#) \*e2, [double](#) min1, [double](#) max1, [double](#) min2, [double](#) max2, [int](#) &status)
- void [BoundaryLoops](#) ([std::vector](#)< [std::vector](#)< [edgeType](#) \* > > &loopvec)
- void [BoundaryLoops](#) ([std::vector](#)< [std::vector](#)< [shared\\_ptr](#)< [edgeType](#) > > > &loopvec)
- void [disjointObjects](#) ([std::vector](#)< [std::vector](#)< [faceType](#) \* > > &grouped\_faces)
- void [cornersAndKinks](#) ([std::vector](#)< [edgeType](#) \* > &vec)
- [const](#) [tpTopologicalInfo](#) & [info](#) ([int](#) n)
  - Returns the raw topological info element.*
- [std::vector](#)< [edgeType](#) \* > [edgesBoundingFace](#) ([faceType](#) \*face) [const](#)
  - Returns all edges bounding a specific face.*
- [const](#) [std::vector](#)< [tpTableEntry](#)< [edgeType](#) > > & [connectedEdges](#) () [const](#)
  - Returns a const reference to a vector representing twin edges.*
- [const](#) void [getInconsistentFaces](#) ([std::vector](#)< [std::pair](#)< [faceType](#) \*, [faceType](#) \* > > &faces)
  - Returns information about face pair with an orientation inconsistency.*
- void [Validate](#) ()
- void [setTable](#) ([const](#) [std::vector](#)< [shared\\_ptr](#)< [faceType](#) > > &faces)
- void [removeFaceFromTable](#) ([shared\\_ptr](#)< [faceType](#) > face)
- void [addFaceToTable](#) ([shared\\_ptr](#)< [faceType](#) > face)
- void [updateTableEntry](#) ([edgeType](#) \*e1, [edgeType](#) \*e2)

## Protected Attributes

- [std::vector](#)< [tpTableEntry](#)< [edgeType](#) > > [table\\_](#)
- [std::vector](#)< [shared\\_ptr](#)< [edgeType](#) > > [edges\\_](#)
- [tpTolerances](#) [tol\\_](#)
- [std::vector](#)< [std::pair](#)< [faceType](#) \*, [faceType](#) \* > > [orientation\\_inconsist\\_](#)

### 29.519.1 Detailed Description

```
template<class edgeType, class faceType>
class Go::tpTopologyTable< edgeType, faceType >
```

[tpTopologyTable](#) - Encapsulates a topology structure. Should be used only for static faces sets.

This class encapsulates a topology structure. It points to external (to this class) surface objects, which are gotten as input to the [PrepareTable\(\)](#) and [ConstructTable\(\)](#) members. It is implemented in terms of [edgeType](#) objects in a half-edge data structure, and many members operate on and/or return pointers to the [edgeType](#) objects owned by the [tpTopologyTable](#) object.

#### Author

Atgeirr F Rasmussen [atgeirr@sintef.no](mailto:atgeirr@sintef.no)

#### See also

[edgeType](#)  
[faceType](#)

Definition at line 161 of file [tpTopologyTable.h](#).

## 29.519.2 Constructor & Destructor Documentation

29.519.2.1 `template<class edgeType , class faceType > Go::tpTopologyTable< edgeType, faceType >::tpTopologyTable ( double tol_gap, double tol_neighbour, double tol_kink, double tol_bend ) [inline]`

Constructor. Detailed description.

Definition at line 179 of file tpTopologyTable.h.

29.519.2.2 `template<class edgeType , class faceType > Go::tpTopologyTable< edgeType, faceType >::~~tpTopologyTable ( ) [inline]`

Destructor. Detailed description.

Definition at line 188 of file tpTopologyTable.h.

## 29.519.3 Member Function Documentation

29.519.3.1 `template<class edgeType , class faceType > void Go::tpTopologyTable< edgeType, faceType >::addEntry ( edgeType * e1, edgeType * e2, double min1, double max1, double min2, double max2, const tpTopologicalInfo & info, int status ) [inline]`

In the AddEntry() function, the min? and max? arguments are the parameter values associated with the endpoints of the curve segments. min1 <= max1 and min2 <= max2. BUT the edges may very well run in different directions, and the point on the first edge corresponding to min1 may be adjacent to either the min2- or the max2- corresponding point on the second edge.

Definition at line 334 of file tpTopologyTable.h.

29.519.3.2 `template<class edgeType , class faceType > void Go::tpTopologyTable< edgeType, faceType >::addFaceToTable ( shared_ptr< faceType > face ) [inline]`

Definition at line 766 of file tpTopologyTable.h.

29.519.3.3 `template<class edgeType , class faceType > void Go::tpTopologyTable< edgeType, faceType >::BoundaryLoops ( std::vector< std::vector< edgeType * > > & loopvec ) [inline]`

Returns all boundary loops. Keep in mind that calling next() on elements on the boundary will not iterate around the boundary, but around the face the edge comes from.

Definition at line 431 of file tpTopologyTable.h.

29.519.3.4 `template<class edgeType , class faceType > void Go::tpTopologyTable< edgeType, faceType >::BoundaryLoops ( std::vector< std::vector< shared_ptr< edgeType > > > & loopvec ) [inline]`

Definition at line 470 of file tpTopologyTable.h.

```
29.519.3.5 template<class edgeType , class faceType > const std::vector<tpTableEntry<edgeType> >&
Go::tpTopologyTable< edgeType, faceType >::connectedEdges () const [inline]
```

Returns a const reference to a vector representing twin edges.

Definition at line 638 of file tpTopologyTable.h.

```
29.519.3.6 template<class edgeType , class faceType > std::vector<edgeType*> Go::tpTopologyTable< edgeType,
faceType >::connectTwins (edgeType * e1, edgeType * e2, double min1, double max1, double min2,
double max2, int & status) [inline]
```

Returns pointer to (vector of) twin edges. Updates topological structure and, if necessary, splits edges.

Definition at line 373 of file tpTopologyTable.h.

```
29.519.3.7 template<class edgeType , class faceType > void Go::tpTopologyTable< edgeType, faceType
>::constructTable (const std::vector< shared_ptr< faceType > > & faces, bool test_orientation = false)
[inline]
```

Definition at line 218 of file tpTopologyTable.h.

```
29.519.3.8 template<class edgeType , class faceType > void Go::tpTopologyTable< edgeType, faceType
>::cornersAndKinks (std::vector< edgeType * > & vec) [inline]
```

Only gives the first of every pair of edges representing a cornering edge or kink edge.

Definition at line 589 of file tpTopologyTable.h.

```
29.519.3.9 template<class edgeType , class faceType > void Go::tpTopologyTable< edgeType, faceType
>::disjointObjects (std::vector< std::vector< faceType * > > & grouped_faces) [inline]
```

Definition at line 523 of file tpTopologyTable.h.

```
29.519.3.10 template<class edgeType , class faceType > std::vector<edgeType*> Go::tpTopologyTable< edgeType,
faceType >::edgesBoundingFace (faceType * face) const [inline]
```

Returns all edges bounding a specific face.

Definition at line 613 of file tpTopologyTable.h.

```
29.519.3.11 template<class edgeType , class faceType > const void Go::tpTopologyTable< edgeType, faceType
>::getInconsistentFaces (std::vector< std::pair< faceType *, faceType * > > & faces) [inline]
```

Returns information about face pair with an orientation inconsistency.

Definition at line 646 of file tpTopologyTable.h.

29.519.3.12 `template<class edgeType , class faceType > tpTolerances Go::tpTopologyTable< edgeType, faceType >::getTolerances ( ) [inline]`

Definition at line 202 of file tpTopologyTable.h.

29.519.3.13 `template<class edgeType , class faceType > const tpTopologicalInfo& Go::tpTopologyTable< edgeType, faceType >::info ( int n ) [inline]`

Returns the raw topological info element.

Definition at line 603 of file tpTopologyTable.h.

29.519.3.14 `template<class edgeType , class faceType > void Go::tpTopologyTable< edgeType, faceType >::prepareTable ( const std::vector< shared_ptr< faceType > > & faces ) [inline]`

Definition at line 209 of file tpTopologyTable.h.

29.519.3.15 `template<class edgeType , class faceType > void Go::tpTopologyTable< edgeType, faceType >::removeFaceFromTable ( shared_ptr< faceType > face ) [inline]`

Definition at line 714 of file tpTopologyTable.h.

29.519.3.16 `template<class edgeType , class faceType > void Go::tpTopologyTable< edgeType, faceType >::setTable ( const std::vector< shared_ptr< faceType > > & faces ) [inline]`

Definition at line 677 of file tpTopologyTable.h.

29.519.3.17 `template<class edgeType , class faceType > void Go::tpTopologyTable< edgeType, faceType >::setTolerances ( const tpTolerances & tol ) [inline]`

Changes the tolerances used in constructing the table.

Definition at line 195 of file tpTopologyTable.h.

29.519.3.18 `template<class edgeType , class faceType > void Go::tpTopologyTable< edgeType, faceType >::updateTableEntry ( edgeType * e1, edgeType * e2 ) [inline]`

Definition at line 875 of file tpTopologyTable.h.

29.519.3.19 `template<class edgeType , class faceType > void Go::tpTopologyTable< edgeType, faceType >::Validate ( ) [inline]`

Definition at line 654 of file tpTopologyTable.h.

### 29.519.4 Member Data Documentation

29.519.4.1 `template<class edgeType , class faceType > std::vector< shared_ptr<edgeType> > Go::tpTopologyTable< edgeType, faceType >::edges_ [protected]`

Definition at line 169 of file tpTopologyTable.h.

29.519.4.2 `template<class edgeType , class faceType > std::vector<std::pair<faceType*, faceType* > > Go::tpTopologyTable< edgeType, faceType >::orientation_inconsist_ [protected]`

Definition at line 172 of file tpTopologyTable.h.

29.519.4.3 `template<class edgeType , class faceType > std::vector< tpTableEntry<edgeType> > Go::tpTopologyTable< edgeType, faceType >::table_ [protected]`

Definition at line 165 of file tpTopologyTable.h.

29.519.4.4 `template<class edgeType , class faceType > tpTolerances Go::tpTopologyTable< edgeType, faceType >::tol_ [protected]`

Definition at line 170 of file tpTopologyTable.h.

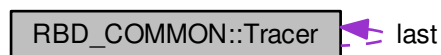
The documentation for this class was generated from the following file:

- [topology/include/GoTools/topology/tpTopologyTable.h](#)

## 29.520 RBD\_COMMON::Tracer Class Reference

```
#include <myexcept.h>
```

Collaboration diagram for RBD\_COMMON::Tracer:



### Public Member Functions

- [Tracer](#) (`const char *`)
- [~Tracer](#) ()
- void [ReName](#) (`const char *`)

## Friends

- class [BaseException](#)
- static [Tracer \\* last](#)
- static void [PrintTrace \(\)](#)
- static void [AddTrace \(\)](#)

### 29.520.1 Detailed Description

Definition at line 66 of file myexcept.h.

### 29.520.2 Constructor & Destructor Documentation

**29.520.2.1** `RBD_COMMON::Tracer::Tracer ( const char * e )` [inline]

Definition at line 101 of file myexcept.h.

**29.520.2.2** `RBD_COMMON::Tracer::~Tracer ( )` [inline]

Definition at line 104 of file myexcept.h.

### 29.520.3 Member Function Documentation

**29.520.3.1** `void Tracer::AddTrace ( )` [static]

Definition at line 116 of file myexcept.cpp.

**29.520.3.2** `void Tracer::PrintTrace ( )` [static]

Definition at line 109 of file myexcept.cpp.

**29.520.3.3** `void RBD_COMMON::Tracer::ReName ( const char * e )` [inline]

Definition at line 106 of file myexcept.h.

### 29.520.4 Friends And Related Function Documentation

**29.520.4.1** `friend class BaseException` [friend]

Definition at line 77 of file myexcept.h.

## 29.520.5 Member Data Documentation

### 29.520.5.1 `Tracer * Tracer::last` `[static]`

Definition at line 76 of file `myexcept.h`.

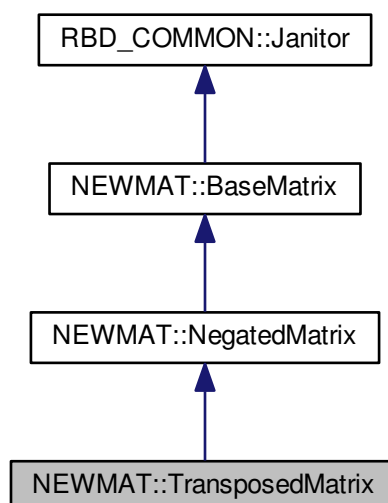
The documentation for this class was generated from the following files:

- [newmat/include/myexcept.h](#)
- [newmat/src/myexcept.cpp](#)

## 29.521 NEWMAT::TransposedMatrix Class Reference

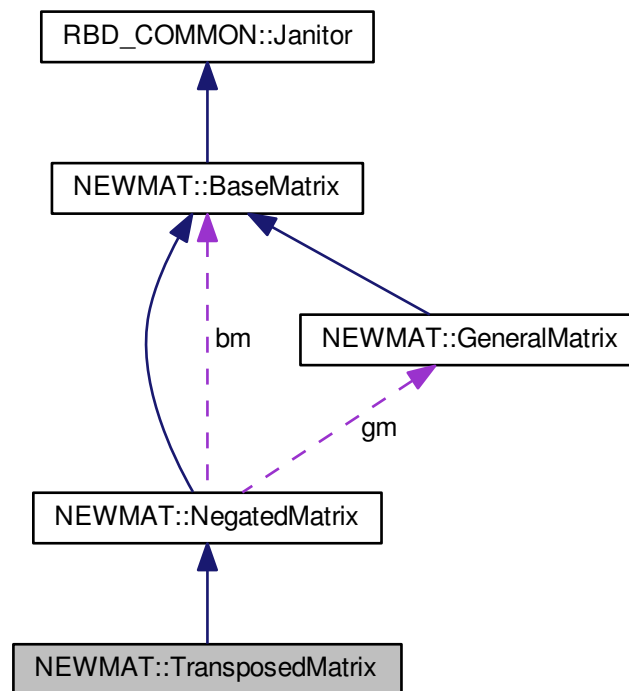
```
#include <newmat.h>
```

Inheritance diagram for `NEWMAT::TransposedMatrix`:





Collaboration diagram for NEWMAT::TransposedMatrix:



### Public Member Functions

- [~TransposedMatrix \(\)](#)
- [GeneralMatrix \\* Evaluate \(MatrixType mt=MatrixTypeUnSp\)](#)
- [MatrixBandWidth BandWidth \(\) const](#)

### Friends

- class [BaseMatrix](#)

### Additional Inherited Members

#### 29.521.1 Detailed Description

Definition at line 1381 of file newmat.h.

#### 29.521.2 Constructor & Destructor Documentation

29.521.2.1 [NEWMAT::TransposedMatrix::~~TransposedMatrix \( \) \[inline\]](#)

Definition at line 1386 of file newmat.h.

### 29.521.3 Member Function Documentation

29.521.3.1 `MatrixBandWidth TransposedMatrix::BandWidth ( ) const` [virtual]

Reimplemented from [NEWMAT::NegatedMatrix](#).

Definition at line 471 of file `newmat4.cpp`.

29.521.3.2 `GeneralMatrix * TransposedMatrix::Evaluate ( MatrixType mt = MatrixTypeUnSp )` [virtual]

Reimplemented from [NEWMAT::NegatedMatrix](#).

Definition at line 278 of file `newmat5.cpp`.

### 29.521.4 Friends And Related Function Documentation

29.521.4.1 `friend class BaseMatrix` [friend]

Definition at line 1384 of file `newmat.h`.

The documentation for this class was generated from the following files:

- [newmat/include/newmat.h](#)
- [newmat/src/newmat4.cpp](#)
- [newmat/src/newmat5.cpp](#)

## 29.522 Go::Triangle Class Reference

```
#include <RegularMesh.h>
```

### Public Member Functions

- [Triangle](#) (int `index`)
- void [setIndex](#) (int `i`)
- int [index](#) () const

### 29.522.1 Detailed Description

Documentation ...

Definition at line 52 of file `RegularMesh.h`.

## 29.522.2 Constructor & Destructor Documentation

### 29.522.2.1 Go::Triangle::Triangle ( int *index* ) [inline], [explicit]

Definition at line 55 of file RegularMesh.h.

## 29.522.3 Member Function Documentation

### 29.522.3.1 int Go::Triangle::index ( ) const [inline]

Definition at line 57 of file RegularMesh.h.

### 29.522.3.2 void Go::Triangle::setIndex ( int *i* ) [inline]

Definition at line 56 of file RegularMesh.h.

The documentation for this class was generated from the following file:

- [gtools-core/include/GoTools/tesselator/RegularMesh.h](#)

## 29.523 hetriang::Triangulation Class Reference

**Triangulation** class for the half-edge data structure with adaption to TTL.

```
#include <ttlTriang.h>
```

### Public Member Functions

- [Triangulation](#) ()
- [~Triangulation](#) ()
- void [createDelaunay](#) (std::vector< [Go::ttlPoint](#) \* >::iterator first, std::vector< [Go::ttlPoint](#) \* >::iterator last)  
*Create Delaunay triangulation from a set of points.*
- [Edge](#) \* [initTwoEnclosingTriangles](#) (std::vector< [Go::ttlPoint](#) \* >::iterator first, std::vector< [Go::ttlPoint](#) \* >::iterator last)↔
- void [swapEdge](#) ([Edge](#) &diagonal)
- [Edge](#) \* [splitTriangle](#) ([Edge](#) &edge, const [Go::ttlPoint](#) &point)
- void [removeTriangle](#) ([Edge](#) &edge)
- void [reverse\\_splitTriangle](#) ([Edge](#) &edge)
- [Dart](#) [createDart](#) ()  
*Create an arbitrary CCW dart.*
- const std::list< [Edge](#) \* > & [getLeadingEdges](#) () const  
*Get a list of "triangles" (one leading half-edge per triangle)*
- int [noTriangles](#) () const
- std::list< [Edge](#) \* > \* [getEdges](#) (bool skip\_boundary\_edges=false) const  
*One half-edge for each arc.*
- void [optimizeDelaunay](#) ()

- Swap edges until the triangulation is Delaunay.*

  - [Edge \\* getConstrainedEdges \(Go::ttlPoint point\) const](#)  
*Returns the start and stop-edge on a constrained edge with spesified nodes.*
  - [bool checkDelaunay \(\) const](#)  
*Check if this triangulation is Delaunay.*
  - [Edge \\* getInteriorNode \(\) const](#)  
*Get an arbitrary interior node (as the source node of the returned edge)*
  - [Edge \\* getBoundaryEdge \(\) const](#)  
*Get an arbitrary boundary edge.*
  - [void printEdges \(std::ofstream &os\) const](#)  
*Print edges for plotting with, e.g., gnuplot.*

### Protected Member Functions

- [void addLeadingEdge \(Edge \\*edge\)](#)
- [bool removeLeadingEdgeFromList \(Edge \\*leadingEdge\)](#)
- [void cleanAll \(\)](#)

### Protected Attributes

- `std::list< Edge \* > leadingEdges_`

### 29.523.1 Detailed Description

[Triangulation](#) class for the half-edge data structure with adaption to TTL.

Definition at line 116 of file `ttlTriang.h`.

### 29.523.2 Constructor & Destructor Documentation

**29.523.2.1** `hetriang::Triangulation::Triangulation ( )` `[inline]`

Definition at line 124 of file `ttlTriang.h`.

**29.523.2.2** `hetriang::Triangulation::~~Triangulation ( )` `[inline]`

Definition at line 125 of file `ttlTriang.h`.

### 29.523.3 Member Function Documentation

**29.523.3.1** `void hetriang::Triangulation::addLeadingEdge ( Edge \* edge )` `[inline]`, `[protected]`

Definition at line 119 of file `ttlTriang.h`.

29.523.3.2 `bool hetriang::Triangulation::checkDelaunay ( ) const`

Check if this triangulation is Delaunay.

29.523.3.3 `void hetriang::Triangulation::cleanAll ( ) [protected]`

29.523.3.4 `Dart hetriang::Triangulation::createDart ( )`

Create an arbitrary CCW dart.

29.523.3.5 `void hetriang::Triangulation::createDelaunay ( std::vector< Go::ttIPoint * >::iterator first, std::vector< Go::ttIPoint * >::iterator last )`

Create Delaunay triangulation from a set of points.

29.523.3.6 `Edge* hetriang::Triangulation::getBoundaryEdge ( ) const`

Get an arbitrary boundary edge.

29.523.3.7 `Edge* hetriang::Triangulation::getConstrainedEdges ( Go::ttIPoint point ) const`

Returns the start and stop-edge on a constrained edge with specified nodes.

29.523.3.8 `std::list<Edge*>* hetriang::Triangulation::getEdges ( bool skip_boundary_edges = false ) const`

One half-edge for each arc.

29.523.3.9 `Edge* hetriang::Triangulation::getInteriorNode ( ) const`

Get an arbitrary interior node (as the source node of the returned edge)

29.523.3.10 `const std::list<Edge*>& hetriang::Triangulation::getLeadingEdges ( ) const [inline]`

Get a list of "triangles" (one leading half-edge per triangle)

Definition at line 148 of file ttITriang.h.

29.523.3.11 `Edge* hetriang::Triangulation::initTwoEnclosingTriangles ( std::vector< Go::ttIPoint * >::iterator first, std::vector< Go::ttIPoint * >::iterator last )`

When using rectangular boundary - loop through all points and expand. (Called from createDelaunay(...) when starting)

29.523.3.12 `int hetriang::Triangulation::noTriangles ( ) const [inline]`

Definition at line 149 of file `ttriang.h`.

29.523.3.13 `void hetriang::Triangulation::optimizeDelaunay ( )`

Swap edges until the triangulation is Delaunay.

29.523.3.14 `void hetriang::Triangulation::printEdges ( std::ofstream & os ) const`

Print edges for plotting with, e.g., `gnuplot`.

29.523.3.15 `bool hetriang::Triangulation::removeLeadingEdgeFromList ( Edge * leadingEdge ) [protected]`

29.523.3.16 `void hetriang::Triangulation::removeTriangle ( Edge & edge )`

29.523.3.17 `void hetriang::Triangulation::reverse_splitTriangle ( Edge & edge )`

29.523.3.18 `Edge* hetriang::Triangulation::splitTriangle ( Edge & edge, const Go::ttlPoint & point )`

29.523.3.19 `void hetriang::Triangulation::swapEdge ( Edge & diagonal )`

## 29.523.4 Member Data Documentation

29.523.4.1 `std::list<Edge*> hetriang::Triangulation::leadingEdges_ [protected]`

Definition at line 118 of file `ttriang.h`.

The documentation for this class was generated from the following file:

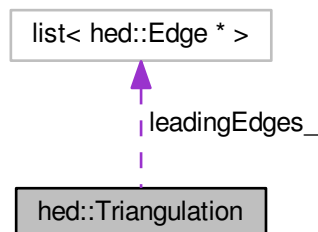
- [compositemodel/include/GoTools/compositemodel/ttriang.h](#)

## 29.524 hed::Triangulation Class Reference

**Triangulation** class for the half-edge data structure with adaption to TTL.

```
#include <HeTriang.h>
```

Collaboration diagram for `hed::Triangulation`:



## Public Member Functions

- [Triangulation](#) ()  
*Default constructor.*
- [Triangulation](#) (const [Triangulation](#) &tr)  
*Copy constructor.*
- [~Triangulation](#) ()  
*Destructor.*
- void [createDelaunay](#) (std::vector< [Node](#) \* >::iterator first, std::vector< [Node](#) \* >::iterator last)  
*Creates a Delaunay triangulation from a set of points.*
- [Edge](#) \* [initTwoEnclosingTriangles](#) (std::vector< [Node](#) \* >::iterator first, std::vector< [Node](#) \* >::iterator last)  
*Creates an initial Delaunay triangulation from two enclosing triangles.*
- void [swapEdge](#) ([Edge](#) &diagonal)  
*Swaps the edge associated with diagonal.*
- [Edge](#) \* [splitTriangle](#) ([Edge](#) &edge, [Node](#) &point)  
*Splits the triangle associated with edge into three new triangles joining at point.*
- void [removeTriangle](#) ([Edge](#) &edge)  
*Removes the boundary triangle associated with edge.*
- void [reverse\\_splitTriangle](#) ([Edge](#) &edge)  
*The reverse operation of removeTriangle.*
- [Dart](#) [createDart](#) ()  
*Creates an arbitrary CCW dart.*
- const list< [Edge](#) \* > & [getLeadingEdges](#) () const  
*Returns a list of "triangles" (one leading half-edge for each triangle)*
- int [noTriangles](#) () const  
*Returns the number of triangles.*
- list< [Edge](#) \* > \* [getEdges](#) (bool skip\_boundary\_edges=false) const  
*Returns a list of half-edges (one half-edge for each arc)*
- void [flagNodes](#) (bool flag) const  
*Sets flag in all the nodes.*
- list< [Node](#) \* > \* [getNodes](#) () const  
*Returns a list of nodes. This function requires TTL\_USE\_NODE\_FLAG to be defined.*
- void [optimizeDelaunay](#) ()  
*Swaps edges until the triangulation is Delaunay (constrained edges are not swapped)*
- bool [checkDelaunay](#) () const  
*Checks if the triangulation is Delaunay.*
- [Edge](#) \* [getInteriorNode](#) () const  
*Returns an arbitrary interior node (as the source node of the returned edge)*
- [Edge](#) \* [getBoundaryEdge](#) () const  
*Returns an arbitrary boundary edge.*
- void [printEdges](#) (std::ofstream &os) const  
*Print edges for plotting with, e.g., gnuplot.*

## Protected Member Functions

- void [addLeadingEdge](#) ([Edge](#) \*edge)
- bool [removeLeadingEdgeFromList](#) ([Edge](#) \*leadingEdge)
- void [cleanAll](#) ()

## Protected Attributes

- list< [Edge](#) \* > [leadingEdges\\_](#)

### 29.524.1 Detailed Description

**Triangulation** class for the half-edge data structure with adaption to TTL.

Definition at line 220 of file HeTriang.h.

### 29.524.2 Constructor & Destructor Documentation

#### 29.524.2.1 `hed::Triangulation::Triangulation ( )` `[inline]`

Default constructor.

Definition at line 230 of file HeTriang.h.

#### 29.524.2.2 `hed::Triangulation::Triangulation ( const Triangulation & tr )` `[inline]`

Copy constructor.

Definition at line 233 of file HeTriang.h.

#### 29.524.2.3 `hed::Triangulation::~~Triangulation ( )` `[inline]`

Destructor.

Definition at line 239 of file HeTriang.h.

### 29.524.3 Member Function Documentation

#### 29.524.3.1 `void hed::Triangulation::addLeadingEdge ( Edge * edge )` `[inline]`, `[protected]`

Definition at line 224 of file HeTriang.h.

#### 29.524.3.2 `bool Triangulation::checkDelaunay ( )` `const`

Checks if the triangulation is Delaunay.

Definition at line 608 of file HeTriang.cpp.

#### 29.524.3.3 `void Triangulation::cleanAll ( )` `[protected]`

Definition at line 363 of file HeTriang.cpp.



#### 29.524.3.4 Dart Triangulation::createDart ( )

Creates an arbitrary CCW dart.

Definition at line 326 of file HeTriang.cpp.

#### 29.524.3.5 void hed::Triangulation::createDelaunay ( std::vector< Node \* >::iterator *first*, std::vector< Node \* >::iterator *last* )

Creates a Delaunay triangulation from a set of points.

#### 29.524.3.6 void hed::Triangulation::flagNodes ( bool *flag* ) const

Sets flag in all the nodes.

#### 29.524.3.7 Edge \* Triangulation::getBoundaryEdge ( ) const

Returns an arbitrary boundary edge.

Definition at line 719 of file HeTriang.cpp.

#### 29.524.3.8 list< Edge \* > \* Triangulation::getEdges ( bool *skip\_boundary\_edges* = false ) const

Returns a list of half-edges (one half-edge for each arc)

Definition at line 423 of file HeTriang.cpp.

#### 29.524.3.9 Edge \* Triangulation::getInteriorNode ( ) const

Returns an arbitrary interior node (as the source node of the returned edge)

Definition at line 681 of file HeTriang.cpp.

#### 29.524.3.10 const list<Edge\*>& hed::Triangulation::getLeadingEdges ( ) const [inline]

Returns a list of "triangles" (one leading half-edge for each triangle)

Definition at line 274 of file HeTriang.h.

#### 29.524.3.11 list<Node\*>\* hed::Triangulation::getNodes ( ) const

Returns a list of nodes. This function requires TTL\_USE\_NODE\_FLAG to be defined.

See also

[Node](#).

**29.524.3.12** `Edge* hed::Triangulation::initTwoEnclosingTriangles ( std::vector< Node * >::iterator first, std::vector< Node * >::iterator last )`

Creates an initial Delaunay triangulation from two enclosing triangles.

**29.524.3.13** `int hed::Triangulation::noTriangles ( ) const` `[inline]`

Returns the number of triangles.

Definition at line 277 of file HeTriang.h.

**29.524.3.14** `void Triangulation::optimizeDelaunay ( )`

Swaps edges until the triangulation is Delaunay (constrained edges are not swapped)

Definition at line 648 of file HeTriang.cpp.

**29.524.3.15** `void hed::Triangulation::printEdges ( std::ofstream & os ) const`

Print edges for plotting with, e.g., gnuplot.

**29.524.3.16** `bool Triangulation::removeLeadingEdgeFromList ( Edge * leadingEdge )` `[protected]`

Definition at line 334 of file HeTriang.cpp.

**29.524.3.17** `void Triangulation::removeTriangle ( Edge & edge )`

Removes the boundary triangle associated with edge.

Definition at line 197 of file HeTriang.cpp.

**29.524.3.18** `void Triangulation::reverse_splitTriangle ( Edge & edge )`

The reverse operation of removeTriangle.

Definition at line 252 of file HeTriang.cpp.

**29.524.3.19** `Edge * Triangulation::splitTriangle ( Edge & edge, Node & point )`

Splits the triangle associated with edge into three new triangles joining at point.

Definition at line 448 of file HeTriang.cpp.

29.524.3.20 void Triangulation::swapEdge ( Edge & *diagonal* )

Swaps the edge associated with diagonal.

Definition at line 542 of file HeTriang.cpp.

#### 29.524.4 Member Data Documentation

29.524.4.1 list<Edge\*> hed::Triangulation::leadingEdges\_ [protected]

Definition at line 223 of file HeTriang.h.

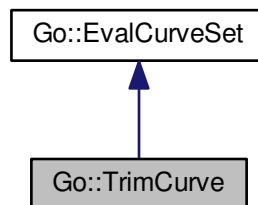
The documentation for this class was generated from the following files:

- [ttl/include/ttl/halfedge/HeTriang.h](#)
- [ttl/src/halfedge/HeTriang.cpp](#)

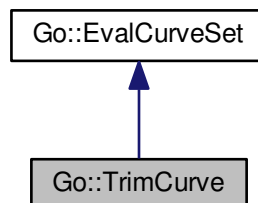
## 29.525 Go::TrimCurve Class Reference

```
#include <TrimCurve.h>
```

Inheritance diagram for Go::TrimCurve:



Collaboration diagram for Go::TrimCurve:



## Public Member Functions

- [TrimCurve](#) ([CurveOnSurface](#) \*bd\_crv)
  - Constructor given the [CurveOnSurface](#) curve representing the trim curve.*
- [TrimCurve](#) ([CurveOnSurface](#) \*bd\_crv, [double](#) start, [double](#) end)
- [TrimCurve](#) ([Point](#) startpt, [Point](#) endpt, [CurveOnSurface](#) \*bd\_crv)
- virtual [~TrimCurve](#) ()
  - virtual destructor ensures safe inheritance*
- [std::vector](#)< [Point](#) > [eval](#) ([double](#) t)
- virtual void [eval](#) ([double](#) t, int n, [std::vector](#)< [std::vector](#)< [Point](#) > > &der)
- virtual [double](#) start ()
- virtual [double](#) end ()
- virtual int dim ()
  - Inherited from [EvalCurveSet::dim\(\)](#).*
- virtual [bool](#) [approximationOK](#) ([double](#) par, [const](#) [std::vector](#)< [Point](#) > &approxpos, [double](#) tol1, [double](#) tol2)
- virtual int [nmbCvs](#) ()

### 29.525.1 Detailed Description

This class represents the curve obtained by projecting a given 3D curve onto a given part of a given 3D surface. Used to improve the accuracy of an already existing trimming curve.

Definition at line 59 of file TrimCurve.h.

### 29.525.2 Constructor & Destructor Documentation

#### 29.525.2.1 Go::TrimCurve::TrimCurve ( [CurveOnSurface](#) \* bd\_crv )

Constructor given the [CurveOnSurface](#) curve representing the trim curve.

#### 29.525.2.2 Go::TrimCurve::TrimCurve ( [CurveOnSurface](#) \* bd\_crv, [double](#) start, [double](#) end )

Constructor given the [CurveOnSurface](#) curve representing the trim curve limited in the parameter domain

#### 29.525.2.3 Go::TrimCurve::TrimCurve ( [Point](#) startpt, [Point](#) endpt, [CurveOnSurface](#) \* bd\_crv )

Constructor given the [CurveOnSurface](#) curve representing the trim curve limited in the geometry space

#### 29.525.2.4 virtual Go::TrimCurve::~~TrimCurve ( ) [virtual]

virtual destructor ensures safe inheritance

### 29.525.3 Member Function Documentation

#### 29.525.3.1 virtual bool Go::TrimCurve::approximationOK ( [double](#) par, [const](#) [std::vector](#)< [Point](#) > & approxpos, [double](#) tol1, [double](#) tol2 ) [virtual]

Inherited from [EvalCurveSet::approximationOK\(\)](#). For this class, the specified tolerances are not used; the internally stored 'epsgeo' value is used as tolerance (this value was specified in the constructor).

## Parameters

<i>par</i>	the parameter at which to check the curve
<i>approxpos</i>	the position we want to check whether or not the curve approximates for parameter 'par'.
<i>tol1</i>	unused
<i>tol2</i>	unused

## Returns

'true' if the curve approximates the point at the parameter, 'false' otherwise.

Implements [Go::EvalCurveSet](#).

**29.525.3.2** `virtual int Go::TrimCurve::dim ( ) [virtual]`

Inherited from [EvalCurveSet::dim\(\)](#).

Implements [Go::EvalCurveSet](#).

**29.525.3.3** `virtual double Go::TrimCurve::end ( ) [virtual]`

End parameter of domain.

## Returns

end parameter of the spline space.

Implements [Go::EvalCurveSet](#).

**29.525.3.4** `std::vector<Point> Go::TrimCurve::eval ( double t ) [virtual]`

Evaluate the curves.

## Parameters

<i>t</i>	parameter in which to evaluate.
----------	---------------------------------

## Returns

the evaluated points for the curve set.

Implements [Go::EvalCurveSet](#).

**29.525.3.5** `virtual void Go::TrimCurve::eval ( double t, int n, std::vector< std::vector< Point > > & der ) [virtual]`

Evaluate the curve derivatives.

## Parameters

<i>t</i>	parameter in which to evaluate.
<i>n</i>	number of derivatives to compute.
<i>der</i>	the evaluated points up to the n'th derivative for the curve set.

Implements [Go::EvalCurveSet](#).

**29.525.3.6** `virtual int Go::TrimCurve::nmbCvs ( ) [virtual]`

The number of curves in the curve set.

## Returns

the number of curves in the curve set.

Implements [Go::EvalCurveSet](#).

**29.525.3.7** `virtual double Go::TrimCurve::start ( ) [virtual]`

Start parameter of domain.

## Returns

start parameter of the spline space.

Implements [Go::EvalCurveSet](#).

The documentation for this class was generated from the following file:

- [gotools-core/include/GoTools/creators/TrimCurve.h](#)

## 29.526 Go::ttlPoint Class Reference

```
#include <ttlPoint.h>
```

### Public Member Functions

- [ttlPoint \(PointIter pnt\\_iter, double x, double y, double z=0.0\)](#)  
*Constructor.*
- [~ttlPoint \(\)](#)  
*Destructor.*
- [const PointIter & pnt\\_iter \(\) const](#)
- [double x \(\) const](#)
- [double y \(\) const](#)
- [double z \(\) const](#)

### 29.526.1 Detailed Description

Utility class for representing a [ftSamplePoint](#) with additional parameter value. Useful when dealing with sets of surfaces treated as one.

Definition at line 51 of file `ttlPoint.h`.

### 29.526.2 Constructor & Destructor Documentation

**29.526.2.1** `Go::ttlPoint::ttlPoint ( PointIter pnt_iter, double x, double y, double z=0.0 )` `[inline]`

Constructor.

Definition at line 56 of file `ttlPoint.h`.

**29.526.2.2** `Go::ttlPoint::~~ttlPoint ( )` `[inline]`

Destructor.

Definition at line 59 of file `ttlPoint.h`.

### 29.526.3 Member Function Documentation

**29.526.3.1** `const PointIter& Go::ttlPoint::pnt_iter ( ) const` `[inline]`

Definition at line 61 of file `ttlPoint.h`.

**29.526.3.2** `double Go::ttlPoint::x ( ) const` `[inline]`

Definition at line 62 of file `ttlPoint.h`.

**29.526.3.3** `double Go::ttlPoint::y ( ) const` `[inline]`

Definition at line 63 of file `ttlPoint.h`.

**29.526.3.4** `double Go::ttlPoint::z ( ) const` `[inline]`

Definition at line 64 of file `ttlPoint.h`.

The documentation for this class was generated from the following file:

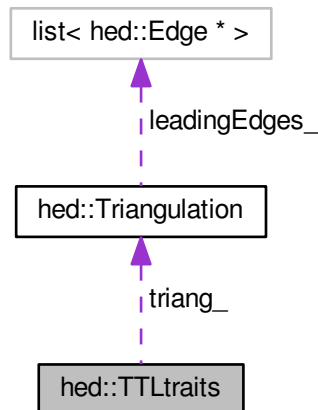
- `compositemodel/include/GoTools/compositemodel/ttlPoint.h`

## 29.527 hed::TTLtraits Struct Reference

**Traits** class (static struct) for the half-edge data structure.

```
#include <HeTraits.h>
```

Collaboration diagram for hed::TTLtraits:



### Public Types

- typedef [double](#) `real_type`

### Static Public Member Functions

#### Geometric Predicates

- static [real\\_type](#) `scalarProduct2d` ([const Dart](#) &v1, [const Dart](#) &v2)
- static [real\\_type](#) `scalarProduct2d` ([const Dart](#) &v, [const Node](#) &p)
- static [real\\_type](#) `crossProduct2d` ([const Dart](#) &v1, [const Dart](#) &v2)
- static [real\\_type](#) `crossProduct2d` ([const Dart](#) &v, [const Node](#) &p)
- static [real\\_type](#) `orient2d` ([const Dart](#) &n1, [const Dart](#) &n2, [const Node](#) &p)
- static [real\\_type](#) `orient2d` ([const Dart](#) &n1, [const Dart](#) &n2, [const Dart](#) &p)

#### Functions for Delaunay Triangulation

- static void `swapEdge` ([Dart](#) &dart)
- static void `splitTriangle` ([Dart](#) &dart, [Node](#) &point)

#### Functions for removing nodes

- static void `reverse_splitTriangle` ([Dart](#) &dart)
- static void `removeBoundaryTriangle` ([Dart](#) &d)



## Static Public Attributes

- static [Triangulation](#) \* `triang_` = NULL

### 29.527.1 Detailed Description

**Traits** class (static struct) for the half-edge data structure.

The member functions are those required by different function templates in the TTL. Documentation is given here to explain what actions should be carried out on the actual data structure as required by the functions in the `ttr` namespace.

The source code of `HeTraits.h` shows how the traits class is implemented for the half-edge data structure.

See also

[Application Programming Interface to TTL \(API\)](#)

Definition at line 70 of file `HeTraits.h`.

### 29.527.2 Member Typedef Documentation

#### 29.527.2.1 typedef double hed::TTLtraits::real\_type

The floating point type used in calculations involving scalar products and cross products.

Definition at line 78 of file `HeTraits.h`.

### 29.527.3 Member Function Documentation

#### 29.527.3.1 static real\_type hed::TTLtraits::crossProduct2d ( const Dart & v1, const Dart & v2 ) [inline], [static]

Cross product between two vectors in the plane represented as darts. The z-component of the cross product is returned.

[ttr\\_util::crossProduct2d](#) can be used.

Definition at line 121 of file `HeTraits.h`.

#### 29.527.3.2 static real\_type hed::TTLtraits::crossProduct2d ( const Dart & v, const Node & p ) [inline], [static]

Cross product between two vectors in the plane. The first vector is represented by a dart `v`, and the second vector has direction from the source node of `v` to the point `p`. The z-component of the cross product is returned.

[ttr\\_util::crossProduct2d](#) can be used.

Definition at line 137 of file `HeTraits.h`.

29.527.3.3 `static real_type hed::TTLtraits::orient2d ( const Dart & n1, const Dart & n2, const Node & p )`  
`[inline],[static]`

Let  $n1$  and  $n2$  be the nodes associated with two darts, and let  $p$  be a point in the plane. Return a positive value if  $n1$ ,  $n2$ , and  $p$  occur in counterclockwise order; a negative value if they occur in clockwise order; and zero if they are collinear.

Definition at line 150 of file HeTraits.h.

29.527.3.4 `static real_type hed::TTLtraits::orient2d ( const Dart & n1, const Dart & n2, const Dart & p )`  
`[inline],[static]`

This is the same predicate as represented with the function above, but with a slightly different interface: The last parameter is given as a dart where the source node of the dart represents a point in the plane. This function is required for constrained triangulation.

Definition at line 166 of file HeTraits.h.

29.527.3.5 `static void hed::TTLtraits::removeBoundaryTriangle ( Dart & d )` `[inline],[static]`

Removes a triangle with an edge at the boundary of the triangulation in the actual data structure

Definition at line 290 of file HeTraits.h.

29.527.3.6 `static void hed::TTLtraits::reverse_splitTriangle ( Dart & dart )` `[inline],[static]`

The reverse operation of [TTLtraits::splitTriangle](#). This function is only required for functions that involve removal of interior nodes; see for example [ttl::removeInteriorNode](#).

Definition at line 281 of file HeTraits.h.

29.527.3.7 `static real_type hed::TTLtraits::scalarProduct2d ( const Dart & v1, const Dart & v2 )` `[inline],`  
`[static]`

Scalar product between two 2D vectors represented as darts.

[ttl\\_util::scalarProduct2d](#) can be used.

Definition at line 93 of file HeTraits.h.

29.527.3.8 `static real_type hed::TTLtraits::scalarProduct2d ( const Dart & v, const Node & p )` `[inline],`  
`[static]`

Scalar product between two 2D vectors. The first vector is represented by a dart  $v$ , and the second vector has direction from the source node of  $v$  to the point  $p$ .

[ttl\\_util::scalarProduct2d](#) can be used.

Definition at line 108 of file HeTraits.h.

29.527.3.9 `static void hed::TTLtraits::splitTriangle ( Dart & dart, Node & point )` `[inline],[static]`

Splits the triangle associated with  $dart$  in the actual data structure into three new triangles joining at  $point$ .

## Parameters

<i>dart</i>	Output: A CCW dart incident with the new node; see the figure.
-------------	----------------------------------------------------------------

Definition at line 257 of file HeTraits.h.

29.527.3.10 `static void hed::TTLtraits::swapEdge ( Dart & dart ) [inline],[static]`

Swaps the edge associated with *dart* in the actual data structure.

## Parameters

<i>dart</i>	Some of the functions require a dart as output. If this is required by the actual function, the dart should be delivered back in a position as seen if it was glued to the edge when swapping (rotating) the edge CCW; see the figure.
-------------	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

## Note

- If the edge is *constrained*, or if it should not be swapped for some other reason, this function need not do the actual swap of the edge.
- Some functions in TTL require that `swapEdge` is implemented such that darts outside the quadrilateral are not affected by the swap.

Definition at line 241 of file HeTraits.h.

## 29.527.4 Member Data Documentation

29.527.4.1 `Triangulation * TTLtraits::triang_ = NULL [static]`

Definition at line 73 of file HeTraits.h.

The documentation for this struct was generated from the following files:

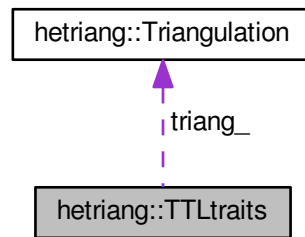
- [ttl/include/ttl/halfedge/HeTraits.h](#)
- [ttl/src/halfedge/HeTriang.cpp](#)

## 29.528 hetriang::TTLtraits Struct Reference

**Traits** class (static struct) for the half-edge data structure.

```
#include <ttlTraits.h>
```

Collaboration diagram for `hetriang::TTLtraits`:



## Public Types

- typedef [double](#) `real_type`

## Static Public Member Functions

### Geometric Predicates

- static [real\\_type](#) `scalarProduct2d` (`const Dart &v1`, `const Dart &v2`)
- static [real\\_type](#) `scalarProduct2d` (`const Dart &v`, `const Go::ttlPoint &p`)
- static [real\\_type](#) `crossProduct2d` (`const Dart &v1`, `const Dart &v2`)
- static [real\\_type](#) `crossProduct2d` (`const Dart &v`, `const Go::ttlPoint &p`)
- static [real\\_type](#) `orient2d` (`const Dart &n1`, `const Dart &n2`, `const Go::ttlPoint &p`)
- static [real\\_type](#) `orient2d` (`const Dart &n1`, `const Dart &n2`, `const Dart &p`)

### Functions for Delaunay Triangulation

- static void `swapEdge` (`Dart &dart`)
- static void `splitTriangle` (`Dart &dart`, `Go::ttlPoint &point`)

### Functions for removing nodes

- static void `reverse_splitTriangle` (`Dart &dart`)
- static void `removeBoundaryTriangle` (`Dart &d`)

## Static Public Attributes

- static [Triangulation](#) \* `triang_`

### 29.528.1 Detailed Description

**Traits** class (static struct) for the half-edge data structure.

The member functions are those required by different function templates in the TTL. Documentation is given here to explain what actions should be carried out on the actual data structure as required by the functions in the ttl namespace.

The source code of [HeTraits.h](#) shows how the traits class is implemented for the half-edge data structure.

See also

[Application Programming Interface to TTL \(API\)](#)

Definition at line 67 of file ttlTraits.h.

### 29.528.2 Member Typedef Documentation

#### 29.528.2.1 typedef double hetriang::TTLtraits::real\_type

The floating point type used in calculations involving scalar products and cross products.

Definition at line 75 of file ttlTraits.h.

### 29.528.3 Member Function Documentation

#### 29.528.3.1 static real\_type hetriang::TTLtraits::crossProduct2d ( const Dart & v1, const Dart & v2 ) [inline], [static]

Cross product between two vectors in the plane represented as darts.

The z-component of the cross product is returned.

[ttl\\_util::crossProduct2d](#) can be used.

Definition at line 102 of file ttlTraits.h.

#### 29.528.3.2 static real\_type hetriang::TTLtraits::crossProduct2d ( const Dart & v, const Go::ttlPoint & p ) [inline], [static]

Cross product between two vectors in the plane.

The first vector is represented by a dart  $v$ , and the second vector has direction from the source node of  $v$  to the point  $p$ .

The z-component of the cross product is returned.

[ttl\\_util::crossProduct2d](#) can be used.

Definition at line 114 of file ttlTraits.h.

**29.528.3.3** `static real_type hetriang::TTLtraits::orient2d ( const Dart & n1, const Dart & n2, const Go::ttlPoint & p )`  
`[inline],[static]`

Let  $n1$  and  $n2$  be the nodes associated with two darts, and let  $p$  be a point in the plane. Return a positive value if  $n1$ ,  $n2$ , and  $p$  occur in counterclockwise order; a negative value if they occur in clockwise order; and zero if they are collinear.

Definition at line 126 of file `ttlTraits.h`.

**29.528.3.4** `static real_type hetriang::TTLtraits::orient2d ( const Dart & n1, const Dart & n2, const Dart & p )`  
`[inline],[static]`

Definition at line 134 of file `ttlTraits.h`.

**29.528.3.5** `static void hetriang::TTLtraits::removeBoundaryTriangle ( Dart & d )` `[inline],[static]`

Removes a triangle with an edge at the boundary of the triangulation in the actual data structure

Definition at line 219 of file `ttlTraits.h`.

**29.528.3.6** `static void hetriang::TTLtraits::reverse_splitTriangle ( Dart & dart )` `[inline],[static]`

The reverse operation of [TTLtraits::splitTriangle](#).

This function is only required for functions that involve removal of interior nodes; see for example [ttl::removeInteriorNode](#).

Definition at line 213 of file `ttlTraits.h`.

**29.528.3.7** `static real_type hetriang::TTLtraits::scalarProduct2d ( const Dart & v1, const Dart & v2 )` `[inline],[static]`

Scalar product between two 2D vectors represented as darts.

[ttl\\_util::scalarProduct2d](#) can be used.

Definition at line 82 of file `ttlTraits.h`.

**29.528.3.8** `static real_type hetriang::TTLtraits::scalarProduct2d ( const Dart & v, const Go::ttlPoint & p )`  
`[inline],[static]`

Scalar product between two 2D vectors.

The first vector is represented by a dart  $v$ , and the second vector has direction from the source node of  $v$  to the point  $p$ .

[ttl\\_util::scalarProduct2d](#) can be used.

Definition at line 93 of file `ttlTraits.h`.

**29.528.3.9** `static void hetriang::TTLtraits::splitTriangle ( Dart & dart, Go::ttlPoint & point )` `[inline],[static]`

Splits the triangle associated with  $dart$  in the actual data structure into three new triangles joining at  $point$ .

## Return values

<i>dart</i>	A CCW dart incident with the new node
-------------	---------------------------------------

Definition at line 197 of file `tTlTraits.h`.

**29.528.3.10** `static void hetriang::TTLtraits::swapEdge ( Dart & dart ) [inline],[static]`

Swaps the edge associated with *dart* in the actual data structure.

## Return values

<i>dart</i>	Some of the functions require a dart as output. If this is required by the actual function, the dart should be delivered back in a position as seen if it was glued to the edge when swapping (rotating) the edge CCW; see the figure.
-------------	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

## Note

Some functions in TTL require that `swapEdge` is implemented such that darts outside the quadrilateral are not affected by the swap.

Definition at line 185 of file `tTlTraits.h`.

**29.528.4 Member Data Documentation**

**29.528.4.1** `Triangulation*` `hetriang::TTLtraits::triang_ [static]`

Definition at line 70 of file `tTlTraits.h`.

The documentation for this struct was generated from the following file:

- `compositemodel/include/GoTools/compositemodel/tTlTraits.h`

**29.529 UnfNodeType Class Reference**

```
#include <PrPathTriangleSeq.h>
```

**Public Member Functions**

- void `printNode ()`

## Public Attributes

- [Go::Vector2D n\\_](#)
- int [origine\\_](#)

### 29.529.1 Detailed Description

[UnfNodeType](#) - used in [PrPathTriangleSeq](#)

Definition at line 70 of file [PrPathTriangleSeq.h](#).

### 29.529.2 Member Function Documentation

29.529.2.1 void [UnfNodeType::printNode](#) ( )

### 29.529.3 Member Data Documentation

29.529.3.1 [Go::Vector2D UnfNodeType::n\\_](#)

Definition at line 73 of file [PrPathTriangleSeq.h](#).

29.529.3.2 int [UnfNodeType::origine\\_](#)

Definition at line 74 of file [PrPathTriangleSeq.h](#).

The documentation for this class was generated from the following file:

- [parametrization/include/GoTools/parametrization/PrPathTriangleSeq.h](#)

## 29.530 Go::CoonsPatchGen::UnKnownError Class Reference

Exception class.

```
#include <CoonsPatchGen.h>
```

### 29.530.1 Detailed Description

Exception class.

Definition at line 63 of file [CoonsPatchGen.h](#).

The documentation for this class was generated from the following file:

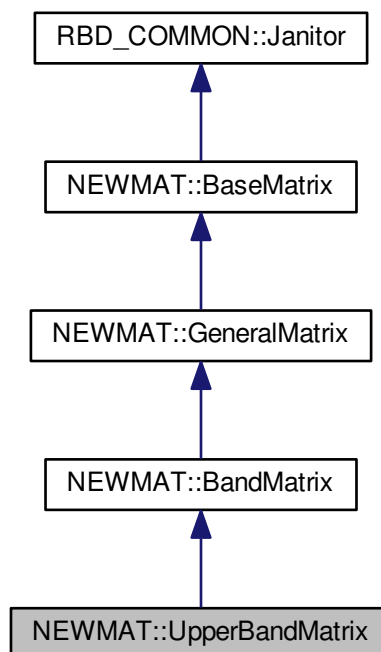
- [gotools-core/include/GoTools/creators/CoonsPatchGen.h](#)



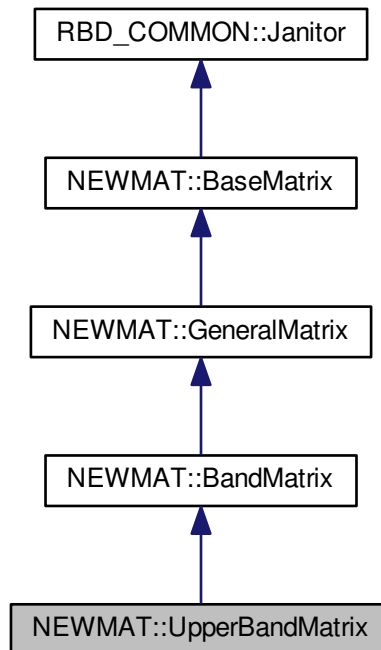
## 29.531 NEWMAT::UpperBandMatrix Class Reference

```
#include <newmat.h>
```

Inheritance diagram for NEWMAT::UpperBandMatrix:



Collaboration diagram for NEWMAT::UpperBandMatrix:



## Public Member Functions

- [UpperBandMatrix \(\)](#)
- [~UpperBandMatrix \(\)](#)
- [UpperBandMatrix \(int n, int ubw\)](#)
- [UpperBandMatrix \(const BaseMatrix &\)](#)
- [void operator= \(const BaseMatrix &\)](#)
- [void operator= \(Real f\)](#)
- [void operator= \(const UpperBandMatrix &m\)](#)
- [MatrixType Type \(\) const](#)
- [UpperBandMatrix \(const UpperBandMatrix &gm\)](#)
- [GeneralMatrix \\* MakeSolver \(\)](#)
- [void Solver \(MatrixColX &, const MatrixColX &\)](#)
- [LogAndSign LogDeterminant \(\) const](#)
- [void ReSize \(int, int, int\)](#)
- [void ReSize \(int n, int ubw\)](#)
- [void ReSize \(const GeneralMatrix &A\)](#)
- [Real & operator\(\) \(int, int\)](#)
- [Real operator\(\) \(int, int\) const](#)
- [Real & element \(int, int\)](#)
- [Real element \(int, int\) const](#)

## Additional Inherited Members

### 29.531.1 Detailed Description

Definition at line 951 of file newmat.h.

### 29.531.2 Constructor & Destructor Documentation

29.531.2.1 `NEWMAT::UpperBandMatrix::UpperBandMatrix ( )` `[inline]`

Definition at line 955 of file newmat.h.

29.531.2.2 `NEWMAT::UpperBandMatrix::~~UpperBandMatrix ( )` `[inline]`

Definition at line 956 of file newmat.h.

29.531.2.3 `NEWMAT::UpperBandMatrix::UpperBandMatrix ( int n, int ubw )` `[inline]`

Definition at line 957 of file newmat.h.

29.531.2.4 `NEWMAT::UpperBandMatrix::UpperBandMatrix ( const BaseMatrix & )`

29.531.2.5 `NEWMAT::UpperBandMatrix::UpperBandMatrix ( const UpperBandMatrix & gm )` `[inline]`

Definition at line 965 of file newmat.h.

### 29.531.3 Member Function Documentation

29.531.3.1 `Real& NEWMAT::UpperBandMatrix::element ( int , int )`

29.531.3.2 `Real NEWMAT::UpperBandMatrix::element ( int , int ) const`

29.531.3.3 `LogAndSign UpperBandMatrix::LogDeterminant ( ) const` `[virtual]`

Reimplemented from [NEWMAT::BandMatrix](#).

Definition at line 412 of file bandmat.cpp.

29.531.3.4 `GeneralMatrix* NEWMAT::UpperBandMatrix::MakeSolver ( )` `[inline],[virtual]`

Reimplemented from [NEWMAT::BandMatrix](#).

Definition at line 966 of file newmat.h.

29.531.3.5 `Real& NEWMAT::UpperBandMatrix::operator()( int, int )`

29.531.3.6 `Real NEWMAT::UpperBandMatrix::operator()( int, int ) const`

29.531.3.7 `void NEWMAT::UpperBandMatrix::operator= ( const BaseMatrix & )`

29.531.3.8 `void NEWMAT::UpperBandMatrix::operator= ( Real f ) [inline]`

Definition at line 961 of file newmat.h.

29.531.3.9 `void NEWMAT::UpperBandMatrix::operator= ( const UpperBandMatrix & m ) [inline]`

Definition at line 962 of file newmat.h.

29.531.3.10 `void NEWMAT::UpperBandMatrix::ReSize ( int, int, int ) [virtual]`

Reimplemented from [NEWMAT::BandMatrix](#).

29.531.3.11 `void NEWMAT::UpperBandMatrix::ReSize ( int n, int ubw ) [inline]`

Definition at line 970 of file newmat.h.

29.531.3.12 `void NEWMAT::UpperBandMatrix::ReSize ( const GeneralMatrix & A ) [inline],[virtual]`

Reimplemented from [NEWMAT::BandMatrix](#).

Definition at line 972 of file newmat.h.

29.531.3.13 `void UpperBandMatrix::Solver ( MatrixColX & mcout, const MatrixColX & mcin ) [virtual]`

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 353 of file bandmat.cpp.

29.531.3.14 `MatrixType UpperBandMatrix::Type ( ) const [virtual]`

Reimplemented from [NEWMAT::BandMatrix](#).

Definition at line 394 of file newmat4.cpp.

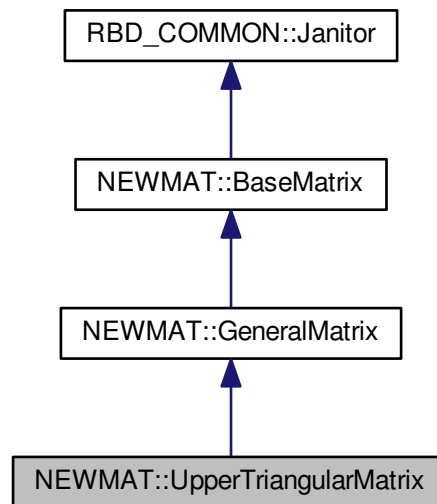
The documentation for this class was generated from the following files:

- [newmat/include/newmat.h](#)
- [newmat/src/bandmat.cpp](#)
- [newmat/src/newmat4.cpp](#)

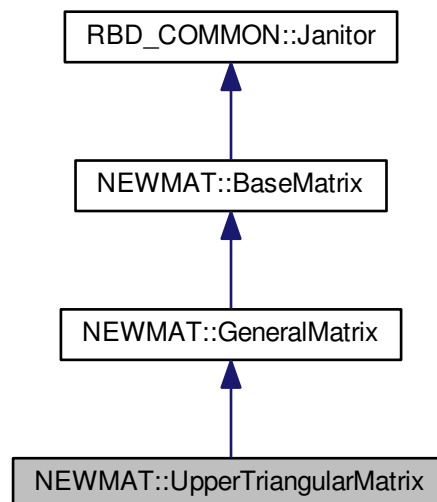
## 29.532 NEWMAT::UpperTriangularMatrix Class Reference

```
#include <newmat.h>
```

Inheritance diagram for NEWMAT::UpperTriangularMatrix:



Collaboration diagram for NEWMAT::UpperTriangularMatrix:



## Public Member Functions

- [UpperTriangularMatrix](#) ()
- [~UpperTriangularMatrix](#) ()
- [UpperTriangularMatrix](#) (ArrayLengthSpecifier)
- void [operator=](#) (const BaseMatrix &)
- void [operator=](#) (const UpperTriangularMatrix &m)
- [UpperTriangularMatrix](#) (const BaseMatrix &)
- [UpperTriangularMatrix](#) (const UpperTriangularMatrix &gm)
- void [operator=](#) (Real f)
- [Real](#) & [operator\(\)](#) (int, int)
- [Real](#) & [element](#) (int, int)
- [Real](#) [operator\(\)](#) (int, int) const
- [Real](#) [element](#) (int, int) const
- [MatrixType](#) [Type](#) () const
- [GeneralMatrix](#) \* [MakeSolver](#) ()
- void [Solver](#) (MatrixColX &, const MatrixColX &)
- [LogAndSign](#) [LogDeterminant](#) () const
- [Real](#) [Trace](#) () const
- void [GetRow](#) (MatrixRowCol &)
- void [GetCol](#) (MatrixRowCol &)
- void [GetCol](#) (MatrixColX &)
- void [RestoreCol](#) (MatrixRowCol &)
- void [RestoreCol](#) (MatrixColX &c)
- void [NextRow](#) (MatrixRowCol &)
- void [ReSize](#) (int)
- void [ReSize](#) (const GeneralMatrix &A)
- [MatrixBandWidth](#) [BandWidth](#) () const

## Additional Inherited Members

### 29.532.1 Detailed Description

Definition at line 670 of file newmat.h.

### 29.532.2 Constructor & Destructor Documentation

29.532.2.1 `NEWMAT::UpperTriangularMatrix::UpperTriangularMatrix ( )` [`inline`]

Definition at line 674 of file newmat.h.

29.532.2.2 `NEWMAT::UpperTriangularMatrix::~~UpperTriangularMatrix ( )` [`inline`]

Definition at line 675 of file newmat.h.

29.532.2.3 `NEWMAT::UpperTriangularMatrix::UpperTriangularMatrix ( ArrayLengthSpecifier )`

29.532.2.4 `NEWMAT::UpperTriangularMatrix::UpperTriangularMatrix ( const BaseMatrix & )`

29.532.2.5 `NEWMAT::UpperTriangularMatrix::UpperTriangularMatrix ( const UpperTriangularMatrix & gm )`  
[inline]

Definition at line 681 of file newmat.h.

### 29.532.3 Member Function Documentation

29.532.3.1 `MatrixBandWidth UpperTriangularMatrix::BandWidth ( ) const` [virtual]

Reimplemented from [NEWMAT::BaseMatrix](#).

Definition at line 406 of file newmat4.cpp.

29.532.3.2 `Real& NEWMAT::UpperTriangularMatrix::element ( int , int )`

29.532.3.3 `Real NEWMAT::UpperTriangularMatrix::element ( int , int ) const`

29.532.3.4 `void NEWMAT::UpperTriangularMatrix::GetCol ( MatrixRowCol & )` [virtual]

Implements [NEWMAT::GeneralMatrix](#).

29.532.3.5 `void NEWMAT::UpperTriangularMatrix::GetCol ( MatrixCoIX & )` [virtual]

Implements [NEWMAT::GeneralMatrix](#).

29.532.3.6 `void UpperTriangularMatrix::GetRow ( MatrixRowCol & mrc )` [virtual]

Implements [NEWMAT::GeneralMatrix](#).

Definition at line 272 of file newmat3.cpp.

29.532.3.7 `LogAndSign UpperTriangularMatrix::LogDeterminant ( ) const` [virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 664 of file newmat8.cpp.

29.532.3.8 `GeneralMatrix* NEWMAT::UpperTriangularMatrix::MakeSolver ( )` [inline],[virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 692 of file newmat.h.

29.532.3.9 void UpperTriangularMatrix::NextRow ( MatrixRowCol & mrc ) [virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 329 of file newmat3.cpp.

29.532.3.10 Real& NEWMAT::UpperTriangularMatrix::operator() ( int , int )

29.532.3.11 Real NEWMAT::UpperTriangularMatrix::operator() ( int , int ) const

29.532.3.12 void NEWMAT::UpperTriangularMatrix::operator= ( const BaseMatrix & )

29.532.3.13 void NEWMAT::UpperTriangularMatrix::operator= ( const UpperTriangularMatrix & m ) [inline]

Definition at line 678 of file newmat.h.

29.532.3.14 void NEWMAT::UpperTriangularMatrix::operator= ( Real f ) [inline]

Definition at line 682 of file newmat.h.

29.532.3.15 void NEWMAT::UpperTriangularMatrix::ReSize ( int )

29.532.3.16 void UpperTriangularMatrix::ReSize ( const GeneralMatrix & A ) [virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 303 of file newmat4.cpp.

29.532.3.17 void NEWMAT::UpperTriangularMatrix::RestoreCol ( MatrixRowCol & ) [virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

29.532.3.18 void NEWMAT::UpperTriangularMatrix::RestoreCol ( MatrixColX & c ) [inline],[virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 700 of file newmat.h.

29.532.3.19 void UpperTriangularMatrix::Solver ( MatrixColX & mcout, const MatrixColX & mcin ) [virtual]

Reimplemented from [NEWMAT::GeneralMatrix](#).

Definition at line 51 of file newmat7.cpp.



29.532.3.20 **Real** UpperTriangularMatrix::Trace ( ) const [virtual]

Reimplemented from [NEWMAT::BaseMatrix](#).

Definition at line 572 of file newmat8.cpp.

29.532.3.21 **MatrixType** UpperTriangularMatrix::Type ( ) const [virtual]

Implements [NEWMAT::GeneralMatrix](#).

Definition at line 387 of file newmat4.cpp.

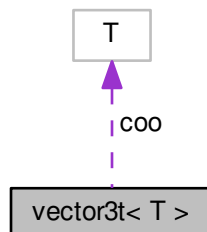
The documentation for this class was generated from the following files:

- [newmat/include/newmat.h](#)
- [newmat/src/newmat3.cpp](#)
- [newmat/src/newmat4.cpp](#)
- [newmat/src/newmat7.cpp](#)
- [newmat/src/newmat8.cpp](#)

## 29.533 vector3t< T > Class Template Reference

```
#include <jonvec.h>
```

Collaboration diagram for vector3t< T >:



## Public Member Functions

- [vector3t](#) ()
- [vector3t](#) (const T a, const T b, const T c)
- [vector3t](#) (const float \*a)
- [vector3t](#) (const double \*a)
- [~vector3t](#) (void)
- [const T \\* raw](#) (void) const
- [T x](#) (void) const
- [T y](#) (void) const
- [T z](#) (void) const
- [T & x](#) (void)
- [T & y](#) (void)
- [T & z](#) (void)
- void [setx](#) (const T a)
- void [sety](#) (const T a)
- void [setz](#) (const T a)
- [bool operator==](#) (const vector3t &v) const
- [bool operator<](#) (const vector3t &v) const
- [bool operator>](#) (const vector3t &v) const
- [bool operator!=](#) (const vector3t &v) const
- [const vector3t operator+](#) (const vector3t &v) const
- [vector3t & operator+=](#) (const vector3t &v)
- [vector3t & operator-=](#) (const vector3t &v)
- [vector3t & operator-=](#) (const T &d)
- [vector3t & operator+=](#) (const T &d)
- [vector3t & operator\\*=](#) (const double x)
- [vector3t & operator/=](#) (const double x)
- [const vector3t operator-](#) (const vector3t &v) const
- [const vector3t operator-](#) (const T &d) const
- [const vector3t operator-](#) (void) const
- [T operator\\*](#) (const vector3t &v) const
- [vector3t operator/](#) (const vector3t &v) const
- [const vector3t & clamp](#) (const vector3t &v0, const vector3t &v1)
- void [min](#) (const vector3t &v)
- void [max](#) (const vector3t &v)
- [vector3t & conv](#) (const vector3t &v)
- [vector3t reciprocal](#) (void) const
- [vector3t & rescale](#) (const vector3t &mi, const vector3t &ma, const vector3t &new\_mi, const vector3t &new\_ma)
- [T length\\_squared](#) (void) const
- [T length](#) (void) const
- void [normalize](#) (void)
- [vector3t normalized](#) (void) const
- void [print](#) (void) const
- void [print2](#) (void) const
- void [print3](#) (void) const
- [T min\\_coo](#) (void) const
- [T max\\_coo](#) (void) const
- [bool dequal](#) (const vector3t< T > &v) const
- [bool dequal2](#) (const vector3t< T > &v) const
- [bool dequal3](#) (const vector3t< T > &v) const
- [bool dequal\\_2d](#) (const vector3t< T > &v) const
- void [rotate\\_xy](#) (const double cosa, const double sina)
- void [rotate\\_xz](#) (const double cosa, const double sina)
- void [rotate\\_yz](#) (const double cosa, const double sina)

## Public Attributes

- T `coo` [3]

## Friends

- `const` friend `vector3t operator*` (`const T &a`, `const vector3t &v`)
- `const` friend `vector3t operator+` (`const double &a`, `const vector3t &v`)
- `const` friend `vector3t operator+` (`const float &a`, `const vector3t &v`)
- T `cosangle` (`const vector3t &v0`, `const vector3t &v1`)
- `vector3t conv2` (`const vector3t &v1`, `const vector3t &v2`)

### 29.533.1 Detailed Description

```
template<typename T>
class vector3t< T >
```

Definition at line 74 of file jonvec.h.

### 29.533.2 Constructor & Destructor Documentation

29.533.2.1 `template<typename T> vector3t< T >::vector3t ( )` [`inline`]

Definition at line 79 of file jonvec.h.

29.533.2.2 `template<typename T> vector3t< T >::vector3t ( const T a, const T b, const T c )` [`inline`]

Definition at line 85 of file jonvec.h.

29.533.2.3 `template<typename T> vector3t< T >::vector3t ( const float * a )` [`inline`]

Definition at line 91 of file jonvec.h.

29.533.2.4 `template<typename T> vector3t< T >::vector3t ( const double * a )` [`inline`]

Definition at line 97 of file jonvec.h.

29.533.2.5 `template<typename T> vector3t< T >::~vector3t ( void )` [`inline`]

Definition at line 116 of file jonvec.h.

### 29.533.3 Member Function Documentation

**29.533.3.1** `template<typename T> const vector3t& vector3t<T>::clamp ( const vector3t<T> & v0, const vector3t<T> & v1 )` `[inline]`

Definition at line 289 of file jonvec.h.

**29.533.3.2** `template<typename T> vector3t& vector3t<T>::conv ( const vector3t<T> & v )` `[inline]`

Definition at line 340 of file jonvec.h.

**29.533.3.3** `template<typename T> bool vector3t<T>::dequal ( const vector3t<T> & v ) const` `[inline]`

Definition at line 424 of file jonvec.h.

**29.533.3.4** `template<typename T> bool vector3t<T>::dequal2 ( const vector3t<T> & v ) const` `[inline]`

Definition at line 429 of file jonvec.h.

**29.533.3.5** `template<typename T> bool vector3t<T>::dequal3 ( const vector3t<T> & v ) const` `[inline]`

Definition at line 437 of file jonvec.h.

**29.533.3.6** `template<typename T> bool vector3t<T>::dequal_2d ( const vector3t<T> & v ) const` `[inline]`

Definition at line 445 of file jonvec.h.

**29.533.3.7** `template<typename T> T vector3t<T>::length ( void ) const` `[inline]`

Definition at line 375 of file jonvec.h.

**29.533.3.8** `template<typename T> T vector3t<T>::length_squared ( void ) const` `[inline]`

Definition at line 370 of file jonvec.h.

**29.533.3.9** `template<typename T> void vector3t<T>::max ( const vector3t<T> & v )` `[inline]`

Definition at line 317 of file jonvec.h.

**29.533.3.10** `template<typename T> T vector3t<T>::max_coo ( void ) const` `[inline]`

Definition at line 419 of file jonvec.h.

29.533.3.11 `template<typename T> void vector3t< T >::min ( const vector3t< T > & v ) [inline]`

Definition at line 311 of file jonvec.h.

29.533.3.12 `template<typename T> T vector3t< T >::min_coo ( void ) const [inline]`

Definition at line 414 of file jonvec.h.

29.533.3.13 `template<typename T> void vector3t< T >::normalize ( void ) [inline]`

Definition at line 381 of file jonvec.h.

29.533.3.14 `template<typename T> vector3t vector3t< T >::normalized ( void ) const [inline]`

Definition at line 389 of file jonvec.h.

29.533.3.15 `template<typename T> bool vector3t< T >::operator!=( const vector3t< T > & v ) const [inline]`

Definition at line 172 of file jonvec.h.

29.533.3.16 `template<typename T> T vector3t< T >::operator*( const vector3t< T > & v ) const [inline]`

Definition at line 269 of file jonvec.h.

29.533.3.17 `template<typename T> vector3t& vector3t< T >::operator*=( const double x ) [inline]`

Definition at line 222 of file jonvec.h.

29.533.3.18 `template<typename T> const vector3t vector3t< T >::operator+ ( const vector3t< T > & v ) const [inline]`

Definition at line 183 of file jonvec.h.

29.533.3.19 `template<typename T> vector3t& vector3t< T >::operator+=( const vector3t< T > & v ) [inline]`

Definition at line 188 of file jonvec.h.

29.533.3.20 `template<typename T> vector3t& vector3t< T >::operator+=( const T & d ) [inline]`

Definition at line 215 of file jonvec.h.

29.533.3.21 `template<typename T> const vector3t vector3t< T >::operator-( const vector3t< T > & v ) const` `[inline]`

Definition at line 235 of file jonvec.h.

29.533.3.22 `template<typename T> const vector3t vector3t< T >::operator-( const T & d ) const` `[inline]`

Definition at line 241 of file jonvec.h.

29.533.3.23 `template<typename T> const vector3t vector3t< T >::operator-( void ) const` `[inline]`

Definition at line 246 of file jonvec.h.

29.533.3.24 `template<typename T> vector3t& vector3t< T >::operator==( const vector3t< T > & v )` `[inline]`

Definition at line 201 of file jonvec.h.

29.533.3.25 `template<typename T> vector3t& vector3t< T >::operator==( const T & d )` `[inline]`

Definition at line 208 of file jonvec.h.

29.533.3.26 `template<typename T> vector3t vector3t< T >::operator/( const vector3t< T > & v ) const` `[inline]`

Definition at line 274 of file jonvec.h.

29.533.3.27 `template<typename T> vector3t& vector3t< T >::operator/=( const double x )` `[inline]`

Definition at line 229 of file jonvec.h.

29.533.3.28 `template<typename T> bool vector3t< T >::operator<( const vector3t< T > & v ) const` `[inline]`

Definition at line 160 of file jonvec.h.

29.533.3.29 `template<typename T> bool vector3t< T >::operator==( const vector3t< T > & v ) const` `[inline]`

Definition at line 150 of file jonvec.h.

29.533.3.30 `template<typename T> bool vector3t< T >::operator>( const vector3t< T > & v ) const` `[inline]`

Definition at line 164 of file jonvec.h.

29.533.3.31 `template<typename T> void vector3t< T >::print ( void ) const [inline]`

Definition at line 394 of file jonvec.h.

29.533.3.32 `template<typename T> void vector3t< T >::print2 ( void ) const [inline]`

Definition at line 401 of file jonvec.h.

29.533.3.33 `template<typename T> void vector3t< T >::print3 ( void ) const [inline]`

Definition at line 408 of file jonvec.h.

29.533.3.34 `template<typename T> const T* vector3t< T >::raw ( void ) const [inline]`

Definition at line 124 of file jonvec.h.

29.533.3.35 `template<typename T> vector3t vector3t< T >::reciprocal ( void ) const [inline]`

Definition at line 356 of file jonvec.h.

29.533.3.36 `template<typename T> vector3t& vector3t< T >::rescale ( const vector3t< T > & mi, const vector3t< T > & ma, const vector3t< T > & new_mi, const vector3t< T > & new_ma ) [inline]`

Definition at line 363 of file jonvec.h.

29.533.3.37 `template<typename T> void vector3t< T >::rotate_xy ( const double cosa, const double sina ) [inline]`

Definition at line 451 of file jonvec.h.

29.533.3.38 `template<typename T> void vector3t< T >::rotate_xz ( const double cosa, const double sina ) [inline]`

Definition at line 460 of file jonvec.h.

29.533.3.39 `template<typename T> void vector3t< T >::rotate_yz ( const double cosa, const double sina ) [inline]`

Definition at line 468 of file jonvec.h.

29.533.3.40 `template<typename T> void vector3t< T >::setx ( const T a ) [inline]`

Definition at line 143 of file jonvec.h.

29.533.3.41 `template<typename T> void vector3t< T >::sety ( const T a ) [inline]`

Definition at line 144 of file jonvec.h.

29.533.3.42 `template<typename T> void vector3t< T >::setz ( const T a ) [inline]`

Definition at line 145 of file jonvec.h.

29.533.3.43 `template<typename T> T vector3t< T >::x ( void ) const [inline]`

Definition at line 133 of file jonvec.h.

29.533.3.44 `template<typename T> T& vector3t< T >::x ( void ) [inline]`

Definition at line 136 of file jonvec.h.

29.533.3.45 `template<typename T> T vector3t< T >::y ( void ) const [inline]`

Definition at line 134 of file jonvec.h.

29.533.3.46 `template<typename T> T& vector3t< T >::y ( void ) [inline]`

Definition at line 137 of file jonvec.h.

29.533.3.47 `template<typename T> T vector3t< T >::z ( void ) const [inline]`

Definition at line 135 of file jonvec.h.

29.533.3.48 `template<typename T> T& vector3t< T >::z ( void ) [inline]`

Definition at line 138 of file jonvec.h.

## 29.533.4 Friends And Related Function Documentation

29.533.4.1 `template<typename T> vector3t conv2 ( const vector3t< T > & v1, const vector3t< T > & v2 ) [friend]`

Definition at line 348 of file jonvec.h.

29.533.4.2 `template<typename T> T cosangle ( const vector3t< T > & v0, const vector3t< T > & v1 ) [friend]`

Definition at line 284 of file jonvec.h.



29.533.4.3 `template<typename T> const friend vector3t operator* ( const T & a, const vector3t< T > & v )`  
[friend]

Definition at line 254 of file jonvec.h.

29.533.4.4 `template<typename T> const friend vector3t operator+ ( const double & a, const vector3t< T > & v )`  
[friend]

Definition at line 260 of file jonvec.h.

29.533.4.5 `template<typename T> const friend vector3t operator+ ( const float & a, const vector3t< T > & v )`  
[friend]

Definition at line 264 of file jonvec.h.

## 29.533.5 Member Data Documentation

29.533.5.1 `template<typename T> T vector3t< T >::coo[3]`

Definition at line 77 of file jonvec.h.

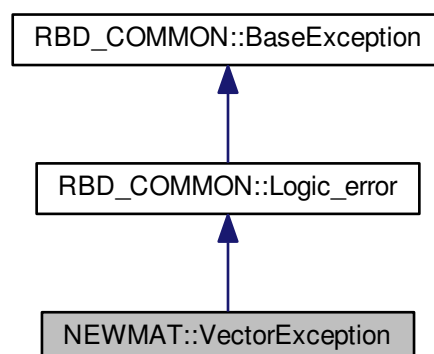
The documentation for this class was generated from the following file:

- [sisl/examples/viewer/include/jonvec.h](#)

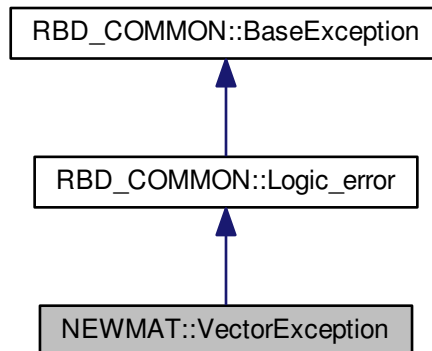
## 29.534 NEWMAT::VectorException Class Reference

```
#include <newmat.h>
```

Inheritance diagram for NEWMAT::VectorException:



Collaboration diagram for NEWMAT::VectorException:



### Public Member Functions

- [VectorException](#) ()
- [VectorException](#) (const [GeneralMatrix](#) &A)

### Static Public Attributes

- static unsigned long [Select](#)

### Additional Inherited Members

#### 29.534.1 Detailed Description

Definition at line 1647 of file `newmat.h`.

#### 29.534.2 Constructor & Destructor Documentation

##### 29.534.2.1 `VectorException::VectorException ( )`

Definition at line 127 of file `newmatex.cpp`.

##### 29.534.2.2 `VectorException::VectorException ( const GeneralMatrix & A )`

Definition at line 134 of file `newmatex.cpp`.

### 29.534.3 Member Data Documentation

#### 29.534.3.1 unsigned long VectorException::Select [static]

Definition at line 1650 of file newmat.h.

The documentation for this class was generated from the following files:

- newmat/include/newmat.h
- newmat/src/newmatex.cpp

## 29.535 Go::Vertex Class Reference

The vertex class represents the vertex entity in a boundary represented solid or face set.

```
#include <Vertex.h>
```

### Public Member Functions

- [Vertex](#) ([Point](#) vertex\_point)
  - Constructor. Give the geometric position of the vertex.*
- [Vertex](#) ([Point](#) vertex\_point, [std::vector](#)< [ftEdge](#) \* > edges)
- [Vertex](#) ([Point](#) vertex\_point, [ftEdge](#) \*edges)
- [Vertex](#) ([ftEdge](#) \*edge, [bool](#) at\_start)
- [~Vertex](#) ()
  - Destructor.*
- void [joinVertex](#) ([shared\\_ptr](#)< [Vertex](#) > other)
- void [addEdge](#) ([ftEdge](#) \*edge)
  - Add a new edge to this vertex. Used in topology build.*
- void [removeEdge](#) ([ftEdge](#) \*edge)
- void [disconnectTwin](#) ([ftEdge](#) \*edge)
- [std::vector](#)< [ftEdge](#) \* > [allEdges](#) () const
  - Returns all edges meeting in this vertex including twins.*
- [std::vector](#)< [ftEdge](#) \* > [uniqueEdges](#) ()
- [std::vector](#)< [ftEdge](#) \* > [uniqueEdges](#) ([Body](#) \*bd)
- int [nmbUniqueEdges](#) ()
- int [nmbUniqueEdges](#) ([Body](#) \*bd)
- [std::vector](#)< [ftEdge](#) \* > [freeEdges](#) ()
  - Get edges which are not associated a face.*
- [ftEdge](#) \* [getEdge](#) (int idx)
- [Point](#) [getVertexPoint](#) ()
  - Get the geometrical position associated to this vertex.*
- void [setVertexPoint](#) ([Point](#) vertex\_point)
  - Set the geometrical position associated to this vertex.*
- [std::vector](#)< [ftSurface](#) \* > [faces](#) () const
  - Get all faces meeting in this vertex.*
- [std::vector](#)< [ftSurface](#) \* > [faces](#) ([Body](#) \*bd) const
  - Get all faces belonging to a given body meeting in this vertex.*
- [std::vector](#)< [std::pair](#)< [ftSurface](#) \*, [Point](#) > > [getFaces](#) ()

- `std::vector< std::pair< ftSurface *, Point > > getFaces (Body *bd)`
- `void averageVertexPos ()`  
*Average corners of spline surfaces corresponding to this vertex.*
- `Point getFacePar (ftSurface *face)`  
*Get parameter of associated face corresponding to vertex.*
- `std::vector< Body * > getBodies ()`  
*Get all bodies meeting in this vertex.*
- `double getDist (shared_ptr< Vertex > other_point)`  
*The distance between this vertex and another vertex.*
- `bool hasEdge (ftEdge *edge) const`  
*Check if the vertex is connected to the given edge.*
- `bool hasFace (ftSurface *face) const`  
*Check if the vertex is adjacent to the given face.*
- `bool hasEdgeSingle (ftEdge *edge) const`
- `bool meetInVertex (ftEdge *e1, ftEdge *e2) const`  
*Check if two given edges meet in this vertex.*
- `bool isBoundaryVertex () const`
- `bool sameEdge (Vertex *other) const`  
*Check if this vertex and the other vertex belongs to the same edge.*
- `bool sameFace (Vertex *other) const`  
*Check if this vertex and the other vertex belongs to the same face.*
- `bool sameUnderlyingSurface (Vertex *other) const`
- `bool connectedToSameVertex (Vertex *other) const`
- `Vertex * getCommonVertex (Vertex *other) const`
- `ftEdge * getCommonEdge (Vertex *other) const`  
*Get the edge associated with two vertices, if any.*
- `ftEdge * getCommonEdgeInFace (Vertex *other, ftSurface *face) const`
- `std::vector< ftSurface * > getCommonFaces (Vertex *other) const`  
*Get the faces associated with two vertices, if any.*
- `std::vector< ftEdge * > getFaceEdges (ftSurface *face) const`  
*Get the edges meeting in this vertex associated with a given face.*
- `bool isCornerInFace (ftSurface *face, double tol)`  
*Check if the vertex is a corner in the given face.*
- `std::vector< shared_ptr< Vertex > > getNextVertex (ftSurface *face) const`  
*Fetch the next vertices in the given face.*
- `void getEdgeDiscontinuities (std::vector< std::pair< ftEdge *, ftEdge * > > &gaps, double tol, std::vector< std::pair< ftEdge *, ftEdge * > > &kinks, double angtol, double angmax) const`
- `void reOrganize ()`  
*Reorganize edges according to twin information.*
- `bool checkVertexTopology ()`  
*Check the consistency of the edge information in this vertex.*

### 29.535.1 Detailed Description

The vertex class represents the vertex entity in a boundary represented solid or face set.

Definition at line 57 of file Vertex.h.

## 29.535.2 Constructor & Destructor Documentation

### 29.535.2.1 Go::Vertex::Vertex ( Point *vertex\_point* )

Constructor. Give the geometric position of the vertex.

### 29.535.2.2 Go::Vertex::Vertex ( Point *vertex\_point*, std::vector< ftEdge \* > *edges* )

Constructor. Give the geometric position of the vertex and associated edges

### 29.535.2.3 Go::Vertex::Vertex ( Point *vertex\_point*, ftEdge \* *edges* )

Constructor. Give the geometric position of the vertex and one associated edge

### 29.535.2.4 Go::Vertex::Vertex ( ftEdge \* *edge*, bool *at\_start* )

Constructor. Give one associated edge and an indication on which of the two vertices belonging to this edge should be constructed

### 29.535.2.5 Go::Vertex::~~Vertex ( )

Destructor.

## 29.535.3 Member Function Documentation

### 29.535.3.1 void Go::Vertex::addEdge ( ftEdge \* *edge* )

Add a new edge to this vertex. Used in topology build.

### 29.535.3.2 std::vector<ftEdge\*> Go::Vertex::allEdges ( ) const

Returns all edges meeting in this vertex including twins.

### 29.535.3.3 void Go::Vertex::averageVertexPos ( )

Average corners of spline surfaces corresponding to this vertex.

### 29.535.3.4 bool Go::Vertex::checkVertexTopology ( )

Check the consistency of the edge information in this vertex.

29.535.3.5 **bool** Go::Vertex::connectedToSameVertex ( **Vertex** \* *other* ) **const**

Check if this vertex and the other vertex are connected to the same vertex

29.535.3.6 **void** Go::Vertex::disconnectTwin ( **ftEdge** \* *edge* )

Given an edge connected to this vertex, remove twin information about the edge in the vertex. Used in connection with topology changes in the associated model

29.535.3.7 **std::vector<ftSurface\*>** Go::Vertex::faces ( ) **const**

Get all faces meeting in this vertex.

29.535.3.8 **std::vector<ftSurface\*>** Go::Vertex::faces ( **Body** \* *bd* ) **const**

Get all faces belonging to a given body meeting in this vertex.

29.535.3.9 **std::vector<ftEdge\*>** Go::Vertex::freeEdges ( )

Get edges which are not associated a face.

29.535.3.10 **std::vector<Body\*>** Go::Vertex::getBodies ( )

Get all bodies meeting in this vertex.

29.535.3.11 **ftEdge\*** Go::Vertex::getCommonEdge ( **Vertex** \* *other* ) **const**

Get the edge associated with two vertices, if any.

29.535.3.12 **ftEdge\*** Go::Vertex::getCommonEdgeInFace ( **Vertex** \* *other*, **ftSurface** \* *face* ) **const**

Fetch the edge, if any, joining this vertex with the vertex other belonging to the specified face

29.535.3.13 **std::vector<ftSurface\*>** Go::Vertex::getCommonFaces ( **Vertex** \* *other* ) **const**

Get the faces associated with two vertices, if any.

29.535.3.14 **Vertex\*** Go::Vertex::getCommonVertex ( **Vertex** \* *other* ) **const**

Fetch the vertex connected (through an edge) to both this and the other vertex, if any

29.535.3.15 `double Go::Vertex::getDist ( shared_ptr< Vertex > other_point ) [inline]`

The distance between this vertex an another vertex.

Definition at line 161 of file Vertex.h.

29.535.3.16 `ftEdge* Go::Vertex::getEdge ( int idx ) [inline]`

Get the specified edge, twin edges are counted once and it is arbitrary which twin edge is returned

Definition at line 120 of file Vertex.h.

29.535.3.17 `void Go::Vertex::getEdgeDiscontinuities ( std::vector< std::pair< ftEdge *, ftEdge * > > & gaps, double tol, std::vector< std::pair< ftEdge *, ftEdge * > > & kinks, double angtol, double angmax ) const`

Collect attached edges where the distance between the endpoints are larger than the specified tolerance or where the curves meet with an angle that are more than the kink tolerance, but less than the corner tolerance

29.535.3.18 `std::vector<ftEdge*> Go::Vertex::getFaceEdges ( ftSurface * face ) const`

Get the edges meeting in this vertex associated with a given face.

29.535.3.19 `Point Go::Vertex::getFacePar ( ftSurface * face )`

Get parameter of associated face corresponding to vertex.

29.535.3.20 `std::vector<std::pair<ftSurface*, Point> > Go::Vertex::getFaces ( )`

Get all faces meeting in this vertex and the parameter value in the face corresponding to the vertex

29.535.3.21 `std::vector<std::pair<ftSurface*, Point> > Go::Vertex::getFaces ( Body * bd )`

Get all faces belonging to a given body meeting in this vertex and the parameter value in the face corresponding to the vertex

29.535.3.22 `std::vector<shared_ptr<Vertex> > Go::Vertex::getNextVertex ( ftSurface * face ) const`

Fetch the next vertices in the given face.

29.535.3.23 `Point Go::Vertex::getVertexPoint ( ) [inline]`

Get the geometrical position associated to this vertex.

Definition at line 126 of file Vertex.h.

**29.535.3.24** `bool Go::Vertex::hasEdge ( ftEdge * edge ) const`

Check if the vertex is connected to the given edge.

**29.535.3.25** `bool Go::Vertex::hasEdgeSingle ( ftEdge * edge ) const`

Check if the vertex is connected to the given edge, and this edge is represented in the vertex with no twin

**29.535.3.26** `bool Go::Vertex::hasFace ( ftSurface * face ) const`

Check if the vertex is adjacent to the given face.

**29.535.3.27** `bool Go::Vertex::isBoundaryVertex ( ) const`

Check if this vertex lies at a model boundary, i.e. is connected to edges with no twin

**29.535.3.28** `bool Go::Vertex::isCornerInFace ( ftSurface * face, double tol )`

Check if the vertex is a corner in the given face.

**29.535.3.29** `void Go::Vertex::joinVertex ( shared_ptr< Vertex > other )`

Vertices belonging to two adjacent edges are represented as one entity. Used in topology build

**29.535.3.30** `bool Go::Vertex::meetInVertex ( ftEdge * e1, ftEdge * e2 ) const`

Check if two given edges meet in this vertex.

**29.535.3.31** `int Go::Vertex::nmbUniqueEdges ( ) [inline]`

Number of unique edges meeting in this vertex, twin edges are counted only once

Definition at line 108 of file Vertex.h.

**29.535.3.32** `int Go::Vertex::nmbUniqueEdges ( Body * bd )`

**29.535.3.33** `void Go::Vertex::removeEdge ( ftEdge * edge )`

Remove an edge from this vertex. Used in connection with topology changes in the associated model



29.535.3.34 `void Go::Vertex::reOrganize ( )`

Reorganize edges according to twin information.

29.535.3.35 `bool Go::Vertex::sameEdge ( Vertex * other ) const`

Check if this vertex and the other vertex belongs to the same edge.

29.535.3.36 `bool Go::Vertex::sameFace ( Vertex * other ) const`

Check if this vertex and the other vertex belongs to the same face.

29.535.3.37 `bool Go::Vertex::sameUnderlyingSurface ( Vertex * other ) const`

Check if this vertex and the other vertex belongs to the same underlying surface (one surface can give rise to several faces)

29.535.3.38 `void Go::Vertex::setVertexPoint ( Point vertex_point ) [inline]`

Set the geometrical position associated to this vertex.

Definition at line 132 of file Vertex.h.

29.535.3.39 `std::vector<ftEdge*> Go::Vertex::uniqueEdges ( )`

Returns all geometrically unique edges meeting in this vertex. One edge is return for a pair of twins.

29.535.3.40 `std::vector<ftEdge*> Go::Vertex::uniqueEdges ( Body * bd )`

Returns all edges belonging to a given body. Twin edges are represented only as one edge

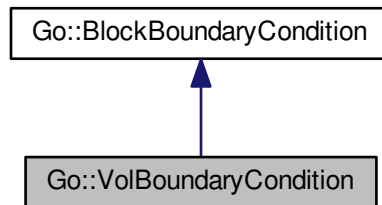
The documentation for this class was generated from the following file:

- `compositemodel/include/GoTools/compositemodel/Vertex.h`

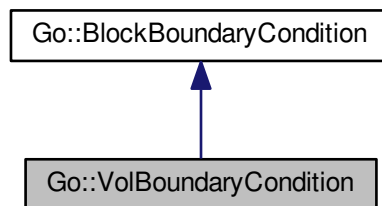
## 29.536 Go::VolBoundaryCondition Class Reference

```
#include <VolBoundaryCondition.h>
```

Inheritance diagram for Go::VolBoundaryCondition:



Collaboration diagram for Go::VolBoundaryCondition:



### Public Member Functions

- [VolBoundaryCondition](#) (int face\_nmb, [BdConditionType](#) type, [BdCondFuncor](#) \*fbd, std::vector< std::pair< double, double > > &domain, [VolSolution](#) \*solution)
- [VolBoundaryCondition](#) (int face\_nmb, [BdConditionType](#) type, const [Point](#) &const\_val, std::vector< std::pair< double, double > > &domain, [VolSolution](#) \*solution)
- virtual [~VolBoundaryCondition](#) ()
- virtual void [getCoefficientsEnumeration](#) (std::vector< int > &local\_enumeration)
- virtual void [getCoefficientsEnumeration](#) (std::vector< int > &local\_enumeration\_bd, std::vector< int > &local\_enumeration\_bd2)
- virtual void [getBdCoefficients](#) (std::vector< std::pair< int, [Point](#) > > &coefs)
- virtual void [getBdCoefficients](#) (std::vector< std::pair< int, [Point](#) > > &coefs\_bd, std::vector< std::pair< int, [Point](#) > > &coefs\_inner)
- void [getBasisFunctions](#) (int index\_of\_Gauss\_point1, int index\_of\_Gauss\_point2, int &const\_dir, vector< double > &basisValues, vector< double > &basisDerivs\_u, vector< double > &basisDerivs\_v) const
- shared\_ptr< [SplineSurface](#) > [getSplineApproximation](#) () const
- virtual void [update](#) ()
- virtual void [updateBoundaryValue](#) ([BdCondFuncor](#) \*fbd)
- int [faceNumber](#) () const
- virtual [tpTolerances](#) [getTolerances](#) () const

## Additional Inherited Members

### 29.536.1 Detailed Description

Definition at line 67 of file VolBoundaryCondition.h.

### 29.536.2 Constructor & Destructor Documentation

29.536.2.1 `Go::VolBoundaryCondition::VolBoundaryCondition ( int face_nmb, BdConditionType type, BdCondFuncor * fbf, std::vector< std::pair< double, double > > & domain, VolSolution * solution )`

29.536.2.2 `Go::VolBoundaryCondition::VolBoundaryCondition ( int face_nmb, BdConditionType type, const Point & const_val, std::vector< std::pair< double, double > > & domain, VolSolution * solution )`

29.536.2.3 `virtual Go::VolBoundaryCondition::~~VolBoundaryCondition ( ) [virtual]`

### 29.536.3 Member Function Documentation

29.536.3.1 `int Go::VolBoundaryCondition::faceNumber ( ) const`

29.536.3.2 `void Go::VolBoundaryCondition::getBasisFunctions ( int index_of_Gauss_point1, int index_of_Gauss_point2, int & const_dir, vector< double > & basisValues, vector< double > & basisDerivs_u, vector< double > & basisDerivs_v ) const`

29.536.3.3 `virtual void Go::VolBoundaryCondition::getBdCoefficients ( std::vector< std::pair< int, Point > > & coefs ) [virtual]`

Implements [Go::BlockBoundaryCondition](#).

29.536.3.4 `virtual void Go::VolBoundaryCondition::getBdCoefficients ( std::vector< std::pair< int, Point > > & coefs_bd, std::vector< std::pair< int, Point > > & coefs_inner ) [virtual]`

Implements [Go::BlockBoundaryCondition](#).

29.536.3.5 `virtual void Go::VolBoundaryCondition::getCoefficientsEnumeration ( std::vector< int > & local_enumeration ) [virtual]`

Implements [Go::BlockBoundaryCondition](#).

29.536.3.6 `virtual void Go::VolBoundaryCondition::getCoefficientsEnumeration ( std::vector< int > & local_enumeration_bd, std::vector< int > & local_enumeration_bd2 ) [virtual]`

Implements [Go::BlockBoundaryCondition](#).

29.536.3.7 `shared_ptr<SplineSurface> Go::VolBoundaryCondition::getSplineApproximation ( ) const`

29.536.3.8 `virtual tpTolerances Go::VolBoundaryCondition::getTolerances ( ) const [virtual]`

Implements [Go::BlockBoundaryCondition](#).

29.536.3.9 `virtual void Go::VolBoundaryCondition::update ( ) [virtual]`

Implements [Go::BlockBoundaryCondition](#).

29.536.3.10 `virtual void Go::VolBoundaryCondition::updateBoundaryValue ( BdCondFuncor * fbd ) [virtual]`

Implements [Go::BlockBoundaryCondition](#).

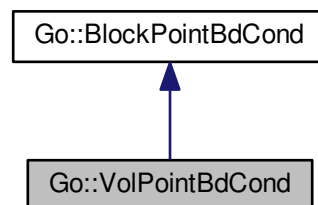
The documentation for this class was generated from the following file:

- [isogeometric\\_model/include/GoTools/isogeometric\\_model/VolBoundaryCondition.h](#)

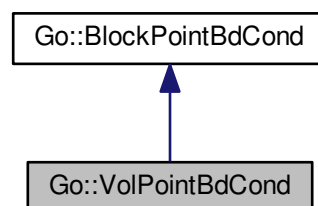
## 29.537 Go::VolPointBdCond Class Reference

```
#include <VolPointBdCond.h>
```

Inheritance diagram for Go::VolPointBdCond:



Collaboration diagram for Go::VolPointBdCond:



## Public Member Functions

- [VolPointBdCond](#) (int face\_nmb, double param[], [Point](#) &condition\_value)
- [~VolPointBdCond](#) ()
- virtual void [getCoefficientsEnumeration](#) (std::vector< int > &local\_enumeration) [const](#)
- virtual [Point](#) [getConditionValue](#) () [const](#)
- double \* [getParam](#) () [const](#)
- virtual void [getInterpolationFactors](#) (std::vector< std::pair< int, double > > &factors) [const](#)
- int [faceNumber](#) () [const](#)

### 29.537.1 Detailed Description

Definition at line 56 of file [VolPointBdCond.h](#).

### 29.537.2 Constructor & Destructor Documentation

29.537.2.1 [Go::VolPointBdCond::VolPointBdCond](#) ( int *face\_nmb*, double *param[]*, [Point](#) & *condition\_value* )

29.537.2.2 [Go::VolPointBdCond::~VolPointBdCond](#) ( )

### 29.537.3 Member Function Documentation

29.537.3.1 int [Go::VolPointBdCond::faceNumber](#) ( ) [const](#)

29.537.3.2 virtual void [Go::VolPointBdCond::getCoefficientsEnumeration](#) ( std::vector< int > & *local\_enumeration* ) [const](#)  
[[virtual](#)]

Implements [Go::BlockPointBdCond](#).

29.537.3.3 virtual [Point](#) [Go::VolPointBdCond::getConditionValue](#) ( ) [const](#) [[virtual](#)]

Implements [Go::BlockPointBdCond](#).

29.537.3.4 virtual void [Go::VolPointBdCond::getInterpolationFactors](#) ( std::vector< std::pair< int, double > > & *factors* )  
[const](#) [[virtual](#)]

Implements [Go::BlockPointBdCond](#).

29.537.3.5 double\* [Go::VolPointBdCond::getParam](#) ( ) [const](#)

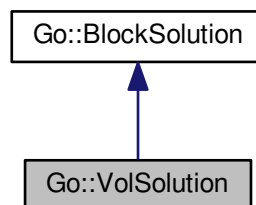
The documentation for this class was generated from the following file:

- [isogeometric\\_model/include/GoTools/isogeometric\\_model/VolPointBdCond.h](#)

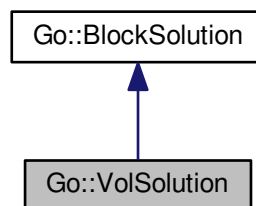
## 29.538 Go::VolSolution Class Reference

```
#include <VolSolution.h>
```

Inheritance diagram for Go::VolSolution:



Collaboration diagram for Go::VolSolution:



### Public Member Functions

- [VolSolution](#) ([IsogeometricVolBlock](#) \*parent, [shared\\_ptr](#)< [SplineVolume](#) > sol\_vol)
- virtual [~VolSolution](#) ()
- virtual [VolSolution](#) \* [asVolSolution](#) ()
- void [addBoundaryCondition](#) (int face\_nmb, [BdConditionType](#) type, [Go::Point](#) &const\_val, [std::vector](#)< [std::pair](#)< [double](#), [double](#) > > &domain)
- void [addBoundaryCondition](#) (int face\_nmb, [BdConditionType](#) type, [BdCondFuncor](#) \*fbd, [std::vector](#)< [std::pair](#)< [double](#), [double](#) > > &domain)
- void [addDirichletPointBdCond](#) ([double](#) param[], [Point](#) &condition\_value)
- int [getNmbOfBoundaryConditions](#) () const
- [shared\\_ptr](#)< [VolBoundaryCondition](#) > [getBoundaryCondition](#) (int index) const
- void [getFaceBoundaryConditions](#) (int face\_number, [std::vector](#)< [shared\\_ptr](#)< [VolBoundaryCondition](#) > > &bd\_cond) const
- void [getFaceBoundaryConditions](#) ([std::vector](#)< [shared\\_ptr](#)< [VolBoundaryCondition](#) > > &bd\_cond) const
- virtual int [getNmbOfPointBdConditions](#) () const

- `shared_ptr< VolPointBdCond > getPointBdCondition (int index) const`
- `void getFacePointBdConditions (int face_number, std::vector< shared_ptr< VolPointBdCond > > &bd_cond) const`
- `void getPointBdCond (std::vector< shared_ptr< VolPointBdCond > > &bd_cond) const`
- `virtual bool matchingSplineSpace (BlockSolution *other) const`
- `virtual void getMatchingCoefficients (BlockSolution *other, std::vector< std::pair< int, int > > &enumeration, int match_pos=0) const`
- `virtual void getBoundaryCoefficients (int boundary, std::vector< int > &enumeration) const`
- `virtual void getBoundaryCoefficients (int boundary, std::vector< int > &enumeration_bd, std::vector< int > &enumeration_bd2) const`
- `virtual void makeMatchingSplineSpace (BlockSolution *other)`
- `virtual void increaseDegree (int new_degree, int pardir)`
- `virtual void insertKnots (const std::vector< int > &knot_intervals, int pardir)`
- `virtual void insertKnots (const std::vector< double > &knots, int pardir)`
- `virtual void erasePreEvaluatedBasisFunctions ()`
- `virtual void performPreEvaluation (std::vector< std::vector< double > > &Gauss_par)`
- `void getBasisFunctions (int index_of_Gauss_point1, int index_of_Gauss_point2, int index_of_Gauss_point3, vector< double > &basisValues, vector< double > &basisDerivs_u, vector< double > &basisDerivs_v, vector< double > &basisDerivs_w) const`
- `void getBasisFunctions (double param1, double param2, double param3, vector< double > &basisValues, vector< double > &basisDerivs_u, vector< double > &basisDerivs_v, vector< double > &basisDerivs_w) const`
- `void getBasisFunctionValues (int basis_func_id_u, int basis_func_id_v, int basis_func_id_w, std::vector< int > &index_of_Gauss_points1, std::vector< int > &index_of_Gauss_points2, std::vector< int > &index_of_Gauss_points3, std::vector< double > &basisValues, std::vector< double > &basisDerivs_u, std::vector< double > &basisDerivs_v, std::vector< double > &basisDerivs_w) const`
- `void getBasisFunctionValues (int basis_func_id_u, int basis_func_id_v, int basis_func_id_w, int knot_ind_u, int knot_ind_v, int knot_ind_w, std::vector< int > &index_of_Gauss_points1, std::vector< int > &index_of_Gauss_points2, std::vector< int > &index_of_Gauss_points3, std::vector< double > &basisValues, std::vector< double > &basisDerivs_u, std::vector< double > &basisDerivs_v, std::vector< double > &basisDerivs_w) const`
- `virtual double getJacobian (std::vector< int > &index_of_Gauss_point) const`
- `virtual void valuesInGaussPoint (const std::vector< int > &index_of_Gauss_point, std::vector< Point > &derivs) const`
- `virtual void setSolutionCoefficients (const std::vector< double > &coefs)`
- `shared_ptr< SplineVolume > getSolutionVolume () const`
- `shared_ptr< SplineVolume > getGeometryVolume () const`
- `void setMinimumDegree (int degree)`
- `void refineToGeometry (int pardir)`
- `virtual int nmbCoefs () const`
- `virtual int nmbCoefs (int pardir) const`
- `virtual int degree (int pardir) const`
- `virtual std::vector< double > knots (int pardir) const`
- `virtual std::vector< double > distinctKnots (int pardir) const`
- `virtual BsplineBasis basis (int pardir) const`
- `virtual int dimension () const`
- `virtual void updateConditions ()`
- `void getGaussParameter (int index_of_Gauss_point1, int index_of_Gauss_point2, int const_dir, double &par1, double &par2) const`
- `virtual tpTolerances getTolerances () const`

### 29.538.1 Detailed Description

Definition at line 84 of file VolSolution.h.

## 29.538.2 Constructor & Destructor Documentation

29.538.2.1 `Go::VolSolution::VolSolution ( IsogeometricVolBlock * parent, shared_ptr< SplineVolume > sol_vol )`

29.538.2.2 `virtual Go::VolSolution::~~VolSolution ( ) [virtual]`

## 29.538.3 Member Function Documentation

29.538.3.1 `void Go::VolSolution::addBoundaryCondition ( int face_nmb, BdConditionType type, Go::Point & const_val, std::vector< std::pair< double, double > > & domain )`

29.538.3.2 `void Go::VolSolution::addBoundaryCondition ( int face_nmb, BdConditionType type, BdCondFuncor * fbd, std::vector< std::pair< double, double > > & domain )`

29.538.3.3 `void Go::VolSolution::addDirichletPointBdCond ( double param[], Point & condition_value )`

29.538.3.4 `virtual VolSolution* Go::VolSolution::asVolSolution ( ) [virtual]`

Reimplemented from [Go::BlockSolution](#).

29.538.3.5 `virtual BsplineBasis Go::VolSolution::basis ( int pardir ) const [virtual]`

Implements [Go::BlockSolution](#).

29.538.3.6 `virtual int Go::VolSolution::degree ( int pardir ) const [virtual]`

Implements [Go::BlockSolution](#).

29.538.3.7 `virtual int Go::VolSolution::dimension ( ) const [virtual]`

Implements [Go::BlockSolution](#).

29.538.3.8 `virtual std::vector<double> Go::VolSolution::distinctKnots ( int pardir ) const [virtual]`

Implements [Go::BlockSolution](#).

29.538.3.9 `virtual void Go::VolSolution::erasePreEvaluatedBasisFunctions ( ) [virtual]`

Implements [Go::BlockSolution](#).



- 29.538.3.10 void Go::VolSolution::getBasisFunctions ( int *index\_of\_Gauss\_point1*, int *index\_of\_Gauss\_point2*, int *index\_of\_Gauss\_point3*, vector< double > & *basisValues*, vector< double > & *basisDerivs\_u*, vector< double > & *basisDerivs\_v*, vector< double > & *basisDerivs\_w* ) const
- 29.538.3.11 void Go::VolSolution::getBasisFunctions ( double *param1*, double *param2*, double *param3*, vector< double > & *basisValues*, vector< double > & *basisDerivs\_u*, vector< double > & *basisDerivs\_v*, vector< double > & *basisDerivs\_w* ) const
- 29.538.3.12 void Go::VolSolution::getBasisFunctionValues ( int *basis\_func\_id\_u*, int *basis\_func\_id\_v*, int *basis\_func\_id\_w*, std::vector< int > & *index\_of\_Gauss\_points1*, std::vector< int > & *index\_of\_Gauss\_points2*, std::vector< int > & *index\_of\_Gauss\_points3*, std::vector< double > & *basisValues*, std::vector< double > & *basisDerivs\_u*, std::vector< double > & *basisDerivs\_v*, std::vector< double > & *basisDerivs\_w* ) const
- 29.538.3.13 void Go::VolSolution::getBasisFunctionValues ( int *basis\_func\_id\_u*, int *basis\_func\_id\_v*, int *basis\_func\_id\_w*, int *knot\_ind\_u*, int *knot\_ind\_v*, int *knot\_ind\_w*, std::vector< int > & *index\_of\_Gauss\_points1*, std::vector< int > & *index\_of\_Gauss\_points2*, std::vector< int > & *index\_of\_Gauss\_points3*, std::vector< double > & *basisValues*, std::vector< double > & *basisDerivs\_u*, std::vector< double > & *basisDerivs\_v*, std::vector< double > & *basisDerivs\_w* ) const
- 29.538.3.14 virtual void Go::VolSolution::getBoundaryCoefficients ( int *boundary*, std::vector< int > & *enumeration* ) const  
[virtual]

Implements [Go::BlockSolution](#).

- 29.538.3.15 virtual void Go::VolSolution::getBoundaryCoefficients ( int *boundary*, std::vector< int > & *enumeration\_bd*, std::vector< int > & *enumeration\_bd2* ) const [virtual]

Implements [Go::BlockSolution](#).

- 29.538.3.16 shared\_ptr<VolBoundaryCondition> Go::VolSolution::getBoundaryCondition ( int *index* ) const
- 29.538.3.17 void Go::VolSolution::getFaceBoundaryConditions ( int *face\_number*, std::vector< shared\_ptr< VolBoundaryCondition > > & *bd\_cond* ) const
- 29.538.3.18 void Go::VolSolution::getFaceBoundaryConditions ( std::vector< shared\_ptr< VolBoundaryCondition > > & *bd\_cond* ) const
- 29.538.3.19 void Go::VolSolution::getFacePointBdConditions ( int *face\_number*, std::vector< shared\_ptr< VolPointBdCond > > & *bd\_cond* ) const
- 29.538.3.20 void Go::VolSolution::getGaussParameter ( int *index\_of\_Gauss\_point1*, int *index\_of\_Gauss\_point2*, int *const\_dir*, double & *par1*, double & *par2* ) const
- 29.538.3.21 shared\_ptr<SplineVolume> Go::VolSolution::getGeometryVolume ( ) const
- 29.538.3.22 virtual double Go::VolSolution::getJacobian ( std::vector< int > & *index\_of\_Gauss\_point* ) const  
[virtual]

Implements [Go::BlockSolution](#).

29.538.3.23 `virtual void Go::VolSolution::getMatchingCoefficients ( BlockSolution * other, std::vector< std::pair< int, int >> & enumeration, int match_pos = 0 ) const` [virtual]

Implements [Go::BlockSolution](#).

29.538.3.24 `int Go::VolSolution::getNmbOfBoundaryConditions ( ) const`

29.538.3.25 `virtual int Go::VolSolution::getNmbOfPointBdConditions ( ) const` [virtual]

Implements [Go::BlockSolution](#).

29.538.3.26 `void Go::VolSolution::getPointBdCond ( std::vector< shared_ptr< VolPointBdCond >> & bd_cond ) const`

29.538.3.27 `shared_ptr<VolPointBdCond> Go::VolSolution::getPointBdCondition ( int index ) const`

29.538.3.28 `shared_ptr<SplineVolume> Go::VolSolution::getSolutionVolume ( ) const`

29.538.3.29 `virtual tpTolerances Go::VolSolution::getTolerances ( ) const` [virtual]

Implements [Go::BlockSolution](#).

29.538.3.30 `virtual void Go::VolSolution::increaseDegree ( int new_degree, int pardir )` [virtual]

Implements [Go::BlockSolution](#).

29.538.3.31 `virtual void Go::VolSolution::insertKnots ( const std::vector< int > & knot_intervals, int pardir )` [virtual]

Implements [Go::BlockSolution](#).

29.538.3.32 `virtual void Go::VolSolution::insertKnots ( const std::vector< double > & knots, int pardir )` [virtual]

Implements [Go::BlockSolution](#).

29.538.3.33 `virtual std::vector<double> Go::VolSolution::knots ( int pardir ) const` [virtual]

Implements [Go::BlockSolution](#).

29.538.3.34 `virtual void Go::VolSolution::makeMatchingSplineSpace ( BlockSolution * other )` [virtual]

Implements [Go::BlockSolution](#).

29.538.3.35 `virtual bool Go::VolSolution::matchingSplineSpace ( BlockSolution * other ) const` [virtual]

Implements [Go::BlockSolution](#).

29.538.3.36 `virtual int Go::VolSolution::nmbCoefs ( ) const` [virtual]

Implements [Go::BlockSolution](#).

29.538.3.37 `virtual int Go::VolSolution::nmbCoefs ( int pardir ) const` [virtual]

Implements [Go::BlockSolution](#).

29.538.3.38 `virtual void Go::VolSolution::performPreEvaluation ( std::vector< std::vector< double > > & Gauss_par )`  
[virtual]

Implements [Go::BlockSolution](#).

29.538.3.39 `void Go::VolSolution::refineToGeometry ( int pardir )`

29.538.3.40 `void Go::VolSolution::setMinimumDegree ( int degree )`

29.538.3.41 `virtual void Go::VolSolution::setSolutionCoefficients ( const std::vector< double > & coefs )` [virtual]

Implements [Go::BlockSolution](#).

29.538.3.42 `virtual void Go::VolSolution::updateConditions ( )` [virtual]

Implements [Go::BlockSolution](#).

29.538.3.43 `virtual void Go::VolSolution::valuesInGaussPoint ( const std::vector< int > & index_of_Gauss_point,  
std::vector< Point > & derivs ) const` [virtual]

Implements [Go::BlockSolution](#).

The documentation for this class was generated from the following file:

- [isogeometric\\_model/include/GoTools/isogeometric\\_model/VolSolution.h](#)

## 29.539 Go::VolumeAdjacency Class Reference

Adjacency analysis of volume models.

```
#include <VolumeAdjacency.h>
```

## Public Member Functions

- [VolumeAdjacency](#) (double gap, double neighbour)  
*Constructor giving tolerances for adjacency analysis.*
- [~VolumeAdjacency](#) ()  
*Destructor.*
- void [setAdjacency](#) (std::vector< shared\_ptr< [Body](#) > > &solids)  
*Perform topology analysis on a set of bodies.*
- void [setAdjacency](#) (std::vector< shared\_ptr< [Body](#) > > &solids, int new\_solid\_pos)

### 29.539.1 Detailed Description

Adjacency analysis of volume models.

Definition at line 52 of file [VolumeAdjacency.h](#).

### 29.539.2 Constructor & Destructor Documentation

#### 29.539.2.1 Go::VolumeAdjacency::VolumeAdjacency ( double gap, double neighbour )

Constructor giving tolerances for adjacency analysis.

#### 29.539.2.2 Go::VolumeAdjacency::~~VolumeAdjacency ( )

Destructor.

### 29.539.3 Member Function Documentation

#### 29.539.3.1 void Go::VolumeAdjacency::setAdjacency ( std::vector< shared\_ptr< [Body](#) > > & solids )

Perform topology analysis on a set of bodies.

#### 29.539.3.2 void Go::VolumeAdjacency::setAdjacency ( std::vector< shared\_ptr< [Body](#) > > & solids, int new\_solid\_pos )

Perform topology analysis on a set of bodies where it is assumed the the topological relationship between the first new\_solid\_pos bodies are known already

The documentation for this class was generated from the following file:

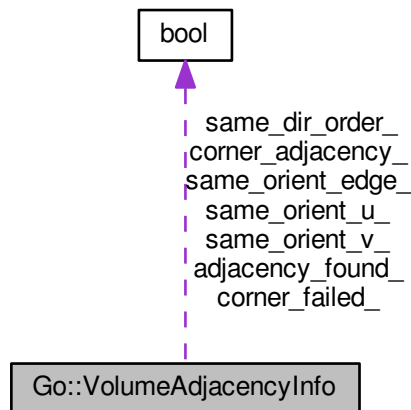
- [trivariatemodel/include/GoTools/trivariatemodel/VolumeAdjacency.h](#)

## 29.540 Go::VolumeAdjacencyInfo Struct Reference

Struct to store information about adjacency relations between two bodies.

```
#include <ftVolume.h>
```

Collaboration diagram for Go::VolumeAdjacencyInfo:



### Public Member Functions

- [VolumeAdjacencyInfo \(\)](#)  
*Constructor.*

### Public Attributes

- [bool adjacency\\_found\\_](#)  
*True if adjacency was found, false if not.*
- [bool corner\\_adjacency\\_](#)  
*True if two volumes meet in along an edge.*
- [int bd\\_idx\\_1\\_](#)  
*Boundary index of first volume. Value 0-5 in order umin, umax, vmin, vmax, wmin, wmax.*
- [int bd\\_idx\\_2\\_](#)  
*Boundary index of second volume. Same interpretation as bd\_idx\_1\_.*
- [int edg\\_idx\\_1\\_](#)  
*Set if corner\_adjacency\_ is true. 0-3 in the order umin, umax, vmin, vmax.*
- [int edg\\_idx\\_2\\_](#)  
*Set if corner\_adjacency\_ is true. 0-3 in the order umin, umax, vmin, vmax.*
- [bool same\\_orient\\_u\\_](#)  
*True if common boundaries are equally oriented in first parameter direction of first surface.*
- [bool same\\_orient\\_v\\_](#)  
*True if common boundaries are equally oriented in second parameter direction of first surface.*
- [bool same\\_orient\\_edge\\_](#)  
*True if corner\_adjacency\_ is found and the boundary curves are equally oriented.*
- [bool same\\_dir\\_order\\_](#)  
*True if u-directions coincide for both boundary surfaces (regardless of orientation)*
- [bool corner\\_failed\\_](#)

### 29.540.1 Detailed Description

Struct to store information about adjacency relations between two bodies.

Definition at line 54 of file ftVolume.h.

### 29.540.2 Constructor & Destructor Documentation

#### 29.540.2.1 `Go::VolumeAdjacencyInfo::VolumeAdjacencyInfo ( ) [inline]`

Constructor.

Definition at line 83 of file ftVolume.h.

### 29.540.3 Member Data Documentation

#### 29.540.3.1 `bool Go::VolumeAdjacencyInfo::adjacency_found_`

True if adjacency was found, false if not.

Definition at line 58 of file ftVolume.h.

#### 29.540.3.2 `int Go::VolumeAdjacencyInfo::bd_idx_1_`

Boundary index of first volume. Value 0-5 in order umin, umax, vmin, vmax, wmin, wmax.

Definition at line 62 of file ftVolume.h.

#### 29.540.3.3 `int Go::VolumeAdjacencyInfo::bd_idx_2_`

Boundary index of second volume. Same interpretation as `bd_idx_1_`.

Definition at line 64 of file ftVolume.h.

#### 29.540.3.4 `bool Go::VolumeAdjacencyInfo::corner_adjacency_`

True if two volumes meet in along an edge.

Definition at line 60 of file ftVolume.h.

#### 29.540.3.5 `bool Go::VolumeAdjacencyInfo::corner_failed_`

True if adjacency is found, user wanted to test if the surfaces meet in corner-to-corner configuration, and the test failed. Otherwise false.

Definition at line 80 of file ftVolume.h.

**29.540.3.6** int Go::VolumeAdjacencyInfo::edg\_idx\_1\_

Set if corner\_adjacency\_ is true. 0-3 in the order umin, umax, vmin, vmax.

Definition at line 66 of file ftVolume.h.

**29.540.3.7** int Go::VolumeAdjacencyInfo::edg\_idx\_2\_

Set if corner\_adjacency\_ is true. 0-3 in the order umin, umax, vmin, vmax.

Definition at line 68 of file ftVolume.h.

**29.540.3.8** bool Go::VolumeAdjacencyInfo::same\_dir\_order\_

True if u-directions coincide for both boundary surfaces (regardless of orientation)

Definition at line 77 of file ftVolume.h.

**29.540.3.9** bool Go::VolumeAdjacencyInfo::same\_orient\_edge\_

True if corner\_adjacency\_ is found and the boundary curves are equally oriented.

Definition at line 75 of file ftVolume.h.

**29.540.3.10** bool Go::VolumeAdjacencyInfo::same\_orient\_u\_

True if common boundaries are equally oriented in first parameter direction of first surface.

Definition at line 70 of file ftVolume.h.

**29.540.3.11** bool Go::VolumeAdjacencyInfo::same\_orient\_v\_

True if common boundaries are equally oriented in second parameter direction of first surface.

Definition at line 72 of file ftVolume.h.

The documentation for this struct was generated from the following file:

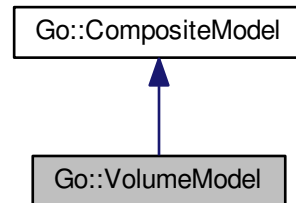
- [trivariatemodel/include/GoTools/trivariatemodel/ftVolume.h](#)

## 29.541 Go::VolumeModel Class Reference

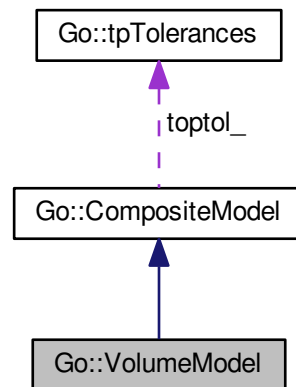
A set of volumes including topology information.

```
#include <VolumeModel.h>
```

Inheritance diagram for Go::VolumeModel:



Collaboration diagram for Go::VolumeModel:



### Public Member Functions

- `VolumeModel` (`std::vector< shared_ptr< ftVolume > > &volumes`, `double gap`, `double neighbour`, `double kink`, `double bend`, `bool adjacency_set=false`)  
*Constructor given a set of volumes and topologic tolerances.*
- `VolumeModel` (`std::vector< shared_ptr< ftVolume > > &volumes`, `double gap`, `double kink`)  
*Constructor given a set of volumes and topologic tolerances.*
- `VolumeModel` (`double gap`, `double neighbour`, `double kink`, `double bend`)  
*Constructor topologic tolerances.*



- [VolumeModel](#) ([const VolumeModel](#) &vm)  
*Copy constructor.*
- virtual [~VolumeModel](#) ()  
*Destructor.*
- virtual [VolumeModel](#) \* [clone](#) () [const](#)  
*Make a copy of the current model.*
- virtual int [nmbEntities](#) () [const](#)  
*Number of simple entities.*
- [shared\\_ptr](#)< [ftVolume](#) > [getBody](#) (int idx) [const](#)  
*Return one body.*
- [shared\\_ptr](#)< [ParamVolume](#) > [getVolume](#) (int idx) [const](#)  
*Return one volume.*
- [shared\\_ptr](#)< [SplineVolume](#) > [getSplineVolume](#) (int idx) [const](#)  
*Return one volume as [SplineVolume](#) if possible.*
- int [getIndex](#) ([shared\\_ptr](#)< [ftVolume](#) > body) [const](#)  
*Given a body in the volume model, return the index of this face.*
- int [getIndex](#) ([ftVolume](#) \*body) [const](#)  
*Given a body in the volume model, return the index of this face.*
- [shared\\_ptr](#)< [ftVolume](#) > [fetchAsSharedPtr](#) ([Body](#) \*body) [const](#)  
*Return a specified body as a shared pointer.*
- virtual void [evaluate](#) (int idx, double par[], [Point](#) &pnt) [const](#)  
*Evaluate position.*
- virtual void [evaluate](#) (int idx, double par[], int nder, [std::vector](#)< [Point](#) > &der) [const](#)  
*Evaluate position and a number of derivatives.*
- virtual void [closestPoint](#) ([Point](#) &pnt, [Point](#) &clo\_pnt, int &idx, double clo\_par[], double &dist)  
*Compute one closest point, interface heritage, not implemented.*
- virtual [shared\\_ptr](#)< [IntResultsModel](#) > [intersect](#) ([const ftLine](#) &line)
- virtual [shared\\_ptr](#)< [IntResultsModel](#) > [intersect\\_plane](#) ([const ftPlane](#) &plane)  
*Intersection with a plane, interface heritage, not implemented.*
- virtual void [extremalPoint](#) ([Point](#) &dir, [Point](#) &clo\_pnt, int &idx, double ext\_par[])
- virtual [BoundingBox](#) [boundingBox](#) ()  
*Bounding box of the entire model.*
- virtual [BoundingBox](#) [boundingBox](#) (int idx) [const](#)  
*Bounding box corresponding to one entity.*
- virtual bool [isDegenerate](#) (int idx) [const](#)
- virtual double [curvature](#) (int idx, double \*par) [const](#)  
*Curvature, interface heritage, not implemented.*
- virtual void [turn](#) (int idx)  
*Interface heritage, not implemented.*
- virtual void [turn](#) ()  
*Interface heritage, not implemented.*
- void [append](#) ([shared\\_ptr](#)< [ftVolume](#) > volume)
- void [append](#) ([std::vector](#)< [shared\\_ptr](#)< [ftVolume](#) > > volumes)
- void [append](#) ([shared\\_ptr](#)< [VolumeModel](#) > anotherModel)
- void [removeSolid](#) ([shared\\_ptr](#)< [ftVolume](#) > vol)  
*Remove one volume from the model.*
- virtual void [tessellate](#) ([std::vector](#)< [shared\\_ptr](#)< [GeneralMesh](#) > > &meshes) [const](#)  
*Tessellate model, interface heritage, not implemented.*
- virtual void [tessellate](#) (int resolution[], [std::vector](#)< [shared\\_ptr](#)< [GeneralMesh](#) > > &meshes) [const](#)  
*Tessellate model, interface heritage, not implemented.*
- virtual void [tessellate](#) (double density, [std::vector](#)< [shared\\_ptr](#)< [GeneralMesh](#) > > &meshes) [const](#)

- Tessellate model, interface heritage, not implemented.*

  - virtual void `tessellatedCtrPolygon` (std::vector< shared\_ptr< [LineCloud](#) > > &ctr\_pol) `const`

*Tessellate model, interface heritage, not implemented.*

  - void `buildTopology` ()
  - void `buildTopology` (shared\_ptr< [ftVolume](#) > body)
  - void `setVertexIdentity` ()
  - void `setBoundarySfs` ()
  - std::vector< shared\_ptr< [ftSurface](#) > > `getBoundaryFaces` () `const`
  - std::vector< shared\_ptr< [ftSurface](#) > > `getBoundaryFaces` (int boundary\_idx) `const`
  - std::vector< shared\_ptr< [ftSurface](#) > > `getUniqueInnerFaces` () `const`
  - `double getApproximationTol` () `const`
  - `bool allSplines` () `const`

*Check if all entities are NURBS.*

  - void `getAllVertices` (std::vector< shared\_ptr< [Vertex](#) > > &vertices) `const`

*Fetch all vertices in the model.*

  - void `getRadialEdges` (std::vector< shared\_ptr< [EdgeVertex](#) > > &rad\_edges) `const`

*Fetch all radial edges in the model.*

  - void `uniqueNonRadialEdges` (std::vector< shared\_ptr< [ftEdge](#) > > &edges) `const`
  - `bool isCornerToCorner` (double tol=DEFAULT\_SPACE\_EPSILON) `const`

*Check if the model has got a corner-to-corner configuration.*

  - void `makeCornerToCorner` (double tol=DEFAULT\_SPACE\_EPSILON)
  - void `makeCommonSplineSpaces` ()

*Ensure that the blocks in the model has got common spline spaces.*

  - void `averageCorrespondingCoefs` ()

*Ensure exact match between corresponding coefficients.*

  - int `nmbBoundaries` () `const`
  - shared\_ptr< [SurfaceModel](#) > `getOuterBoundary` (int idx) `const`

*Fetch specified boundary of this model.*

  - std::vector< shared\_ptr< [VolumeModel](#) > > `getConnectedModels` ()

*Divide this model into connected volume model.*

  - void `regularizeBdShells` ()
  - void `replaceNonRegVolumes` (int degree=3, int split\_mode=1)

*Replaces volumes that are not regular with sets of regular volumes.*

  - `bool checkModelTopology` ()

*Debug.*

## Additional Inherited Members

### 29.541.1 Detailed Description

A set of volumes including topology information.

Definition at line 54 of file `VolumeModel.h`.

### 29.541.2 Constructor & Destructor Documentation

29.541.2.1 `Go::VolumeModel::VolumeModel ( std::vector< shared_ptr< ftVolume > > &volumes, double gap, double neighbour, double kink, double bend, bool adjacency_set = false )`

Constructor given a set of volumes and topologic tolerances.

29.541.2.2 `Go::VolumeModel::VolumeModel ( std::vector< shared_ptr< ftVolume > > & volumes, double gap, double kink )`

Constructor given a set of volumes and topologic tolerances.

29.541.2.3 `Go::VolumeModel::VolumeModel ( double gap, double neighbour, double kink, double bend )`

Constructor topologic tolerances.

29.541.2.4 `Go::VolumeModel::VolumeModel ( const VolumeModel & vm )`

Copy constructor.

29.541.2.5 `virtual Go::VolumeModel::~~VolumeModel ( ) [virtual]`

Destructor.

### 29.541.3 Member Function Documentation

29.541.3.1 `bool Go::VolumeModel::allSplines ( ) const`

Check if all entities are NURBS.

29.541.3.2 `void Go::VolumeModel::append ( shared_ptr< ftVolume > volume )`

Append a new body to the volume model. The body is included in the topological structure

29.541.3.3 `void Go::VolumeModel::append ( std::vector< shared_ptr< ftVolume > > volumes )`

Append a vector of bodies to the volume model. The bodies are included in the topological structure

29.541.3.4 `void Go::VolumeModel::append ( shared_ptr< VolumeModel > anotherModel )`

Append all bodies from another volume model. The bodies are included in the topological structure

29.541.3.5 `void Go::VolumeModel::averageCorrespondingCoefs ( )`

Ensure exact match between corresponding coefficients.

29.541.3.6 virtual **BoundingBox** Go::VolumeModel::boundingBox ( ) [virtual]

Bounding box of the entire model.

Implements [Go::CompositeModel](#).

29.541.3.7 virtual **BoundingBox** Go::VolumeModel::boundingBox ( int *idx* ) const [virtual]

Bounding box corresponding to one entity.

Implements [Go::CompositeModel](#).

29.541.3.8 void Go::VolumeModel::buildTopology ( )

Construct the topology information regarding the input geometry Construct the topology information regarding the input geometry for all pairs of volumes

29.541.3.9 void Go::VolumeModel::buildTopology ( shared\_ptr< ftVolume > *body* )

Construct the topology information regarding the input geometry involving a given volume

29.541.3.10 bool Go::VolumeModel::checkModelTopology ( )

Debug.

29.541.3.11 virtual **VolumeModel\*** Go::VolumeModel::clone ( ) const [inline],[virtual]

Make a copy of the current model.

Reimplemented from [Go::CompositeModel](#).

Definition at line 77 of file VolumeModel.h.

29.541.3.12 virtual void Go::VolumeModel::closestPoint ( Point & *pnt*, Point & *clo\_pnt*, int & *idx*, double *clo\_par*[], double & *dist* ) [virtual]

Compute one closest point, interface heritage, not implemented.

Implements [Go::CompositeModel](#).

29.541.3.13 virtual double Go::VolumeModel::curvature ( int *idx*, double \* *par* ) const [virtual]

[Curvature](#), interface heritage, not implemente.

Implements [Go::CompositeModel](#).

29.541.3.14 `virtual void Go::VolumeModel::evaluate ( int idx, double par[], Point & pnt ) const` `[virtual]`

Evaluate position.

Implements [Go::CompositeModel](#).

29.541.3.15 `virtual void Go::VolumeModel::evaluate ( int idx, double par[], int nder, std::vector< Point > & der ) const` `[virtual]`

Evaluate position and a number of derivatives.

Implements [Go::CompositeModel](#).

29.541.3.16 `virtual void Go::VolumeModel::extremalPoint ( Point & dir, Point & ext_pnt, int & idx, double ext_par[] )` `[virtual]`

Extremal point(s) in a given direction

Parameters

<i>dir</i>	Direction
<i>ext_pnt</i>	Found extremal point
<i>idx</i>	Index of curve or surface where the extremal point is found
<i>ext_par</i> []	Parameter value of extremal point

Implements [Go::CompositeModel](#).

29.541.3.17 `shared_ptr<ftVolume> Go::VolumeModel::fetchAsSharedPtr ( Body * body ) const`

Return a specified body as a shared pointer.

29.541.3.18 `void Go::VolumeModel::getAllVertices ( std::vector< shared_ptr< Vertex > > & vertices ) const`

Fetch all vertices in the model.

29.541.3.19 `double Go::VolumeModel::getApproximationTol ( ) const` `[inline]`

Return approximation tolerance.

Returns

Approximation tolerance.

Definition at line 220 of file VolumeModel.h.

29.541.3.20 `shared_ptr<ftVolume> Go::VolumeModel::getBody ( int idx ) const`

Return one body.

29.541.3.21 `std::vector<shared_ptr<ftSurface> > Go::VolumeModel::getBoundaryFaces ( ) const`

Fetch all faces at the outer boundary of this model

29.541.3.22 `std::vector<shared_ptr<ftSurface> > Go::VolumeModel::getBoundaryFaces ( int boundary_idx ) const`

Fetch all faces at one of the boundaries of this model

Parameters

<i>boundary_idx</i>	Index of one boundary
---------------------	-----------------------

Return values

<i>faces</i>	Vector of pointers to the faces at one boundary.
--------------	--------------------------------------------------

29.541.3.23 `std::vector<shared_ptr<VolumeModel> > Go::VolumeModel::getConnectedModels ( )`

Divide this model into connected volume model.

29.541.3.24 `int Go::VolumeModel::getIndex ( shared_ptr< ftVolume > body ) const`

Given a body in the volume model, return the index of this face.

29.541.3.25 `int Go::VolumeModel::getIndex ( ftVolume * body ) const`

Given a body in the volume model, return the index of this face.

29.541.3.26 `shared_ptr<SurfaceModel> Go::VolumeModel::getOuterBoundary ( int idx ) const`

Fetch specified boundary of this model.

29.541.3.27 `void Go::VolumeModel::getRadialEdges ( std::vector< shared_ptr< EdgeVertex > > & rad_edges ) const`

Fetch all radial edges in the model.

29.541.3.28 `shared_ptr<SplineVolume> Go::VolumeModel::getSplineVolume ( int idx ) const`

Return one volume as [SplineVolume](#) if possible.

29.541.3.29 `std::vector<shared_ptr<ftSurface>> Go::VolumeModel::getUniqueInnerFaces ( ) const`

Fetch all interval unique inner faces in this model, i.e. a [ftSurface](#) for each boundary face with a twin, only one of the surfaces in the pair is returned

29.541.3.30 `shared_ptr<ParamVolume> Go::VolumeModel::getVolume ( int idx ) const`

Return one volume.

29.541.3.31 `virtual shared_ptr<IntResultsModel> Go::VolumeModel::intersect ( const ftLine & line ) [virtual]`

Intersection with a line, interface heritage, not implemented. Expected output is points, probably one point. Curves can occur in special configurations.

Implements [Go::CompositeModel](#).

29.541.3.32 `virtual shared_ptr<IntResultsModel> Go::VolumeModel::intersect_plane ( const ftPlane & plane ) [virtual]`

Intersection with a plane, interface heritage, not implemented.

Implements [Go::CompositeModel](#).

29.541.3.33 `bool Go::VolumeModel::isCornerToCorner ( double tol = DEFAULT_SPACE_EPSILON ) const`

Check if the model has got a corner-to-corner configuration.

29.541.3.34 `virtual bool Go::VolumeModel::isDegenerate ( int idx ) const [virtual]`

Whether one particular entity is degenerate, interface heritage, not implemented

Implements [Go::CompositeModel](#).

29.541.3.35 `void Go::VolumeModel::makeCommonSplineSpaces ( )`

Ensure that the blocks in the model has got common spline spaces.

29.541.3.36 `void Go::VolumeModel::makeCornerToCorner ( double tol = DEFAULT_SPACE_EPSILON )`

Ensure that the blocks in the model meet in a corner-to-corner configuration

29.541.3.37 `int Go::VolumeModel::nmbBoundaries ( ) const`

Return the number of boundaries of a volume set (including holes).

29.541.3.38 `virtual int Go::VolumeModel::nmbEntities ( ) const` [virtual]

Number of simple entities.

Implements [Go::CompositeModel](#).

29.541.3.39 `void Go::VolumeModel::regularizeBdShells ( )`

Update the model by regularizing all boundary shells, i.e. all faces in all boundary shells of all connected volumes should be 4-sided, and no T-joints are allowed

29.541.3.40 `void Go::VolumeModel::removeSolid ( shared_ptr< ftVolume > vol )`

Remove one volume from the model.

29.541.3.41 `void Go::VolumeModel::replaceNonRegVolumes ( int degree = 3, int split_mode = 1 )`

Replaces volumes that are not regular with sets of regular volumes.

29.541.3.42 `void Go::VolumeModel::setBoundarySfs ( )`

Compute boundary shells

29.541.3.43 `void Go::VolumeModel::setVertexIdentity ( )`

Add topology information regarding degenerate volumes meeting along an edge

29.541.3.44 `virtual void Go::VolumeModel::tessellate ( std::vector< shared_ptr< GeneralMesh > > & meshes ) const`  
[virtual]

Tessellate model, interface heritage, not implemented.

Implements [Go::CompositeModel](#).

29.541.3.45 `virtual void Go::VolumeModel::tessellate ( int resolution[], std::vector< shared_ptr< GeneralMesh > > & meshes ) const` [virtual]

Tessellate model, interface heritage, not implemented.

Implements [Go::CompositeModel](#).



```
29.541.3.46 virtual void Go::VolumeModel::tessellate (double density, std::vector< shared_ptr< GeneralMesh > > &
 meshes) const [virtual]
```

Tessellate model, interface heritage, not implemented.

Implements [Go::CompositeModel](#).

```
29.541.3.47 virtual void Go::VolumeModel::tesselatedCtrPolygon (std::vector< shared_ptr< LineCloud > > & ctr_pol)
 const [virtual]
```

Tessellate model, interface heritage, not implemented.

Implements [Go::CompositeModel](#).

```
29.541.3.48 virtual void Go::VolumeModel::turn (int idx) [virtual]
```

Interface heritage, not implemented.

Implements [Go::CompositeModel](#).

```
29.541.3.49 virtual void Go::VolumeModel::turn () [virtual]
```

Interface heritage, not implemented.

Implements [Go::CompositeModel](#).

```
29.541.3.50 void Go::VolumeModel::uniqueNonRadialEdges (std::vector< shared_ptr< ftEdge > > & edges) const
```

Fetch edges where no radial edge exist, i.e. there is no adjacent volume along any boundary surface meeting in this edge. Only one occurrence in an edge, twin-edge pair is returned

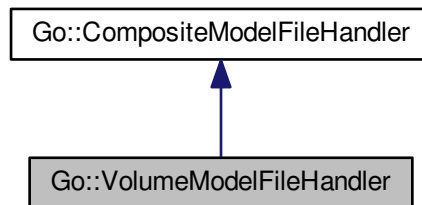
The documentation for this class was generated from the following file:

- [trivariatemodel/include/GoTools/trivariatemodel/VolumeModel.h](#)

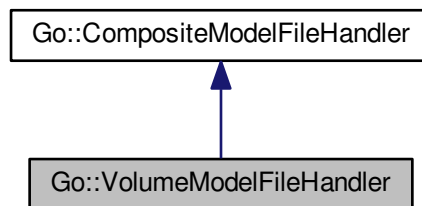
## 29.542 Go::VolumeModelFileHandler Class Reference

```
#include <VolumeModelFileHandler.h>
```

Inheritance diagram for Go::VolumeModelFileHandler:



Collaboration diagram for Go::VolumeModelFileHandler:



### Public Member Functions

- [VolumeModelFileHandler](#) ()
- [~VolumeModelFileHandler](#) ()
- void [writeVolume](#) (const shared\_ptr< [ftVolume](#) > &body, std::ostream &os, int body\_id=-1, bool faces=true)
- shared\_ptr< [ftVolume](#) > [readVolume](#) (const char \*filein, int id=-1)
- void [writeVolumeModel](#) ([VolumeModel](#) &vol\_model, std::ostream &os)
- shared\_ptr< [VolumeModel](#) > [readVolumeModel](#) (const char \*filein)

### Additional Inherited Members

#### 29.542.1 Detailed Description

Definition at line 54 of file [VolumeModelFileHandler.h](#).

## 29.542.2 Constructor & Destructor Documentation

29.542.2.1 `Go::VolumeModelFileHandler::VolumeModelFileHandler ( )`

29.542.2.2 `Go::VolumeModelFileHandler::~~VolumeModelFileHandler ( )`

## 29.542.3 Member Function Documentation

29.542.3.1 `shared_ptr<ftVolume> Go::VolumeModelFileHandler::readVolume ( const char * filein, int id = -1 )`

29.542.3.2 `shared_ptr<VolumeModel> Go::VolumeModelFileHandler::readVolumeModel ( const char * filein )`

29.542.3.3 `void Go::VolumeModelFileHandler::writeVolume ( const shared_ptr< ftVolume > & body, std::ostream & os, int body_id = -1, bool faces = true )`

29.542.3.4 `void Go::VolumeModelFileHandler::writeVolumeModel ( VolumeModel & vol_model, std::ostream & os )`

The documentation for this class was generated from the following file:

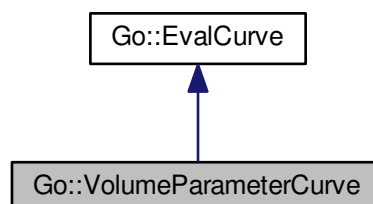
- [trivariatemodel/include/GoTools/trivariatemodel/VolumeModelFileHandler.h](#)

## 29.543 Go::VolumeParameterCurve Class Reference

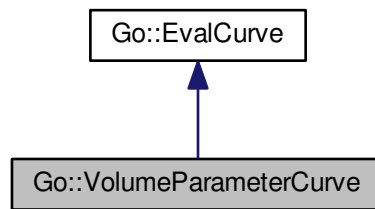
An evaluator based curve representing the parameter domain curve in a given volume which represents the same curve as a given space curve. Project a geometry curve into the parameter domain of a volume.

```
#include <VolumeParameterCurve.h>
```

Inheritance diagram for `Go::VolumeParameterCurve`:



Collaboration diagram for Go::VolumeParameterCurve:



## Public Member Functions

- `VolumeParameterCurve` (`shared_ptr< ParamVolume > vol`, `shared_ptr< ParamCurve > crv`)
- `VolumeParameterCurve` (`shared_ptr< ParamVolume > vol`, `shared_ptr< ParamCurve > crv`, `shared_ptr< Point > par1`, `shared_ptr< Point > par2`)
- `virtual ~VolumeParameterCurve ()`  
*virtual destructor ensures save inheritance*
- `virtual Point eval (double t) const`
- `virtual void eval (double t, int n, Point der[]) const`
- `virtual double start () const`
- `virtual double end () const`
- `virtual int dim () const`
- `virtual bool approximationOK (double par, Point approxpos, double tol1, double tol2) const`

### 29.543.1 Detailed Description

An evaluator based curve representing the parameter domain curve in a given volume which represents the same curve as a given space curve. Project a geometry curve into the parameter domain of a volume.

Definition at line 58 of file `VolumeParameterCurve.h`.

### 29.543.2 Constructor & Destructor Documentation

29.543.2.1 `Go::VolumeParameterCurve::VolumeParameterCurve ( shared_ptr< ParamVolume > vol, shared_ptr< ParamCurve > crv )`

Constructor

Parameters

<code>vol</code>	volume in which the curve lies
<code>crv</code>	space curve representing the same curve

29.543.2.2 `Go::VolumeParameterCurve::VolumeParameterCurve ( shared_ptr< ParamVolume > vol, shared_ptr< ParamCurve > crv, shared_ptr< Point > par1, shared_ptr< Point > par2 )`

Constructor

Parameters

<i>vol</i>	volume in which the curve lies
<i>crv</i>	space curve representing the same curve
<i>par1</i>	startparameter of the final curve <code>par1 &gt;= crv-&gt;startparam()</code>
<i>par2</i>	endparameter of the final curve <code>par1 &lt;= crv-&gt;endparam()</code>

29.543.2.3 `virtual Go::VolumeParameterCurve::~~VolumeParameterCurve ( ) [virtual]`

virtual destructor ensures save inheritance

### 29.543.3 Member Function Documentation

29.543.3.1 `virtual bool Go::VolumeParameterCurve::approximationOK ( double par, Point approxpos, double tol1, double tol2 ) const [virtual]`

Check if the curve, evaluated at a given parameter, approximates a given position within a given tolerance.

Parameters

<i>par</i>	the parameter at which to check the curve
<i>approxpos</i>	the position we want to check whether or not the curve approximates for parameter 'par'.
<i>tol1</i>	approximation tolerance.
<i>tol2</i>	another approximation tolerance (its use is defined by some of the derived classes.

Returns

'true' if the curve approximates the point at the parameter, 'false' otherwise.

Implements [Go::EvalCurve](#).

29.543.3.2 `virtual int Go::VolumeParameterCurve::dim ( ) const [virtual]`

Get the dimension of the space in which the curve lies.

Returns

the space dimension of the curve.

Implements [Go::EvalCurve](#).

29.543.3.3 virtual double Go::VolumeParameterCurve::end ( ) const [virtual]

Get the end parameter of the curve.

#### Returns

the end parameter of the curve.

Implements [Go::EvalCurve](#).

29.543.3.4 virtual Point Go::VolumeParameterCurve::eval ( double t ) const [virtual]

Evaluate a point on the curve for a given parameter

#### Parameters

<i>t</i>	the parameter for which to evaluate the curve.
----------	------------------------------------------------

#### Returns

the evaluated point

Implements [Go::EvalCurve](#).

29.543.3.5 virtual void Go::VolumeParameterCurve::eval ( double t, int n, Point der[] ) const [virtual]

Evaluate a point and a certain number of derivatives on the curve for a given parameter.

#### Parameters

<i>t</i>	the parameter for which to evaluate the curve.
<i>n</i>	the number of derivatives (0 or more)

#### Return values

<i>der</i>	pointer to an array of Points where the result will be written. The position will be stored first, then the first derivative (tangent), then the second, etc.. <b>NB:</b> For most (all) derived classes of ' <a href="#">EvalCurve</a> ', the implementation actually only supports the computation of one derivative, i.e. if $n > 1$ , only one derivative will be computed anyway.
------------	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Implements [Go::EvalCurve](#).

29.543.3.6 virtual double Go::VolumeParameterCurve::start ( ) const [virtual]

Get the start parameter of the curve.

**Returns**

the start parameter of the curve.

Implements [Go::EvalCurve](#).

The documentation for this class was generated from the following file:

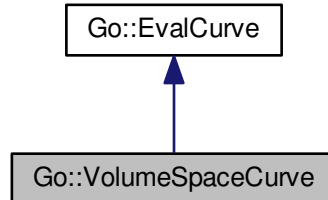
- [trivariate/include/GoTools/trivariate/VolumeParameterCurve.h](#)

## 29.544 Go::VolumeSpaceCurve Class Reference

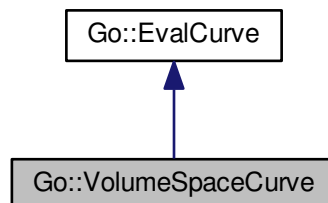
An evaluator based curve representing the space curve corresponding to parameter domain curve in a given volume. Compute the space curve corresponding to a curve in the parameter domain of a volume.

```
#include <VolumeSpaceCurve.h>
```

Inheritance diagram for Go::VolumeSpaceCurve:



Collaboration diagram for Go::VolumeSpaceCurve:



## Public Member Functions

- [VolumeSpaceCurve](#) (shared\_ptr< [ParamVolume](#) > vol, shared\_ptr< [ParamCurve](#) > crv)
- virtual [~VolumeSpaceCurve](#) ()  
*virtual destructor ensures save inheritance*
- virtual [Point](#) [eval](#) (double t) const
- virtual void [eval](#) (double t, int n, [Point](#) der[]) const
- virtual double [start](#) () const
- virtual double [end](#) () const
- virtual int [dim](#) () const
- virtual bool [approximationOK](#) (double par, [Point](#) approxpos, double tol1, double tol2) const

### 29.544.1 Detailed Description

An evaluator based curve representing the space curve corresponding to parameter domain curve in a given volume. Compute the space curve corresponding to a curve in the parameter domain of a volume.

Definition at line 59 of file [VolumeSpaceCurve.h](#).

### 29.544.2 Constructor & Destructor Documentation

29.544.2.1 `Go::VolumeSpaceCurve::VolumeSpaceCurve ( shared_ptr< ParamVolume > vol, shared_ptr< ParamCurve > crv )`

Constructor

Parameters

<i>vol</i>	volume in which the curve lies
<i>crv</i>	curve in the parameter domain of the given volume

29.544.2.2 `virtual Go::VolumeSpaceCurve::~~VolumeSpaceCurve ( ) [virtual]`

virtual destructor ensures save inheritance

### 29.544.3 Member Function Documentation

29.544.3.1 `virtual bool Go::VolumeSpaceCurve::approximationOK ( double par, Point approxpos, double tol1, double tol2 ) const [virtual]`

Check if the curve, evaluated at a given parameter, approximates a given position within a given tolerance.

Parameters

<i>par</i>	the parameter at which to check the curve
<i>approxpos</i>	the position we want to check whether or not the curve approximates for parameter 'par'.
<i>tol1</i>	approximation tolerance.
<i>tol2</i>	another approximation tolerance (its use is defined by some of the derived classes.



**Returns**

'true' if the curve approximates the point at the parameter, 'false' otherwise.

Implements [Go::EvalCurve](#).

**29.544.3.2** `virtual int Go::VolumeSpaceCurve::dim ( ) const [virtual]`

Get the dimension of the space in which the curve lies.

**Returns**

the space dimension of the curve.

Implements [Go::EvalCurve](#).

**29.544.3.3** `virtual double Go::VolumeSpaceCurve::end ( ) const [virtual]`

Get the end parameter of the curve.

**Returns**

the end parameter of the curve.

Implements [Go::EvalCurve](#).

**29.544.3.4** `virtual Point Go::VolumeSpaceCurve::eval ( double t ) const [virtual]`

Evaluate a point on the curve for a given parameter

**Parameters**

<i>t</i>	the parameter for which to evaluate the curve.
----------	------------------------------------------------

**Returns**

the evaluated point

Implements [Go::EvalCurve](#).

**29.544.3.5** `virtual void Go::VolumeSpaceCurve::eval ( double t, int n, Point der[] ) const [virtual]`

Evaluate a point and a certain number of derivatives on the curve for a given parameter.

**Parameters**

<i>t</i>	the parameter for which to evaluate the curve.
<i>n</i>	the number of derivatives (0 or more)

## Return values

<i>der</i>	pointer to an array of Points where the result will be written. The position will be stored first, then the first derivative (tangent), then the second, etc.. <b>NB:</b> For most (all) derived classes of 'EvalCurve', the implementation actually only supports the computation of one derivative, i.e. if $n > 1$ , only one derivative will be computed anyway.
------------	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Implements [Go::EvalCurve](#).

**29.544.3.6** `virtual double Go::VolumeSpaceCurve::start ( ) const [virtual]`

Get the start parameter of the curve.

## Returns

the start parameter of the curve.

Implements [Go::EvalCurve](#).

The documentation for this class was generated from the following file:

- [trivariate/include/GoTools/trivariate/VolumeSpaceCurve.h](#)

## 29.545 Go::Zero\_Parameter\_Span\_Error Class Reference

Error object used internally in [IntersectionCurve](#).

```
#include <IntersectionCurve.h>
```

### 29.545.1 Detailed Description

Error object used internally in [IntersectionCurve](#).

Definition at line 60 of file [IntersectionCurve.h](#).

The documentation for this class was generated from the following file:

- [intersections/include/GoTools/intersections/IntersectionCurve.h](#)

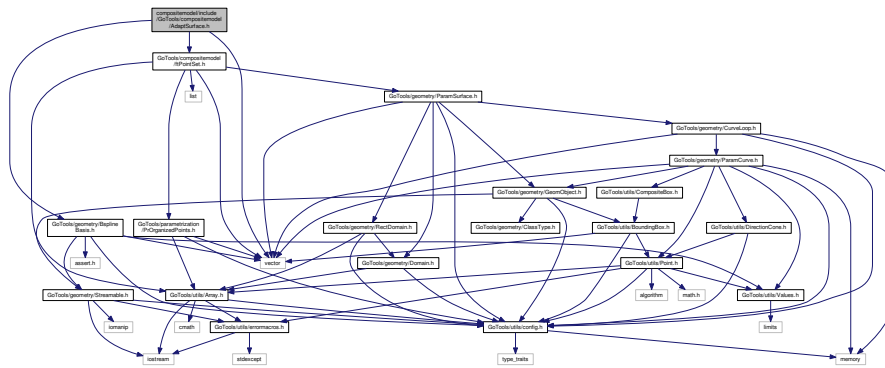
# Chapter 30

## File Documentation

### 30.1 compositemodel/include/GoTools/compositemodel/AdaptSurface.h File Reference

```
#include "GoTools/geometry/BsplineBasis.h"
#include "GoTools/compositemodel/ftPointSet.h"
#include <vector>
```

Include dependency graph for AdaptSurface.h:



#### Namespaces

- [Go](#)
- [Go::AdaptSurface](#)

#### Functions

- `shared_ptr< SplineSurface > Go::AdaptSurface::approxInSplineSpace (shared_ptr< ParamSurface > surf, shared_ptr< SplineSurface > surf2, double tol)`
- `std::vector< shared_ptr< SplineSurface > > Go::AdaptSurface::expressInSameSplineSpace (shared_ptr< ParamSurface > surf1, shared_ptr< ParamSurface > surf2, double tol)`
- `bool Go::AdaptSurface::getCornerCorrespondance (shared_ptr< ParamSurface > surf1, shared_ptr< ParamSurface > surf2, int &idx, bool &turned)`
- `std::vector< shared_ptr< SplineCurve > > Go::AdaptSurface::curveApprox (shared_ptr< ParamCurve > cvs[], int nmb_cvs, const BsplineBasis &init_basis, double tol)`

- `std::vector< shared_ptr< SplineCurve > >` [Go::AdaptSurface::curveApprox](#) (`shared_ptr< ParamCurve >` cvs[], `int` nmb\_cvs, `double` tol, `double` degree=3)
- `shared_ptr< SplineSurface >` [Go::AdaptSurface::adaptSurface](#) (`shared_ptr< ParamSurface >` surf, `shared_ptr< SplineSurface >` init\_surf, `double` tol)
- `double` [Go::AdaptSurface::parameterizePoints](#) (`shared_ptr< SplineSurface >` init\_surf, `shared_ptr< ftPointSet >` points, `std::vector< int >` corner)
- `double` [Go::AdaptSurface::projectPoints](#) (`shared_ptr< SplineSurface >` surf, `shared_ptr< ftPointSet >` points)

*Parameterize by projection.*

- `shared_ptr< SplineSurface >` [Go::AdaptSurface::doApprox](#) (`shared_ptr< SplineSurface >` init\_surf, `int` max\_iter, `shared_ptr< ftPointSet >` points, `double` tol, `double` &max\_error, `double` &mean\_error)
- `void` [Go::AdaptSurface::createTriangulation](#) (`shared_ptr< ParamSurface >` surf, `const` RectDomain &dom, `shared_ptr< ftPointSet >` &points, `vector< int >` &corner, `bool` consider\_joint=true, `int` nmb=-1)

*Fetch data and create triangulation.*

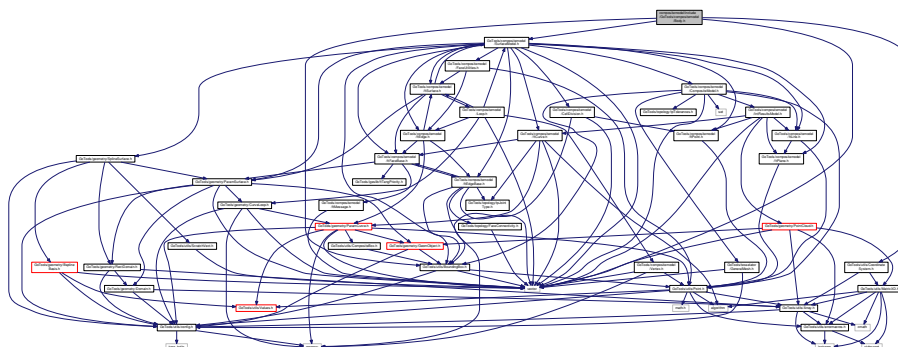
- `void` [Go::AdaptSurface::getBoundaryData](#) (`shared_ptr< ParamSurface >` surf, `const` RectDomain &dom, `int` nmb\_sample, `shared_ptr< ftPointSet >` points, `std::vector< int >` &corner)
- `void` [Go::AdaptSurface::getInnerData](#) (`shared_ptr< ParamSurface >` surf, `const` RectDomain &dom, `int` nmb\_sample, `shared_ptr< ftPointSet >` points, `bool` consider\_joint=true)
- `void` [Go::AdaptSurface::updatePointTopology](#) (`shared_ptr< ParamSurface >` surf, `ftPointSet` &points)

*Compute point set topology.*

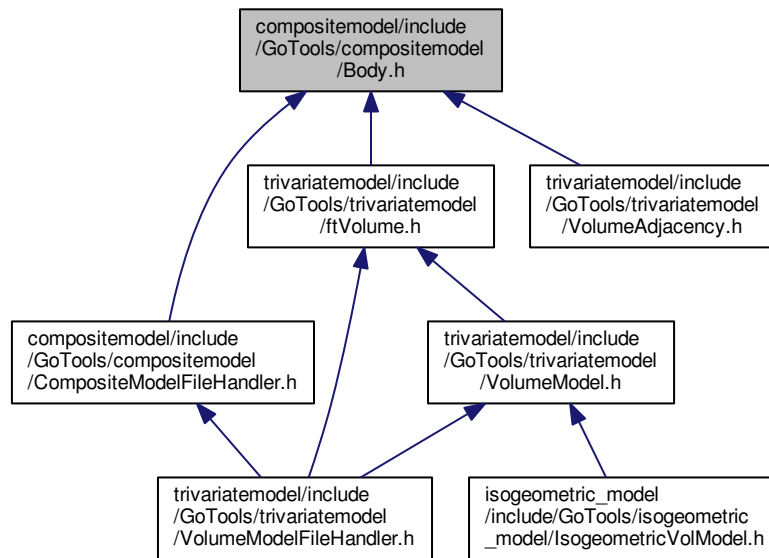
## 30.2 compositemodel/include/GoTools/compositemodel/Body.h File Reference

```
#include "GoTools/compositemodel/SurfaceModel.h"
#include "GoTools/utils/CoordinateSystem.h"
#include "GoTools/utils/BoundingBox.h"
#include <vector>
```

Include dependency graph for Body.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [Go::Body](#)

*A boundary represented solid.*

## Namespaces

- [Go](#)

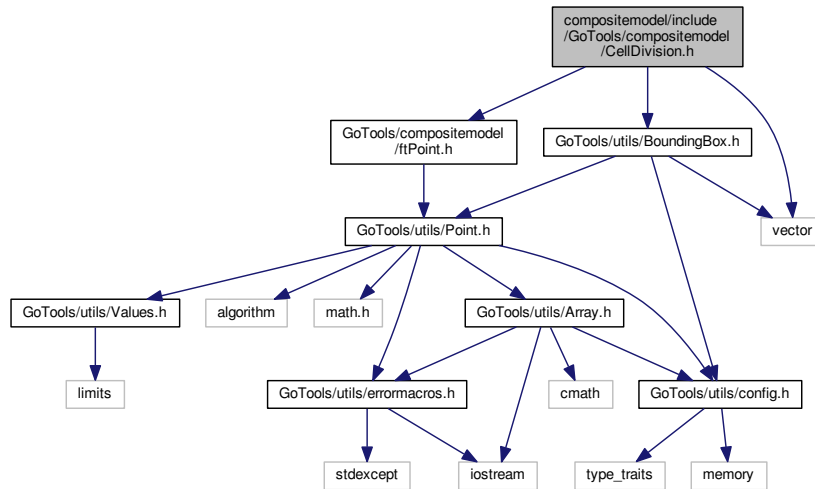
## 30.3 compositemodel/include/GoTools/compositemodel/CellDivision.h File Reference

```

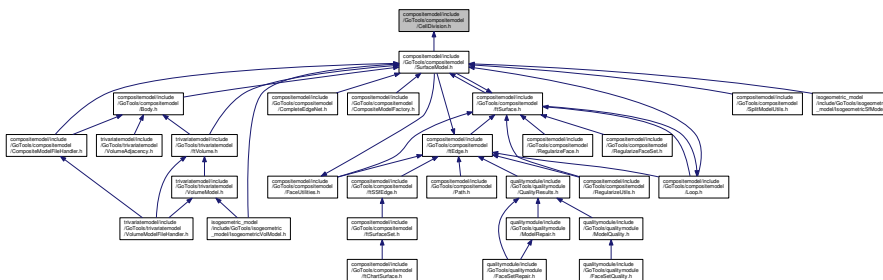
#include "GoTools/compositemodel/ftPoint.h"
#include "GoTools/utils/BoundingBox.h"
#include <vector>

```

Include dependency graph for CellDivision.h:



This graph shows which files directly or indirectly include this file:



## Classes

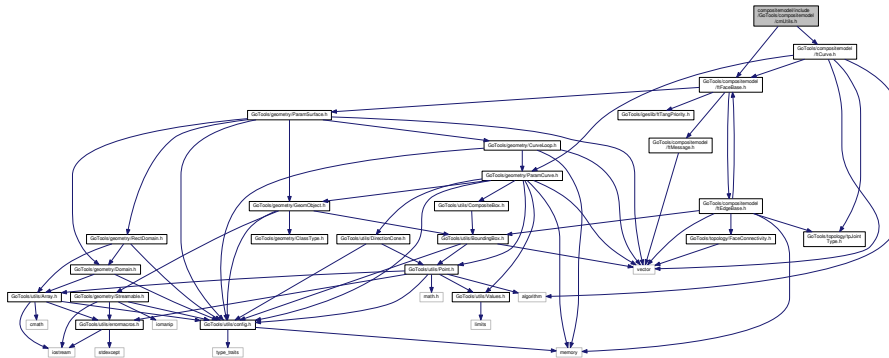
- class [Go::ftCell](#)  
*Helper class handling one of the cells in class [CellDivision](#).*
- class [Go::ftCellInfo](#)  
*Function object for sorting ftCells.*
- class [Go::CellDivision](#)

## Namespaces

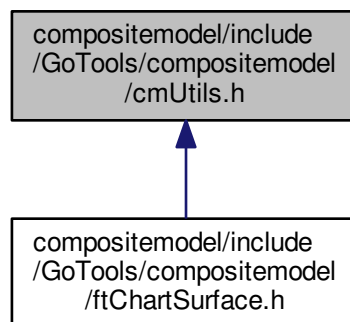
- [Go](#)

## 30.4 compositemodel/include/GoTools/compositemodel/cmUtils.h File Reference

```
#include "GoTools/compositemodel/ftFaceBase.h"
#include "GoTools/compositemodel/ftCurve.h"
Include dependency graph for cmUtils.h:
```



This graph shows which files directly or indirectly include this file:



### Classes

- struct [Go::RotationInfo](#)

### Namespaces

- [Go](#)
- [Go::cmUtils](#)

*Various utility functions for the compositemodel module.*

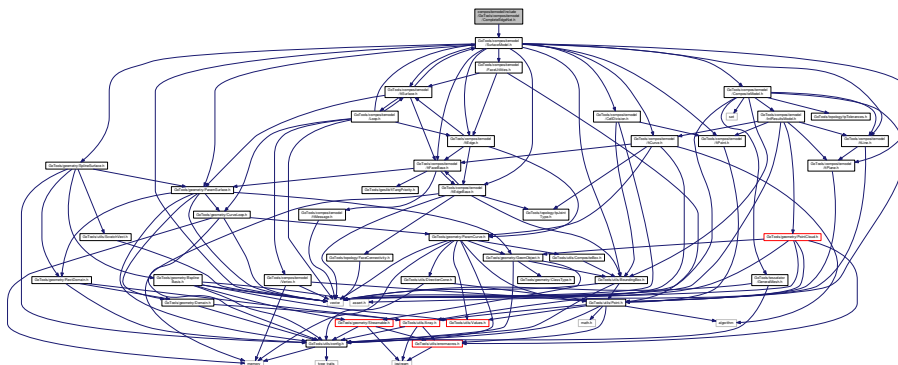
## Functions

- `double Go::cmUtils::estimatedCurveLength` (ftEdgeBase \*edge, int nmb\_samples=4)  
*Estimate the curve length corresponding to an edge.*
- RectDomain `Go::cmUtils::geometricParamDomain` (ParamSurface \*sf)  
*Domain of surface is rescaled according to surface lengths in geometry space.*
- `bool Go::cmUtils::abovePlane` (Go::Point pt, Go::Point plane\_pt, Go::Point normal)  
*Return true if pt lies above plane as defined.*
- `void Go::cmUtils::extendWithDegBd` (std::vector< int > &corner, std::vector< shared\_ptr< Go::ParamCurve > > &bd\_curves, std::vector< shared\_ptr< Go::ParamCurve > > &cross\_curves, int idxmin)
- `void Go::cmUtils::updateWithNewCorners` (const std::vector< Go::ftEdgeBase \* > &outer\_loop, std::vector< int > &corners, const std::vector< Go::Point > &add\_corner\_pts, double gap)  
*Assuming input corner pts lie inside gap of outer\_loop, and not in the middle of a segment.*
- `bool Go::cmUtils::insideBdTrain` (const Go::Vector2D &sample\_pt, const std::vector< Go::Vector2D > &bd\_train)  
*bd\_train must be a simple loop, direction may be either.*
- `void Go::cmUtils::reparametrizeBdCvs` (const std::vector< shared\_ptr< Go::SplineCurve > > &bd\_cvs, double appr\_tol, std::vector< shared\_ptr< Go::SplineCurve > > &new\_bd\_cvs)
- `void Go::cmUtils::reparametrizeBdCvs2` (const std::vector< shared\_ptr< Go::SplineCurve > > &bd\_cvs, double appr\_tol, std::vector< shared\_ptr< Go::SplineCurve > > &new\_bd\_cvs)
- `void Go::cmUtils::splitEdgesInCorners` (std::vector< shared\_ptr< Go::ftEdgeBase > > &edges, const std::vector< Go::Point > &corner\_pts, double gap)  
*interior of an edge. To be called prior to connection with twin.*
- `void Go::cmUtils::cwOrientation` (std::vector< Go::ftEdgeBase \* > &meeting\_edges, std::vector< bool > &start, double angle\_tol=1e-05)
- `void Go::cmUtils::cwOrientation2` (std::vector< Go::ftEdgeBase \* > &meeting\_edges, std::vector< bool > &start)
- `std::vector< std::pair< shared_ptr< Go::ParamCurve >, Go::ftFaceBase * > >` `Go::cmUtils::getG1FaceCurves` (Go::ftCurve &bd\_curve)  
*Divide the input curve into G1 segments lying on one surface only (input expected to lie on bd).*
- `double Go::cmUtils::ccwAngle` (Go::Point first\_leg, Go::Point second\_leg, Go::Point \*normal=NULL)
- `std::vector< int >` `Go::cmUtils::removeInnerCorners` (const std::vector< Go::ftEdgeBase \* > &outer\_loop, std::vector< int > &corners)

## 30.5 compositemodel/include/GoTools/compositemodel/CompleteEdgeNet.h File Reference

```
#include "GoTools/compositemodel/SurfaceModel.h"
```

Include dependency graph for CompleteEdgeNet.h:





## Classes

- class [Go::CompleteEdgeNet](#)

Complete the edge net of a [SurfaceModel](#). Fetches the wire frame model corresponding to a surface model, and extends it such that the extended wire frame will become the wire frame model corresponding to a volume model have the given surface model as its outer boundary. The extension to the wireframe is represented as pairs of vertices where these vertices lie at the endpoints of the missing edges. NB! This solution is currently not expected to handle all configurations.

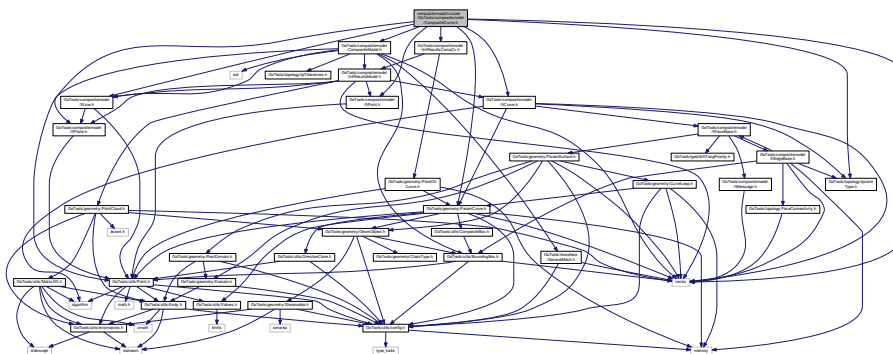
## Namespaces

- [Go](#)

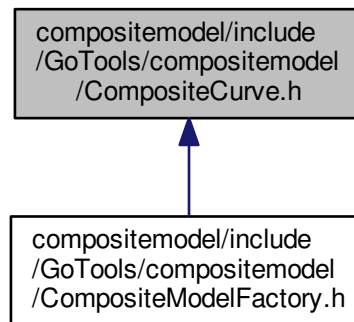
## 30.6 compositemodel/include/GoTools/compositemodel/CompositeCurve.h File Reference

```
#include "GoTools/topology/tpJointType.h"
#include "GoTools/compositemodel/CompositeModel.h"
#include "GoTools/geometry/ParamCurve.h"
#include "GoTools/compositemodel/ftPoint.h"
#include "GoTools/compositemodel/ftCurve.h"
#include "GoTools/compositemodel/ftPlane.h"
#include "GoTools/compositemodel/ftLine.h"
#include "GoTools/compositemodel/IntResultsCompCv.h"
#include <vector>
```

Include dependency graph for CompositeCurve.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [Go::CompositeCurve](#)

## Namespaces

- [Go](#)

### 30.7 compositemodel/include/GoTools/compositemodel/compositemodel-doxymain.h File Reference

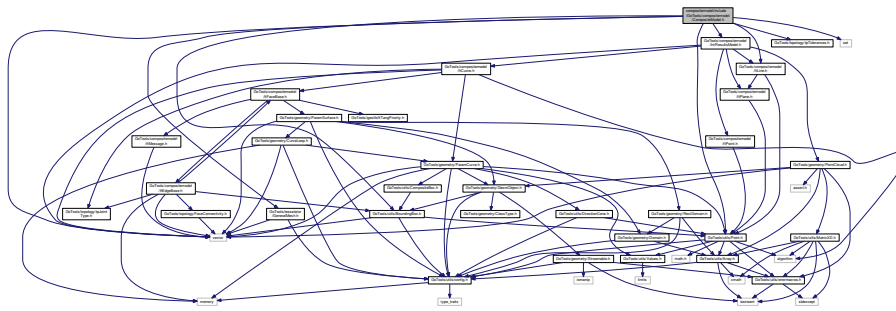
### 30.8 compositemodel/include/GoTools/compositemodel/CompositeModel.h File Reference

```

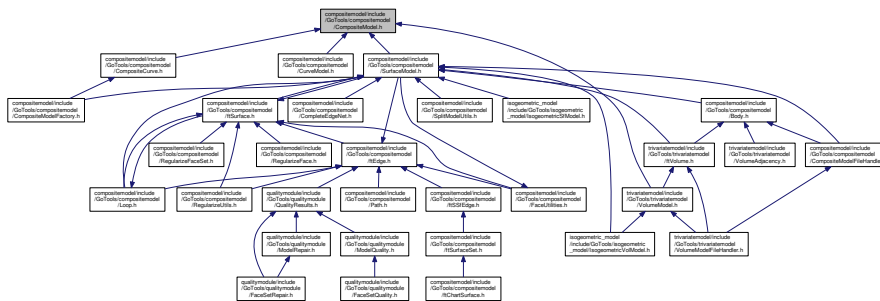
#include "GoTools/utils/Point.h"
#include "GoTools/utils/BoundingBox.h"
#include "GoTools/topology/tpTolerances.h"
#include "GoTools/tesselator/GeneralMesh.h"
#include "GoTools/compositemodel/ftLine.h"
#include "GoTools/compositemodel/IntResultsModel.h"
#include <vector>
#include <set>

```

Include dependency graph for CompositeModel.h:



This graph shows which files directly or indirectly include this file:



### Classes

- class [Go::CompositeModel](#)

### Namespaces

- [Go](#)

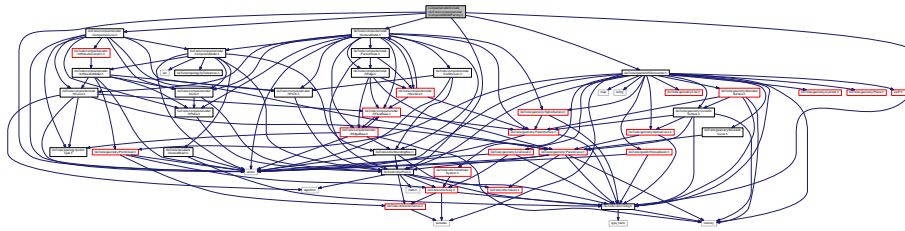
### Enumerations

- enum [Go::closestPointLevel](#) { [Go::LOCAL\\_SEARCH](#) = 1, [Go::SEMI\\_LOCAL\\_SEARCH](#), [Go::GLOBAL\\_SEARCH](#) }

## 30.9 compositemodel/include/GoTools/compositemodel/CompositeModelFactory.h File Reference

```
#include "GoTools/utils/config.h"
#include "GoTools/compositemodel/SurfaceModel.h"
#include "GoTools/compositemodel/CompositeCurve.h"
#include "GoTools/igeslib/IGESconverter.h"
#include "GoTools/utils/Point.h"
#include <vector>
```

Include dependency graph for CompositeModelFactory.h:



## Classes

- class [Go::CompositeModelFactory](#)

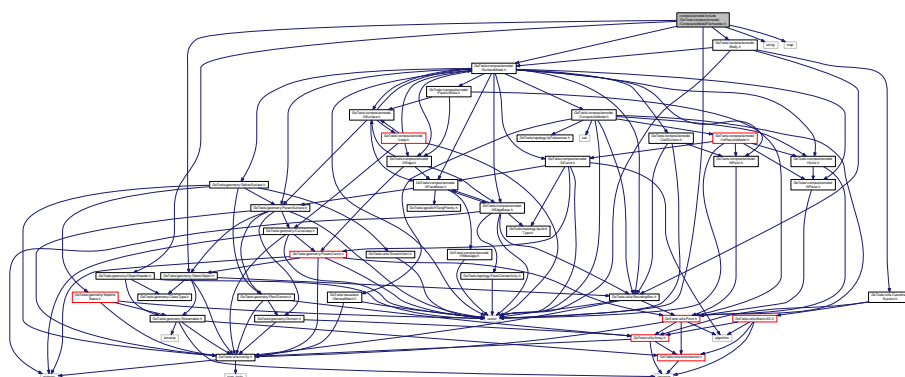
## Namespaces

- [Go](#)

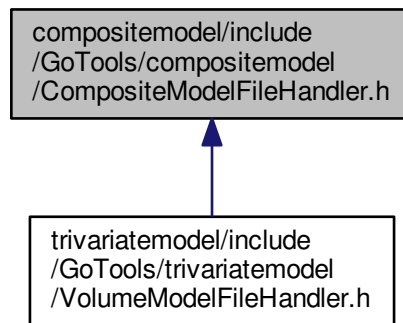
## 30.10 compositemodel/include/GoTools/compositemodel/CompositeModelFileHandler.h File Reference

```
#include "GoTools/geometry/ObjectHeader.h"
#include "GoTools/geometry/GeomObject.h"
#include "GoTools/compositemodel/SurfaceModel.h"
#include "GoTools/compositemodel/Body.h"
#include <string>
#include <vector>
#include <map>
```

Include dependency graph for CompositeModelFileHandler.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [Go::CompositeModelFileHandler](#)
- struct [Go::CompositeModelFileHandler::sfcvinfo](#)

## Namespaces

- [Go](#)

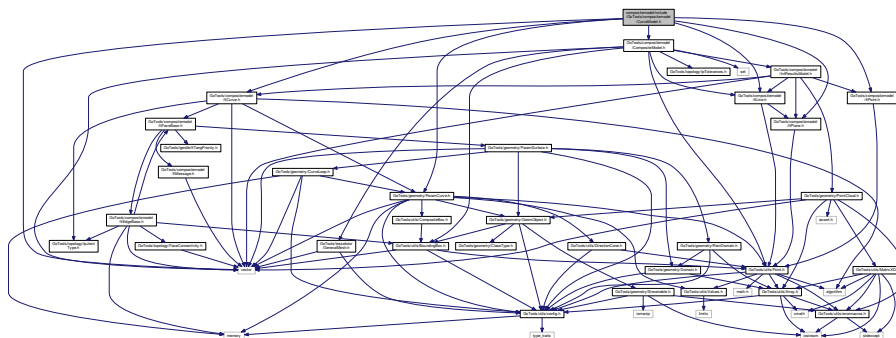
## 30.11 compositemodel/include/GoTools/compositemodel/CurveModel.h File Reference

```

#include "GoTools/compositemodel/CompositeModel.h"
#include "GoTools/geometry/ParamCurve.h"
#include "GoTools/compositemodel/ftPoint.h"
#include "GoTools/compositemodel/ftCurve.h"
#include "GoTools/compositemodel/ftPlane.h"
#include "GoTools/compositemodel/ftLine.h"
#include <vector>

```

Include dependency graph for CurveModel.h:



## Classes

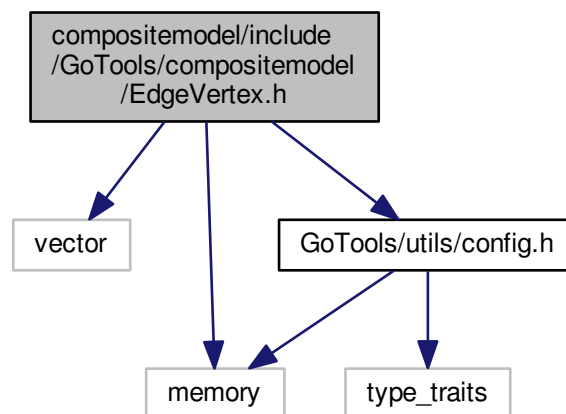
- class [Go::CurveModel](#)

## Namespaces

- [Go](#)

## 30.12 compositemodel/include/GoTools/compositemodel/EdgeVertex.h File Reference

```
#include <vector>
#include <memory>
#include "GoTools/utils/config.h"
Include dependency graph for EdgeVertex.h:
```



## Classes

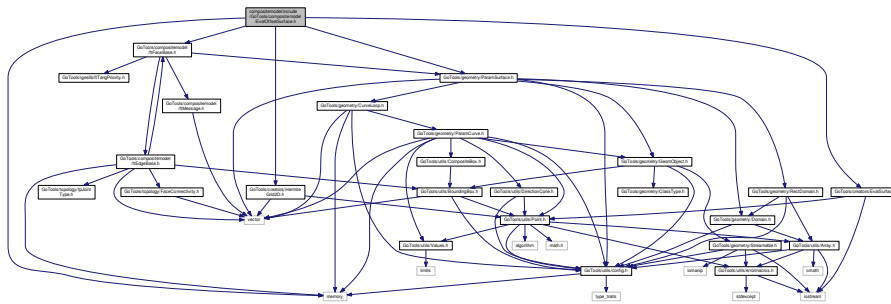
- class [Go::EdgeVertex](#)

## Namespaces

- [Go](#)

## 30.13 compositemodel/include/GoTools/compositemodel/EvalOffsetSurface.h File Reference

```
#include <memory>
#include "GoTools/compositemodel/ftFaceBase.h"
#include "GoTools/geometry/ParamSurface.h"
#include "GoTools/creators/EvalSurface.h"
#include "GoTools/creators/HermiteGrid2D.h"
Include dependency graph for EvalOffsetSurface.h:
```



### Classes

- class [Go::BaseSurface](#)
- class [Go::EvalOffsetSurface](#)

### Namespaces

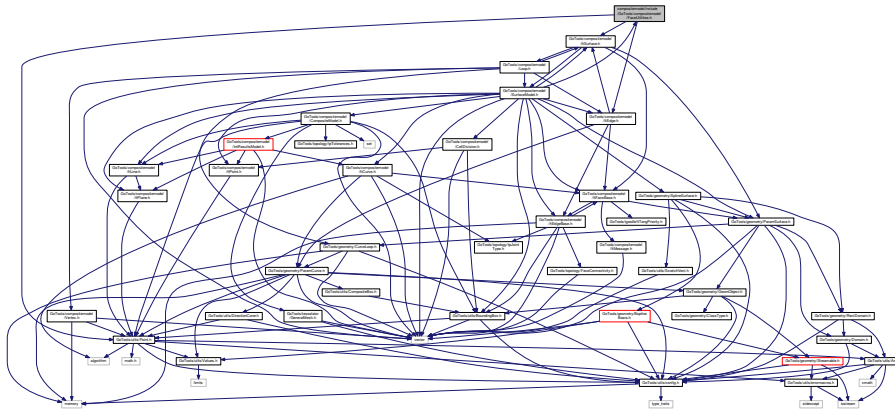
- [Go](#)

## 30.14 compositemodel/include/GoTools/compositemodel/examples\_compositemodel\_↔doxygen.h File Reference

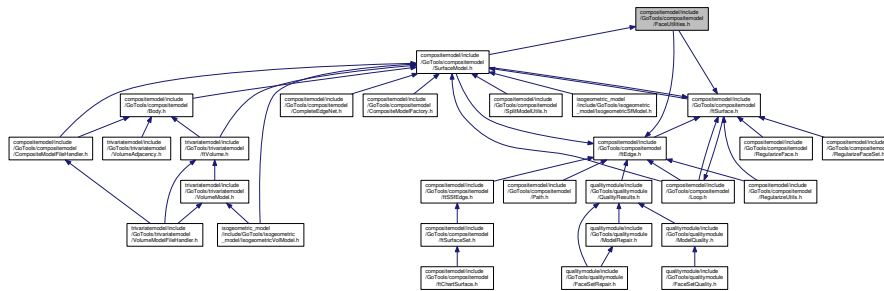
## 30.15 compositemodel/include/GoTools/compositemodel/FaceUtilities.h File Reference

```
#include "GoTools/utils/Point.h"
#include "GoTools/compositemodel/ftSurface.h"
#include "GoTools/compositemodel/ftEdge.h"
```

Include dependency graph for FaceUtilities.h:



This graph shows which files directly or indirectly include this file:



## Classes

- struct [Go::SamplePointData](#)

Sample data related to one face. The struct contains one point in a point set with information about position, associated surface normal and curvature. Points lying on the face boundary know about its associated edge. If the surface normal and curvature information regarding boundary points is not unique, the associated data is set to be equal to `MAX_DOUBLE`.

## Namespaces

- [Go](#)
- [Go::FaceUtilities](#)

## Functions

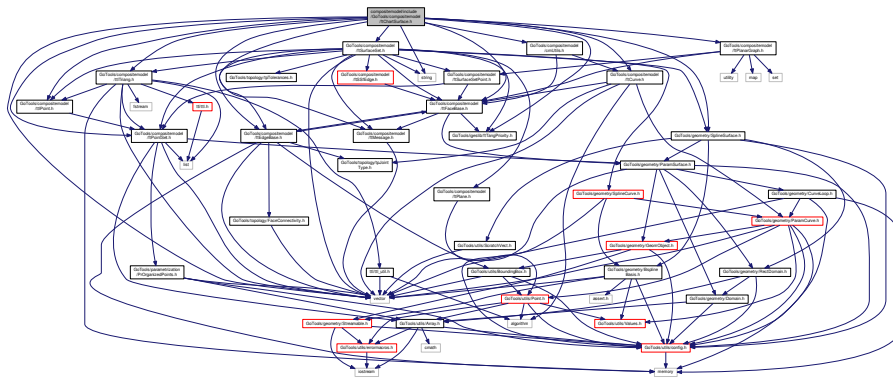
- void [Go::FaceUtilities::getBoundaryData](#) (ftSurface \*face, int nmb\_sample, std::vector< SamplePointData > &sample\_points)
- void [Go::FaceUtilities::getInnerData](#) (ftSurface \*face, int nmb\_sample\_u, int nmb\_sample\_v, std::vector< SamplePointData > &sample\_points)
- bool [Go::FaceUtilities::enforceCoLinearity](#) (ftSurface \*face1, ftEdge \*edge1, ftSurface \*face2, double tol, double ang\_tol)
- bool [Go::FaceUtilities::enforceVxCoLinearity](#) (shared\_ptr< Vertex > vx, double tol, double ang\_tol)



## 30.16 compositemodel/include/GoTools/compositemodel/ftChartSurface.h File Reference

```
#include <vector>
#include <string>
#include "GoTools/compositemodel/ftMessage.h"
#include "GoTools/igeslib/ftTangPriority.h"
#include "GoTools/compositemodel/ftPointSet.h"
#include "GoTools/compositemodel/ftFaceBase.h"
#include "GoTools/compositemodel/ftEdgeBase.h"
#include "GoTools/geometry/SplineSurface.h"
#include "GoTools/compositemodel/ttlPoint.h"
#include "GoTools/compositemodel/ttlTriang.h"
#include "GoTools/compositemodel/ftSurfaceSet.h"
#include "GoTools/compositemodel/ftCurve.h"
#include "GoTools/compositemodel/cmUtils.h"
#include "GoTools/compositemodel/ftPlanarGraph.h"
```

Include dependency graph for ftChartSurface.h:



### Classes

- class [Go::ftChartSurface](#)

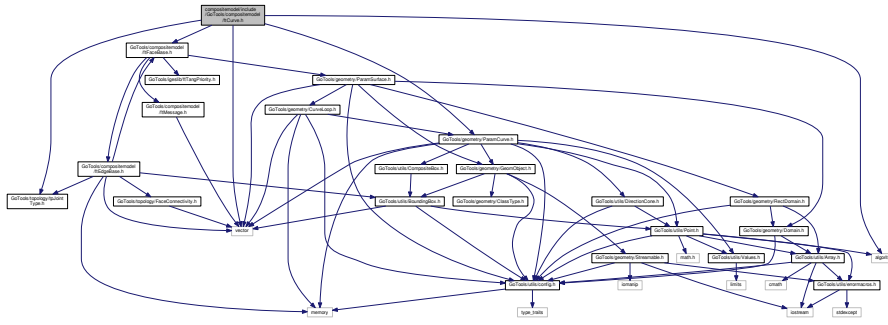
### Namespaces

- [Go](#)

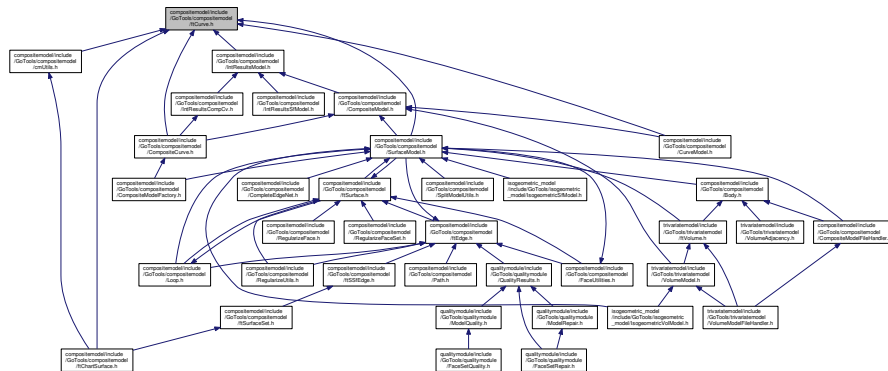
## 30.17 compositemodel/include/GoTools/compositemodel/ftCurve.h File Reference

```
#include <algorithm>
#include <vector>
#include "GoTools/geometry/ParamCurve.h"
#include "GoTools/topology/tpJointType.h"
#include "GoTools/compositemodel/ftFaceBase.h"
```

Include dependency graph for `ftCurve.h`:



This graph shows which files directly or indirectly include this file:



## Classes

- class `Go::ftCurveSegment`
- class `Go::ftCurve`

## Namespaces

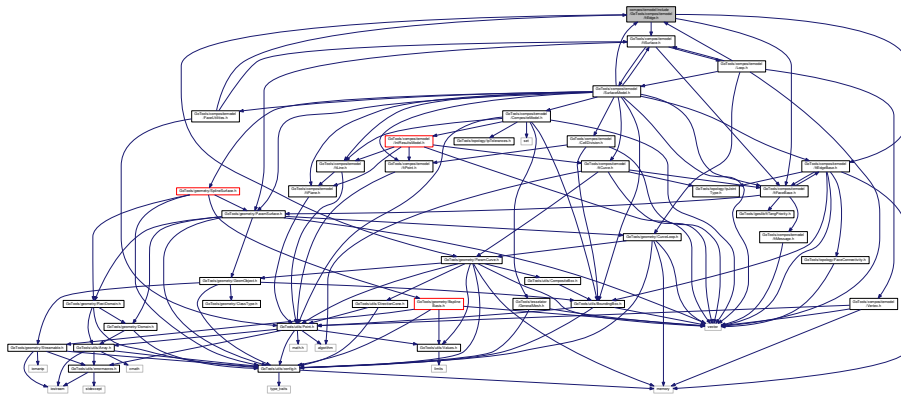
- `Go`

## Enumerations

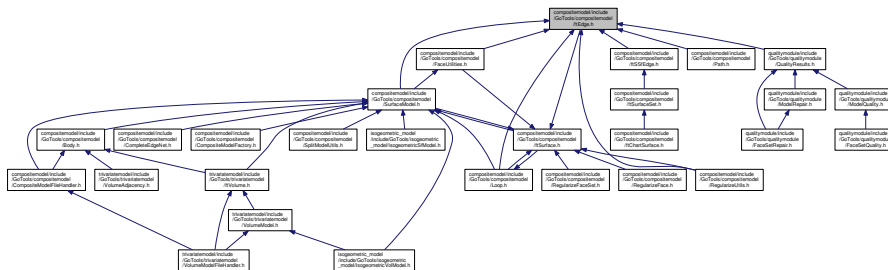
- enum `Go::ftCurveType` {  
`Go::CURVE_NOTYPE = -1`, `Go::CURVE_INTERSECTION`, `Go::CURVE_EDGE`, `Go::CURVE_KINK`,  
`Go::CURVE_CORNER`, `Go::CURVE_GAP`, `Go::CURVE_SINGULAR` }

## 30.18 compositemodel/include/GoTools/compositemodel/ftEdge.h File Reference

```
#include "GoTools/compositemodel/ftEdgeBase.h"
#include "GoTools/geometry/ParamCurve.h"
#include "GoTools/compositemodel/ftSurface.h"
#include "GoTools/compositemodel/ftFaceBase.h"
Include dependency graph for ftEdge.h:
```



This graph shows which files directly or indirectly include this file:



## Classes

- class [Go::ftEdge](#)

The *ftEdge* is a half-edge implementation of a topological data structure.

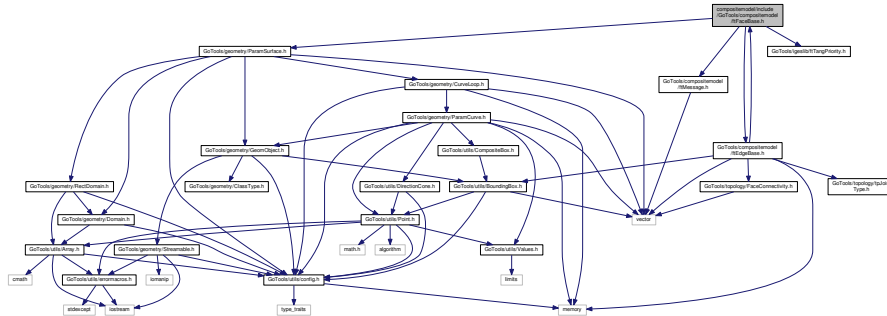
## Namespaces

- [Go](#)

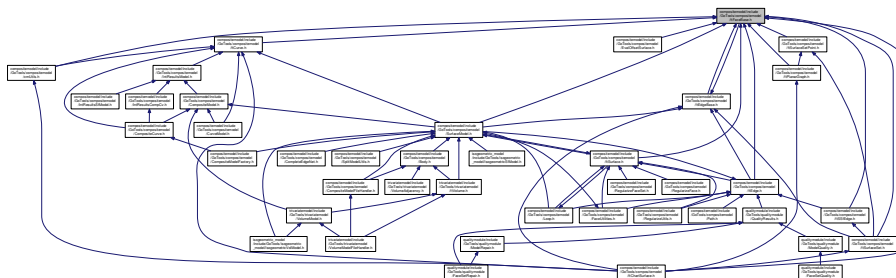


## 30.20 compositemodel/include/GoTools/compositemodel/ftFaceBase.h File Reference

```
#include "GoTools/compositemodel/ftEdgeBase.h"
#include "GoTools/geometry/ParamSurface.h"
#include "GoTools/compositemodel/ftMessage.h"
#include "GoTools/igeslib/ftTangPriority.h"
Include dependency graph for ftFaceBase.h:
```



This graph shows which files directly or indirectly include this file:



### Classes

- class [Go::ftFaceBase](#)

### Namespaces

- [Go](#)

## 30.21 compositemodel/include/GoTools/compositemodel/ftLine.h File Reference

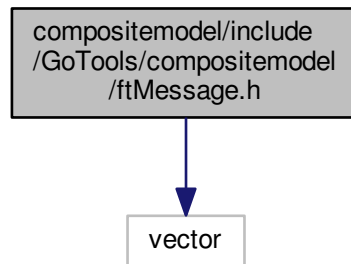
```
#include "GoTools/utils/Point.h"
#include "GoTools/compositemodel/ftPlane.h"
```



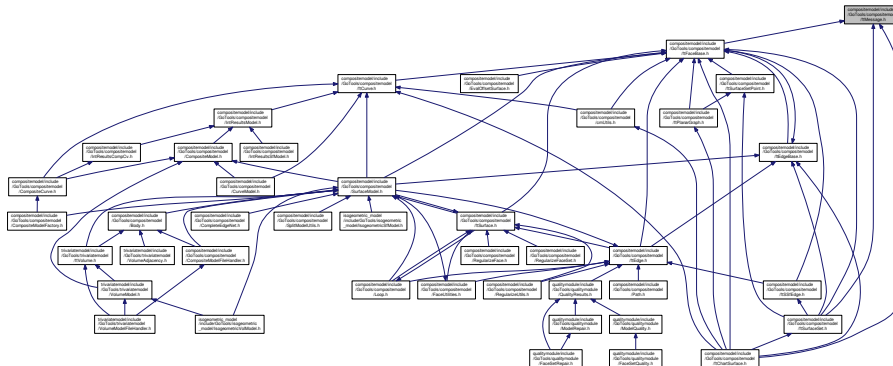
## 30.22 compositemodel/include/GoTools/compositemodel/ftMessage.h File Reference

```
#include <vector>
```

Include dependency graph for ftMessage.h:



This graph shows which files directly or indirectly include this file:



### Classes

- class [Go::ftMessage](#)

### Namespaces

- [Go](#)

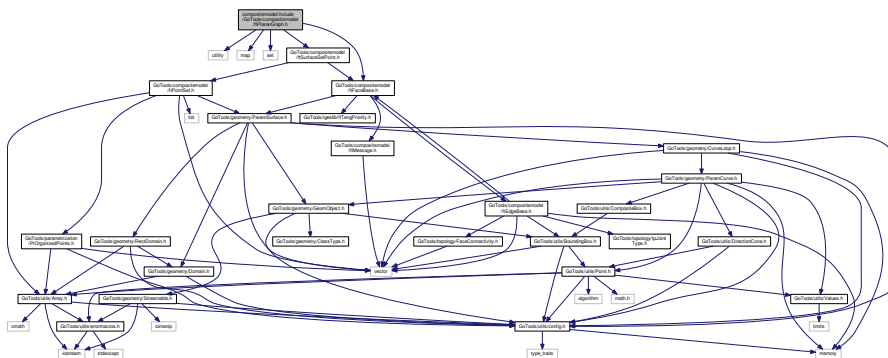
## Enumerations

- enum Go::ftmessages {
  - Go::FT\_NOT\_IMPLEMENTED = -99, Go::FT\_NO\_DATA, Go::FT\_BAD\_INPUT\_FILE, Go::FT\_ERROR\_IN\_READ\_IGES,
  - Go::FT\_DISCONTINUITY, Go::FT\_UNEXPECTED\_DATA\_TYPE, Go::FT\_NON\_4\_SIDED\_SURF, Go::FT\_WRONG\_NO\_OF\_BOUNDARIES,
  - Go::FT\_INSUFFICIENT\_MESH\_CURVES, Go::FT\_ERROR\_IN\_TOPOLOGY\_ANALYSIS, Go::FT\_TOPOLOGY\_PROBLEM, Go::FT\_ERROR\_IN\_SURFACE\_CREATION,
  - Go::FT\_ERROR\_IN\_PARAMETERIZE, Go::FT\_ERROR\_IN\_SURFACE\_TRIMMING, Go::FT\_NOT\_SUPPORTED, Go::FT\_NOT\_SPLINE\_SURF,
  - Go::FT\_ERROR\_IN\_SURFACE\_GRIDDING, Go::FT\_NEIGHBOUR\_GRID\_SIZE\_DIFFER, Go::FT\_OK = 0,
  - Go::FT\_APPROX\_ERROR\_TOO\_LARGE,
  - Go::FT\_COULD\_NOT\_REFINE\_SURFACE, Go::FT\_FEATURE\_ERROR\_TOO\_LARGE, Go::FT\_DEGENERATE\_PATCH\_CREATED, Go::FT\_SURFACE\_ALREADY\_CREATED,
  - Go::FT\_DEGENERATE\_SURFACE, Go::FT\_UNEXPECTED\_INPUT\_OBJECT\_IGNORED, Go::FT\_FACES\_OVERLAP, Go::FT\_COULD\_NOT\_SMOOTH\_SURFACE,
  - Go::FT\_COULD\_NOT\_SMOOTH\_SURFACESET, Go::FT\_NO\_SMOOTHING, Go::FT\_FAILED\_CREATING\_GRAPH, Go::FT\_GRID\_NOT\_CREATED,
  - Go::FT\_SURFACE\_NOT\_MODIFIED }

## 30.23 compositemodel/include/GoTools/compositemodel/ftPlanarGraph.h File Reference

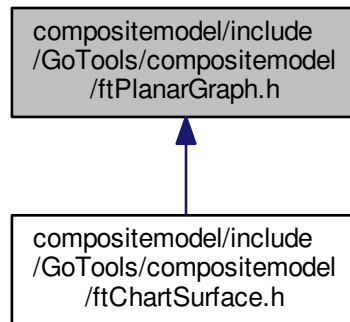
```
#include <utility>
#include <map>
#include <set>
#include "GoTools/compositemodel/ftFaceBase.h"
#include "GoTools/compositemodel/ftSurfaceSetPoint.h"
```

Include dependency graph for ftPlanarGraph.h:





This graph shows which files directly or indirectly include this file:



## Classes

- class [Go::ftGraphEdge](#)
- class [Go::ftSearchNode](#)
- class [Go::ftPlanarGraph](#)

## Namespaces

- [Go](#)

## Typedefs

- typedef `std::vector< ftSearchNode >::iterator` [Go::GraphIter](#)

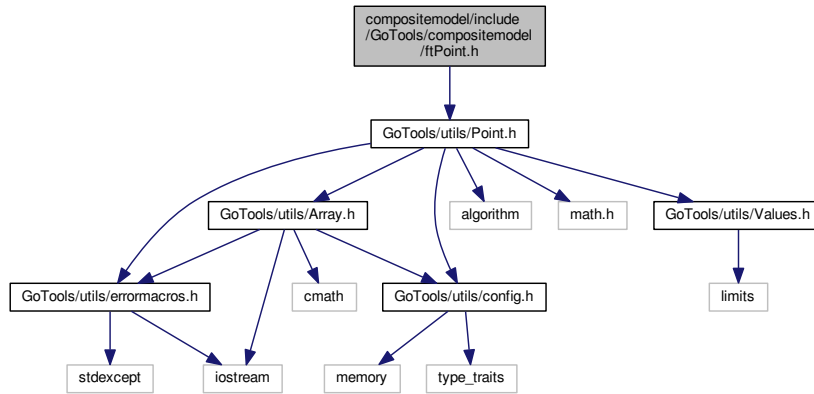
## Functions

- [double Go::areaTriangle](#) (`const Vector2D &corner1`, `const Vector2D &corner2`, `const Vector2D &corner3`)

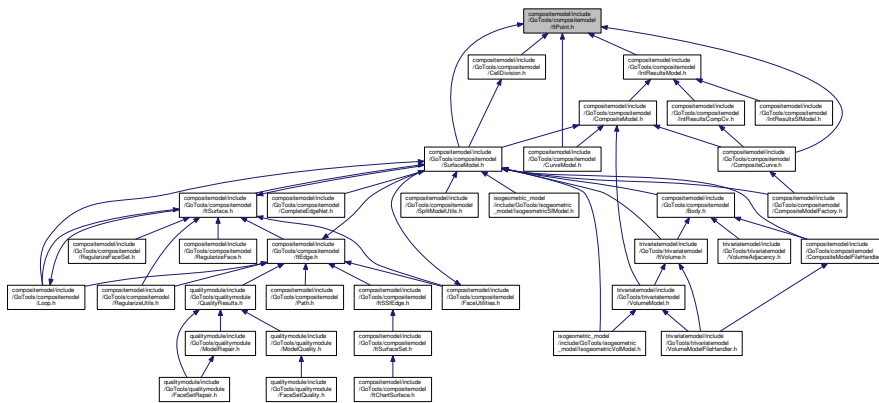


## 30.25 compositemodel/include/GoTools/compositemodel/ftPoint.h File Reference

```
#include "GoTools/Utils/Point.h"
Include dependency graph for ftPoint.h:
```



This graph shows which files directly or indirectly include this file:



### Classes

- class [Go::ftPoint](#)

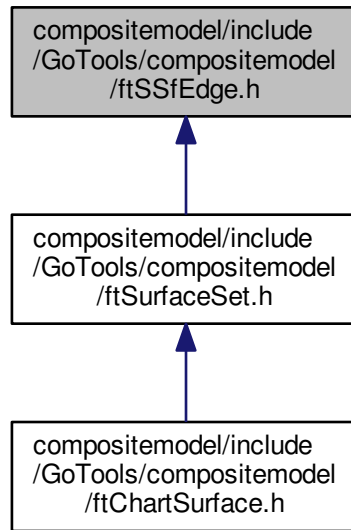
### Namespaces

- [Go](#)





This graph shows which files directly or indirectly include this file:



### Classes

- class [Go::ftSSfEdge](#)

### Namespaces

- [Go](#)

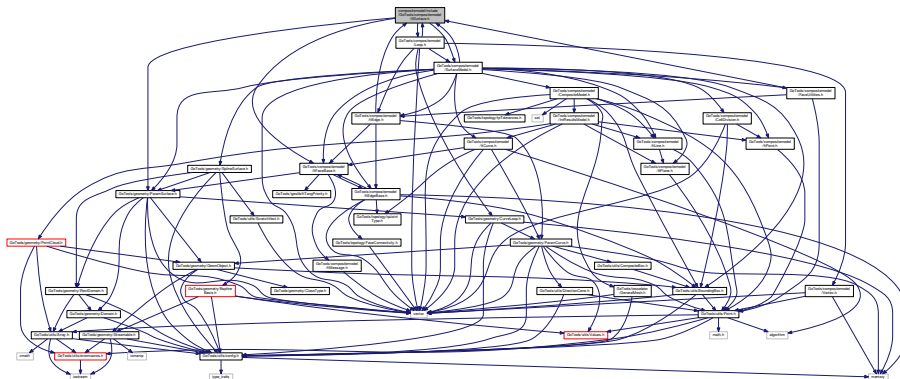
## 30.29 compositemodel/include/GoTools/compositemodel/ftSurface.h File Reference

```

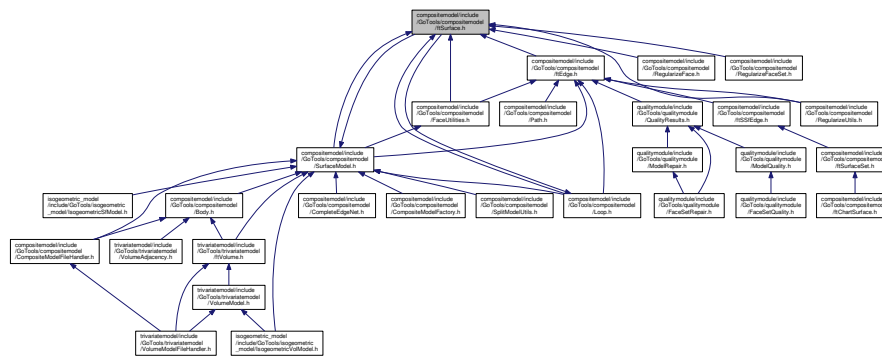
#include "GoTools/compositemodel/ftFaceBase.h"
#include "GoTools/geometry/ParamSurface.h"
#include "GoTools/compositemodel/Loop.h"
#include "GoTools/compositemodel/SurfaceModel.h"

```

Include dependency graph for ftSurface.h:



This graph shows which files directly or indirectly include this file:



## Classes

- struct [Go::AdjacencyInfo](#)  
*Struct to store information about adjacency relations between two faces.*
- class [Go::ftSurface](#)

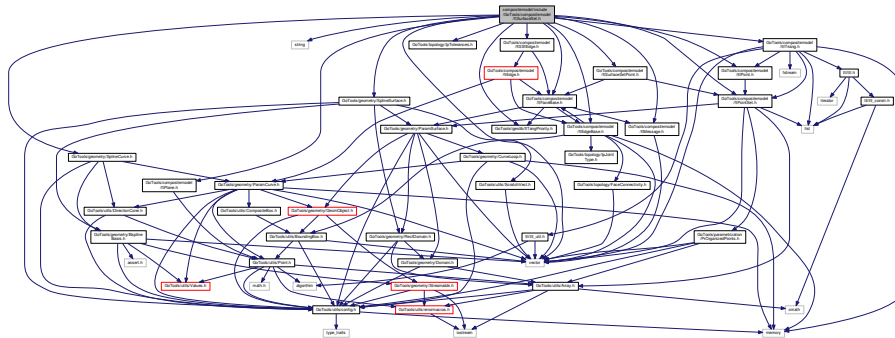
## Namespaces

- [Go](#)

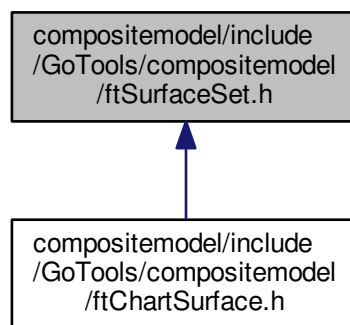
## 30.30 compositemodel/include/GoTools/compositemodel/ftSurfaceSet.h File Reference

```
#include <vector>
#include <string>
#include "GoTools/compositemodel/ftMessage.h"
#include "GoTools/igeslib/ftTangPriority.h"
#include "GoTools/compositemodel/ftPointSet.h"
#include "GoTools/topology/tpTolerances.h"
#include "GoTools/compositemodel/ftFaceBase.h"
#include "GoTools/compositemodel/ftEdgeBase.h"
#include "GoTools/geometry/SplineSurface.h"
#include "GoTools/geometry/SplineCurve.h"
#include "GoTools/compositemodel/ttlPoint.h"
#include "GoTools/compositemodel/ttlTriang.h"
#include "GoTools/compositemodel/ftSurfaceSetPoint.h"
#include "GoTools/compositemodel/ftPlane.h"
#include "GoTools/compositemodel/ftSSfEdge.h"
```

Include dependency graph for `ftSurfaceSet.h`:



This graph shows which files directly or indirectly include this file:



## Classes

- class [Go::ftSurfaceSet](#)

## Namespaces

- [Go](#)

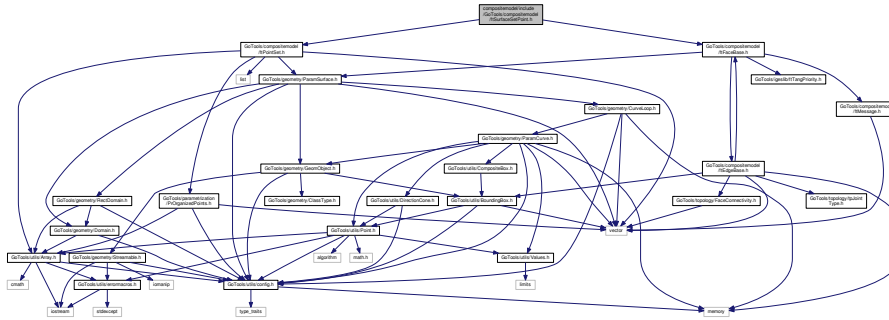
## Typedefs

- typedef `std::pair< std::pair< double, double >, std::pair< ftFaceBase *, int > >` [Go::BoundaryPiece](#)

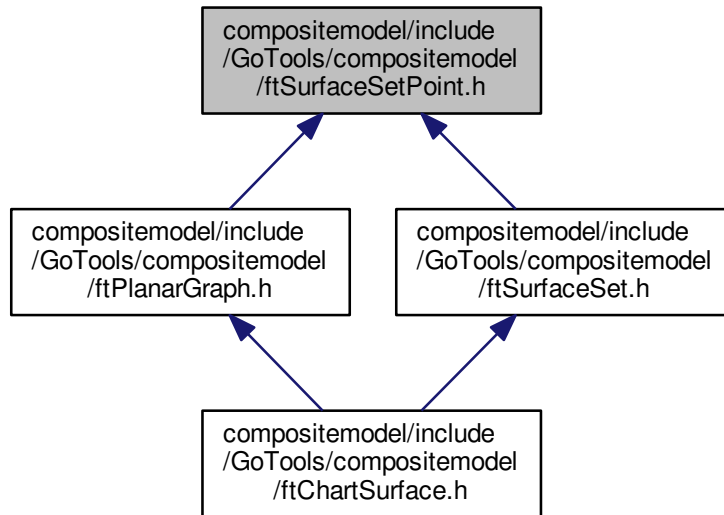


### 30.31 compositemodel/include/GoTools/compositemodel/ftSurfaceSetPoint.h File Reference

```
#include "GoTools/compositemodel/ftPointSet.h"
#include "GoTools/compositemodel/ftFaceBase.h"
Include dependency graph for ftSurfaceSetPoint.h:
```



This graph shows which files directly or indirectly include this file:



#### Classes

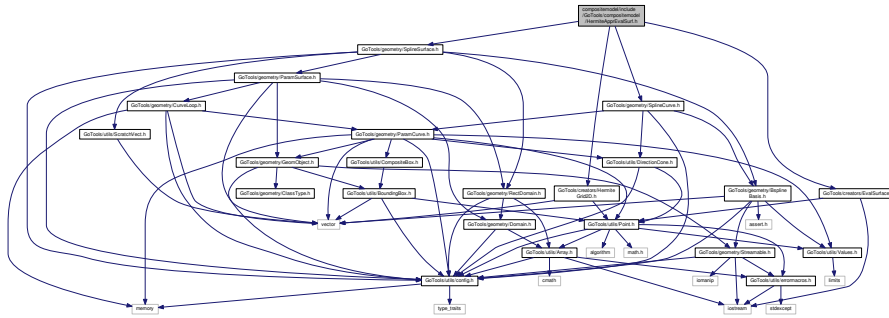
- class [Go::ftSurfaceSetPoint](#)

#### Namespaces

- [Go](#)

### 30.32 compositemodel/include/GoTools/compositemodel/HermiteApprEvalSurf.h File Reference

```
#include "GoTools/creators/EvalSurface.h"
#include "GoTools/creators/HermiteGrid2D.h"
#include "GoTools/geometry/SplineSurface.h"
#include "GoTools/geometry/SplineCurve.h"
Include dependency graph for HermiteApprEvalSurf.h:
```



#### Classes

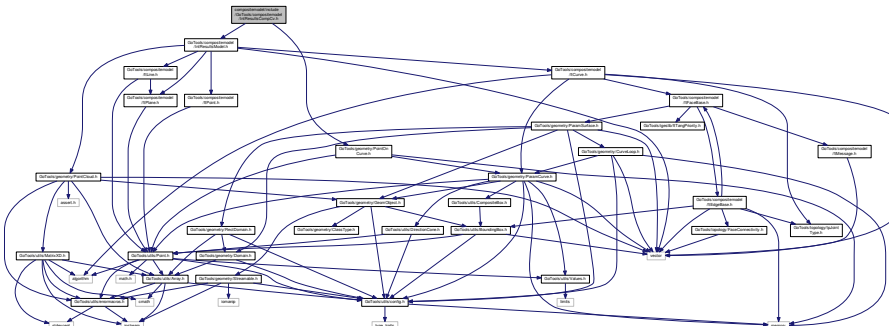
- class [Go::HermiteApprEvalSurf](#)

#### Namespaces

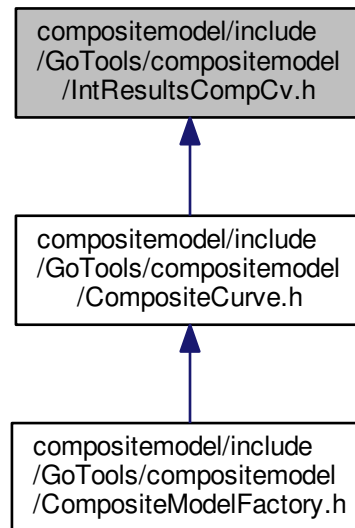
- [Go](#)

### 30.33 compositemodel/include/GoTools/compositemodel/IntResultsCompCv.h File Reference

```
#include "GoTools/compositemodel/IntResultsModel.h"
#include "GoTools/geometry/PointOnCurve.h"
Include dependency graph for IntResultsCompCv.h:
```



This graph shows which files directly or indirectly include this file:



## Classes

- class [Go::IntResultsCompCv](#)

## Namespaces

- [Go](#)

## 30.34 compositemodel/include/GoTools/compositemodel/IntResultsModel.h File Reference

```
#include "GoTools/compositemodel/ftPoint.h"
#include "GoTools/compositemodel/ftCurve.h"
#include "GoTools/compositemodel/ftPlane.h"
#include "GoTools/compositemodel/ftLine.h"
#include "GoTools/geometry/PointCloud.h"
#include <vector>
```

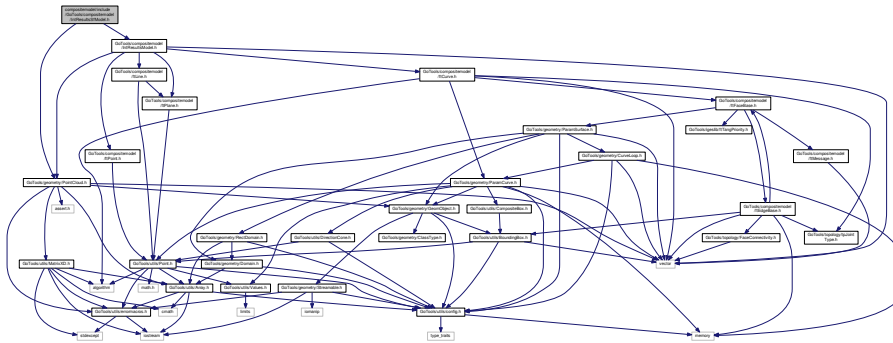


## 30.35 compositemodel/include/GoTools/compositemodel/IntResultsSfModel.h File Reference

```
#include "GoTools/compositemodel/IntResultsModel.h"
```

```
#include "GoTools/geometry/PointCloud.h"
```

Include dependency graph for IntResultsSfModel.h:



### Classes

- class [Go::IntResultsSfModel](#)

### Namespaces

- [Go](#)

## 30.36 compositemodel/include/GoTools/compositemodel/Loop.h File Reference

```
#include "GoTools/geometry/CurveLoop.h"
```

```
#include "GoTools/compositemodel/Vertex.h"
```

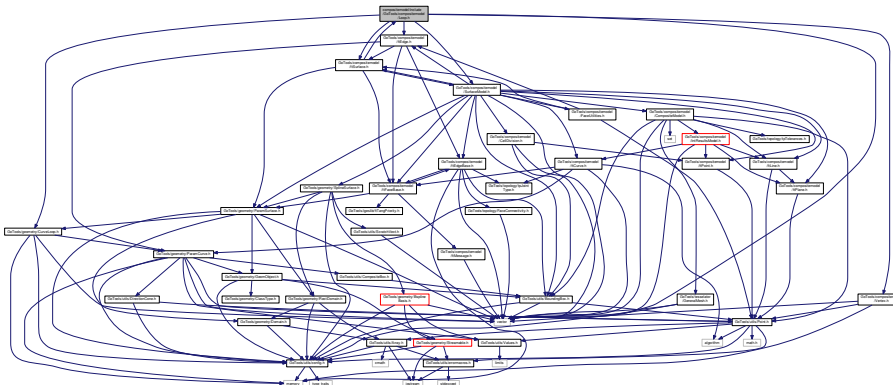
```
#include "GoTools/compositemodel/ftEdge.h"
```

```
#include "GoTools/compositemodel/ftSurface.h"
```

```
#include "GoTools/compositemodel/SurfaceModel.h"
```

```
#include <vector>
```

Include dependency graph for Loop.h:





## Namespaces

- [Go](#)
- [Go::OffsetSurfaceUtils](#)

## Enumerations

- enum [OffsetSurfaceStatus](#) {  
 OFFSET\_OK = 0, OFFSET\_FAILED = 1, SELF\_INTERSECTING\_INTERIOR = 2, SELF\_INTERSECTING\_BOUNDARY = 3,  
 TOLERANCE\_ERROR = 4, NOT\_FOUR\_CORNERS = 5, NON\_ISO\_KINK\_CURVE = 6 }

## Functions

- [OffsetSurfaceStatus](#) [Go::OffsetSurfaceUtils::offsetSurfaceSet](#) (const std::vector< shared\_ptr< ParamSurface > > &param\_sfs, double offset\_dist, double epsgeo, shared\_ptr< SplineSurface > &offset\_sf)

### 30.37.1 Enumeration Type Documentation

#### 30.37.1.1 enum OffsetSurfaceStatus

##### Enumerator

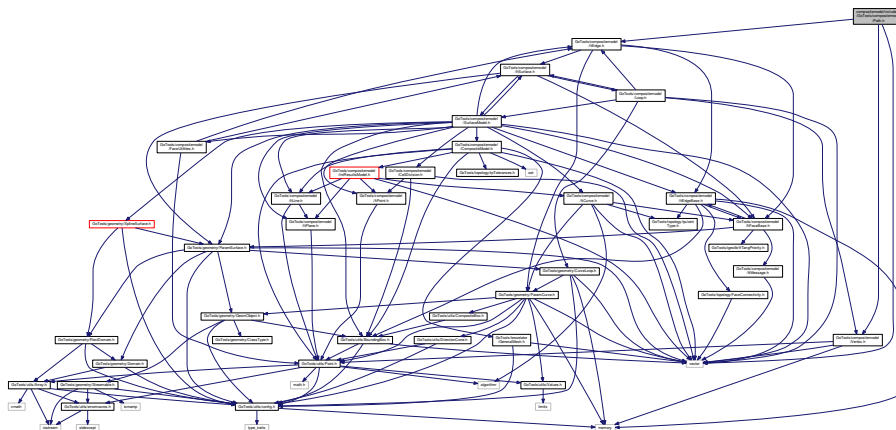
**OFFSET\_OK**  
**OFFSET\_FAILED**  
**SELF\_INTERSECTING\_INTERIOR**  
**SELF\_INTERSECTING\_BOUNDARY**  
**TOLERANCE\_ERROR**  
**NOT\_FOUR\_CORNERS**  
**NON\_ISO\_KINK\_CURVE**

Definition at line 47 of file OffsetSurfaceUtils.h.

## 30.38 compositemodel/include/GoTools/compositemodel/Path.h File Reference

```
#include "GoTools/compositemodel/Vertex.h"
#include "GoTools/compositemodel/ftEdge.h"
#include <vector>
```

Include dependency graph for Path.h:



## Namespaces

- [Go](#)
- [Go::Path](#)

*Functions related to sequence of edges.*

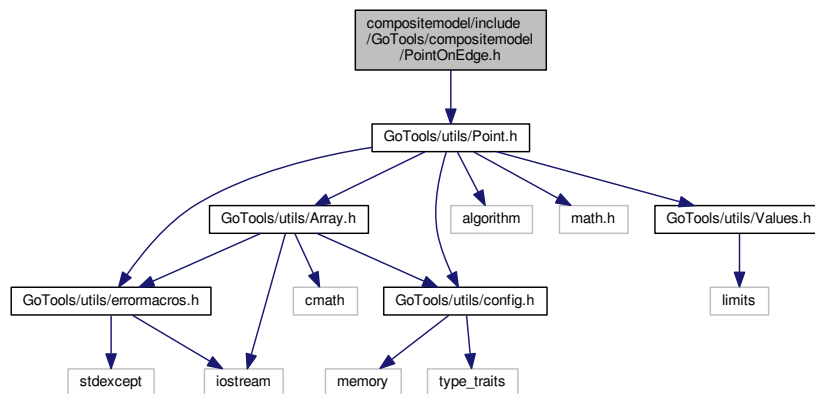
## Functions

- [bool Go::Path::estimateHoleInfo](#) (`const std::vector< ftEdge * > &edges`, `Point &centre`, `Point &axis`, `double &radius`, `double &angle`)  
*sequence*
- `void` [Go::Path::classifyCorners](#) (`const std::vector< ftEdge * > &edges`, `double tol`, `std::vector< shared_ptr< Vertex > > &corner`, `std::vector< shared_ptr< Vertex > > &non_corner`)
- `std::vector< ftEdge * >` [Go::Path::identifyLoop](#) (`std::vector< ftEdge * > edges`, `shared_ptr< Vertex > vx`)
- `void` [Go::Path::closestPoint](#) (`std::vector< ftEdge * > edges`, `const Point &pt`, `int &clo_ind`, `double &clo_par`, `Point &clo_pt`, `double &clo_dist`)
- `void` [Go::Path::getEdgeCurves](#) (`std::vector< ftEdge * > &loop`, `std::vector< shared_ptr< ParamCurve > > &space_cvs`, `std::vector< Point > &joint_points`, `double eps`, `double tol`, `bool corner_in_Tjoint=true`)
- `std::vector< ftEdge * >` [Go::Path::edgeChain](#) (`ftEdge *edg`, `double angtol`, `shared_ptr< Vertex > &v1`, `shared_ptr< Vertex > &v2`)

## 30.39 compositemodel/include/GoTools/compositemodel/PointOnEdge.h File Reference

```
#include "GoTools/utils/Point.h"
```

Include dependency graph for PointOnEdge.h:



## Classes

- class [Go::PointOnEdge](#)

## Namespaces

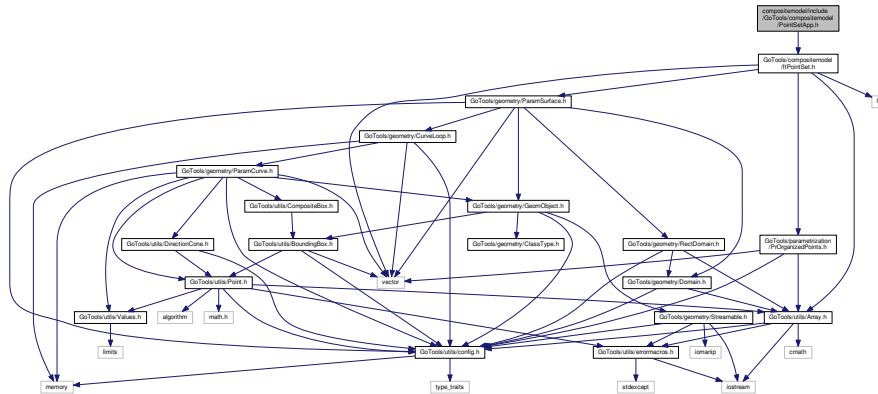
- [Go](#)



## 30.40 compositemodel/include/GoTools/compositemodel/PointSetApp.h File Reference

```
#include "GoTools/compositemodel/ftPointSet.h"
```

Include dependency graph for PointSetApp.h:



### Namespaces

- [Go](#)
- [Go::PointSetApp](#)
  - *Point set and triangulation applications.*

### Functions

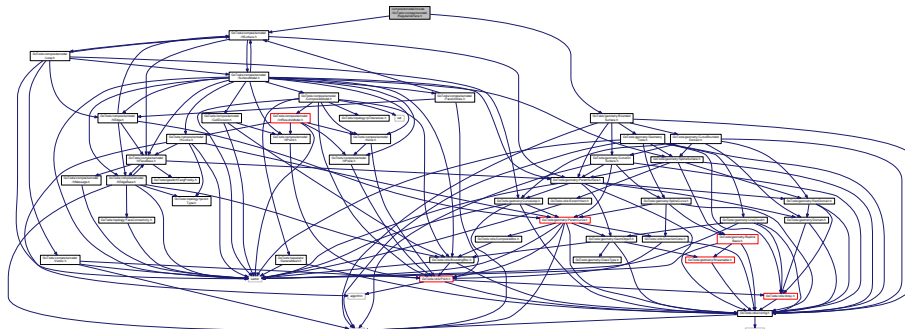
- `void Go::PointSetApp::parameterizeTriang (const double *xyz_points, int nmbp, const int *triangles, int nmbt, std::vector< double > &uv_pars)`
  - *Parameterize triangulated point set.*
- `bool Go::PointSetApp::recognizeBoundary (shared_ptr< ftPointSet > &triang, std::vector< int > &corner_ix)`
  - *Service functionality for parameterizeTriang.*
- `bool Go::PointSetApp::recognizeCornerNodes (const shared_ptr< ftPointSet > &triang, const std::vector< int > &bd_nodes, std::vector< int > &corner_ix)`

## 30.41 compositemodel/include/GoTools/compositemodel/RegularizeFace.h File Reference

```
#include "GoTools/compositemodel/ftSurface.h"
```

```
#include "GoTools/geometry/BoundedSurface.h"
```

Include dependency graph for RegularizeFace.h:



## Classes

- class [Go::RegularizeFace](#)

*Split one face into a number of 4-sided domains without inner trimming. This class is intended for use in block structuring. One face with possible inner and outer trimming is split according to certain rules to result in a face set with 4 sided faces although faces with less than 4 sides can occur. A side is defined as a piece of the face boundary between two corners or between vertices where there are more than one adjacent face. The trimmed surfaces being output from this class can later be approximated by spline surfaces. The splitting procedure is recursive.*

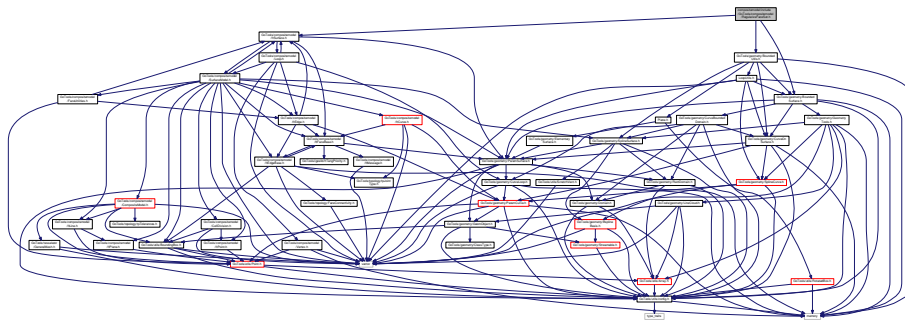
## Namespaces

- [Go](#)

### 30.42 compositemodel/include/GoTools/compositemodel/RegularizeFaceSet.h File Reference

```
#include "GoTools/compositemodel/ftSurface.h"
#include "GoTools/geometry/BoundedSurface.h"
#include "GoTools/geometry/BoundedUtils.h"
```

Include dependency graph for RegularizeFaceSet.h:



## Classes

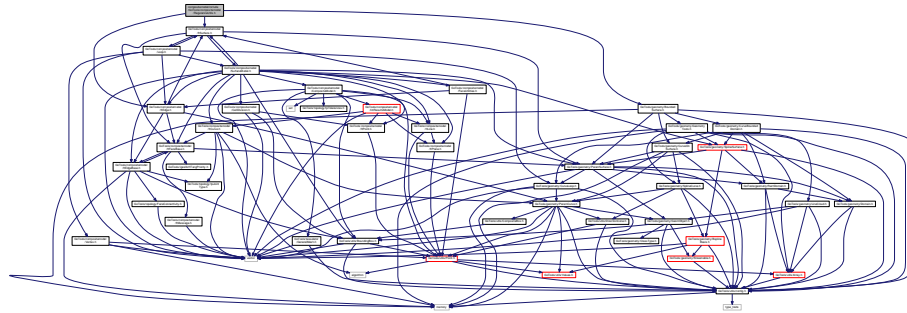
- class [Go::RegularizeFaceSet](#)

## Namespaces

- [Go](#)

## 30.43 compositemodel/include/GoTools/compositemodel/RegularizeUtils.h File Reference

```
#include "GoTools/compositemodel/ftSurface.h"
#include "GoTools/compositemodel/ftEdge.h"
#include "GoTools/geometry/BoundedSurface.h"
Include dependency graph for RegularizeUtils.h:
```



### Namespaces

- [Go](#)
- [Go::RegularizeUtils](#)

Utility functionality for [RegularizeFace](#) and [RegularizeFaceSet](#).

### Functions

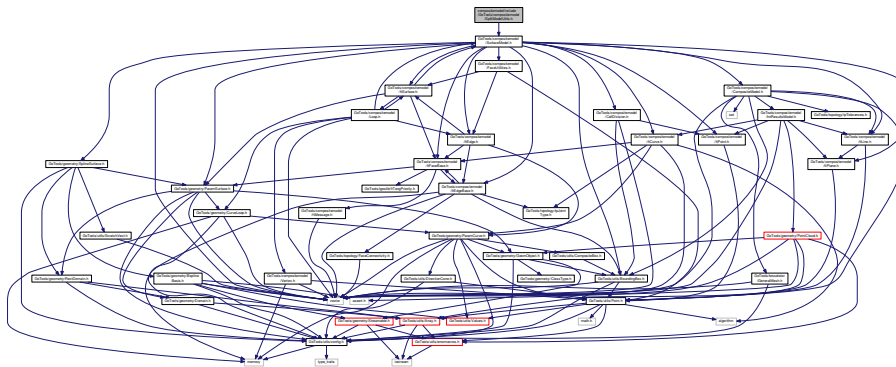
- `std::vector< shared_ptr< ftSurface > >` [Go::RegularizeUtils::divideVertex](#) (`shared_ptr< ftSurface > face`, `shared_ptr< Vertex > vx`, `std::vector< shared_ptr< Vertex > > &cand_vx`, `ftEdge *cand_edge`, `std::vector< shared_ptr< Vertex > > &prio_vx`, `double epsge`, `double tol2`, `double angtol`, `double bend`, `std::vector< shared_ptr< Vertex > > &non_corner`, `const Point &centre`, `const Point &axis`, `bool strong=false`)
- `std::vector< shared_ptr< CurveOnSurface > >` [Go::RegularizeUtils::findVertexSplit](#) (`shared_ptr< ftSurface > face`, `shared_ptr< Vertex > vx`, `std::vector< shared_ptr< Vertex > > &cand_vx`, `ftEdge *cand_edge`, `std::vector< shared_ptr< Vertex > > &prio_vx`, `double epsge`, `double tol2`, `double angtol`, `double bend`, `std::vector< shared_ptr< Vertex > > &non_corner`, `const Point &centre`, `const Point &axis`, `shared_ptr< BoundedSurface > &bd_sf`, `bool strong=false`)
- `std::vector< shared_ptr< ftSurface > >` [Go::RegularizeUtils::createFaces](#) (`std::vector< shared_ptr< BoundedSurface > > &sub_sfs`, `shared_ptr< ftSurface > face`, `double epsge`, `double tol2`, `double angtol`, `std::vector< shared_ptr< Vertex > > non_corner`)
- `void` [Go::RegularizeUtils::getDivisionPlane](#) (`shared_ptr< ftSurface > face`, `shared_ptr< Vertex > vx`, `double epsge`, `Point &pnt`, `Point &normal`)
- `void` [Go::RegularizeUtils::getClosestBoundaryPar](#) (`shared_ptr< ftSurface > face`, `shared_ptr< Vertex > vx`, `std::vector< shared_ptr< ParamCurve > > &vx_cvs`, `const Point &pnt`, `double epsge`, `int &close_idx`, `double &close_dist`, `Point &close_par`, `int loop_idx=-1`)
- `bool` [Go::RegularizeUtils::cornerInShortestPath](#) (`shared_ptr< Vertex > vx1`, `shared_ptr< Vertex > vx2`, `shared_ptr< ftSurface > face`, `double angtol`)
- `bool` [Go::RegularizeUtils::getPath](#) (`ftEdge *edg`, `shared_ptr< Vertex > vx`, `shared_ptr< Vertex > last`, `shared_ptr< ftSurface > face`, `std::vector< ftEdge * > &path`)
- `int` [Go::RegularizeUtils::noExtension](#) (`shared_ptr< Vertex > vx`, `ftSurface *face`, `shared_ptr< Vertex > &vx2`, `std::pair< Point, Point > &co_par1`, `std::pair< Point, Point > &co_par2`, `int &dir1`, `int &dir2`, `double &val1`, `double &val2`, `double angtol`, `bool check_constant_curve`)

- `bool Go::RegularizeUtils::mergeSituationContinuation` (ftSurface \*init\_face, shared\_ptr< Vertex > vx, ftEdge \*edge, double angtol)
- `double Go::RegularizeUtils::getMaxParFrac` (shared\_ptr< ftSurface > face)
- `int Go::RegularizeUtils::selectCandVx` (shared\_ptr< ftSurface > face, shared\_ptr< Vertex > vx, const Point &in\_vec, vector< shared\_ptr< Vertex > > cand\_vx, RectDomain &dom, double epsge, double angtol, const Point &centre, const Point &normal, std::vector< shared\_ptr< ParamCurve > > &vx\_cvs, double close\_dist, const Point &close\_pt, double &cyl\_rad, bool strong=false)
- `void Go::RegularizeUtils::adjustTrimSeg` (std::vector< shared\_ptr< CurveOnSurface > > &trim\_segments, Point \*parval1, Point \*parval2, shared\_ptr< ftSurface > face, shared\_ptr< BoundedSurface > &bd\_sf, std::vector< shared\_ptr< Vertex > > &non\_corner, double tol, double epsge)
- `void Go::RegularizeUtils::checkTrimSeg` (std::vector< shared\_ptr< CurveOnSurface > > &trim\_segments, std::vector< shared\_ptr< Vertex > > &next\_vxs, const Point &vx\_point, const Point &other\_pt, double epsge)
- `void Go::RegularizeUtils::checkTrimSeg2` (std::vector< shared\_ptr< CurveOnSurface > > &trim\_segments, const Point &vx\_par1, const Point &vx\_par2, double epsge)
- `void Go::RegularizeUtils::checkTrimSeg3` (std::vector< shared\_ptr< CurveOnSurface > > &trim\_segments, const Point &vx\_par1, const Point &vx\_par2, double epsge)
- `void Go::RegularizeUtils::checkTrimConfig` (shared\_ptr< ftSurface > face, std::vector< shared\_ptr< CurveOnSurface > > &trim\_segments, shared\_ptr< Vertex > vx, std::vector< shared\_ptr< Vertex > > &corners, double epsge)
- `ftEdge * Go::RegularizeUtils::getOppositeBoundaryPar` (shared\_ptr< ftSurface > face, shared\_ptr< Vertex > vx, std::vector< shared\_ptr< Vertex > > &corners, double epsge, Point &point, double &par, double &dist)
- `Point Go::RegularizeUtils::getInVec` (shared\_ptr< Vertex > vx, shared\_ptr< ftSurface > face)
- `shared_ptr< ParamCurve > Go::RegularizeUtils::checkStrightParCv` (shared\_ptr< ftSurface > face, const Point &pos1, const Point &pos2, double epsge)
- `shared_ptr< ParamCurve > Go::RegularizeUtils::checkStrightParCv` (shared\_ptr< ftSurface > face, shared\_ptr< Vertex > vx1, shared\_ptr< Vertex > vx2, double epsge)
- `shared_ptr< ParamCurve > Go::RegularizeUtils::checkStrightParCv` (shared\_ptr< ftSurface > face, shared\_ptr< Vertex > vx1, const Point &mid, double epsge)
- `bool Go::RegularizeUtils::checkPath` (shared\_ptr< Vertex > vx1, shared\_ptr< Vertex > vx2, shared\_ptr< Vertex > vx, shared\_ptr< ftSurface > face, double angtol)
- `bool Go::RegularizeUtils::checkRegularity` (std::vector< shared\_ptr< Vertex > > &cand\_vx, shared\_ptr< ftSurface > face, bool checkConvex=true)
- `std::vector< shared_ptr< Vertex > > Go::RegularizeUtils::endVxInChain` (shared\_ptr< ftSurface > face, ftSurface \*face1, ftSurface \*face2, shared\_ptr< Vertex > vx, shared\_ptr< Vertex > prev, shared\_ptr< Vertex > vx0, std::vector< shared\_ptr< Vertex > > &met\_already)
- `int Go::RegularizeUtils::traverseUntilTJoint` (std::vector< ftSurface \* > vx\_faces, shared\_ptr< Vertex > vx, shared\_ptr< Vertex > &vx2, std::vector< ftSurface \* > &vx\_faces2)
- `void Go::RegularizeUtils::angleInEndpoints` (shared\_ptr< CurveOnSurface > seg, shared\_ptr< Vertex > vx1, shared\_ptr< Vertex > vx2, shared\_ptr< ftSurface > face, double &min\_ang1, double &min\_ang2)
- `void Go::RegularizeUtils::getSourceCvs` (std::vector< shared\_ptr< ftEdge > > &all\_edg, std::vector< shared\_ptr< ParamCurve > > &all\_cvs)

### 30.44 compositemodel/include/GoTools/compositemodel/SplitModelUtils.h File Reference

```
#include "GoTools/compositemodel/SurfaceModel.h"
```

Include dependency graph for SplitModelUtils.h:



## Namespaces

- [Go](#)
- [Go::SplitModelUtils](#)

*Utility functionality for splitting of surface models.*

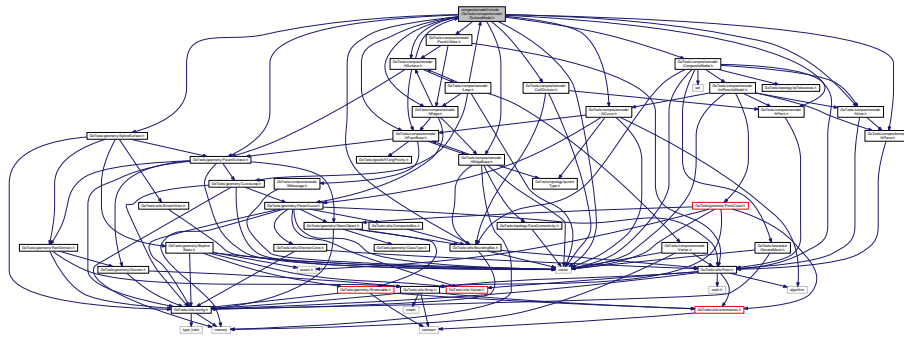
## Functions

- void [Go::SplitModelUtils::splitInFreeCorners](#) (shared\_ptr< SurfaceModel > sfmodel, const Point &pnt, const Point &axis)
- void [Go::SplitModelUtils::splitInNonCorners](#) (shared\_ptr< SurfaceModel > sfmodel, const Point &pnt, const Point &axis)
- void [Go::SplitModelUtils::splitInOuterVertices](#) (shared\_ptr< SurfaceModel > sfmodel, shared\_ptr< ftSurface > face, const Point &pnt, const Point &axis)

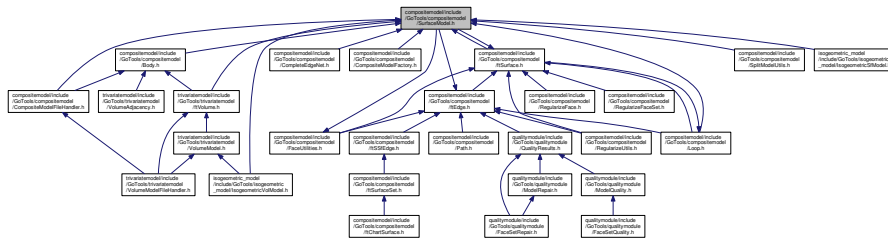
## 30.45 compositemodel/include/GoTools/compositemodel/SurfaceModel.h File Reference

```
#include "GoTools/compositemodel/CompositeModel.h"
#include "GoTools/compositemodel/ftSurface.h"
#include "GoTools/compositemodel/ftFaceBase.h"
#include "GoTools/compositemodel/CellDivision.h"
#include "GoTools/compositemodel/ftCurve.h"
#include "GoTools/compositemodel/ftPoint.h"
#include "GoTools/utils/BoundingBox.h"
#include "GoTools/geometry/ParamSurface.h"
#include "GoTools/compositemodel/ftEdgeBase.h"
#include "GoTools/compositemodel/ftEdge.h"
#include "GoTools/geometry/SplineSurface.h"
#include "GoTools/compositemodel/ftPlane.h"
#include "GoTools/compositemodel/ftLine.h"
#include "GoTools/compositemodel/FaceUtilities.h"
#include <vector>
```

Include dependency graph for SurfaceModel.h:



This graph shows which files directly or indirectly include this file:



Classes

- class [Go::SurfaceModel](#)

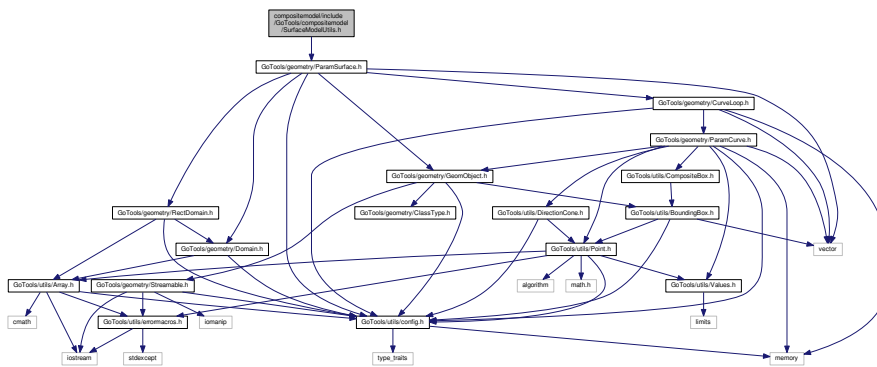
Namespaces

- [Go](#)

30.46 compositemodel/include/GoTools/compositemodel/SurfaceModelUtils.h File Reference

```
#include "GoTools/geometry/ParamSurface.h"
```

Include dependency graph for SurfaceModelUtils.h:



## Namespaces

- [Go](#)
- [Go::SurfaceModelUtils](#)

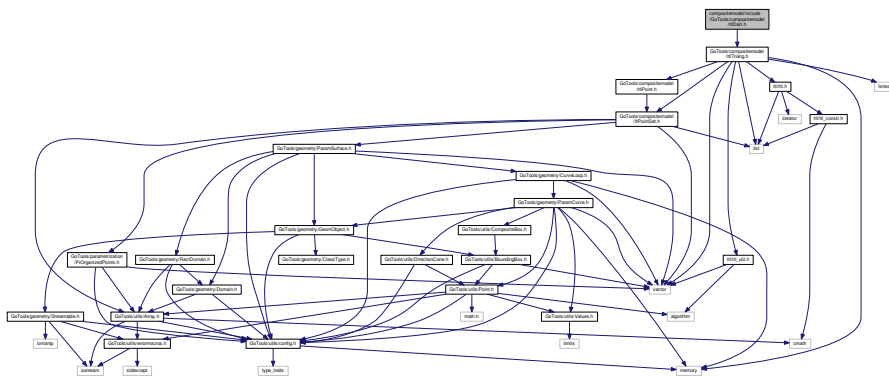
*Utility functionality for splitting of surface models.*

## Functions

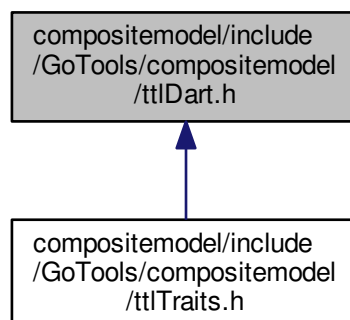
- `std::vector< shared_ptr< ParamSurface > >` [Go::SurfaceModelUtils::checkClosedFaces](#) (`shared_ptr< ParamSurface >` `surface`, `double tol`)

## 30.47 compositemodel/include/GoTools/compositemodel/ttIDart.h File Reference

```
#include "GoTools/compositemodel/ttITriang.h"
Include dependency graph for ttIDart.h:
```



This graph shows which files directly or indirectly include this file:



Classes

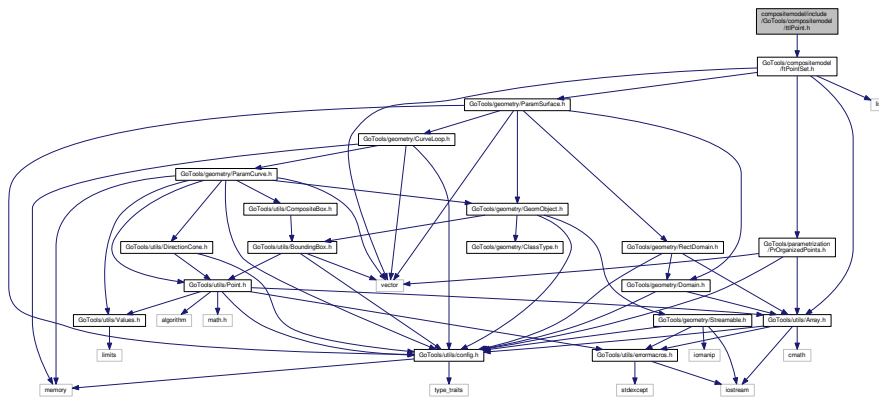
- class [hetriang::Dart](#)  
*Dart* class for the half-edge data structure.

Namespaces

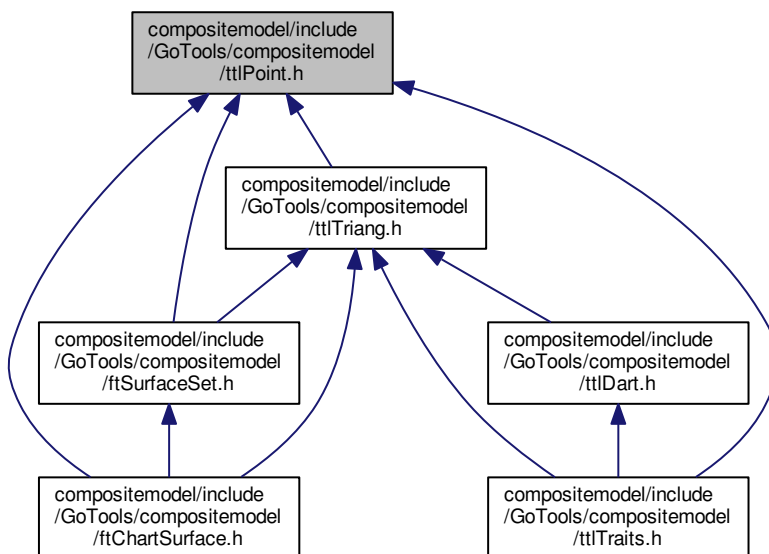
- [hetriang](#)

30.48 `compositemodel/include/GoTools/compositemodel/ttlPoint.h` File Reference

```
#include "GoTools/compositemodel/ftPointSet.h"
Include dependency graph for ttlPoint.h:
```



This graph shows which files directly or indirectly include this file:





## Classes

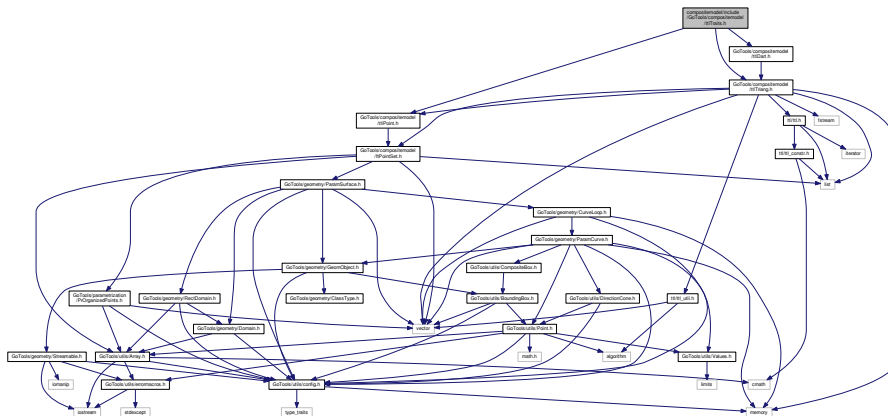
- class [Go::ttlPoint](#)

## Namespaces

- [Go](#)

## 30.49 compositemodel/include/GoTools/compositemodel/ttlTraits.h File Reference

```
#include "GoTools/compositemodel/ttlTriang.h"
#include "GoTools/compositemodel/ttlDart.h"
#include "GoTools/compositemodel/ttlPoint.h"
Include dependency graph for ttlTraits.h:
```



## Classes

- struct [hetriang::TTLtraits](#)  
*Traits* class (static struct) for the half-edge data structure.

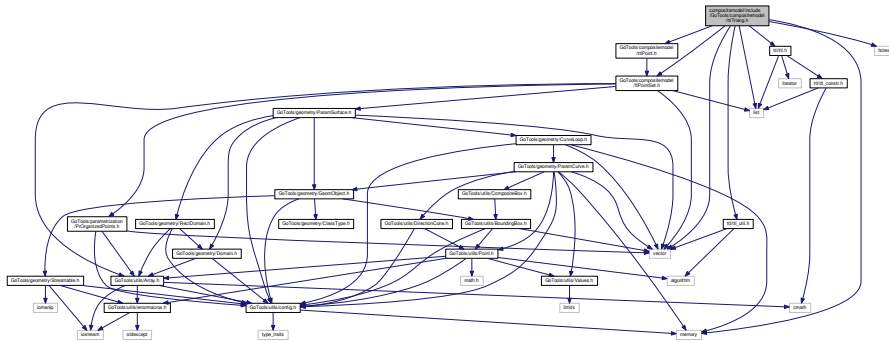
## Namespaces

- [hetriang](#)

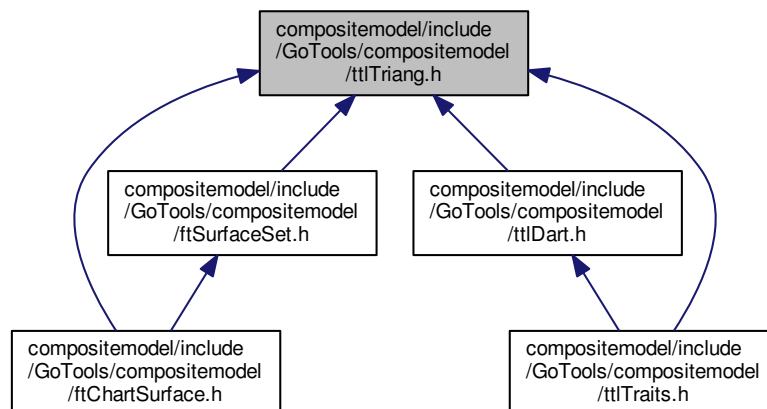
### 30.50 compositemodel/include/GoTools/compositemodel/ttlTriang.h File Reference

```
#include "GoTools/compositemodel/ttlPoint.h"
#include "ttl/ttl.h"
#include "ttl/ttl_util.h"
#include <memory>
#include "GoTools/compositemodel/ftPointSet.h"
#include <fstream>
#include <list>
#include <vector>
```

Include dependency graph for ttlTriang.h:



This graph shows which files directly or indirectly include this file:



#### Classes

- class [hetriang::Node](#)  
*Node* class in the half-edge data structure
- class [hetriang::Edge](#)  
*Edge* class in the half-edge data structure
- class [hetriang::Triangulation](#)  
*Triangulation* class for the half-edge data structure with adaption to TTL.



## Namespaces

- [Go](#)

## 30.52 doc/html/2dpoly\_\_for\_\_s2m\_8h.js File Reference

## 30.53 doc/html/\_adapt\_surface\_8h.js File Reference

### Variables

- [var \\_adapt\\_surface\\_8h](#)

### 30.53.1 Variable Documentation

#### 30.53.1.1 var \_adapt\_surface\_8h

#### Initial value:

```
=
[
 ["adaptSurface", "_adapt_surface_8h.html#a1194b8ffa451b12a281d211e6910d768", null],
 ["approxInSplineSpace", "_adapt_surface_8h.html#a15b06081712dfe9ad4f7d754df64d753", null],
 ["createTriangulation", "_adapt_surface_8h.html#a07973f4fcc58c148043c348d2ad9ddc4", null],
 ["curveApprox", "_adapt_surface_8h.html#a63fc23e20b1e1b0f82145a1d0283cc3c", null],
 ["curveApprox", "_adapt_surface_8h.html#a4797ff542210b4d3ff90953ed305125c", null],
 ["doApprox", "_adapt_surface_8h.html#a95451a902921add6b13b4b7e8f6786cd", null],
 ["expressInSameSplineSpace", "_adapt_surface_8h.html#ae881321f771124da370e9d03089c7f3f", null],
 ["getBoundaryData", "_adapt_surface_8h.html#aadc2781aa90fda2a39b619fc7e976dab", null],
 ["getCornerCorrespondance", "_adapt_surface_8h.html#a40427a58f1173e0f10f116e6b53df53f", null],
 ["getInnerData", "_adapt_surface_8h.html#a09c6dc39b4a998c11e43e86b4e1abb23", null],
 ["parameterizePoints", "_adapt_surface_8h.html#af42c3ae424ec3855831bab07058990e5", null],
 ["projectPoints", "_adapt_surface_8h.html#a4b86cf9be37a6421d44f34651dfca151", null],
 ["updatePointTopology", "_adapt_surface_8h.html#a738ceb1f612a11517883f525ec39b853", null]
]
```

Definition at line 1 of file `_adapt_surface_8h.js`.

## 30.54 doc/html/\_array\_8h.js File Reference

### Variables

- [var \\_array\\_8h](#)

### 30.54.1 Variable Documentation

#### 30.54.1.1 var \_array\_8h

#### Initial value:

```
=
[
 ["Array", "class_go_1_l_array.html", "class_go_1_l_array"],
 ["Vector2D", "_array_8h.html#a68a1cf4bb38d0d831736d42eafbd4027", null],
 ["Vector3D", "_array_8h.html#aa67ec96c4a16e8e493785fceed12f68f", null],
 ["Vector4D", "_array_8h.html#ae69a93ef6927e772507d639d15296506", null],
 ["operator*", "_array_8h.html#a77570f74726be3fd2242a14aeeee0f2", null],
 ["operator<<", "_array_8h.html#ad9be98128b71d8f9a4a03cf253a40cb4", null],
 ["operator>>", "_array_8h.html#a43041127ef073e892f26e9c1189687bc", null]
]
```

Definition at line 1 of file `_array_8h.js`.

## 30.55 doc/html/\_bary\_coord\_system\_8h.js File Reference

### Variables

- var [\\_bary\\_coord\\_system\\_8h](#)

### 30.55.1 Variable Documentation

#### 30.55.1.1 var \_bary\_coord\_system\_8h

##### Initial value:

```
=
[
 ["BaryCoordSystem", "class_go_1_1_bary_coord_system.html", "class_go_1_1_bary_coord_system"],
 ["BaryCoordSystem2D", "_bary_coord_system_8h.html#a12cca84099b202b44209f71f6c9dd20c", null],
 ["BaryCoordSystem3D", "_bary_coord_system_8h.html#a4fd447cf6cb7a456837fe15c599f8077", null],
 ["operator*", "_bary_coord_system_8h.html#a700a91754d00b7aa70c2b1eaf4042c0a", null],
 ["operator<<", "_bary_coord_system_8h.html#abf8183ddde075dcb20899e97cc598ce4", null],
 ["operator>>", "_bary_coord_system_8h.html#a8a299fe67cc03dbd5b87a8d3a2654c04", null]
]
```

Definition at line 1 of file `_bary_coord_system_8h.js`.

## 30.56 doc/html/\_bd\_condition\_type\_8h.js File Reference

### Variables

- var [\\_bd\\_condition\\_type\\_8h](#)

### 30.56.1 Variable Documentation

#### 30.56.1.1 var \_bd\_condition\_type\_8h

##### Initial value:

```
=
[
 ["BdConditionType", "_bd_condition_type_8h.html#a2ef4e46477cac9d1ab25de9f9be20fed7", [
 ["UNKNOWN", "
 _bd_condition_type_8h.html#a2ef4e46477cac9d1ab25de9f9be20fed7a42f3d93048535945ea47f21d2c21dcc9", null],
 ["ZERO_DIRICHLET", "
 _bd_condition_type_8h.html#a2ef4e46477cac9d1ab25de9f9be20fed7aec9786d0435848f55a9401da630edbaaf", null],
 ["CONSTANT_DIRICHLET", "
 _bd_condition_type_8h.html#a2ef4e46477cac9d1ab25de9f9be20fed7aeb0fe5f686259362f9c5a83be92b506", null],
 ["DIRICHLET", "
 _bd_condition_type_8h.html#a2ef4e46477cac9d1ab25de9f9be20fed7a5c504f4a19f84fedb7d51aa7db55592e", null],
 ["ZERO_NEUMANN", "
 _bd_condition_type_8h.html#a2ef4e46477cac9d1ab25de9f9be20fed7a5a1060dcd56fe6e7058351bcf342e1c5", null],
 ["NEUMANN", "
 _bd_condition_type_8h.html#a2ef4e46477cac9d1ab25de9f9be20fed7aed6f828e7a7040280d06e6d5dc58f405", null],
 ["SYMMETRY", "
 _bd_condition_type_8h.html#a2ef4e46477cac9d1ab25de9f9be20fed7aa17bd98b807d8cba21c3ac83184ebcc8", null]
]
]
```

Definition at line 1 of file `_bd_condition_type_8h.js`.

## 30.57 doc/html/\_bernstein\_multi\_8h.js File Reference

### Variables

- var [\\_bernstein\\_multi\\_8h](#)

### 30.57.1 Variable Documentation

#### 30.57.1.1 var \_bernstein\_multi\_8h

##### Initial value:

```
=
[
 ["BernsteinMulti", "class_go_1_1_bernstein_multi.html", "class_go_1_1_bernstein_multi"],
 ["operator*", "_bernstein_multi_8h.html#ac0f9b420a54655ea7980c36b3ae7f353", null],
 ["operator*", "_bernstein_multi_8h.html#a9f71cb0740b68956099216ff4c312f21", null],
 ["operator+", "_bernstein_multi_8h.html#a14bfe575b4f197508ab34231812f5ea8", null],
 ["operator+", "_bernstein_multi_8h.html#ae8d9cb217494b8cff37f175f027d59c2", null],
 ["operator+", "_bernstein_multi_8h.html#a0e45c5beac215ac505022b71f0b684c7", null],
 ["operator+", "_bernstein_multi_8h.html#a3d0cd82e4ab401500bb3570ba24c8daf", null],
 ["operator-", "_bernstein_multi_8h.html#acc8be3d61d13a0e91dd2ae40508d88c8", null],
 ["operator-", "_bernstein_multi_8h.html#a5ac33326e97d2a9169a7309946df94bb", null],
 ["operator-", "_bernstein_multi_8h.html#a06d2d7a07f377da83f913da084c772a7", null],
 ["operator/", "_bernstein_multi_8h.html#a765ac7dff2f0f33a9f0729bb81be373c", null],
 ["operator<<", "_bernstein_multi_8h.html#a7d42865a64dda880f50b6f1b1269149a", null],
 ["operator>>", "_bernstein_multi_8h.html#aeeba7e48ed668867a1d5b4f8737ee320", null]
]
```

Definition at line 1 of file `_bernstein_multi_8h.js`.

## 30.58 doc/html/\_bernstein\_poly\_8h.js File Reference

### Variables

- var [\\_bernstein\\_poly\\_8h](#)

### 30.58.1 Variable Documentation

#### 30.58.1.1 var \_bernstein\_poly\_8h

##### Initial value:

```
=
[
 ["BernsteinPoly", "class_go_1_1_bernstein_poly.html", "class_go_1_1_bernstein_poly"],
 ["operator*", "_bernstein_poly_8h.html#ab84c1aeaf05397990e367ff6bb3daee", null],
 ["operator*", "_bernstein_poly_8h.html#a534b6eb239dcfcf018979c254bbb1c8c", null],
 ["operator+", "_bernstein_poly_8h.html#a3f522b731263ee4398a9b674fedd8fe9", null],
 ["operator+", "_bernstein_poly_8h.html#abfbbfd20652b030cf66e8bcba814a937f", null],
 ["operator+", "_bernstein_poly_8h.html#a12433ab3180672fb9aaa23bbd9d4991b", null],
 ["operator+", "_bernstein_poly_8h.html#adf2c5b270def59383fdd88d04d25c031", null],
 ["operator-", "_bernstein_poly_8h.html#ae21cec482ad8abcb7c1c31b415135b05", null],
 ["operator-", "_bernstein_poly_8h.html#a380e74b8844f2dc85ae5050949c7c531", null],
 ["operator-", "_bernstein_poly_8h.html#af0b73d456cbdc62ef108b8363701eec6", null],
 ["operator/", "_bernstein_poly_8h.html#a7e243a4bc3bb92b66c6ba61f6bca49609", null],
 ["operator<<", "_bernstein_poly_8h.html#a26f2da4ffc29574a70eb27d25685613a", null],
 ["operator>>", "_bernstein_poly_8h.html#affdf8734ba785122985dc1df2a895d5d", null]
]
```

Definition at line 1 of file `_bernstein_poly_8h.js`.

## 30.59 doc/html/\_bernstein\_tetrahedral\_poly\_8h.js File Reference

### Variables

- var [\\_bernstein\\_tetrahedral\\_poly\\_8h](#)

### 30.59.1 Variable Documentation

#### 30.59.1.1 var\_bernstein\_tetrahedral\_poly\_8h

##### Initial value:

```
=
[
 ["BernsteinTetrahedralPoly", "class_go_1_1_bernstein_tetrahedral_poly.html", "
 class_go_1_1_bernstein_tetrahedral_poly"],
 ["operator*", "_bernstein_tetrahedral_poly_8h.html#a34364609c743b4fcee9d139fe06bcaba", null],
 ["operator*", "_bernstein_tetrahedral_poly_8h.html#a9330192cbed096f7cf60b9bc39e3b30a", null],
 ["operator*", "_bernstein_tetrahedral_poly_8h.html#a4d430a1dec1cd20d8645e134a7cf8876", null],
 ["operator+", "_bernstein_tetrahedral_poly_8h.html#a49e5fb7c3f78de991afa7bd7794e8c00", null],
 ["operator+", "_bernstein_tetrahedral_poly_8h.html#ac75490e2444989c2bed34ed7767cb8d1", null],
 ["operator+", "_bernstein_tetrahedral_poly_8h.html#a04a4947cb295e771b77eea322055303e", null],
 ["operator-", "_bernstein_tetrahedral_poly_8h.html#a81be2d89e20687a9eecdadc88a3ac1ba", null],
 ["operator-", "_bernstein_tetrahedral_poly_8h.html#a3b9904b3f6bf5bbca9aaff94e9d6eb30", null],
 ["operator-", "_bernstein_tetrahedral_poly_8h.html#aaf2cf0fc7acla3fb7a2e021fd9f5edd5", null],
 ["operator/", "_bernstein_tetrahedral_poly_8h.html#aec9c6cca3303cbf364ab26bebbb17bf", null],
 ["operator<<", "_bernstein_tetrahedral_poly_8h.html#a177f619335169e255b7e4eb6c9815833", null],
 ["operator>>", "_bernstein_tetrahedral_poly_8h.html#a652c787b70b5bad83b2268ecbe334ecb", null]
]
```

Definition at line 1 of file `_bernstein_tetrahedral_poly_8h.js`.

## 30.60 doc/html/\_bernstein\_triangular\_poly\_8h.js File Reference

### Variables

- var [\\_bernstein\\_triangular\\_poly\\_8h](#)

### 30.60.1 Variable Documentation

#### 30.60.1.1 var\_bernstein\_triangular\_poly\_8h

##### Initial value:

```
=
[
 ["BernsteinTriangularPoly", "class_go_1_1_bernstein_triangular_poly.html", "
 class_go_1_1_bernstein_triangular_poly"],
 ["operator*", "_bernstein_triangular_poly_8h.html#ae593da703f098a6bd5fa8c9dbfa838a5", null],
 ["operator*", "_bernstein_triangular_poly_8h.html#a509d5c4d7f34b09dacf71a0c67a634f3", null],
 ["operator*", "_bernstein_triangular_poly_8h.html#aae2d82c5477642cefa315e60a59a6eec", null],
 ["operator+", "_bernstein_triangular_poly_8h.html#acf6016b168fbc78b01314dd06d655aca", null],
 ["operator+", "_bernstein_triangular_poly_8h.html#a462d4e25cb00e65fd8fca6c7400837d1", null],
 ["operator+", "_bernstein_triangular_poly_8h.html#acf99d1417b42eb82f7990ddb3441fb35", null],
 ["operator-", "_bernstein_triangular_poly_8h.html#aea92d93bae625ac0410a2c5020c30806", null],
 ["operator-", "_bernstein_triangular_poly_8h.html#a6d318aba24152e98f772f8a98fb1855c", null],
 ["operator-", "_bernstein_triangular_poly_8h.html#a0d2fbaac82708424fc51afed89b553b1", null],
 ["operator/", "_bernstein_triangular_poly_8h.html#a85d0e20143ab2df2c6f76ac6d3b48b70", null],
 ["operator<<", "_bernstein_triangular_poly_8h.html#a38161e9c347c6ecf6c10c5376879c4b4", null],
 ["operator>>", "_bernstein_triangular_poly_8h.html#a0dc991998e8beac11b48c8334d17f3ef", null]
]
```

Definition at line 1 of file `_bernstein_triangular_poly_8h.js`.

## 30.61 doc/html/\_bernstein\_utils\_8h.js File Reference

### Variables

- var [\\_bernstein\\_utils\\_8h](#)

### 30.61.1 Variable Documentation

#### 30.61.1.1 var\_bernstein\_utils\_8h

##### Initial value:

```
=
[
 ["bernsteinToSpline", "_bernstein_utils_8h.html#a0bed81e44e84d11e8556824c25049ead", null],
 ["bernsteinToSpline", "_bernstein_utils_8h.html#a1dac32e6ed6c19bf932e4bbade79f344", null],
 ["spline_to_bernstein", "_bernstein_utils_8h.html#a098bb6ff4bac316ae10df8d2b653fa56", null],
 ["spline_to_bernstein", "_bernstein_utils_8h.html#a7547008e8009792bceb9ac9b7096ea27", null],
 ["spline_to_bernstein", "_bernstein_utils_8h.html#ada87974215b573178fd8c13c5f0aacf5", null],
 ["spline_to_bernstein", "_bernstein_utils_8h.html#a4a48f88a6bf4b9175e410bec9debba54", null],
 ["splineToBernstein", "_bernstein_utils_8h.html#a5482851c26f0ec5c7121245fad4b1a7d", null],
 ["splineToBernstein", "_bernstein_utils_8h.html#a1c5ba6de69a967161d8c4b7d84e711c5", null]
]
```

Definition at line 1 of file `_bernstein_utils_8h.js`.

## 30.62 doc/html/\_bezier\_triangle\_8h.js File Reference

### Variables

- var [\\_bezier\\_triangle\\_8h](#)

### 30.62.1 Variable Documentation

#### 30.62.1.1 var\_bezier\_triangle\_8h

##### Initial value:

```
=
[
 ["BezierTriangle", "class_go_1_1_bezier_triangle.html", "class_go_1_1_bezier_triangle"],
 ["__power", "_bezier_triangle_8h.html#a97b81ceb78c6f342eb91ad62ba86260f", null],
 ["__power", "_bezier_triangle_8h.html#ac866b28c409b5c4b6b613e17fd3b5c85", null],
 ["identity_element", "_bezier_triangle_8h.html#a78e37a8fb419e182957f4be0adacl03", null],
 ["identity_element", "_bezier_triangle_8h.html#a0f2e93c7362c26ce388c62a5abc1be52", null],
 ["power", "_bezier_triangle_8h.html#a9f0e2aadelc38elfcefed969201e3b15", null],
 ["power", "_bezier_triangle_8h.html#a3753fc712c0a734234d3a33045379baf", null]
]
```

Definition at line 1 of file `_bezier_triangle_8h.js`.



## 30.63 doc/html/\_bounded\_utils\_8h.js File Reference

### Variables

- var [\\_bounded\\_utils\\_8h](#)

#### 30.63.1 Variable Documentation

##### 30.63.1.1 var \_bounded\_utils\_8h

Definition at line 1 of file `_bounded_utils_8h.js`.

## 30.64 doc/html/\_bounding\_box\_8h.js File Reference

### Variables

- var [\\_bounding\\_box\\_8h](#)

#### 30.64.1 Variable Documentation

##### 30.64.1.1 var \_bounding\_box\_8h

#### Initial value:

```
=
[
 ["BoundingBox", "class_go_1_1_bounding_box.html", "class_go_1_1_bounding_box"],
 ["operator<<", "_bounding_box_8h.html#abc8494ee04093d37f0ff0ffeac48fbcf", null],
 ["operator>>", "_bounding_box_8h.html#ae5b1a7ef34c842092e5cd985a94fc314", null]
]
```

Definition at line 1 of file `_bounding_box_8h.js`.

## 30.65 doc/html/\_bspline\_basis\_8h.js File Reference

### Variables

- var [\\_bspline\\_basis\\_8h](#)

#### 30.65.1 Variable Documentation

##### 30.65.1.1 var \_bspline\_basis\_8h

#### Initial value:

```
=
[
 ["B splineBasis", "class_go_1_1_bspline_basis.html", "class_go_1_1_bspline_basis"],
 ["CHECK", "_bspline_basis_8h.html#a70246e9ac30d8d3e1fba7ca97cd5f4e8", null]
]
```

Definition at line 1 of file `_bspline_basis_8h.js`.

## 30.66 doc/html/\_class\_type\_8h.js File Reference

### Variables

- var [\\_class\\_type\\_8h](#)

### 30.66.1 Variable Documentation

#### 30.66.1.1 var \_class\_type\_8h

Definition at line 1 of file `_class_type_8h.js`.

## 30.67 doc/html/\_closest\_point\_8h.js File Reference

### Variables

- var [\\_closest\\_point\\_8h](#)

### 30.67.1 Variable Documentation

#### 30.67.1.1 var \_closest\_point\_8h

#### Initial value:

```
=
[
 ["AlgorithmChoice", "_closest_point_8h.html#a9ec4373c3aa2905cb547164492f510d1", [
 ["GEOMETRICAL", "_closest_point_8h.html#a9ec4373c3aa2905cb547164492f510d1a4877e954b6cbc33a24efca42fad0cb20", null],
 ["FUNCTIONAL", "_closest_point_8h.html#a9ec4373c3aa2905cb547164492f510d1a0aed87aad2d610531dda3628ea995d8", null]
]],
 ["closestPtCurves", "_closest_point_8h.html#ab76861f84a1fd1ae98cc29dc3b596e59", null],
 ["closestPtCurves", "_closest_point_8h.html#a1d9b397185e29aa817062a44374f9fd5", null],
 ["closestPtCurves2D", "_closest_point_8h.html#a0ab4ffc60e1bca0e0dbb8458d648b9ee", null],
 ["closestPtCurveSurf", "_closest_point_8h.html#a62dbb0e73daa2ba25c33d030bfc1d8d6", null],
 ["closestPtCurveSurf", "_closest_point_8h.html#afeee87804e5a35525c7017d7fcc95c03", null],
 ["closestPtSurfSurfPlane", "_closest_point_8h.html#afe9bbf94f33a5d46b1f7fda8aa0d6bea", null],
 ["closestPtSurfSurfPlaneFunctional", "_closest_point_8h.html#a789a3fc2d43b9234327ebf3bbd3e88c3", null
],
 ["closestPtSurfSurfPlaneGeometrical", "_closest_point_8h.html#a6788ce489922788dc484c43db18c94d4", null
]
]
```

Definition at line 1 of file `_closest_point_8h.js`.

## 30.68 doc/html/\_closest\_point\_utils\_8h.js File Reference

### Variables

- var [\\_closest\\_point\\_utils\\_8h](#)

## 30.68.1 Variable Documentation

### 30.68.1.1 var\_closest\_point\_utils\_8h

#### Initial value:

```
=
[
 ["SurfaceData", "class_go_1_lbox_structuring_1_1_surface_data.html", "
 class_go_1_lbox_structuring_1_1_surface_data"],
 ["SubSurfaceBoundingBox", "class_go_1_lbox_structuring_1_1_sub_surface_bounding_box.html", "
 class_go_1_lbox_structuring_1_1_sub_surface_bounding_box"],
 ["BoundingBoxStructure", "class_go_1_lbox_structuring_1_1_bounding_box_structure.html", "
 class_go_1_lbox_structuring_1_1_bounding_box_structure"],
 ["closestDistances", "_closest_point_utils_8h.html#aea642a6b9559e8ff39dd5d06bc69daac", null],
 ["closestPointCalculations", "_closest_point_utils_8h.html#a6d2dcf73d3785040a689bacc2516eafd", null],
 ["closestPointCalculations", "_closest_point_utils_8h.html#aed2f2b4f85218d4301ea69afc4cc93e2", null],
 ["closestPoints", "_closest_point_utils_8h.html#a8744167826e0e05a35f0de72d2db0fc2", null],
 ["closestPointSingleCalculation", "_closest_point_utils_8h.html#a40d2980555c13353db074c570229503a",
 null],
 ["closestSignedDistances", "_closest_point_utils_8h.html#a8c08c7f6202af07431b1c5a39241e4a2", null],
 ["closestVectorsOld", "_closest_point_utils_8h.html#a26d674d4f146b5604e861ca42c7e4a1f", null],
 ["preProcessClosestVectors", "_closest_point_utils_8h.html#ad64c20172c59d962caacba9501b70331", null]
]
```

Definition at line 1 of file \_closest\_point\_utils\_8h.js.

## 30.69 doc/html/\_coincidence\_8h.js File Reference

### Variables

- var [\\_coincidence\\_8h](#)

## 30.69.1 Variable Documentation

### 30.69.1.1 var\_coincidence\_8h

#### Initial value:

```
=
[
 ["checkCoincide", "_coincidence_8h.html#ab80f651343b83068f28ab4ec23776c49", null],
 ["checkCoincide", "_coincidence_8h.html#a10dcd8f8c2a2f6fd35df74a0266d8c880", null],
 ["checkCoincide", "_coincidence_8h.html#a3a1af9a2b135c4c6a285f482e809844a", null],
 ["checkCoincide", "_coincidence_8h.html#a34ff95b9167b7f556960a27927eef9ff", null],
 ["checkCoincide", "_coincidence_8h.html#ae73e9dd57a9d70a61e617ad0df4bcdal", null],
 ["determineCoincidenceRegion", "_coincidence_8h.html#ad30a2b7f5dec7fb482802e897c5474f5", null],
 ["determineCoincidenceRegion", "_coincidence_8h.html#a8c5497d7799246ea2be86f130063dce8", null],
 ["evalDistCurve", "_coincidence_8h.html#a7068ed17a79ef424dcb5f438ecd3762e", null],
 ["evalProjectedCurve", "_coincidence_8h.html#a9170e06176ec0daa7fa91ed5f26315f1", null],
 ["stepLength", "_coincidence_8h.html#a0a19ab60016163f2c2ba15983e0752b93", null],
 ["stepLength", "_coincidence_8h.html#aa9484c16f49fc09bc9fe0d8a1e1c7a13", null]
]
```

Definition at line 1 of file \_coincidence\_8h.js.

## 30.70 doc/html/\_composite\_model\_8h.js File Reference

### Variables

- var [\\_composite\\_model\\_8h](#)

### 30.70.1 Variable Documentation

#### 30.70.1.1 var \_composite\_model\_8h

##### Initial value:

```
=
[
 ["CompositeModel", "class_go_1_1_composite_model.html", "class_go_1_1_composite_model"],
 ["closestPointLevel", "_composite_model_8h.html#a64e7b23868d471d8f59b8e280437195e", [
 ["LOCAL_SEARCH", "
 _composite_model_8h.html#a64e7b23868d471d8f59b8e280437195ea67da27ebc2731782484f21d1d9acb8fd", null],
 ["SEMI_LOCAL_SEARCH", "
 _composite_model_8h.html#a64e7b23868d471d8f59b8e280437195eabd2e6a95880a37b4ccf6522c7805dc25", null],
 ["GLOBAL_SEARCH", "
 _composite_model_8h.html#a64e7b23868d471d8f59b8e280437195ea31db71b7b0faac9b18302d7d0b14a861", null]
]]
]
```

Definition at line 1 of file `_composite_model_8h.js`.

## 30.71 doc/html/\_constraint\_definitions\_8h.js File Reference

### Variables

- var [\\_constraint\\_definitions\\_8h](#)

### 30.71.1 Variable Documentation

#### 30.71.1.1 var \_constraint\_definitions\_8h

##### Initial value:

```
=
[
 ["sideConstraint", "struct_go_1_1side_constraint.html", "struct_go_1_1side_constraint"],
 ["sideConstraintSet", "struct_go_1_1side_constraint_set.html", "struct_go_1_1side_constraint_set"],
 ["sideConstraint", "_constraint_definitions_8h.html#ab5e42d330d6c5ac6e51b604a52c0d4d7", null],
 ["sideConstraintSet", "_constraint_definitions_8h.html#af57c06077dc46e0ee165b4f9c38280e", null]
]
```

Definition at line 1 of file `_constraint_definitions_8h.js`.

## 30.72 doc/html/\_coons\_patch\_gen\_8h.js File Reference

### Variables

- var [\\_coons\\_patch\\_gen\\_8h](#)

### 30.72.1 Variable Documentation

#### 30.72.1.1 var \_coons\_patch\_gen\_8h

##### Initial value:

```
=
[
 ["UnknownError", "class_go_1_1_coons_patch_gen_1_1_un_known_error.html", null],
 ["addMissingCrossCurves", "_coons_patch_gen_8h.html#abcd8b6517d8699c3c73b92e7ce9c013e", null],
 ["blendcoef", "_coons_patch_gen_8h.html#a426bdc367559559cf17aa3530a88bcba", null],
 ["createCoonsPatch", "_coons_patch_gen_8h.html#ad0f0aa95e3833ecc132f1db245905386", null],
 ["createCoonsPatch", "_coons_patch_gen_8h.html#ae26c2c1d02c51914a7b7260cd94fb3a1", null],
 ["createCoonsPatch", "_coons_patch_gen_8h.html#a11bdc62865ac134427133be8b578fe74", null],
 ["createGordonSurface", "_coons_patch_gen_8h.html#afde30acdf5a34b0cd4aced7fb4828e3a", null],
 ["createGordonSurface", "_coons_patch_gen_8h.html#a2451696905b72a1619c5de84123114c6", null],
 ["doCreateSurface", "_coons_patch_gen_8h.html#a2b5b8f0ea79803cdba7b57c72b3f8586", null],
 ["fixCrossEndPts", "_coons_patch_gen_8h.html#a2e83dc35642c7eb4a09ea8a04fd92a0d", null],
 ["getCrossTangs", "_coons_patch_gen_8h.html#a047980670e3c2c6125cd0519d7e47773", null],
 ["getTangBlends", "_coons_patch_gen_8h.html#ad8b1e16dcd7894194102a93640f2e5c9", null],
 ["hermit", "_coons_patch_gen_8h.html#ae25a1d1703dd7b45951b442709729a1b", null],
 ["loftSurface", "_coons_patch_gen_8h.html#ab3f83eae4e1e27072374f7a49086b3aa", null],
 ["loftSurface", "_coons_patch_gen_8h.html#aa09af1b3d20cb24b6fbfbf0be7004ee0", null],
 ["loftSurface", "_coons_patch_gen_8h.html#a2bd9185e55b6b12c43eba27576555083", null],
 ["makeLoftParams", "_coons_patch_gen_8h.html#a2c1ad6de3b84a80a1a257a12b0e7db7", null],
 ["reparamBoundaryCurve", "_coons_patch_gen_8h.html#a59c8a7a874409e9ef473635fef96d3e4", null],
 ["sortMeshCurves", "_coons_patch_gen_8h.html#abdf9064bb03437bd206503ecd2448f02", null],
 ["splitMeshCurves", "_coons_patch_gen_8h.html#ac10aaf26bd04falaac5fd7f3aaebc127", null],
 ["tpSurface", "_coons_patch_gen_8h.html#a4c8a39b9e4bf8c741ed4cebd9361c19", null]
]
```

Definition at line 1 of file `_coons_patch_gen_8h.js`.

## 30.73 doc/html/\_coons\_patch\_volume\_gen\_8h.js File Reference

### Variables

- var [\\_coons\\_patch\\_volume\\_gen\\_8h](#)

### 30.73.1 Variable Documentation

#### 30.73.1.1 var \_coons\_patch\_volume\_gen\_8h

##### Initial value:

```
=
[
 ["createCoonsPatch", "_coons_patch_volume_gen_8h.html#a54726922b1e5f3b36c95b3145c5dde86", null],
 ["createCoonsPatchDirectly", "_coons_patch_volume_gen_8h.html#ab367998ce6c2d5aeb2b90ff687e583c8", null],
 ["edge_curves_equal", "_coons_patch_volume_gen_8h.html#af3266d664434db11a0a73a5ba524e652", null],
 ["get_corners", "_coons_patch_volume_gen_8h.html#adec8c92f76747dd5e4f15b2360334cce", null],
 ["push_corners", "_coons_patch_volume_gen_8h.html#a16da1675eb8b2d9e6c5dd8884bc7cbf4", null]
]
```

Definition at line 1 of file `_coons_patch_volume_gen_8h.js`.

## 30.74 doc/html/\_creators\_offset\_utils\_8h.js File Reference

### Variables

- var [\\_creators\\_offset\\_utils\\_8h](#)

### 30.74.1 Variable Documentation

#### 30.74.1.1 var \_creators\_offset\_utils\_8h

##### Initial value:

```
=
[
 ["_OFFSETUTILS_H", "_creators_offset_utils_8h.html#ad9fc2da1082c23bb539309710e76122f", null],
 ["blend_sl421", "_creators_offset_utils_8h.html#ab2f4932c752729805bde2446a5482514", null]
]
```

Definition at line 1 of file `_creators_offset_utils_8h.js`.

## 30.75 doc/html/\_creators\_utils\_8h.js File Reference

### Variables

- var [\\_creators\\_utils\\_8h](#)

### 30.75.1 Variable Documentation

#### 30.75.1.1 var \_creators\_utils\_8h

##### Initial value:

```
=
[
 ["createCrossTangent", "_creators_utils_8h.html#a829df5c9ecc1975b683e7529602bbf88", null],
 ["createCrossTangent", "_creators_utils_8h.html#a1d6a75a5ced06b87b421c4b910e63a8e", null],
 ["fixSeemCurves", "_creators_utils_8h.html#a9848d6bdd8b82ae9c36383395c445923", null],
 ["fixTrimCurves", "_creators_utils_8h.html#a4317c830c62ef48865e1f85ca29d1230", null],
 ["getParametricCurve", "_creators_utils_8h.html#a5e5bf433f7a17e559ff81dc9f1ef086b", null],
 ["projectCurvePoint", "_creators_utils_8h.html#a1aa70202d3e7396d3d092b486bcf7d01", null],
 ["projectCurvePoint", "_creators_utils_8h.html#ab40a9a304dc881074d88899b764d2d6e", null],
 ["projectPoint", "_creators_utils_8h.html#a73525d8b34b7e17d72599f5a2d0b6857", null]
]
```

Definition at line 1 of file `_creators_utils_8h.js`.

## 30.76 doc/html/\_curvature\_8h.js File Reference

### Variables

- var [\\_curvature\\_8h](#)

## 30.76.1 Variable Documentation

### 30.76.1.1 var \_curvature\_8h

#### Initial value:

```
=
[
 ["curvatureRadiusPoints", "_curvature_8h.html#a55b1e94cb2383c1231b3e031cbf56947", null],
 ["minimalCurvatureRadius", "_curvature_8h.html#acd9ba3a40afd2f42b3dafc68b687486b", null]
]
```

Definition at line 1 of file \_curvature\_8h.js.

## 30.77 doc/html/\_curvature\_analysis\_8h.js File Reference

### Variables

- [var \\_curvature\\_analysis\\_8h](#)

## 30.77.1 Variable Documentation

### 30.77.1.1 var \_curvature\_analysis\_8h

#### Initial value:

```
=
[
 ["computeFirstFundamentalForm", "_curvature_analysis_8h.html#a8cf34a592d0ad2d00c92fa04a165f4e4", null],
 ["computeSecondFundamentalForm", "_curvature_analysis_8h.html#a4bb9d690d56c189557a77a653e391d11", null],
 ["curvatures", "_curvature_analysis_8h.html#a727500a1c59435cfe0369d9a8d1c7d39", null],
 ["evaluateMinCurvatureRadius", "_curvature_analysis_8h.html#a6fc6900c4d1f5efb51cba2d0aca99a30", null]
 ,
 ["minimalCurvatureRadius", "_curvature_analysis_8h.html#a0cd3083e864936b349b65b62169067e9", null],
 ["principalCurvatures", "_curvature_analysis_8h.html#aaaf22d9e6cd82c03dd1239c90a0ee6468", null]
]
```

Definition at line 1 of file \_curvature\_analysis\_8h.js.

## 30.78 doc/html/\_curvature\_utils\_8h.js File Reference

### Variables

- [var \\_curvature\\_utils\\_8h](#)

### 30.78.1 Variable Documentation

#### 30.78.1.1 var \_curvature\_utils\_8h

##### Initial value:

```
=
[
 ["curvatureRadius", "_curvature_utils_8h.html#a8074e8bf6506a7cbcf018ab8a8947bc0", null],
 ["getHermiteData", "_curvature_utils_8h.html#a9b46ce19f0418597ccc67ee4e6cdef3b", null],
 ["stepLenFromRadius", "_curvature_utils_8h.html#a5706ecc2f2b2821adf457094dbbecdef", null],
 ["tanLenFromRadius", "_curvature_utils_8h.html#a87059cc6cddba0b64ae502b9c4d52a52", null]
]
```

Definition at line 1 of file \_curvature\_utils\_8h.js.

## 30.79 doc/html/\_curve\_creators\_8h.js File Reference

### Variables

- var [\\_curve\\_creators\\_8h](#)

### 30.79.1 Variable Documentation

#### 30.79.1.1 var \_curve\_creators\_8h

##### Initial value:

```
=
[
 ["approxCurves", "_curve_creators_8h.html#a704191d8280a68d26847b450ba04bf94", null],
 ["blend", "_curve_creators_8h.html#a2dafc0b5183e58d1fcf86e891e02a5da", null],
 ["createCircle", "_curve_creators_8h.html#a158fcda72251457a1d13f4fa86fcb28c", null],
 ["curveApprox", "_curve_creators_8h.html#a8dc25e7453ec9dcab07d0f51a868dea5", null],
 ["curveApprox", "_curve_creators_8h.html#a7f3047d4b2671a0942d339addb2d3220", null],
 ["insertParamDomain", "_curve_creators_8h.html#aadeeb32226dba4888e5cfcbca2071b86", null],
 ["liftParameterCurve", "_curve_creators_8h.html#a3900fe644f843d5692303908b21676d2", null],
 ["multCurveWithFunction", "_curve_creators_8h.html#a7c2c75aa8328743594f5ae6a31ccf332", null],
 ["offsetCurve", "_curve_creators_8h.html#a9c415126f128a6bf9fd1f368dc7309ed", null],
 ["projectCurve", "_curve_creators_8h.html#a598c8ebac32febb03783a4756746b26b", null],
 ["projectSpaceCurve", "_curve_creators_8h.html#aa227f8ea8e1d21bb27d60c8f4cf0a998", null]
]
```

Definition at line 1 of file \_curve\_creators\_8h.js.

## 30.80 doc/html/\_curve\_interpolator\_8h.js File Reference

### Variables

- var [\\_curve\\_interpolator\\_8h](#)



### 30.80.1 Variable Documentation

#### 30.80.1.1 var \_curve\_interpolator\_8h

**Initial value:**

```
=
[
 ["regularInterpolation", "_curve_interpolator_8h.html#a164216ef97184a5250a9c934745e8e42", null]
]
```

Definition at line 1 of file \_curve\_interpolator\_8h.js.

## 30.81 doc/html/\_curve\_loop\_8h.js File Reference

### Variables

- var [\\_curve\\_loop\\_8h](#)

### 30.81.1 Variable Documentation

#### 30.81.1.1 var \_curve\_loop\_8h

**Initial value:**

```
=
[
 ["CurveLoop", "class_go_1_1_curve_loop.html", "class_go_1_1_curve_loop"],
 ["computeLoopGap", "_curve_loop_8h.html#a8ec77aba0c49d2e5543a016f2c51cd10", null]
]
```

Definition at line 1 of file \_curve\_loop\_8h.js.

## 30.82 doc/html/\_direction2\_d\_8h.js File Reference

### Variables

- var [\\_direction2\\_d\\_8h](#)

### 30.82.1 Variable Documentation

#### 30.82.1.1 var \_direction2\_d\_8h

**Initial value:**

```
=
[
 ["Direction2D", "_direction2_d_8h.html#a821a750d4a6740898072cf759246eb87", [
 ["XFIXED", "_direction2_d_8h.html#a821a750d4a6740898072cf759246eb87a1e4145becb15ce54486dd57c554a8716", null],
 ["YFIXED", "_direction2_d_8h.html#a821a750d4a6740898072cf759246eb87a344e219d46e24ae18ec2ff7fe11afddd", null]
]]
]
```

Definition at line 1 of file \_direction2\_d\_8h.js.

## 30.83 doc/html/\_face\_utilities\_8h.js File Reference

### Variables

- [var \\_face\\_utilities\\_8h](#)

### 30.83.1 Variable Documentation

#### 30.83.1.1 var \_face\_utilities\_8h

##### Initial value:

```
=
[
 ["SamplePointData", "struct_go_1_1_sample_point_data.html", "struct_go_1_1_sample_point_data"],
 ["enforceCoLinearity", "_face_utilities_8h.html#a9885c0102c47f161b0c3bd1b23b30e0f", null],
 ["enforceVxCoLinearity", "_face_utilities_8h.html#a2ca0231f2063214726c5b93c20f36ec5", null],
 ["getBoundaryData", "_face_utilities_8h.html#a2655415c3d346862dbb5007d42416ad4", null],
 ["getInnerData", "_face_utilities_8h.html#ad48da9124efcdee9b255bb48e7fc0c98", null]
]
```

Definition at line 1 of file `_face_utilities_8h.js`.

## 30.84 doc/html/\_factory\_8h.js File Reference

### Variables

- [var \\_factory\\_8h](#)

### 30.84.1 Variable Documentation

#### 30.84.1.1 var \_factory\_8h

##### Initial value:

```
=
[
 ["Creator", "class_go_1_1_creator.html", "class_go_1_1_creator"],
 ["ConcreteCreator", "class_go_1_1_concrete_creator.html", "class_go_1_1_concrete_creator"],
 ["Factory", "class_go_1_1_factory.html", "class_go_1_1_factory"],
 ["Registrar", "class_go_1_1_registrator.html", "class_go_1_1_registrator"],
 ["Register", "_factory_8h.html#a8ac9e95a1dc83d595e751bea0df8561c", null]
]
```

Definition at line 1 of file `_factory_8h.js`.

## 30.85 doc/html/\_gap\_removal\_8h.js File Reference

### Variables

- [var \\_gap\\_removal\\_8h](#)

### 30.85.1 Variable Documentation

#### 30.85.1.1 var \_gap\_removal\_8h

##### Initial value:

```
=
[
 ["checkBoundaryDist", "_gap_removal_8h.html#a043aa4c44fc86fbbb6230a156a6b002c", null],
 ["getBoundarySamples", "_gap_removal_8h.html#ae5b097a95c42dd286dbcb88743554902", null],
 ["getCoefConstraints", "_gap_removal_8h.html#a2460c4084b5a9190b9720d1078933098", null],
 ["getSplineAndBd", "_gap_removal_8h.html#a88637576f47e7963ed4adf44298fa568", null],
 ["modifyAtVertex", "_gap_removal_8h.html#ac9fd81db44f84ff78587e1ff2eadb215", null],
 ["modifySplines", "_gap_removal_8h.html#a56726c7530054fc24c0a8d001e5e4b1e", null],
 ["modifySplineSf", "_gap_removal_8h.html#a3407664edbdaf9abdaa458e69347b1a3", null],
 ["removeGapSpline", "_gap_removal_8h.html#a8122e2ec705cf19564b4ec418d495cca", null],
 ["removeGapSpline2", "_gap_removal_8h.html#a8fa43c39c2e495472d42b225ae6ecc80", null],
 ["removeGapSplineTrim", "_gap_removal_8h.html#adc46385088155f36ced81c2df5ca414a", null],
 ["removeGapTrim", "_gap_removal_8h.html#a8757b1449a3f2e65b785488d315d5ec9", null],
 ["replaceCurvePiece", "_gap_removal_8h.html#ab6bf0bd8f417de8ace6050351f67f480", null]
]
```

Definition at line 1 of file \_gap\_removal\_8h.js.

## 30.86 doc/html/\_gap\_removal\_volume\_8h.js File Reference

### Variables

- var [\\_gap\\_removal\\_volume\\_8h](#)

### 30.86.1 Variable Documentation

#### 30.86.1.1 var \_gap\_removal\_volume\_8h

##### Initial value:

```
=
[
 ["removeGapSpline", "_gap_removal_volume_8h.html#a825b245aaed99dbfc9d2de9d743da7f6", null]
]
```

Definition at line 1 of file \_gap\_removal\_volume\_8h.js.

## 30.87 doc/html/\_general\_function\_minimizer\_8h.js File Reference

### Variables

- var [\\_general\\_function\\_minimizer\\_8h](#)

### 30.87.1 Variable Documentation

#### 30.87.1.1 `var _general_function_minimizer_8h`

**Initial value:**

```
=
[
 ["FunctionMinimizer", "class_go_1_1_function_minimizer.html", "class_go_1_1_function_minimizer"],
 ["FunctionMinimizer", "class_go_1_1_function_minimizer.html", "class_go_1_1_function_minimizer"],
 ["minimise_conjugated_gradient", "
 _general_function_minimizer_8h.html#a96d9737a699a1a929eb4dbaef18cce0e", null]
]
```

Definition at line 1 of file `_general_function_minimizer_8h.js`.

## 30.88 `doc/html/_general_function_minimizer__implementation_8h.js` File Reference

### Variables

- [var `\_general\_function\_minimizer\_\_implementation\_8h`](#)

### 30.88.1 Variable Documentation

#### 30.88.1.1 `var _general_function_minimizer__implementation_8h`

**Initial value:**

```
=
[
 ["minimise_conjugated_gradient", "
 _general_function_minimizer__implementation_8h.html#a96d9737a699a1a929eb4dbaef18cce0e", null]
]
```

Definition at line 1 of file `_general_function_minimizer__implementation_8h.js`.

## 30.89 `doc/html/_geom_object_8h.js` File Reference

### Variables

- [var `\_geom\_object\_8h`](#)

### 30.89.1 Variable Documentation

#### 30.89.1.1 var \_geom\_object\_8h

##### Initial value:

```
=
[
 ["GeomObject", "class_go_1_1_geom_object.html", "class_go_1_1_geom_object"],
 ["MAJOR_VERSION", "_geom_object_8h.html#a8c682769e20547696ce06a2d6b3ce871", null],
 ["MINOR_VERSION", "_geom_object_8h.html#a4f9cf4f5710b80dd24fb029acdd69637", null]
]
```

Definition at line 1 of file \_geom\_object\_8h.js.

## 30.90 doc/html/\_geometry\_tools\_8h.js File Reference

### Variables

- var [\\_geometry\\_tools\\_8h](#)

### 30.90.1 Variable Documentation

#### 30.90.1.1 var \_geometry\_tools\_8h

Definition at line 1 of file \_geometry\_tools\_8h.js.

## 30.91 doc/html/\_go\_intersections\_8h.js File Reference

### Variables

- var [\\_go\\_intersections\\_8h](#)

### 30.91.1 Variable Documentation

#### 30.91.1.1 var \_go\_intersections\_8h

##### Initial value:

```
=
[
 ["pretop_UNDEF", "
_go_intersections_8h.html#a0589ebbe872cac278faf8159c61b8292a3ead4181c54e7d4f7e0add68c20ad134", null],
 ["pretop_IN", "
_go_intersections_8h.html#a0589ebbe872cac278faf8159c61b8292a418cdf83cdf220855b9ed6eec2a3f68d", null],
 ["pretop_OUT", "
_go_intersections_8h.html#a0589ebbe872cac278faf8159c61b8292a6656387d8b452630d832f8a6ba20c79d", null],
 ["pretop_ON", "
_go_intersections_8h.html#a0589ebbe872cac278faf8159c61b8292a57715e2bd060af960bad6c7fa862d717", null],
 ["pretop_AT", "
_go_intersections_8h.html#a0589ebbe872cac278faf8159c61b8292ab0740d5b95cb39891d4fbc9e8ae72315", null],
 ["closestPtCurves", "
_go_intersections_8h.html#ae9f16856595109b8546a03c4ffe7068e", null],
 ["intersect2Dcurves", "
_go_intersections_8h.html#afc5de86c01d5dccfb0415a124e393277", null],
 ["intersectCurvePoint", "
_go_intersections_8h.html#af156ae3bfdbf586c1a54f12b76729bfff", null],
 ["intersectcurves", "
_go_intersections_8h.html#a8437da0e9da0f9f2d90d4cba7e7a0423", null],
 ["intersectCurveSurf", "
_go_intersections_8h.html#a3012f3eebc00a086ef19161f6421169d", null]
]
```

Definition at line 1 of file \_go\_intersections\_8h.js.

## 30.92 doc/html/\_go\_read\_write\_8cpp.js File Reference

### Variables

- var [\\_go\\_read\\_write\\_8cpp](#)

### 30.92.1 Variable Documentation

#### 30.92.1.1 var \_go\_read\_write\_8cpp

#### Initial value:

```
=
[
 ["readGoCurve", "_go_read_write_8cpp.html#a8c022a190ba41f0a616981f639c61e3c", null],
 ["readGoPoints", "_go_read_write_8cpp.html#ab2870835b9150e3469d9751597e7b5e9", null],
 ["readGoSurface", "_go_read_write_8cpp.html#ab0a168989d4c5eb986377c2a619d36a6", null],
 ["writeGoCurve", "_go_read_write_8cpp.html#a604ab007ccebada02c1f7f0e6292843d", null],
 ["writeGoPoints", "_go_read_write_8cpp.html#aab52ace77579fb664a483672206f5bd1", null],
 ["writeGoSurface", "_go_read_write_8cpp.html#aafcea5555e7f0afdb947bf425ccc594a", null]
]
```

Definition at line 1 of file `_go_read_write_8cpp.js`.

## 30.93 doc/html/\_go\_read\_write\_8h.js File Reference

### Variables

- var [\\_go\\_read\\_write\\_8h](#)

### 30.93.1 Variable Documentation

#### 30.93.1.1 var \_go\_read\_write\_8h

#### Initial value:

```
=
[
 ["readGoCurve", "_go_read_write_8h.html#a4c192b15e3015a57c0e0fed19c063758", null],
 ["readGoPoints", "_go_read_write_8h.html#ab2870835b9150e3469d9751597e7b5e9", null],
 ["readGoSurface", "_go_read_write_8h.html#a4c3063b821ee6123a7f18062e57f1acb", null],
 ["writeGoCurve", "_go_read_write_8h.html#a604ab007ccebada02c1f7f0e6292843d", null],
 ["writeGoPoints", "_go_read_write_8h.html#aab52ace77579fb664a483672206f5bd1", null],
 ["writeGoSurface", "_go_read_write_8h.html#a8c958736c880b2f546acccd6a41a49ab", null]
]
```

Definition at line 1 of file `_go_read_write_8h.js`.

## 30.94 doc/html/\_go\_tools\_\_version\_8h.js File Reference

### Variables

- var [\\_go\\_tools\\_\\_version\\_8h](#)

#### 30.94.1 Variable Documentation

##### 30.94.1.1 var \_go\_tools\_\_version\_8h

###### Initial value:

```
=
[
 ["GO_VERSION_MAJOR", "_go_tools__version_8h.html#acf93ae65e8651c85a2237918a9801eff", null],
 ["GO_VERSION_MINOR", "_go_tools__version_8h.html#aa78bb3e4b1fbd25ebc24743214ff1ff9", null],
 ["GO_VERSION_PATCH", "_go_tools__version_8h.html#a6ac408a33e44fad62ff96ca978a90045", null]
]
```

Definition at line 1 of file `_go_tools__version_8h.js`.

## 30.95 doc/html/\_hahns\_surface\_gen\_8h.js File Reference

### Variables

- var [\\_hahns\\_surface\\_gen\\_8h](#)

#### 30.95.1 Variable Documentation

##### 30.95.1.1 var \_hahns\_surface\_gen\_8h

###### Initial value:

```
=
[
 ["constructHahnsSurface", "_hahns_surface_gen_8h.html#a5178eba7f94853022b2b3fdca849a1bd", null],
 ["constructPolygonialSurface", "_hahns_surface_gen_8h.html#a6528cba4d1c2401b1e2dda281e08cbd9", null]
]
```

Definition at line 1 of file `_hahns_surface_gen_8h.js`.

## 30.96 doc/html/\_he\_triang\_8h.js File Reference

### Variables

- var [\\_he\\_triang\\_8h](#)

## 30.96.1 Variable Documentation

### 30.96.1.1 var\_he\_triang\_8h

#### Initial value:

```
=
[
 ["Node", "classhed_l_l_node.html", "classhed_l_l_node"],
 ["Edge", "classhed_l_l_edge.html", "classhed_l_l_edge"],
 ["Triangulation", "classhed_l_l_triangulation.html", "classhed_l_l_triangulation"],
 ["TTL_USE_NODE_FLAG", "_he_triang_8h.html#ab98540b0e1ab0f43a73fa85e3bf2a8c3", null],
 ["TTL_USE_NODE_ID", "_he_triang_8h.html#a4693e6332f64f266bae5663937fcb284", null]
]
```

Definition at line 1 of file `_he_triang_8h.js`.

## 30.97 doc/html/\_i\_g\_e\_sconverter\_8h.js File Reference

### Variables

- var [\\_i\\_g\\_e\\_sconverter\\_8h](#)

## 30.97.1 Variable Documentation

### 30.97.1.1 var\_i\_g\_e\_sconverter\_8h

#### Initial value:

```
=
[
 ["IGESheader", "struct_go_l_l_i_g_e_sheader.html", "struct_go_l_l_i_g_e_sheader"],
 ["EntityList", "struct_go_l_l_entity_list.html", "struct_go_l_l_entity_list"],
 ["IGESdirentry", "struct_go_l_l_i_g_e_sdirentry.html", "struct_go_l_l_i_g_e_sdirentry"],
 ["IGESconverter", "class_go_l_l_i_g_e_sconverter.html", "class_go_l_l_i_g_e_sconverter"],
 ["FileFormat", "_i_g_e_sconverter_8h.html#alb060f2a0c10fd0bc766c070bf3f86d4", [
 ["go", "_i_g_e_sconverter_8h.html#alb060f2a0c10fd0bc766c070bf3f86d4a727b71e7d5e25b99928ed913a31437f0", null],
 ["disp", "_i_g_e_sconverter_8h.html#alb060f2a0c10fd0bc766c070bf3f86d4afdb0b616372654f2b42dbf875f740077", null],
 ["IGES", "_i_g_e_sconverter_8h.html#alb060f2a0c10fd0bc766c070bf3f86d4ae0747afbdd38bb230962db3f51df27d5", null]
]],
 ["IGESSection", "_i_g_e_sconverter_8h.html#a95d383314d9ef7d277424ee672473063", [
 ["S", "_i_g_e_sconverter_8h.html#a95d383314d9ef7d277424ee672473063a9175c6d2bfb458461de6f564a532dda5", null],
 ["G", "_i_g_e_sconverter_8h.html#a95d383314d9ef7d277424ee672473063aa32eca3219a522f885088233d0cf8aa3", null],
 ["D", "_i_g_e_sconverter_8h.html#a95d383314d9ef7d277424ee672473063ada8cdab2225fa4f764ba2ff6d553f3d4", null],
 ["P", "_i_g_e_sconverter_8h.html#a95d383314d9ef7d277424ee672473063ad1ce8b353acf910f8871322784488407", null],
 ["T", "_i_g_e_sconverter_8h.html#a95d383314d9ef7d277424ee672473063ad80deac00a5f9e9bea01b087b3bc6474", null],
 ["E", "_i_g_e_sconverter_8h.html#a95d383314d9ef7d277424ee672473063a67661f5f4694b7e01c81e2e32d5841d0", null]
]]
]
```

Definition at line 1 of file `_i_g_e_sconverter_8h.js`.



## 30.98 doc/html/\_implicit\_utils\_8h.js File Reference

### Variables

- var `_implicit_utils_8h`

### 30.98.1 Variable Documentation

#### 30.98.1.1 var `_implicit_utils_8h`

##### Initial value:

```
=
[
 ["cart_to_bary", "_implicit_utils_8h.html#a2bf45a2d4003691b4f59200a7736c65f", null],
 ["cart_to_bary", "_implicit_utils_8h.html#a815f7e0de1d83ead4b066ad1e2284c79", null],
 ["cart_to_bary", "_implicit_utils_8h.html#af9b3d94ca3aldab9168732b5ca4e8bfa", null],
 ["cart_to_bary", "_implicit_utils_8h.html#ab88f74a141a96d124d15bc01408e9505", null],
 ["create_bary_coord_system2D", "_implicit_utils_8h.html#a7b801275b11d7667cc1f5d45307f48e4", null],
 ["create_bary_coord_system3D", "_implicit_utils_8h.html#a70845387423ca16803f43291a2a13639", null],
 ["create_bary_coord_system3D", "_implicit_utils_8h.html#a9a44a4097f20e78aad8ace234f9c62a", null],
 ["create_bary_coord_system3D", "_implicit_utils_8h.html#ad432da218425ec5badf70eed364e4ea8", null],
 ["create_bary_coord_system3D", "_implicit_utils_8h.html#ab9661f3a938911bb440140c1114dcd8f", null],
 ["make_implicit_gauss", "_implicit_utils_8h.html#af861508aa242c5f3b156080f33c84cfc", null],
 ["make_implicit_svd", "_implicit_utils_8h.html#a7a3f7dd2036591326e15e39ebbf5bc3c", null],
 ["make_matrix", "_implicit_utils_8h.html#ab49c4c7fb3851dc49e43bd8e458a558b", null],
 ["make_matrix", "_implicit_utils_8h.html#aca403dc7adf6d81bbd91965d8fc32052", null],
 ["make_matrix", "_implicit_utils_8h.html#a2fa490ef57992e3592810cb53e20e73d", null]
]
```

Definition at line 1 of file `_implicit_utils_8h.js`.

## 30.99 doc/html/\_int\_results\_model\_8h.js File Reference

### Variables

- var `_int_results_model_8h`

### 30.99.1 Variable Documentation

#### 30.99.1.1 var `_int_results_model_8h`

##### Initial value:

```
=
[
 ["IntResultsModel", "class_go_1_1_int_results_model.html", "class_go_1_1_int_results_model"],
 ["IntersectionType", "_int_results_model_8h.html#a8dbaf65eacdc80d8d1714419e976ed2b", [
 ["No_Type", "
 _int_results_model_8h.html#a8dbaf65eacdc80d8d1714419e976ed2babcd670721a68d8943fefaf6f071c7ab7", null],
 ["SurfaceModel_SurfaceModel", "
 _int_results_model_8h.html#a8dbaf65eacdc80d8d1714419e976ed2baee87611effea029dfe473a4624a95dfa", null],
 ["SurfaceModel_Plane", "
 _int_results_model_8h.html#a8dbaf65eacdc80d8d1714419e976ed2baf9621d4293551c45e350597f13946246", null],
 ["SurfaceModel_Line", "
 _int_results_model_8h.html#a8dbaf65eacdc80d8d1714419e976ed2ba0dfb696f6e60baec701c53148a04bbe2", null],
 ["CompositeCurve_CompositeCurve", "
 _int_results_model_8h.html#a8dbaf65eacdc80d8d1714419e976ed2ba5fba73d495653899d9d0ec04ab7f8b0d", null],
 ["CompositeCurve_Plane", "
 _int_results_model_8h.html#a8dbaf65eacdc80d8d1714419e976ed2ba3d2cd6b6e2bfd2faf5d66a14ccc2202e", null],
 ["CompositeCurve_Line", "
 _int_results_model_8h.html#a8dbaf65eacdc80d8d1714419e976ed2ba19c967c2cfc4511f6b597999263cafd", null]
]]
]
```

Definition at line 1 of file `_int_results_model_8h.js`.

## 30.100 doc/html/\_integrate\_8h.js File Reference

### Variables

- var [\\_integrate\\_8h](#)

### 30.100.1 Variable Documentation

#### 30.100.1.1 var \_integrate\_8h

##### Initial value:

```
=
[
 ["GaussQuadInner", "_integrate_8h.html#a44435785e8233b6bfff4b3d2d8c320057", null],
 ["GaussQuadInner2", "_integrate_8h.html#a1893f2d3c867396837b2f3954afcdcde", null],
 ["GaussQuadInnerFlat", "_integrate_8h.html#a2905af84f191007e0597a1d73c1ffc4d", null],
 ["GaussQuadInnerRational", "_integrate_8h.html#aca2dfd3b6d0a8c31bed3dceb1a0b8029", null],
 ["GaussQuadValues", "_integrate_8h.html#adcab00389069ae725ccfe0581664ab68", null]
]
```

Definition at line 1 of file `_integrate_8h.js`.

## 30.101 doc/html/\_integration\_8h.js File Reference

### Variables

- var [\\_integration\\_8h](#)

### 30.101.1 Variable Documentation

#### 30.101.1.1 var \_integration\_8h

##### Initial value:

```
=
[
 ["gaussian_quadrature", "_integration_8h.html#ab4c730eb8950bb42bc147c581693c86a", null],
 ["gaussian_quadrature2D", "_integration_8h.html#af9c631beedbb3ccd175b8029a6fa9983", null],
 ["simpsons_rule", "_integration_8h.html#a6b7d95725bdace6e2cbf5dbc10547c63", null],
 ["simpsons_rule2D", "_integration_8h.html#ae1b029fed0d7cca1c0902c042a484ca9", null],
 ["trapezoidal", "_integration_8h.html#a370d3e95d3757108e4a7cca878010cc4", null]
]
```

Definition at line 1 of file `_integration_8h.js`.

## 30.102 doc/html/\_intersection\_curve\_8h.js File Reference

### Variables

- var [\\_intersection\\_curve\\_8h](#)

## 30.102.1 Variable Documentation

### 30.102.1.1 var \_intersection\_curve\_8h

#### Initial value:

```
=
[
 ["Zero_Parameter_Span_Error", "class_go_1_1_zero__parameter__span__error.html", null],
 ["IntersectionCurve", "class_go_1_1_intersection_curve.html", "class_go_1_1_intersection_curve"],
 ["DegeneratedIntersectionCurve", "class_go_1_1_degenerated_intersection_curve.html", "
class_go_1_1_degenerated_intersection_curve"],
 ["NonEvaluableIntersectionCurve", "class_go_1_1_non_evaluable_intersection_curve.html", "
class_go_1_1_non_evaluable_intersection_curve"],
 ["IsoparametricIntersectionCurve", "class_go_1_1_isoparametric_intersection_curve.html", "
class_go_1_1_isoparametric_intersection_curve"],
 ["InterpolatedIntersectionCurve", "class_go_1_1_interpolated_intersection_curve.html", "
class_go_1_1_interpolated_intersection_curve"],
 ["EstimateDirection", "_intersection_curve_8h.html#a73ae93efa43d949b788530375bae4a09", [
 ["FORWARDS", "
_intersection_curve_8h.html#a73ae93efa43d949b788530375bae4a09adafe5a0010991f3c64bf4a3a0fa7e0bf", null],
 ["BACKWARDS", "
_intersection_curve_8h.html#a73ae93efa43d949b788530375bae4a09ab28d938a8509d39f43be6186d739198c", null]
]],
 ["EvalKind", "_intersection_curve_8h.html#a11cd92eeab3979030879414ab681ae8f", [
 ["SPACECURVE", "
_intersection_curve_8h.html#a11cd92eeab3979030879414ab681ae8fac704b2922cc7feca52f5239ec619b22", null],
 ["PARAMCURVE_1", "
_intersection_curve_8h.html#a11cd92eeab3979030879414ab681ae8fab087de473bf3a94dfe7e5517ff3fca7e", null],
 ["PARAMCURVE_2", "
_intersection_curve_8h.html#a11cd92eeab3979030879414ab681ae8fa5d0237f746518b9794e553b00c211c3d", null]
]],
 ["TangentDomain", "_intersection_curve_8h.html#a683c7cd0653c849d16af423f40a6daf6", [
 ["GEOM", "
_intersection_curve_8h.html#a683c7cd0653c849d16af423f40a6daf6a2eab3dfbfcbb69fb143144ec699e952e", null],
 ["PARAM1", "
_intersection_curve_8h.html#a683c7cd0653c849d16af423f40a6daf6a44e72e0b76476af77c20052f24d3b5be", null],
 ["PARAM2", "
_intersection_curve_8h.html#a683c7cd0653c849d16af423f40a6daf6a7c4a6ef82dbc05c929c42e8d776d0afe", null]
]],
 ["constructIntersectionCurve", "_intersection_curve_8h.html#a627a2c2616aab16215e86047f7ccf721", null]
]
```

Definition at line 1 of file \_intersection\_curve\_8h.js.

## 30.103 doc/html/\_intersection\_interface\_8h.js File Reference

### Variables

- var [\\_intersection\\_interface\\_8h](#)

### 30.103.1 Variable Documentation

#### 30.103.1.1 var \_intersection\_interface\_8h

#### Initial value:

```
=
[
 ["intersectCurves", "_intersection_interface_8h.html#a053ccbbd7119260227c1c95fd6f6f898d", null]
]
```

Definition at line 1 of file \_intersection\_interface\_8h.js.

## 30.104 doc/html/\_intersection\_point\_utils\_8h.js File Reference

### Variables

- [var \\_intersection\\_point\\_utils\\_8h](#)

### 30.104.1 Variable Documentation

#### 30.104.1.1 var \_intersection\_point\_utils\_8h

##### Initial value:

```
=
[
 ["CachedInterval", "struct_go_1_1_cached_interval.html", "struct_go_1_1_cached_interval"],
 ["IntPtInfo", "struct_go_1_1_int_pt_info.html", "struct_go_1_1_int_pt_info"],
 ["IntPtClassification", "_intersection_point_utils_8h.html#a96d472ee6c1174d35dc179b133c76bde", [
 ["DIR_UNDEF", "
 _intersection_point_utils_8h.html#a96d472ee6c1174d35dc179b133c76bdea820ac70fd33c863aebcf7a3b0bc691c9", null],
 ["DIR_IN", "
 _intersection_point_utils_8h.html#a96d472ee6c1174d35dc179b133c76bdea2b229b86fc7c49a888f4c85de0fd0023", null],
 ["DIR_OUT", "
 _intersection_point_utils_8h.html#a96d472ee6c1174d35dc179b133c76bdea867efb07d41953d6d8e3b6ee17a45772", null],
 ["DIR_PARALLEL", "
 _intersection_point_utils_8h.html#a96d472ee6c1174d35dc179b133c76bdea28b39001ab0ad9baeb5e05efd209f940", null],
 ["DIR_PERPENDICULAR", "
 _intersection_point_utils_8h.html#a96d472ee6c1174d35dc179b133c76bdea0f096a32edf421a2627ec6d6e20382e6", null],
 ["DIR_HIGHLY_SINGULAR", "
 _intersection_point_utils_8h.html#a96d472ee6c1174d35dc179b133c76bdea9aead62bfaaac96e0f966c90ef04f2c", null],
 ["DIR_TOUCH", "
 _intersection_point_utils_8h.html#a96d472ee6c1174d35dc179b133c76bdea734d14ac74f992b6632803bd13715769", null]
]],
 ["IntPtLocation", "_intersection_point_utils_8h.html#a7d692aadd79b236fb05b5b4135cd0ad", [
 ["LOC_OUTSIDE_BOTH", "
 _intersection_point_utils_8h.html#a7d692aadd79b236fb05b5b4135cd0ada91b8e290e0e2555d6ble96dfae8e4bae", null],
 ["LOC_INSIDE_BOTH", "
 _intersection_point_utils_8h.html#a7d692aadd79b236fb05b5b4135cd0ada7d4f28281cf301a763ffcfdf526f3ba2", null],
 ["LOC_EDGE_ONE", "
 _intersection_point_utils_8h.html#a7d692aadd79b236fb05b5b4135cd0ada0cc998fe7d568329a67a5af22270fae8", null],
 ["LOC_CORNER_ONE", "
 _intersection_point_utils_8h.html#a7d692aadd79b236fb05b5b4135cd0ada65dc06de763e9db4a605a22c74ab0303", null],
 ["LOC_CORNER_BOTH", "
 _intersection_point_utils_8h.html#a7d692aadd79b236fb05b5b4135cd0adad56be127936a25a10c7b50cfafaf85cc", null],
 ["LOC_EDGE_BOTH", "
 _intersection_point_utils_8h.html#a7d692aadd79b236fb05b5b4135cd0ada3db4d0a9d71b8d50fd87fdc59664a13c", null]
]]
]
```

Definition at line 1 of file `_intersection_point_utils_8h.js`.

## 30.105 doc/html/\_intersection\_pool\_utils\_8h.js File Reference

### Variables

- [var \\_intersection\\_pool\\_utils\\_8h](#)

## 30.105.1 Variable Documentation

### 30.105.1.1 var \_intersection\_pool\_utils\_8h

#### Initial value:

```
=
[
 ["raw_pointer_comp", "struct_go_1_1raw_pointer_comp.html", "struct_go_1_1raw_pointer_comp"],
 ["CrossesValue", "class_go_1_1crosses_value.html", "class_go_1_1crosses_value"],
 ["TestInDomain", "class_go_1_1_test_in_domain.html", "class_go_1_1_test_in_domain"],
 ["ClosestPointCalculator", "class_go_1_1_closest_point_calculator.html", "
 class_go_1_1_closest_point_calculator"],
 ["LockedDirDistFunc", "class_go_1_1_locked_dir_dist_func.html", "class_go_1_1_locked_dir_dist_func"],
 ["ConnectionFunc", "class_go_1_1_connection_func.html", "class_go_1_1_connection_func"],
 ["add_reachables_from", "_intersection_pool_utils_8h.html#a1c2a7b126148ae101e23c71206465583", null],
 ["compare_first", "_intersection_pool_utils_8h.html#aed15b783b1b9e9bf85d8deb556769b12", null],
 ["debug_write_line", "_intersection_pool_utils_8h.html#a4ee59f17c50de9de31d0c0633edbead0", null],
 ["debug_write_point", "_intersection_pool_utils_8h.html#a983ea40b1fe9434be55eae4604fb78fa", null],
 ["determine_seed", "_intersection_pool_utils_8h.html#adc66af32945c78547a7c1ccfe32b30e", null],
 ["estimate_seed_by_interpolation", "_intersection_pool_utils_8h.html#a06aa59135575f74e179b54b2858ecce8",
 null],
 ["extract_chains", "_intersection_pool_utils_8h.html#af507107da1913d723fbc4d0644e05bcf", null],
 ["find_point_in", "_intersection_pool_utils_8h.html#ab8ca76589d4df7db161ea9b2b35fb9aa", null],
 ["flip_intersecting_objects", "_intersection_pool_utils_8h.html#a4ea368ab0db3908730d7f33280e06979",
 null],
 ["link_is_iso", "_intersection_pool_utils_8h.html#ab4676c710e9cff8f203774226e04d62c", null],
 ["link_is_iso_in", "_intersection_pool_utils_8h.html#a8c71cefd2478e57738916762cfe04f3", null],
 ["link_is_iso_in_other_than", "_intersection_pool_utils_8h.html#afe19a667acd7c32f6db8f89e52383132",
 null],
 ["make_curve_on_surface", "_intersection_pool_utils_8h.html#ae90d3ebafba8423b99cfc59cd2a6db8b", null],
 ["no_parent", "_intersection_pool_utils_8h.html#a660520ff0c2a9c2d6a864bddcc33902e", null]
]
```

Definition at line 1 of file `_intersection_pool_utils_8h.js`.

## 30.106 doc/html/\_intersection\_utils\_8h.js File Reference

### Variables

- [var \\_intersection\\_utils\\_8h](#)

## 30.106.1 Variable Documentation

### 30.106.1.1 var \_intersection\_utils\_8h

#### Initial value:

```
=
[
 ["create1DSplineCurve", "_intersection_utils_8h.html#a72ee921f021fd4690b379cae8bb7478a", null],
 ["create1DSplineSurface", "_intersection_utils_8h.html#a9e358b78add198705d85a29d66d57334", null],
 ["distImplRepresentationCompFunction", "_intersection_utils_8h.html#a24ea8ac591f45d41f4f2282b3bf0fcbcd",
 null],
 ["insertCvInAlgc", "_intersection_utils_8h.html#af3f3b40f4a0e288b1fe629c4f1847071", null],
 ["insertSfInAlgsf", "_intersection_utils_8h.html#a245382b2c052cd3413fdb7faff70259a", null],
 ["insertSfInAlgsf2", "_intersection_utils_8h.html#a7c62b2c4f7331ca7d279d62124704334", null],
 ["insertSfInImplObj", "_intersection_utils_8h.html#ad00de047b913ad413503340696b85985", null],
 ["splineCurveProduct", "_intersection_utils_8h.html#a6efa5b96b45c2db8fb49c36641323a49", null],
 ["splineSurfaceProduct", "_intersection_utils_8h.html#a917c5516673720296cbf9a7e46494983", null]
]
```

Definition at line 1 of file `_intersection_utils_8h.js`.

## 30.107 doc/html/\_l\_r\_approx\_app\_8h.js File Reference

### Variables

- var [\\_l\\_r\\_approx\\_app\\_8h](#)

### 30.107.1 Variable Documentation

#### 30.107.1.1 var \_l\_r\_approx\_app\_8h

#### Initial value:

```
=
[
 ["categorizeCloudFromDist", "_l_r_approx_app_8h.html#a3b71548b7d81a481c49c82ca532893c0", null],
 ["categorizeCloudFromDist_omp", "_l_r_approx_app_8h.html#ae38f52047f571473587231c3fa039ccc", null],
 ["classifyCloudFromDist", "_l_r_approx_app_8h.html#a08d13cf5029d545c3fd3430617fc602a", null],
 ["classifyCloudFromDist_omp", "_l_r_approx_app_8h.html#a48badf20be4b88af31bcf8e400350e43", null],
 ["computeDistPointSpline", "_l_r_approx_app_8h.html#ab1842cd630d4f77d1495159c9a38ab62", null],
 ["computeDistPointSpline_omp", "_l_r_approx_app_8h.html#ad363e9bc8c6ea04d20503d6d488010f7", null],
 ["pointCloud2Spline", "_l_r_approx_app_8h.html#ae174c4352de0a200875762dd350b6973", null]
]
```

Definition at line 1 of file `_l_r_approx_app_8h.js`.

## 30.108 doc/html/\_l\_r\_b\_spline2\_d\_8h.js File Reference

### Variables

- var [\\_l\\_r\\_b\\_spline2\\_d\\_8h](#)

### 30.108.1 Variable Documentation

#### 30.108.1.1 var \_l\_r\_b\_spline2\_d\_8h

#### Initial value:

```
=
[
 ["LRBSpline2D", "class_go_1_1_l_r_b_spline2_d.html", "class_go_1_1_l_r_b_spline2_d"],
 ["operator<<", "_l_r_b_spline2_d_8h.html#a6e7d6c2c402da2b60203f8cc4e44758d", null],
 ["operator>>", "_l_r_b_spline2_d_8h.html#ade1cc2538c3d77f4806b29b5c8c1d4a5", null]
]
```

Definition at line 1 of file `_l_r_b_spline2_d_8h.js`.

## 30.109 doc/html/\_l\_r\_b\_spline2\_d\_utils\_8h.js File Reference

### Variables

- var [\\_l\\_r\\_b\\_spline2\\_d\\_utils\\_8h](#)

### 30.109.1 Variable Documentation

#### 30.109.1.1 var \_l\_r\_b\_spline2\_d\_utils\_8h

**Initial value:**

```
=
[
 ["derive_knots", "_l_r_b_spline2_d_utils_8h.html#aad3f20e5e947ef8bbf0874462afd783", null],
 ["split_function", "_l_r_b_spline2_d_utils_8h.html#a9743e75fcc6fd16a8b87ae72df567fee", null],
 ["split_several", "_l_r_b_spline2_d_utils_8h.html#a4f0db6999c8dd9b90c306a83d081b409", null],
 ["try_split_once", "_l_r_b_spline2_d_utils_8h.html#aff31a580cd1acee9f5725b42e1bed14c", null]
]
```

Definition at line 1 of file `_l_r_b_spline2_d_utils_8h.js`.

## 30.110 doc/html/\_l\_r\_benchmark\_utils\_8h.js File Reference

### Variables

- var [\\_l\\_r\\_benchmark\\_utils\\_8h](#)

### 30.110.1 Variable Documentation

#### 30.110.1.1 var \_l\_r\_benchmark\_utils\_8h

**Initial value:**

```
=
[
 ["benchmarkSfRefinement", "_l_r_benchmark_utils_8h.html#a4dc3c0bcb0187ac1fab862cb1a2d8bd8", null]
]
```

Definition at line 1 of file `_l_r_benchmark_utils_8h.js`.

## 30.111 doc/html/\_l\_r\_spline\_eval\_grid\_8h.js File Reference

### Variables

- var [\\_l\\_r\\_spline\\_eval\\_grid\\_8h](#)

### 30.111.1 Variable Documentation

#### 30.111.1.1 var \_l\_r\_spline\_eval\_grid\_8h

**Initial value:**

```
=
[
 ["LRSplineEvalGrid", "class_go_1_1_l_r_spline_eval_grid.html", "class_go_1_1_l_r_spline_eval_grid"],
 ["_LRSPLINEEVAVGGRID_H", "_l_r_spline_eval_grid_8h.html#a861377e9079b564d42abe23579c6992b", null],
 ["simpleElement", "_l_r_spline_eval_grid_8h.html#a046535108edecaad94e9960634734291", null]
]
```

Definition at line 1 of file `_l_r_spline_eval_grid_8h.js`.

## 30.112 doc/html/\_l\_r\_spline\_m\_b\_a\_8h.js File Reference

### Variables

- var [\\_l\\_r\\_spline\\_m\\_b\\_a\\_8h](#)

### 30.112.1 Variable Documentation

#### 30.112.1.1 var \_l\_r\_spline\_m\_b\_a\_8h

##### Initial value:

```
=
[
 ["add_contribution", "_l_r_spline_m_b_a_8h.html#a7746ce5f0e960417c31cbe369951206d", null],
 ["add_contribution2", "_l_r_spline_m_b_a_8h.html#a64d38b9b4499f43124da69e5459b51b2", null],
 ["MBADistAndUpdate", "_l_r_spline_m_b_a_8h.html#a736bab9513cf83f3b0c1143b8b908ee2", null],
 ["MBADistAndUpdate_omp", "_l_r_spline_m_b_a_8h.html#afb4e83788879084765598ed037e3645a", null],
 ["MBAUpdate", "_l_r_spline_m_b_a_8h.html#ac8d9dfeb51ced6380e5cda7fa0de4244", null],
 ["MBAUpdate", "_l_r_spline_m_b_a_8h.html#a08e09eba8aede0ca573ff517431bae02", null],
 ["MBAUpdate_omp", "_l_r_spline_m_b_a_8h.html#af434dc2388f448370a7f5c0b7a7d2032", null]
]
```

Definition at line 1 of file `_l_r_spline_m_b_a_8h.js`.

## 30.113 doc/html/\_l\_r\_spline\_plot\_utils\_8h.js File Reference

### Variables

- var [\\_l\\_r\\_spline\\_plot\\_utils\\_8h](#)

### 30.113.1 Variable Documentation

#### 30.113.1.1 var \_l\_r\_spline\_plot\_utils\_8h

##### Initial value:

```
=
[
 ["writePostscriptMesh", "_l_r_spline_plot_utils_8h.html#af1c6e27bb10c5d76b4ec155ab8676270", null]
]
```

Definition at line 1 of file `_l_r_spline_plot_utils_8h.js`.

## 30.114 doc/html/\_l\_r\_spline\_utils\_8h.js File Reference

### Variables

- var [\\_l\\_r\\_spline\\_utils\\_8h](#)



## 30.114.1 Variable Documentation

### 30.114.1.1 var \_l\_r\_spline\_utils\_8h

#### Initial value:

```
=
[
 ["support_compare", "struct_go_l_l_l_r_spline_utils_l_lsupport__compare.html", "
 struct_go_l_l_l_r_spline_utils_l_lsupport__compare"],
 ["all_meshlines_uniform", "_l_r_spline_utils_8h.html#a07152a994fb9e3ae035eccabb9805c7e", null],
 ["compute_alpha", "_l_r_spline_utils_8h.html#a228ca5293bc3a0bae51d2b7953a49035", null],
 ["compute_greville", "_l_r_spline_utils_8h.html#a8532bcd90e52ae33172b90c3c300c0a4", null],
 ["compute_greville", "_l_r_spline_utils_8h.html#a2f34316e93c29d331be4c0183c923d55", null],
 ["distributeDataPoints", "_l_r_spline_utils_8h.html#a6229f7ced8a32103f1ec4dbb22bcef81", null],
 ["elementLineClouds", "_l_r_spline_utils_8h.html#a6e2646ca71a91443d9c22be20d888fca", null],
 ["elementOK", "_l_r_spline_utils_8h.html#a5b786578b3a0884a647accd13f42c8c5", null],
 ["fullTensorProductSurface", "_l_r_spline_utils_8h.html#af51b22a5cf3d0d58e94ff7df2ad08a04", null],
 ["identify_elements_from_mesh", "_l_r_spline_utils_8h.html#a0df66adf1511debd6e5042d890989278", null],
 ["increment_knotvec_indices", "_l_r_spline_utils_8h.html#adcc7711bae33c4f0667d81ae588a0074", null],
 ["insert_basis_function", "_l_r_spline_utils_8h.html#ae7dfe0e2e77cf611c53f0a79f17fea8d", null],
 ["insert_knots", "_l_r_spline_utils_8h.html#a9d8e64a570cb768edb77aaa84e09f234", null],
 ["insertParameterFunctions", "_l_r_spline_utils_8h.html#a61429b0a4d1e40fdc60284308aldc744", null],
 ["iteratively_split", "_l_r_spline_utils_8h.html#a33904736dc5f4c872c7b44395508c556", null],
 ["iteratively_split2", "_l_r_spline_utils_8h.html#adb184c6cdc35d1c98f115ab272caa6d9", null],
 ["knots_to_insert", "_l_r_spline_utils_8h.html#a4f98f5746feb419caf049335e2b1eda7", null],
 ["locate_interval", "_l_r_spline_utils_8h.html#a64d8cc60b5fdafd020a79f9311b1cb80", null],
 ["mostComparableBspline", "_l_r_spline_utils_8h.html#a491ae8a5130f39b70f87060471f2e402", null],
 ["refine_mesh", "_l_r_spline_utils_8h.html#abcbcc13651a9f0e9994af4d59f1c7f98", null],
 ["set_uniform_meshlines", "_l_r_spline_utils_8h.html#af5002165fefef51c705e3e4c3bd02975", null],
 ["support_equal", "_l_r_spline_utils_8h.html#aaa14a37f51bb036b0f06f0d0ba85d47f", null],
 ["tensor_split", "_l_r_spline_utils_8h.html#a871bce06468326aff1c18d91d0ae5d31", null],
 ["update_elements_with_single_bspline", "_l_r_spline_utils_8h.html#a8a31c141bc46e6ca969a556ae80abdc",
 null]
]
```

Definition at line 1 of file `_l_r_spline_utils_8h.js`.

## 30.115 doc/html/\_l\_r\_surf\_smooth\_l\_s\_8h.js File Reference

### Variables

- var `_l_r_surf_smooth_l_s_8h`

## 30.115.1 Variable Documentation

### 30.115.1.1 var \_l\_r\_surf\_smooth\_l\_s\_8h

#### Initial value:

```
=
[
 ["LRSurfSmoothLS", "class_go_l_l_l_r_surf_smooth_l_s.html", "class_go_l_l_l_r_surf_smooth_l_s"],
 ["BsplineIndexMap", "_l_r_surf_smooth_l_s_8h.html#a4b15ecc9104dfd8fd5bc9cd2a0e525f1", null]
]
```

Definition at line 1 of file `_l_r_surf_smooth_l_s_8h.js`.

## 30.116 doc/html/\_l\_r\_surf\_stitch\_8h.js File Reference

### Variables

- var [\\_l\\_r\\_surf\\_stitch\\_8h](#)

### 30.116.1 Variable Documentation

#### 30.116.1.1 var \_l\_r\_surf\_stitch\_8h

##### Initial value:

```
=
[
 ["averageCorner", "_l_r_surf_stitch_8h.html#afbc9bf8c16995c9cef03b90fbd5999aa", null],
 ["averageEdge", "_l_r_surf_stitch_8h.html#afb9fbc8819dfd87d85203a9f458ed43e", null],
 ["averageEdge", "_l_r_surf_stitch_8h.html#a4d41fc38a785302cb538150369cd6173", null],
 ["defineRefinements", "_l_r_surf_stitch_8h.html#a33e9db4b176dbaf56d9f31830c206a1c", null],
 ["extractBoundaryBsplines", "_l_r_surf_stitch_8h.html#a3509ee922d6de21fdd642ce066d4a024", null],
 ["extractMissingKnots", "_l_r_surf_stitch_8h.html#a823159db9ecb7a6a0a50ab76bad1fdad", null],
 ["fetchEdgeCorners", "_l_r_surf_stitch_8h.html#a448922c6cf420b46b0e5008d85734979", null]
]
```

Definition at line 1 of file [\\_l\\_r\\_surf\\_stitch\\_8h.js](#).

## 30.117 doc/html/\_l\_u\_decomp\_8h.js File Reference

### Variables

- var [\\_l\\_u\\_decomp\\_8h](#)

### 30.117.1 Variable Documentation

#### 30.117.1.1 var \_l\_u\_decomp\_8h

##### Initial value:

```
=
[
 ["backwardSubstitution", "_l_u_decomp_8h.html#a4ba62414926e61f2b132c414e2379013", null],
 ["backwardSubstitution", "_l_u_decomp_8h.html#a93246224796027668dd33bdfaef663c3", null],
 ["forwardSubstitution", "_l_u_decomp_8h.html#a5c8cea4b85360943935b959f1ad2a0b2", null],
 ["forwardSubstitution", "_l_u_decomp_8h.html#aebc4d929d2c713f7d25009045116bf33", null],
 ["LUDecomp", "_l_u_decomp_8h.html#a77c8de2bdc12a47b62ef5c306ac0e46d", null],
 ["LUsolveSystem", "_l_u_decomp_8h.html#afd9ea9a56159c9a5841819ebf13280b8", null]
]
```

Definition at line 1 of file [\\_l\\_u\\_decomp\\_8h.js](#).

## 30.118 doc/html/\_l\_u\_decomp\_\_implementation\_8h.js File Reference

### Variables

- [var \\_l\\_u\\_decomp\\_\\_implementation\\_8h](#)

### 30.118.1 Variable Documentation

#### 30.118.1.1 var \_l\_u\_decomp\_\_implementation\_8h

##### Initial value:

```
=
[
 ["backwardSubstitution", "_l_u_decomp__implementation_8h.html#a4ba62414926e61f2b132c414e2379013", null
],
 ["backwardSubstitution", "_l_u_decomp__implementation_8h.html#a93246224796027668dd33bdfaef663c3", null
],
 ["forwardSubstitution", "_l_u_decomp__implementation_8h.html#a5c8cea4b85360943935b959f1ad2a0b2", null
],
 ["forwardSubstitution", "_l_u_decomp__implementation_8h.html#aebc4d929d2c713f7d25009045116bf33", null
],
 ["LUDecomp", "_l_u_decomp__implementation_8h.html#a77c8de2bdc12a47b62ef5c306ac0e46d", null],
 ["LUSolveSystem", "_l_u_decomp__implementation_8h.html#afd9ea9a56159c9a5841819ebf13280b8", null]
]
```

Definition at line 1 of file `_l_u_decomp__implementation_8h.js`.

## 30.119 doc/html/\_lin\_dep\_utils\_8h.js File Reference

### Variables

- [var \\_lin\\_dep\\_utils\\_8h](#)

### 30.119.1 Variable Documentation

#### 30.119.1.1 var \_lin\_dep\_utils\_8h

##### Initial value:

```
=
[
 ["isPeelable", "_lin_dep_utils_8h.html#a9bb1ad93f4a1f5b2bc48d5e1288af238", null],
 ["unpeelableBasisFunctions", "_lin_dep_utils_8h.html#ad22f2f47c7074aa2f6471aaf157dbfac", null]
]
```

Definition at line 1 of file `_lin_dep_utils_8h.js`.

## 30.120 doc/html/\_link\_type\_8h.js File Reference

### Variables

- var [\\_link\\_type\\_8h](#)

### 30.120.1 Variable Documentation

#### 30.120.1.1 var \_link\_type\_8h

Definition at line 1 of file `_link_type_8h.js`.

## 30.121 doc/html/\_loft\_surface\_creator\_8h.js File Reference

### Variables

- var [\\_loft\\_surface\\_creator\\_8h](#)

### 30.121.1 Variable Documentation

#### 30.121.1.1 var \_loft\_surface\_creator\_8h

#### Initial value:

```
=
[
 ["loftNonrationalSurface", "_loft_surface_creator_8h.html#a7e7b440ee8deaae5b11b2454ed8f393c", null],
 ["loftRationalSurface", "_loft_surface_creator_8h.html#a218592ed0e749089fe755dfb3e10467a", null],
 ["loftSurface", "_loft_surface_creator_8h.html#a93373c803ae43537fda8932169fa7b61", null],
 ["loftSurface", "_loft_surface_creator_8h.html#aebcf1dd13d08c8ed0341a21b3c7756a9", null],
 ["loftSurface", "_loft_surface_creator_8h.html#a9df745ae35b798c2bb5d7a22d4af9bd9", null],
 ["loftSurfaceFromUnifiedCurves", "_loft_surface_creator_8h.html#a2c955837128d789b492833df74848a98",
 null],
 ["makeLoftParams", "_loft_surface_creator_8h.html#a5631ec03801ac3e5a52be64db868656f", null],
 ["unifiedCurvesCopy", "_loft_surface_creator_8h.html#a303f3544a76e2144b0d0987685b69b44", null]
]
```

Definition at line 1 of file `_loft_surface_creator_8h.js`.

## 30.122 doc/html/\_loft\_volume\_creator\_8h.js File Reference

### Variables

- var [\\_loft\\_volume\\_creator\\_8h](#)

### 30.122.1 Variable Documentation

#### 30.122.1.1 var\_loft\_volume\_creator\_8h

**Initial value:**

```
=
[
 ["loftNonrationalVolume", "_loft_volume_creator_8h.html#a34f3b27a135302c551ec19c45a3901f0", null],
 ["loftRationalVolume", "_loft_volume_creator_8h.html#a3ee08ba99531b1ae4728705e095fff4d", null],
 ["loftVolume", "_loft_volume_creator_8h.html#aa9817ce767b96df73b86878cafe9932b", null],
 ["loftVolume", "_loft_volume_creator_8h.html#adde5d396516a8767313e5bf62e550bde", null],
 ["loftVolumeFromUnifiedSurfaces", "_loft_volume_creator_8h.html#acd616a5bc92b035d68e779b7ea75e77f",
 null],
 ["makeLoftParams", "_loft_volume_creator_8h.html#a03918bf5487c862b3da02fe4bac3fe65", null],
 ["unifiedSurfacesCopy", "_loft_volume_creator_8h.html#ad7376df24386f9bb6190d0c6e45e395b", null]
]
```

Definition at line 1 of file \_loft\_volume\_creator\_8h.js.

## 30.123 doc/html/\_loop\_utils\_8h.js File Reference

**Variables**

- var [\\_loop\\_utils\\_8h](#)

### 30.123.1 Variable Documentation

#### 30.123.1.1 var\_loop\_utils\_8h

**Initial value:**

```
=
[
 ["firstLoopInsideSecond", "_loop_utils_8h.html#a73a5c859c00aab6b90a93157513e85ff", null],
 ["loopIsCCW", "_loop_utils_8h.html#aafeccd2cd443da8bdc169c2ff9c6ealb", null],
 ["loopIsCCW", "_loop_utils_8h.html#aac158dbe7059be8ad43ba7c24cc1bb1b", null],
 ["makeLoopCCW", "_loop_utils_8h.html#a5fld7e6ed80391391cf4a36af089c7c5", null],
 ["paramIsCCW", "_loop_utils_8h.html#a969af355fbf48c6d254a34f3cc6f7f3e", null],
 ["representAsSurfaceCurves", "_loop_utils_8h.html#a366b3c54d773b4f9daf1ab9a8b17e5b6", null]
]
```

Definition at line 1 of file \_loop\_utils\_8h.js.

## 30.124 doc/html/\_matrix\_x\_d\_8h.js File Reference

**Variables**

- var [\\_matrix\\_x\\_d\\_8h](#)

### 30.124.1 Variable Documentation

#### 30.124.1.1 var \_matrix\_x\_d\_8h

##### Initial value:

```
=
[
 ["MatrixXD", "class_go_1_1_matrix_x_d.html", "class_go_1_1_matrix_x_d"],
 ["operator<<", "_matrix_x_d_8h.html#a25d3656c853c19fe5fce21df614e373e", null]
]
```

Definition at line 1 of file \_matrix\_x\_d\_8h.js.

### 30.125 doc/html/\_mesh2\_d\_8h.js File Reference

#### Variables

- var [\\_mesh2\\_d\\_8h](#)

### 30.125.1 Variable Documentation

#### 30.125.1.1 var \_mesh2\_d\_8h

##### Initial value:

```
=
[
 ["GPos", "struct_go_1_1_g_pos.html", "struct_go_1_1_g_pos"],
 ["Mesh2D", "class_go_1_1_mesh2_d.html", "class_go_1_1_mesh2_d"],
 ["flip", "_mesh2_d_8h.html#abe2f7726ba7cf956959bc53d26ee25e6", null],
 ["operator<<", "_mesh2_d_8h.html#a9e098e4d619a15a57b7c68d10b03b649", null],
 ["operator<<", "_mesh2_d_8h.html#a631596c966521e4f4598e51fd68fbb0d", null],
 ["operator>>", "_mesh2_d_8h.html#a40434f0c0180c0391fb6d93a157b3630", null],
 ["operator>>", "_mesh2_d_8h.html#a1a6ab68e4a1a08677d6759e93b7af7a1", null]
]
```

Definition at line 1 of file \_mesh2\_d\_8h.js.

### 30.126 doc/html/\_mesh2\_d\_utils\_8h.js File Reference

#### Variables

- var [\\_mesh2\\_d\\_utils\\_8h](#)

### 30.126.1 Variable Documentation

#### 30.126.1.1 var\_mesh2\_d\_utils\_8h

##### Initial value:

```
=
[
 ["first_larger_knotvalue_ix", "_mesh2_d_utils_8h.html#a1461b9e821e4e653a1863fbb11ff8183", null],
 ["identify_patch_lower_left", "_mesh2_d_utils_8h.html#a70ff355fcf2aa04dabab17c8387ce6d3", null],
 ["identify_patch_upper_right", "_mesh2_d_utils_8h.html#a118220e0b56c3f07a13dc8ba42a393cb", null],
 ["last_nonlarger_knotvalue_ix", "_mesh2_d_utils_8h.html#a8a567523d54d76c3a532ce9a9749f58e", null],
 ["search_downwards_for_nonzero_multiplicity", "
_mesh2_d_utils_8h.html#a568514426907f3a343f44a6c4e71909b", null],
 ["search_downwards_for_nonzero_multiplicity", "
_mesh2_d_utils_8h.html#a7493fce8d41b25d29e13a60a1a76b4c8", null],
 ["search_upwards_for_nonzero_multiplicity", "_mesh2_d_utils_8h.html#ae42a4542f261f229730e4748a381acab"
, null],
 ["search_upwards_for_nonzero_multiplicity", "_mesh2_d_utils_8h.html#addf7d8865462ca277c8952be864aac73"
, null]
]
```

Definition at line 1 of file \_mesh2\_d\_utils\_8h.js.

## 30.127 doc/html/\_modify\_surf\_8h.js File Reference

### Variables

- [var\\_modify\\_surf\\_8h](#)

### 30.127.1 Variable Documentation

#### 30.127.1.1 var\_modify\_surf\_8h

##### Initial value:

```
=
[
 ["enforceCoefCoLinearity", "_modify_surf_8h.html#a137bcf3d2c27c7d2b389f6e64a5d69de", null],
 ["enforceVxCoefCoLinearity", "_modify_surf_8h.html#a3913d84da475ce0939ad502c7a6ae95a", null],
 ["replaceBoundary", "_modify_surf_8h.html#aeacca79de366536b18da66a6029feac6", null]
]
```

Definition at line 1 of file \_modify\_surf\_8h.js.

## 30.128 doc/html/\_offset\_surface\_utils\_8h.js File Reference

### Variables

- [var\\_offset\\_surface\\_utils\\_8h](#)

### 30.128.1 Variable Documentation

#### 30.128.1.1 var\_offset\_surface\_utils\_8h

##### Initial value:

```
=
[
 ["OffsetSurfaceStatus", "_offset_surface_utils_8h.html#a49c34a8a952371206ac5113543270589", [
 ["OFFSET_OK", "
 _offset_surface_utils_8h.html#a49c34a8a952371206ac5113543270589addc758c86d9a578e0dcb248819d592af", null],
 ["OFFSET_FAILED", "
 _offset_surface_utils_8h.html#a49c34a8a952371206ac5113543270589adc3d327e65f0c46656bfc9e175fcc151", null],
 ["SELF_INTERSECTING_INTERIOR", "
 _offset_surface_utils_8h.html#a49c34a8a952371206ac5113543270589a26493d54b0525be972c6alb180e4c674", null],
 ["SELF_INTERSECTING_BOUNDARY", "
 _offset_surface_utils_8h.html#a49c34a8a952371206ac5113543270589a2b8e889d4d696085c18036c41a5feee8", null],
 ["TOLERANCE_ERROR", "
 _offset_surface_utils_8h.html#a49c34a8a952371206ac5113543270589a89c7beb3296ffda86f726e61362f7c9c", null],
 ["NOT_FOUR_CORNERS", "
 _offset_surface_utils_8h.html#a49c34a8a952371206ac5113543270589a1db463f259d94e558afdf3af00a006af", null],
 ["NON_ISO_KINK_CURVE", "
 _offset_surface_utils_8h.html#a49c34a8a952371206ac5113543270589a8f44525c3493edeeef93a3948eabe7bee", null]
]],
 ["offsetSurfaceSet", "_offset_surface_utils_8h.html#a3558e89477dcc6ad2d2df5bc15722492", null]
]
```

Definition at line 1 of file `_offset_surface_utils_8h.js`.

## 30.129 doc/html/\_param\_surface\_8h.js File Reference

### Variables

- var [\\_param\\_surface\\_8h](#)

### 30.129.1 Variable Documentation

#### 30.129.1.1 var\_param\_surface\_8h

##### Initial value:

```
=
[
 ["ParamSurface", "class_go_1_1_param_surface.html", "class_go_1_1_param_surface"],
 ["degenerate_info", "struct_go_1_1_param_surface_1_1degenerate__info.html", "
 struct_go_1_1_param_surface_1_1degenerate__info"],
 ["IteratorType", "_param_surface_8h.html#a241482b98287c4f940c75ef70671e05c", [
 ["Iterator_parametric", "
 _param_surface_8h.html#a241482b98287c4f940c75ef70671e05cac62977f430391a4f0a8643ae07a5daf4", null],
 ["Iterator_geometric", "
 _param_surface_8h.html#a241482b98287c4f940c75ef70671e05cacd4a1ab60b6688a79afb6336b99c936e", null]
]]
]
```

Definition at line 1 of file `_param_surface_8h.js`.



## 30.130 doc/html/\_path\_8h.js File Reference

### Variables

- [var \\_path\\_8h](#)

#### 30.130.1 Variable Documentation

##### 30.130.1.1 var \_path\_8h

###### Initial value:

```
=
[
 ["classifyCorners", "_path_8h.html#a31a5b4b8ed77447803c7c0c1bb32f9ed", null],
 ["closestPoint", "_path_8h.html#a15c625456455d04a4fabe5d1181712ac", null],
 ["edgeChain", "_path_8h.html#aa64387b9b36ec80612d05a560428a4ec", null],
 ["estimateHoleInfo", "_path_8h.html#af1ad3c3df0c9672c5686b3547ad95f7c", null],
 ["getEdgeCurves", "_path_8h.html#a8458aecd1ab5d8362a7137e2ea2280f54", null],
 ["identifyLoop", "_path_8h.html#a12301ef59c50bee8128a06243290bdfb", null]
]
```

Definition at line 1 of file \_path\_8h.js.

## 30.131 doc/html/\_point\_8h.js File Reference

### Variables

- [var \\_point\\_8h](#)

#### 30.131.1 Variable Documentation

##### 30.131.1.1 var \_point\_8h

###### Initial value:

```
=
[
 ["Point", "class_go_1_1_point.html", "class_go_1_1_point"],
 ["operator*", "_point_8h.html#a641dacf21690f2dd5a46a496b9dc0f8c", null],
 ["operator<<", "_point_8h.html#a66d59df4e32729691185f369f0b5a393", null],
 ["operator<<<", "_point_8h.html#af96ef1e897f0ac52ed4773621c805862", null],
 ["operator==", "_point_8h.html#a1a0e7dde1c83f7b616816d78ac11f3a", null],
 ["operator>>", "_point_8h.html#ad51bfc00ff13f4956bb91947e21ace00", null]
]
```

Definition at line 1 of file \_point\_8h.js.

## 30.132 doc/html/\_point\_cloud\_8h.js File Reference

### Variables

- [var \\_point\\_cloud\\_8h](#)

### 30.132.1 Variable Documentation

#### 30.132.1.1 var \_point\_cloud\_8h

##### Initial value:

```
=
[
 ["PointCloud", "class_go_1_1_point_cloud.html", "class_go_1_1_point_cloud"],
 ["PointCloud3D", "_point_cloud_8h.html#aca0adb8106a4ffc358eeffeb11ef851", null],
 ["PointCloud4D", "_point_cloud_8h.html#ac0a110c4920004f774150b3e9ccbfe64", null]
]
```

Definition at line 1 of file `_point_cloud_8h.js`.

## 30.133 doc/html/\_point\_sequence\_8h.js File Reference

### Variables

- [var \\_point\\_sequence\\_8h](#)

### 30.133.1 Variable Documentation

#### 30.133.1.1 var \_point\_sequence\_8h

##### Initial value:

```
=
[
 ["PointSequence", "class_go_1_1_point_sequence.html", "class_go_1_1_point_sequence"],
 ["PointSequenceType", "_point_sequence_8h.html#a8fd32afd3a4d4892e4020b0580de70e1", [
 ["PSTPoint", "_point_sequence_8h.html#a8fd32afd3a4d4892e4020b0580de70e1a4eea7b565319ed23e1e46c7b519bcfe2", null],
 ["PSTPointTangent", "_point_sequence_8h.html#a8fd32afd3a4d4892e4020b0580de70e1a44f2da44ddde83fedb06c67050ddc6e4", null],
 ["PSTScattered", "_point_sequence_8h.html#a8fd32afd3a4d4892e4020b0580de70e1ae7ab6c2a6415887b29d8152b39a92f92", null]
]]
]
```

Definition at line 1 of file `_point_sequence_8h.js`.

## 30.134 doc/html/\_point\_set\_app\_8h.js File Reference

### Variables

- var [\\_point\\_set\\_app\\_8h](#)

#### 30.134.1 Variable Documentation

##### 30.134.1.1 var \_point\_set\_app\_8h

###### Initial value:

```
=
[
 ["parameterizeTriang", "_point_set_app_8h.html#a6f49e135ca05e5513ab3eba8657ef11c", null],
 ["recognizeBoundary", "_point_set_app_8h.html#ae0f70c8f26bce38c4c89331df6b42b", null],
 ["recognizeCornerNodes", "_point_set_app_8h.html#a90a3b8a6f1f566bfa0cd02a5d0c6b669", null]
]
```

Definition at line 1 of file `_point_set_app_8h.js`.

## 30.135 doc/html/\_pr\_dijkstra\_8h.js File Reference

### Variables

- var [\\_pr\\_dijkstra\\_8h](#)

#### 30.135.1 Variable Documentation

##### 30.135.1.1 var \_pr\_dijkstra\_8h

###### Initial value:

```
=
[
 ["HeapNode", "class_heap_node.html", "class_heap_node"],
 ["Dijkstra", "class_dijkstra.html", "class_dijkstra"],
 ["GraphType", "_pr_dijkstra_8h.html#a206245534b423169d06a05a296e3a169", null],
 ["HeapType", "_pr_dijkstra_8h.html#afb8a42e8b8299d2995279be8fdaa601d", null]
]
```

Definition at line 1 of file `_pr_dijkstra_8h.js`.

## 30.136 doc/html/\_pr\_geodesics\_8h.js File Reference

### Variables

- var [\\_pr\\_geodesics\\_8h](#)

### 30.136.1 Variable Documentation

#### 30.136.1.1 var \_pr\_geodesics\_8h

##### Initial value:

```
=
[
 ["EPS", "_pr_geodesics_8h.html#a6ebf6899d6c1c8b7b9d09be872c05aae", null],
 ["decreasing_length", "_pr_geodesics_8h.html#ad90da015119c685931562203c8200235", null],
 ["DijkstraPT", "_pr_geodesics_8h.html#a9de5f006c7cd1bbfd2ea6311ea4661b0", null],
 ["dist2D", "_pr_geodesics_8h.html#ac7a9fdb0b3584a5707d06f653b651ab2", null],
 ["dist3D", "_pr_geodesics_8h.html#a502010751565dfcadf85ed74bd03519f", null],
 ["GeodesicPT", "_pr_geodesics_8h.html#a6433488647d99aa6adb0b4bc31a47d9e", null],
 ["length_polygonal_path", "_pr_geodesics_8h.html#ab184bcc756b99d5c32bf480c0aea576d", null],
 ["next_pivot", "_pr_geodesics_8h.html#a6ede54ba38a09b1d4b49a554e67dc011", null],
 ["pivot_in_new_list", "_pr_geodesics_8h.html#a981247420304886270592d3d91486f99", null],
 ["pivot_in_new_list", "_pr_geodesics_8h.html#a23201cc2fec144cd1354936bdad5c66a", null],
 ["point3D_from_indice", "_pr_geodesics_8h.html#aeb7addad1f7db7745aea8aa2aa9c90b", null],
 ["sqr", "_pr_geodesics_8h.html#a22f4e97ed7ca9e188dc4e074a1b2e5a7", null],
 ["tr_crossed_by_path_vertices", "_pr_geodesics_8h.html#a7e37f49d81d4457bf2d7a377a632cbe7", null],
 ["tr_sequence_around_vertex", "_pr_geodesics_8h.html#ab2fecc2891044c9570fdeacc7a90857a", null],
 ["tr_sequence_around_vertex", "_pr_geodesics_8h.html#a75863d5fc529fb0973bcb4038d4bb313", null],
 ["update_triangle_sequence_around_pivot", "_pr_geodesics_8h.html#adb6c85b9ff49747043f7aa470c39abbc",
 null]
]
```

Definition at line 1 of file `_pr_geodesics_8h.js`.

## 30.137 doc/html/\_pr\_interface\_8h.js File Reference

### Variables

- var [\\_pr\\_interface\\_8h](#)

### 30.137.1 Variable Documentation

#### 30.137.1.1 var \_pr\_interface\_8h

##### Initial value:

```
=
[
 ["print", "_pr_interface_8h.html#a702a39c3e329cb93bb7a836a3eaf7067", null],
 ["print", "_pr_interface_8h.html#a9a7a38c0f5e4dfe4319360a310a86586", null],
 ["print", "_pr_interface_8h.html#ab0c8b3ddf235719dfb256c12066f013d", null],
 ["print", "_pr_interface_8h.html#a8484caf5c67beb9ab23948df6a1449a9", null],
 ["print", "_pr_interface_8h.html#a85c471224a04b8237c9d5f3a84f886df", null],
 ["print_edges_lengths", "_pr_interface_8h.html#ac98a994b8e67dc020d74321deb989a7c", null],
 ["print_lengths", "_pr_interface_8h.html#aa6bdc7b0e7b969e7d7b931a69991b156", null],
 ["printGeomviewPath", "_pr_interface_8h.html#acd00bb4ad8271eeb58b22911102c06e3", null],
 ["printGeomviewPath", "_pr_interface_8h.html#af4f60a12113a1e3abe4f406f4aa5eeab", null],
 ["printGeomviewTriangleSequence", "_pr_interface_8h.html#a53ab7ace7d913995e2b712bb5024ba5e", null],
 ["printGeomviewTriangleSequence", "_pr_interface_8h.html#a1f3a70ddb35f5d312f04be4732de18b2", null],
 ["printGeomviewTriangulation", "_pr_interface_8h.html#a46dc65442ecbad8914bb88ae6c21fb78", null],
 ["printGeomviewTriangulation", "_pr_interface_8h.html#a78613b67dc369c1f1c5b8bd2f41bf817", null]
]
```

Definition at line 1 of file `_pr_interface_8h.js`.

## 30.138 doc/html/\_pr\_multi\_dijkstra\_8h.js File Reference

### Variables

- var [\\_pr\\_multi\\_dijkstra\\_8h](#)

### 30.138.1 Variable Documentation

#### 30.138.1.1 var \_pr\_multi\_dijkstra\_8h

##### Initial value:

```
=
[
 ["HeapNode2", "class_heap_node2.html", "class_heap_node2"],
 ["MultiDijkstra", "class_multi_dijkstra.html", "class_multi_dijkstra"],
 ["GraphType", "_pr_multi_dijkstra_8h.html#a206245534b423169d06a05a296e3a169", null],
 ["Point", "_pr_multi_dijkstra_8h.html#a264185c815a47814791f68e7c6da85ad", null],
 ["HeapType2", "_pr_multi_dijkstra_8h.html#a07dcabcf75f750257e24dd71571ec318", null]
]
```

Definition at line 1 of file `_pr_multi_dijkstra_8h.js`.

## 30.139 doc/html/\_pr\_param\_util\_8h.js File Reference

### Variables

- var [\\_pr\\_param\\_util\\_8h](#)

### 30.139.1 Variable Documentation

#### 30.139.1.1 var \_pr\_param\_util\_8h

##### Initial value:

```
=
[
 ["area", "_pr_param_util_8h.html#a88d3831b85d9d3912354527e5fe6d73c", null],
 ["area", "_pr_param_util_8h.html#a64c674e467409817ea2a3f3d259ff21f", null],
 ["baryCoords", "_pr_param_util_8h.html#a2723a4fb8d9d4e9806b85d24be903113", null],
 ["baryCoords0", "_pr_param_util_8h.html#a031f05ae9429c7ef17f461c7b12b4cbf", null],
 ["cotangent", "_pr_param_util_8h.html#a1494b41fd0552963267eebd1178f96f5", null],
 ["det", "_pr_param_util_8h.html#a67274299dba3338b5ca9a1dc2fe43ea5", null],
 ["polarCoords", "_pr_param_util_8h.html#ae4fbc9e0feac9dd18f35463c670a19a", null],
 ["polarCoords", "_pr_param_util_8h.html#a38cf5694f3655997dc7b8f64787a5e55", null],
 ["tanThetaOverTwo", "_pr_param_util_8h.html#acf689ef17a246312b527f1c2b8fa1266", null]
]
```

Definition at line 1 of file `_pr_param_util_8h.js`.

## 30.140 doc/html/\_pr\_parametrize\_bdy\_8h.js File Reference

### Variables

- [var \\_pr\\_parametrize\\_bdy\\_8h](#)

### 30.140.1 Variable Documentation

#### 30.140.1.1 var \_pr\_parametrize\_bdy\_8h

##### Initial value:

```
=
[
 ["PrParametrizeBdy", "class_pr_parametrize_bdy.html", "class_pr_parametrize_bdy"],
 ["PrBdyParamKind", "_pr_parametrize_bdy_8h.html#ac4f0ea96d827517d164d141b76822905", [
 ["PrCHORDLENGTHBDY", "
 _pr_parametrize_bdy_8h.html#ac4f0ea96d827517d164d141b76822905a01024afd7e94cb195b4956f794df5ea3", null],
 ["PrCENTRIPETAL", "
 _pr_parametrize_bdy_8h.html#ac4f0ea96d827517d164d141b76822905a9d8ef4f0fb695440e1e12f49b448ee1c", null],
 ["PrUNIFBDY", "
 _pr_parametrize_bdy_8h.html#ac4f0ea96d827517d164d141b76822905ac0af8acf5a5382cb8d17efec95c6f194", null]
]]
]
```

Definition at line 1 of file `_pr_parametrize_bdy_8h.js`.

## 30.141 doc/html/\_pr\_parametrize\_int\_8h.js File Reference

### Variables

- [var \\_pr\\_parametrize\\_int\\_8h](#)

### 30.141.1 Variable Documentation

#### 30.141.1.1 var \_pr\_parametrize\_int\_8h

##### Initial value:

```
=
[
 ["PrParametrizeInt", "class_pr_parametrize_int.html", "class_pr_parametrize_int"],
 ["PrParamStartVector", "_pr_parametrize_int_8h.html#ab93ec75afebf4127286a826175e2184e", [
 ["PrBARYCENTRE", "
 _pr_parametrize_int_8h.html#ab93ec75afebf4127286a826175e2184eac1e230601ee7949ef1e75f1a4439f4e7", null],
 ["PrFROMUV", "
 _pr_parametrize_int_8h.html#ab93ec75afebf4127286a826175e2184ea2740fa09d2d03e4c2d55733fd8ec3e36", null]
]]
]
```

Definition at line 1 of file `_pr_parametrize_int_8h.js`.

## 30.142 doc/html/\_pr\_path\_triangle\_seq\_8h.js File Reference

### Variables

- var [\\_pr\\_path\\_triangle\\_seq\\_8h](#)

#### 30.142.1 Variable Documentation

##### 30.142.1.1 var \_pr\_path\_triangle\_seq\_8h

Definition at line 1 of file \_pr\_path\_triangle\_seq\_8h.js.

## 30.143 doc/html/\_pr\_texture\_8h.js File Reference

### Variables

- var [\\_pr\\_texture\\_8h](#)

#### 30.143.1 Variable Documentation

##### 30.143.1.1 var \_pr\_texture\_8h

#### Initial value:

```
=
[
 ["printTexture", "_pr_texture_8h.html#a05539445948b94bfbceeafb09ca79355", null]
]
```

Definition at line 1 of file \_pr\_texture\_8h.js.

## 30.144 doc/html/\_pr\_wavelet\_util\_8h.js File Reference

### Variables

- var [\\_pr\\_wavelet\\_util\\_8h](#)

#### 30.144.1 Variable Documentation

##### 30.144.1.1 var \_pr\_wavelet\_util\_8h

#### Initial value:

```
=
[
 ["theta", "_pr_wavelet_util_8h.html#a15e73be8d8f9ce9ea4cbd71ee2b9c980", null]
]
```

Definition at line 1 of file \_pr\_wavelet\_util\_8h.js.

## 30.145 doc/html/\_quality\_utils\_8h.js File Reference

### Variables

- var [\\_quality\\_utils\\_8h](#)

### 30.145.1 Variable Documentation

#### 30.145.1.1 var \_quality\_utils\_8h

##### Initial value:

```
=
[
 ["estimateArea", "_quality_utils_8h.html#a46ec6241e74820a157e314d4b163e1c2", null],
 ["estimateLoopArea", "_quality_utils_8h.html#a0fd632c6b50602b49201e247ebaf6b45", null],
 ["hasIndistinctKnots", "_quality_utils_8h.html#a9b42768776ea5a799f5d58baedf493ac", null],
 ["isSliverFace", "_quality_utils_8h.html#aed4c9016d1223c82a3b20dae423dddab", null],
 ["isSliverFace", "_quality_utils_8h.html#a104103b9742a9dddbe928b7703d93273", null],
 ["isSliverFace", "_quality_utils_8h.html#a75c27d806accb47bcd5808bb0b131e1b", null],
 ["isSliverFace2", "_quality_utils_8h.html#ad0b8815d663271c1ea92a74d0a6250cf", null]
]
```

Definition at line 1 of file `_quality_utils_8h.js`.

## 30.146 doc/html/\_rational\_8h.js File Reference

### Variables

- var [\\_rational\\_8h](#)

### 30.146.1 Variable Documentation

#### 30.146.1.1 var \_rational\_8h

##### Initial value:

```
=
[
 ["Rational", "class_go_1_1_rational.html", "class_go_1_1_rational"],
 ["operator*", "_rational_8h.html#acd00661011348959429ac61eba54450b", null],
 ["operator+", "_rational_8h.html#af4bb35b00be1c5d14445424567d40924", null],
 ["operator-", "_rational_8h.html#a2288c9a51f97c6dcfb70326c4966295f", null],
 ["operator/", "_rational_8h.html#acf19e1992bd9553c19f5c1592f12017d", null],
 ["operator<<", "_rational_8h.html#a73534fc637d495b35fcf6a638c44ff9f", null]
]
```

Definition at line 1 of file `_rational_8h.js`.



## 30.147 doc/html/\_registration\_utils\_8h.js File Reference

### Variables

- var [\\_registration\\_utils\\_8h](#)

### 30.147.1 Variable Documentation

#### 30.147.1.1 var \_registration\_utils\_8h

##### Initial value:

```
=
[
 ["RegistrationInput", "struct_go_1_1_registration_input.html", "struct_go_1_1_registration_input"],
 ["RegistrationResult", "struct_go_1_1_registration_result.html", "struct_go_1_1_registration_result"]
,
 ["RegistrationReturnType", "_registration_utils_8h.html#a4d9923f0997091e744526f5654084584", [
 ["RegistrationOK", "
 _registration_utils_8h.html#a4d9923f0997091e744526f5654084584ad83c7e38a4aeb95705671d8fe4e7a0d6", null],
 ["TooFewPoints", "
 _registration_utils_8h.html#a4d9923f0997091e744526f5654084584a409700864df743fb5ed29a95821ab1ad", null],
 ["PointSetSizeDiff", "
 _registration_utils_8h.html#a4d9923f0997091e744526f5654084584abcdec83ae9dd6bb083acac7e3b218dc3", null],
 ["AreaTooSmall", "
 _registration_utils_8h.html#a4d9923f0997091e744526f5654084584a8c50f932f2a5486c74f9faf803baf700", null],
 ["SolveFailed", "
 _registration_utils_8h.html#a4d9923f0997091e744526f5654084584ac56050e3b34602e91f4081c84c3afb45", null]
]],
 ["addToLinearSystem", "_registration_utils_8h.html#a4a7c74a330ce13c1ce246a2d334db386", null],
 ["fineRegistration", "_registration_utils_8h.html#a32bf7c4fa290cdb0dea3a82d785a2f70", null],
 ["rawRegistration", "_registration_utils_8h.html#aac556b609d575079c24b6f3d3b768298", null],
 ["registration", "_registration_utils_8h.html#a4831812139dcffc44f486feded3f0600", null]
]
```

Definition at line 1 of file `_registration_utils_8h.js`.

## 30.148 doc/html/\_regularize\_utils\_8h.js File Reference

### Variables

- var [\\_regularize\\_utils\\_8h](#)

### 30.148.1 Variable Documentation

#### 30.148.1.1 var \_regularize\_utils\_8h

##### Initial value:

```
=
[
 ["adjustTrimSeg", "_regularize_utils_8h.html#ab03a9de2d4a3e76a7247e01bec34ff46", null],
 ["angleInEndpoints", "_regularize_utils_8h.html#a3e9e023534852388e4323b7a30e5807a", null],
 ["checkPath", "_regularize_utils_8h.html#ab8cd502e3d254c59f19410df90a80891", null],
 ["checkRegularity", "_regularize_utils_8h.html#a1e063d06793ab7a215005e3d3bad9f09", null],
 ["checkStrightParCv", "_regularize_utils_8h.html#a1a9f23d6f619e174b85019c24be683d", null],
 ["checkStrightParCv", "_regularize_utils_8h.html#ac80e95c7deb858902ea893bd3ac7a7ba", null],
 ["checkStrightParCv", "_regularize_utils_8h.html#a5cda5168b0b2ad44605f837c6a1b15aa", null],
 ["checkTrimConfig", "_regularize_utils_8h.html#a54800724845f163385fc11ac8d1e0cc7", null],
 ["checkTrimSeg", "_regularize_utils_8h.html#a51bfd8a074e30aed6c7712bcbe3b6372", null],
 ["checkTrimSeg2", "_regularize_utils_8h.html#a92514dd3409f3fa07f0f2ab0afb1e0d1", null],
 ["checkTrimSeg3", "_regularize_utils_8h.html#acd73683d4b80f51509b69ef8b76b7e02", null],
 ["cornerInShortestPath", "_regularize_utils_8h.html#a2ac2fec0ce7a594a0e83ffa0ea9b5197", null],
 ["createFaces", "_regularize_utils_8h.html#aefc7214e2be4a3f087be18f363157bec", null],
 ["divideVertex", "_regularize_utils_8h.html#aea0fd3969995ee455abcc04a86cadcf3", null],
 ["endVxInChain", "_regularize_utils_8h.html#af82e4906c7be047230b5879bc4ed455e", null],
 ["findVertexSplit", "_regularize_utils_8h.html#a63e13e3c414de2c43a43df75f223a55c", null],
 ["getClosestBoundaryPar", "_regularize_utils_8h.html#ae1e743485ecfb9b61a6ae15e4caa8664", null],
 ["getDivisionPlane", "_regularize_utils_8h.html#a6752635dc2d95db3b61325b9a2d1a7d6", null],
 ["getInVec", "_regularize_utils_8h.html#a63d9dccb8f8acce97747220e6eca", null],
 ["getMaxParFrac", "_regularize_utils_8h.html#ad0da2d6a5b73bacc05d93ef2ca29976", null],
 ["getOppositeBoundaryPar", "_regularize_utils_8h.html#ac4e103ee529c635f3199b7b403b570fb", null],
 ["getPath", "_regularize_utils_8h.html#a811b72efa742fd4bc2c8efe21c5e394e", null],
 ["getSourceCvs", "_regularize_utils_8h.html#ae7142e1d8173437374fa4fa19fd2fee9", null],
 ["mergeSituationContinuation", "_regularize_utils_8h.html#a844ac90341d1013832c0da880bf35565", null],
 ["noExtension", "_regularize_utils_8h.html#a1acc4dafb5c79c9a62212fbac4cd9cd5", null],
 ["selectCandVx", "_regularize_utils_8h.html#a53772d53ee11874fd34f994cfb14a679", null],
 ["traverseUntilTJoint", "_regularize_utils_8h.html#a7030be3138658e7497c5de416f2a6986", null]
]

```

Definition at line 1 of file `_regularize_utils_8h.js`.

## 30.149 doc/html/\_s\_i\_s\_lconversion\_8h.js File Reference

### Variables

- var `_s_i_s_lconversion_8h`

### 30.149.1 Variable Documentation

#### 30.149.1.1 var `_s_i_s_lconversion_8h`

##### Initial value:

```
=
[
 ["Curve2SISL", "_s_i_s_lconversion_8h.html#a2491f37cd3da574fc4bd84462f353008", null],
 ["Curve2SISL_rat", "_s_i_s_lconversion_8h.html#a4e36ec92b881249512da1f32d48b0c8b", null],
 ["GoSurf2SISL", "_s_i_s_lconversion_8h.html#a3d92b383daa4ed962792b8df5796e7a2", null],
 ["SISLCurve2Go", "_s_i_s_lconversion_8h.html#a014388c3978c2cb31649a1f7e1b17bbb", null],
 ["SISLSurf2Go", "_s_i_s_lconversion_8h.html#af854c9d2a4faa28d3a0e053710a1fbb8", null]
]

```

Definition at line 1 of file `_s_i_s_lconversion_8h.js`.

## 30.150 doc/html/\_singular\_8h.js File Reference

### Variables

- var `_singular_8h`

### 30.150.1 Variable Documentation

#### 30.150.1.1 var \_singular\_8h

**Initial value:**

```
=
[
 ["vanishingNormal", "_singular_8h.html#a26f9cae0b0a16fb49ea296f92bb38d88", null],
 ["vanishingTangent", "_singular_8h.html#af3cf03b5484e64f91845595487db1843", null]
]
```

Definition at line 1 of file \_singular\_8h.js.

## 30.151 doc/html/\_singularity\_classification\_8h.js File Reference

### Variables

- var [\\_singularity\\_classification\\_8h](#)

### 30.151.1 Variable Documentation

#### 30.151.1.1 var \_singularity\_classification\_8h

**Initial value:**

```
=
[
 ["SingularityClassification", "_singularity_classification_8h.html#a2d40b57e4fc6eed03a9fbeebe181e7c5",
 [
 ["NO_SING", "
_singularity_classification_8h.html#a2d40b57e4fc6eed03a9fbeebe181e7c5ab1559d7dcf7654af2bcec501f5312732", null],
 ["INIT_SING", "
_singularity_classification_8h.html#a2d40b57e4fc6eed03a9fbeebe181e7c5a490091ee185383778d6cd7293288d611", null],
 ["DIVIDED_SING", "
_singularity_classification_8h.html#a2d40b57e4fc6eed03a9fbeebe181e7c5a98b637c26d37d470fcfbb5260774f8f9", null],
 ["KEEP_SING", "
_singularity_classification_8h.html#a2d40b57e4fc6eed03a9fbeebe181e7c5a5950a0471c4dedf23ebab3d521693e9e", null]
]
]
]
```

Definition at line 1 of file \_singularity\_classification\_8h.js.

## 30.152 doc/html/\_singularity\_type\_8h.js File Reference

### Variables

- var [\\_singularity\\_type\\_8h](#)

### 30.152.1 Variable Documentation

#### 30.152.1.1 var\_singularity\_type\_8h

##### Initial value:

```
=
[
 ["SingularityType", "_singularity_type_8h.html#a3f21d9ac1d33fe9bf8364573e0126af2", [
 ["ORDINARY_POINT", "_singularity_type_8h.html#a3f21d9ac1d33fe9bf8364573e0126af2a1fdcba6fec487885e75718a98dd830b0", null],
 ["TANGENTIAL_POINT", "_singularity_type_8h.html#a3f21d9ac1d33fe9bf8364573e0126af2a7a751867640d143dba6d355e3ad044ce", null],
 ["ISOLATED_POINT", "_singularity_type_8h.html#a3f21d9ac1d33fe9bf8364573e0126af2a1b8fe22dbe7702bdc99dd252ffc2ebc4", null],
 ["BRANCH_POINT", "_singularity_type_8h.html#a3f21d9ac1d33fe9bf8364573e0126af2a268fe557333f706f171980d85a3c8897", null],
 ["HIGHER_ORDER_POINT", "_singularity_type_8h.html#a3f21d9ac1d33fe9bf8364573e0126af2a22f487388b26506dd36247ceb5747b41", null]
]]
]
```

Definition at line 1 of file `_singularity_type_8h.js`.

## 30.153 doc/html/\_smooth\_volume\_8h.js File Reference

### Variables

- var [\\_smooth\\_volume\\_8h](#)

### 30.153.1 Variable Documentation

#### 30.153.1.1 var\_smooth\_volume\_8h

##### Initial value:

```
=
[
 ["SmoothVolume", "class_go_1_1_smooth_volume.html", "class_go_1_1_smooth_volume"],
 ["CoefStatus", "_smooth_volume_8h.html#a24654e7a71d69eef31552a072adeecc4", [
 ["CoefFree", "_smooth_volume_8h.html#a24654e7a71d69eef31552a072adeecc4acc3d70575051b1013dd5c4c2c320b903", null],
 ["CoefKnown", "_smooth_volume_8h.html#a24654e7a71d69eef31552a072adeecc4a29f03e2de4b5d85782b35c411364c8b9", null],
 ["CoefAvoid", "_smooth_volume_8h.html#a24654e7a71d69eef31552a072adeecc4ad3f55b5236e42142ec74305c3c1835eb", null],
 ["CoefOther", "_smooth_volume_8h.html#a24654e7a71d69eef31552a072adeecc4a0a7f75ddd8adc10ac2f890df80d7bc96", null]
]]
]
```

Definition at line 1 of file `_smooth_volume_8h.js`.

## 30.154 doc/html/\_spline\_debug\_utils\_8h.js File Reference

### Variables

- var [\\_spline\\_debug\\_utils\\_8h](#)

### 30.154.1 Variable Documentation

#### 30.154.1.1 var spline\_debug\_utils\_8h

##### Initial value:

```
=
[
 ["objsToFile", "_spline_debug_utils_8h.html#a9ab9fe829e55d0a4c64537b7197c90fc", null],
 ["objToFile", "_spline_debug_utils_8h.html#a025275ced8cb3a4d858e5e688459f7fb", null],
 ["writeCvsOnSf", "_spline_debug_utils_8h.html#aff508623d7e8af4d9e60b981c4c1b5ab", null],
 ["writeCvsOnSf", "_spline_debug_utils_8h.html#a471ce0dcc864ec1469d15bee77abda47", null],
 ["writeOuterBoundaryLoop", "_spline_debug_utils_8h.html#adf3b1ea9fdee5b49ee982b196835f1b1", null],
 ["writeSISLFormat", "_spline_debug_utils_8h.html#a471ce0dcc864ec1469d15bee77abda47", null],
 ["writeSpaceParamCurve", "_spline_debug_utils_8h.html#ae7ea73070ed893f7422c24ae0d119cb6", null],
 ["writeSpaceParamCurve", "_spline_debug_utils_8h.html#a2a3d10df7cbb2f020a44b51a545c6a86", null],
 ["writeTrimmedInfo", "_spline_debug_utils_8h.html#aa182fa5ddd1962ded096b5752708aaf8", null]
]
```

Definition at line 1 of file `_spline_debug_utils_8h.js`.

## 30.155 doc/html/\_spline\_utils\_8h.js File Reference

### Variables

- var [spline\\_utils\\_8h](#)

### 30.155.1 Variable Documentation

#### 30.155.1.1 var spline\_utils\_8h

##### Initial value:

```
=
[
 ["closest_in_array", "_spline_utils_8h.html#ad0b75a04db51bcb78fa2bb19bcb4aa33", null],
 ["closest_on_line_segment", "_spline_utils_8h.html#a456841bb1f236d2a425fc1f9a1c1e99d", null],
 ["closest_on_rectgrid", "_spline_utils_8h.html#a4c07e5d5c7c1d6a0079a44ba2d68b5a1", null],
 ["closest_on_rectgrid", "_spline_utils_8h.html#af4911fb18e062dc33cb1603708e21e1f", null],
 ["closest_on_triangle", "_spline_utils_8h.html#a9ad707052b437c98fd4a59138e492bf6", null],
 ["curve_ratder", "_spline_utils_8h.html#ad82474684fdffe163a98b99b593db9", null],
 ["extractBezierCoefs", "_spline_utils_8h.html#aad065b5050899dec56c43aa786357e06", null],
 ["insertKnots", "_spline_utils_8h.html#a1fe5eaadddb2004e581998d5029ab59c", null],
 ["make_coef_array_from_rational_coefs", "_spline_utils_8h.html#a3deb0f86f557d5c4f297605ff0f31c63", null],
 ["osloalg", "_spline_utils_8h.html#aa1539751b135c0ae6707325744eab7ff", null],
 ["refinedBezierCoefsCubic", "_spline_utils_8h.html#a98130202c020c0d6aa19511a69875bb3", null],
 ["refineToBezier", "_spline_utils_8h.html#a707a5dd4e9d2b3a25a0daf3df4ddfad7", null],
 ["refmatrix", "_spline_utils_8h.html#a19d359dc0c04f2c2e3f4a517129c1526", null],
 ["splineToBezierTransfMat", "_spline_utils_8h.html#a4cb10616760a1995e3ec2fa8f1814111", null],
 ["surface_ratder", "_spline_utils_8h.html#a1a2e159f4d5ff53835263180aa4f0202", null],
 ["transpose_array", "_spline_utils_8h.html#aa8a0841b35a489065578a166a2e03a39", null]
]
```

Definition at line 1 of file `_spline_utils_8h.js`.

## 30.156 doc/html/\_spline\_volume\_8h.js File Reference

### Variables

- var [\\_spline\\_volume\\_8h](#)

### 30.156.1 Variable Documentation

#### 30.156.1.1 var \_spline\_volume\_8h

##### Initial value:

```
=
[
 ["BasisPts", "struct_go_1_1_basis_pts.html", "struct_go_1_1_basis_pts"],
 ["BasisDerivs", "struct_go_1_1_basis_derivs.html", "struct_go_1_1_basis_derivs"],
 ["BasisDerivs2", "struct_go_1_1_basis_derivs2.html", "struct_go_1_1_basis_derivs2"],
 ["SplineVolume", "class_go_1_1_spline_volume.html", "class_go_1_1_spline_volume"],
 ["SPLINE_VOLUME_BD_SFS_SIZE", "_spline_volume_8h.html#a78ac980199fce248a693bcac021a6099", null],
 ["SPLINE_VOLUME_DEGEN_SIZE", "_spline_volume_8h.html#a530569cc4a9966aade6a64be07cceb7", null],
 ["SPLINE_VOLUME_PERIOD_INFO_SIZE", "_spline_volume_8h.html#a91a9dbdfe8d48f1fdcc34d596dee702b", null]
]
```

Definition at line 1 of file `_spline_volume_8h.js`.

## 30.157 doc/html/\_split\_model\_utils\_8h.js File Reference

### Variables

- var [\\_split\\_model\\_utils\\_8h](#)

### 30.157.1 Variable Documentation

#### 30.157.1.1 var \_split\_model\_utils\_8h

##### Initial value:

```
=
[
 ["splitInFreeCorners", "_split_model_utils_8h.html#a731cf99590470ea5781efcc8cc0897f7", null],
 ["splitInNonCorners", "_split_model_utils_8h.html#a5fb661cefef94184cefa2bec1e650ed9", null],
 ["splitInOuterVertices", "_split_model_utils_8h.html#a749ca80f7d32a01b5743998073e878b8", null]
]
```

Definition at line 1 of file `_split_model_utils_8h.js`.

## 30.158 doc/html/\_stream\_utils\_8h.js File Reference

### Variables

- var [\\_stream\\_utils\\_8h](#)

### 30.158.1 Variable Documentation

#### 30.158.1.1 var \_stream\_utils\_8h

##### Initial value:

```
=
[
 ["object_from_stream", "_stream_utils_8h.html#a4894fce509ab613c659adfb8c6e4f3a4", null],
 ["object_from_stream", "_stream_utils_8h.html#aa32125ce457acfe089493557cb692bb5", null],
 ["object_from_stream", "_stream_utils_8h.html#a096e67b6673d7d6479006279ce415576", null],
 ["object_from_stream", "_stream_utils_8h.html#a17a755168f4d23a504d416a2dd99f2bd", null],
 ["object_to_stream", "_stream_utils_8h.html#aa01c0a039e65b4114a243eddb38ed4b3", null],
 ["object_to_stream", "_stream_utils_8h.html#ab156ea451457461fae5f88814546be88", null],
 ["object_to_stream", "_stream_utils_8h.html#af6f93dc84b2f7d738c9860431bdb56dd", null],
 ["object_to_stream", "_stream_utils_8h.html#a6f0c13c5750e8dd6c8fe94e05cda5e99", null]
]
```

Definition at line 1 of file \_stream\_utils\_8h.js.

## 30.159 doc/html/\_streamable\_8h.js File Reference

### Variables

- var [\\_streamable\\_8h](#)

### 30.159.1 Variable Documentation

#### 30.159.1.1 var \_streamable\_8h

##### Initial value:

```
=
[
 ["Streamable", "class_go_1_1_streamable.html", "class_go_1_1_streamable"],
 ["EofException", "class_go_1_1_streamable_1_1_eof_exception.html", null],
 ["operator<<", "_streamable_8h.html#aa5c798f53d1918fbc856ed61004fd34c", null],
 ["operator>>", "_streamable_8h.html#aab82a782826598596c233196f3b2a89c", null]
]
```

Definition at line 1 of file \_streamable\_8h.js.

## 30.160 doc/html/\_subdivision\_classification\_8h.js File Reference

### Variables

- var [\\_subdivision\\_classification\\_8h](#)

### 30.160.1 Variable Documentation

#### 30.160.1.1 var \_subdivision\_classification\_8h

##### Initial value:

```
=
[
 ["SubdivisionClassification", "_subdivision_classification_8h.html#a20f9976341cc350b6e0d38e92eca1583",
 [
 ["CANNOT_SUBDIVIDE", "
 _subdivision_classification_8h.html#a20f9976341cc350b6e0d38e92eca1583af713558562644ccbc38448cc0ccd431f", null],
 ["DIVIDE_DEG", "
 _subdivision_classification_8h.html#a20f9976341cc350b6e0d38e92eca1583a7024a78dfc8b657dfcd157ef0e0b1886", null],
 ["DIVIDE_CRITICAL", "
 _subdivision_classification_8h.html#a20f9976341cc350b6e0d38e92eca1583a5a94412a23dc475840ac49836f9acacc", null],
 ["DIVIDE_HIGH_SING", "
 _subdivision_classification_8h.html#a20f9976341cc350b6e0d38e92eca1583a8c4ad2ea3ed668d02477f0b10b55d6ad", null],
 ["DIVIDE_SING", "
 _subdivision_classification_8h.html#a20f9976341cc350b6e0d38e92eca1583aff8eb5e454216f614a6f2debdd17b98", null],
 ["DIVIDE_KNOT", "
 _subdivision_classification_8h.html#a20f9976341cc350b6e0d38e92eca1583a261b563ba14323519361642ffd4fb8e6", null],
 ["DIVIDE_INT", "
 _subdivision_classification_8h.html#a20f9976341cc350b6e0d38e92eca1583afc505e9630be475c3842fb7b43f30d94", null],
 ["DIVIDE_PAR", "
 _subdivision_classification_8h.html#a20f9976341cc350b6e0d38e92eca1583a4ad3fd9c982e0d60e139cf3d8fc16ea8", null]
]
]
]
```

Definition at line 1 of file `_subdivision_classification_8h.js`.

## 30.161 doc/html/\_surface\_creators\_8h.js File Reference

### Variables

- var [\\_surface\\_creators\\_8h](#)

### 30.161.1 Variable Documentation

#### 30.161.1.1 var \_surface\_creators\_8h

##### Initial value:

```
=
[
 ["createSmoothTransition", "_surface_creators_8h.html#a858bb397e81281be0bb543508e3c56b9", null],
 ["insertParamDomain", "_surface_creators_8h.html#a0f06ceb8b7da47dfd27594a3cdbf95f2", null],
 ["mergeRationalParts", "_surface_creators_8h.html#a81f9af572c3f6d8bdcbdd20d832c1fc9", null],
 ["multiDBezierPatches", "_surface_creators_8h.html#a443311afde0b95d1e72801ca0fa91881", null],
 ["multiDSurfaces", "_surface_creators_8h.html#ae95a71e47c11aa9f423046405e5c3750", null],
 ["separateRationalParts", "_surface_creators_8h.html#a9e2f8f7e84cb01abe72ae79427cc86cb", null]
]
```

Definition at line 1 of file `_surface_creators_8h.js`.



## 30.162 doc/html/\_surface\_interpolator\_8h.js File Reference

### Variables

- [var \\_surface\\_interpolator\\_8h](#)

#### 30.162.1 Variable Documentation

##### 30.162.1.1 var \_surface\_interpolator\_8h

###### Initial value:

```
=
[
 ["regularInterpolation", "_surface_interpolator_8h.html#afd90b4917533a9d8939d5838d712c229", null]
]
```

Definition at line 1 of file \_surface\_interpolator\_8h.js.

## 30.163 doc/html/\_surface\_model\_utils\_8h.js File Reference

### Variables

- [var \\_surface\\_model\\_utils\\_8h](#)

#### 30.163.1 Variable Documentation

##### 30.163.1.1 var \_surface\_model\_utils\_8h

###### Initial value:

```
=
[
 ["checkClosedFaces", "_surface_model_utils_8h.html#a679b4c93fe452dc5bf2f31994c818ed5", null]
]
```

Definition at line 1 of file \_surface\_model\_utils\_8h.js.

## 30.164 doc/html/\_surface\_on\_volume\_tools\_8h.js File Reference

### Variables

- [var \\_surface\\_on\\_volume\\_tools\\_8h](#)

### 30.164.1 Variable Documentation

#### 30.164.1.1 var \_surface\_on\_volume\_tools\_8h

##### Initial value:

```
=
[
 ["getOuterBoundaryLoop", "_surface_on_volume_tools_8h.html#aa585ddfed4894e8954b4bc33c814af11", null]
]
```

Definition at line 1 of file \_surface\_on\_volume\_tools\_8h.js.

## 30.165 doc/html/\_surface\_tools\_8h.js File Reference

### Variables

- var [\\_surface\\_tools\\_8h](#)

### 30.165.1 Variable Documentation

#### 30.165.1.1 var \_surface\_tools\_8h

##### Initial value:

```
=
[
 ["absolutelyAllBoundarySfLoops", "_surface_tools_8h.html#af4323213d23da8bc5ef4f5fcab4cdac0", null],
 ["allBoundarySfLoops", "_surface_tools_8h.html#a38c1b35dedce6b995bdab79c0a099b62", null],
 ["checkCoefCoLinearity", "_surface_tools_8h.html#af45aae277730c2f7765ff4b23fe60b7e", null],
 ["checkSurfaceClosed", "_surface_tools_8h.html#a97544e15edd15cd8313c96485074e73a", null],
 ["cornerToCornerSfs", "_surface_tools_8h.html#ac57f6942894cf2f49d6a91c885f53de4", null],
 ["estimateTangentLength", "_surface_tools_8h.html#ab6a68a15d4d30497549a58d65e6c15d3", null],
 ["getCoefEnumeration", "_surface_tools_8h.html#a448a3866411fde253b4482c8ea938061", null],
 ["getCoefEnumeration", "_surface_tools_8h.html#aa86f25d2c74cbb60ef6188c5074089b1", null],
 ["getCornerCoefEnum", "_surface_tools_8h.html#a0b1837d7023cbd191257b94fc88b203f", null],
 ["getCorrCoefEnum", "_surface_tools_8h.html#a0b280b6e8aefe8519547fbbf4ecc516f", null],
 ["getParEpsilon", "_surface_tools_8h.html#aa550ccbd55b0427b0e0e603389387568", null],
 ["getSfAdjacencyInfo", "_surface_tools_8h.html#a69d645f167d5c3e0a151413329d77fc4", null],
 ["iterateCornerPos", "_surface_tools_8h.html#a862c0c530045ad4b59fbf91051719ad6", null],
 ["outerBoundarySfLoop", "_surface_tools_8h.html#a0b4b094f6b39fe9d2496b2a04d9c76df", null],
 ["parameterizeByBaseSurf", "_surface_tools_8h.html#ad9e5bc87e70ff17ed3bb6bf877b9facf", null],
 ["surface_seedfind", "_surface_tools_8h.html#aac26a6206c258b5b821dd45b536a8b59", null],
 ["surfaceClosed", "_surface_tools_8h.html#a4d9d51de1000caccldd29b7a96636176", null]
]
```

Definition at line 1 of file \_surface\_tools\_8h.js.

## 30.166 doc/html/\_tesselator\_utils\_8h.js File Reference

### Variables

- var [\\_tesselator\\_utils\\_8h](#)

### 30.166.1 Variable Documentation

#### 30.166.1.1 var \_tesselator\_utils\_8h

##### Initial value:

```
=
[
 ["getCtrPol", "_tesselator_utils_8h.html#a686ee084c6285418ea0f03c7eee6e331", null],
 ["getResolution", "_tesselator_utils_8h.html#a07334c0026ebfb7a56738e4bf1957c29", null]
]
```

Definition at line 1 of file \_tesselator\_utils\_8h.js.

## 30.167 doc/html/\_utils\_8h.js File Reference

### Variables

- var [\\_utils\\_8h](#)

### 30.167.1 Variable Documentation

#### 30.167.1.1 var \_utils\_8h

##### Initial value:

```
=
[
 ["go_iterator_traits", "struct_go_1_lgo_iterator_traits.html", "struct_go_1_lgo_iterator_traits"]
 ,
 ["go_iterator_traits< T * >", "struct_go_1_lgo_iterator_traits_3_01_t_01_5_01_4.html", "
 struct_go_1_lgo_iterator_traits_3_01_t_01_5_01_4"],
 ["go_iterator_traits< const T * >", "struct_go_1_lgo_iterator_traits_3_01const_01_t_01_5_01_4.html",
 "struct_go_1_lgo_iterator_traits_3_01const_01_t_01_5_01_4"],
 ["distance_squared", "_utils_8h.html#ad522e8e98b924e9e0226f4f4111144f1", null],
 ["eatwhite", "_utils_8h.html#a43780996b9bc44a2d902858e953f8050", null],
 ["inner", "_utils_8h.html#a277ca01c9889efb36be1d32af5545484", null],
 ["normalize", "_utils_8h.html#accf22745624858161d45d2d05f5a3511", null],
 ["sum", "_utils_8h.html#ad9e8c423ald270b6b015f3a09a3c30c9", null],
 ["sum_squared", "_utils_8h.html#a48fa5a9b5a428653667b7cf046ab6920", null]
]
```

Definition at line 1 of file \_utils\_8h.js.

## 30.168 doc/html/\_values\_8h.js File Reference

### Variables

- var [\\_values\\_8h](#)

### 30.168.1 Variable Documentation

#### 30.168.1.1 var \_values\_8h

##### Initial value:

```
=
[
 ["DEFAULT_PARAMETER_EPSILON", "_values_8h.html#ad93873aa87d054e3af416fee52eeaa8b", null],
 ["DEFAULT_SPACE_EPSILON", "_values_8h.html#a946cce20e8506c27fd9f46c09b39319e", null],
 ["M_PI", "_values_8h.html#ad5fc6b96e5441dd61200dd09dd9e363c", null],
 ["MAXDOUBLE", "_values_8h.html#a29ffe35f4b863b738f5bb836908b5ce0", null],
 ["MAXINT", "_values_8h.html#a85349090d85e7f334d4b3fd1f14a05c5", null]
]
```

Definition at line 1 of file \_values\_8h.js.

## 30.169 doc/html/\_volume\_interpolator\_8h.js File Reference

### Variables

- var [\\_volume\\_interpolator\\_8h](#)

### 30.169.1 Variable Documentation

#### 30.169.1.1 var \_volume\_interpolator\_8h

##### Initial value:

```
=
[
 ["regularInterpolation", "_volume_interpolator_8h.html#ac000136504b5a8b84959ac84b929ae0d", null]
]
```

Definition at line 1 of file \_volume\_interpolator\_8h.js.

## 30.170 doc/html/\_volume\_model\_creator\_8h.js File Reference

### Variables

- var [\\_volume\\_model\\_creator\\_8h](#)

### 30.170.1 Variable Documentation

#### 30.170.1.1 var \_volume\_model\_creator\_8h

##### Initial value:

```
=
[
 ["createRotationalModel", "_volume_model_creator_8h.html#a854a55f713caabe38fd56ffc32264e5a", null],
 ["linearSweptModel", "_volume_model_creator_8h.html#a81121da55c79b1600b933a236f4e641b", null]
]
```

Definition at line 1 of file \_volume\_model\_creator\_8h.js.

## 30.171 doc/html/\_volume\_model\_file\_handler\_8h.js File Reference

### Variables

- [var \\_volume\\_model\\_file\\_handler\\_8h](#)

### 30.171.1 Variable Documentation

#### 30.171.1.1 var \_volume\_model\_file\_handler\_8h

##### Initial value:

```
=
[
 ["VolumeModelFileHandler", "class_go_1_1_volume_model_file_handler.html", "
 class_go_1_1_volume_model_file_handler"],
 ["_VOLUMEFILEHANDLER_H", "_volume_model_file_handler_8h.html#a1d5505728388942e4ea90d4be08daaea", null
]
]
```

Definition at line 1 of file `_volume_model_file_handler_8h.js`.

## 30.172 doc/html/\_volume\_tools\_8h.js File Reference

### Variables

- [var \\_volume\\_tools\\_8h](#)

### 30.172.1 Variable Documentation

#### 30.172.1.1 var \_volume\_tools\_8h

##### Initial value:

```
=
[
 ["analyzePeriodicity", "_volume_tools_8h.html#ad73a8f4284b1f2a672a0fa6c9885400d", null],
 ["approxVolParamCurve", "_volume_tools_8h.html#aa500ad08b2e111caedb2c15ba3c6999f", null],
 ["cornerToCornerVols", "_volume_tools_8h.html#adfdf5b99d592845dbbaa634c59b84d69", null],
 ["getBoundarySurface", "_volume_tools_8h.html#a6bc0b6c84bdabced076fa1cb0abb7bc3", null],
 ["getBoundarySurfaces", "_volume_tools_8h.html#a873af9dd9666e4292bedf42a52fc5839", null],
 ["getCorrCoefVolEnum", "_volume_tools_8h.html#a064a74ed96b0c805c7a6645500042ce4", null],
 ["getOrientedBoundarySurface", "_volume_tools_8h.html#a58b40a61c979ea4daf9ebb65da13a062", null],
 ["getOrientedBoundarySurfaces", "_volume_tools_8h.html#a128d7e1806cb12df2b8083e063f66dfb", null],
 ["getVolAdjacencyInfo", "_volume_tools_8h.html#ad2f7e61bc55d47793d03770659881c06", null],
 ["getVolBdCoefEnumeration", "_volume_tools_8h.html#ab5f6e24e913f17d375c1cbcffda61869", null],
 ["getVolCoefEnumeration", "_volume_tools_8h.html#a46225b6df1dd3860f5f351d7a0212984", null],
 ["getVolCoefEnumeration", "_volume_tools_8h.html#aff10f42549713cd1da90b201b41ea056", null],
 ["liftVolParamCurve", "_volume_tools_8h.html#a31ab8609766412bdd39243eaaafd1f530", null],
 ["projectVolParamCurve", "_volume_tools_8h.html#a2567b57a3c7c6e2240e86cab87dc7392", null],
 ["representCurveAsVolume", "_volume_tools_8h.html#a28dd9bc785e8364b1a681187208edd6e", null],
 ["representSurfaceAsVolume", "_volume_tools_8h.html#ad17cd9703b11abe9bfb2a20f211d5ab8", null],
 ["representVolumeAsCurve", "_volume_tools_8h.html#acb3006dd8acc2a77965132805c6654f7", null],
 ["representVolumeAsSurface", "_volume_tools_8h.html#a8d05413d8ale3c4a965e18494ffbe092", null],
 ["volCommonSplineSpace", "_volume_tools_8h.html#a044797df4ccc3301dc4543b9224f1e50", null]
]
```

Definition at line 1 of file `_volume_tools_8h.js`.

## 30.173 doc/html/\_volumes\_8h.js File Reference

### Variables

- var [\\_volumes\\_8h](#)

### 30.173.1 Variable Documentation

#### 30.173.1.1 var \_volumes\_8h

#### Initial value:

```
=
[
 ["area", "_volumes_8h.html#a8cc8f744af2fc5541e493ffdd3d2e1f5", null],
 ["area", "_volumes_8h.html#a04f0f159a17bcc67285bb7349debd945", null],
 ["determinantOf", "_volumes_8h.html#a19c9bd461ffdf353431464a6fe0c80b5", null],
 ["determinantOf", "_volumes_8h.html#aa003b2f42cc209edef837b35294badfc", null],
 ["signed_area", "_volumes_8h.html#a99bb3e0367f54a9f8a328e22c4e6d2b2", null],
 ["simplex_volume", "_volumes_8h.html#a6d1e64227d0bbaea95363f58701082e5", null],
 ["volume", "_volumes_8h.html#ad8c173fcf99167307d4691d039ba47b7", null]
]
```

Definition at line 1 of file `_volumes_8h.js`.

## 30.174 doc/html/annotated\_dup.js File Reference

### Variables

- var [annotated\\_dup](#)

### 30.174.1 Variable Documentation

#### 30.174.1.1 var annotated\_dup

Definition at line 1 of file `annotated_dup.js`.

## 30.175 doc/html/aux2\_8cpp.js File Reference

### Variables

- var [aux2\\_8cpp](#)

### 30.175.1 Variable Documentation

#### 30.175.1.1 var aux2\_8cpp

**Initial value:**

```
=
[
 ["tic", "aux2_8cpp.html#ace51fd7d8435f88c8b2b3bcf64367673", null],
 ["toc", "aux2_8cpp.html#aec20b9a2b3b99a34f91fc9b8e1d83225", null],
 ["jon_timer", "aux2_8cpp.html#ab6b3a0f92f7304c85e25c9422c4c959c", null]
]
```

Definition at line 1 of file aux2\_8cpp.js.

## 30.176 doc/html/aux2\_8h.js File Reference

### Variables

- var [aux2\\_8h](#)

### 30.176.1 Variable Documentation

#### 30.176.1.1 var aux2\_8h

Definition at line 1 of file aux2\_8h.js.

## 30.177 doc/html/bandmat\_8cpp.js File Reference

### Variables

- var [bandmat\\_8cpp](#)

### 30.177.1 Variable Documentation

#### 30.177.1.1 var bandmat\_8cpp

**Initial value:**

```
=
[
 ["REPORT", "bandmat_8cpp.html#a787099914a94ed31fa544b985b55752f", null],
 ["WANT_MATH", "bandmat_8cpp.html#a8335b327d416f961a5dbaa0e058cc515", null],
 ["square", "bandmat_8cpp.html#ae0c42a7620957500708a95cbb31e4260", null]
]
```

Definition at line 1 of file bandmat\_8cpp.js.

## 30.178 doc/html/binom\_8h.js File Reference

### Variables

- var [binom\\_8h](#)

### 30.178.1 Variable Documentation

#### 30.178.1.1 var binom\_8h

##### Initial value:

```
=
[
 ["binom", "binom_8h.html#a5b80ecab78ea5ff1030dacc17d799161", null],
 ["factorial", "binom_8h.html#af2dale127f042c70cdb76c5b48158f8d", null],
 ["quadrinomial", "binom_8h.html#a28ad91e9ccb0d4796b3c226c7a9f847c", null],
 ["trinomial", "binom_8h.html#ac36f51acb739431294b2f88a8e6f65fb", null]
]
```

Definition at line 1 of file binom\_8h.js.

## 30.179 doc/html/boolean\_8h.js File Reference

### Variables

- var [boolean\\_8h](#)

### 30.179.1 Variable Documentation

#### 30.179.1.1 var boolean\_8h

##### Initial value:

```
=
[
 ["bool", "classbool.html", "classbool"],
 ["bool_LIB", "boolean_8h.html#ae034ee292f37bf7f1a5488c0afe12cfd", null],
 ["false", "boolean_8h.html#aa9a4c04243afd8c8f2274245a5d39635", null],
 ["true", "boolean_8h.html#aad3977b1b821fa4804b65c37a188066a", null]
]
```

Definition at line 1 of file boolean\_8h.js.

## 30.180 doc/html/brent\_\_minimize\_8h.js File Reference

### Variables

- var [brent\\_\\_minimize\\_8h](#)



### 30.180.1 Variable Documentation

#### 30.180.1.1 var brent\_\_minimize\_8h

##### Initial value:

```
=
[
 ["Fun2Fun", "class_go_1_1_fun2_fun.html", "class_go_1_1_fun2_fun"],
 ["brent_minimize", "brent__minimize_8h.html#a1bee23d40e575050d659ee12bc9d0f55", null]
]
```

Definition at line 1 of file brent\_\_minimize\_8h.js.

## 30.181 doc/html/checks\_8h.js File Reference

### Variables

- var [checks\\_8h](#)

### 30.181.1 Variable Documentation

#### 30.181.1.1 var checks\_8h

##### Initial value:

```
=
[
 ["compare_seq", "checks_8h.html#a53dfe6870adf2a25c6f827e7947c11f7", null],
 ["interval_overlap", "checks_8h.html#a55d10f76746c762b77ed74ed73ad8eec", null],
 ["nondecreasing", "checks_8h.html#a55281b6a4d49b457d96336b75076cbdd", null],
 ["nonincreasing", "checks_8h.html#acd8f6d5e30019f5e7d039dd550311a52", null],
 ["strictly_decreasing", "checks_8h.html#a6504eff7b2aa3dd41986e71dca7db979", null],
 ["strictly_decreasing", "checks_8h.html#a6b95d20d57a8288cffd972814ab7cf16", null],
 ["strictly_decreasing", "checks_8h.html#ae83af5c77cc2f4fe843bc34c7dd188cc", null],
 ["strictly_increasing", "checks_8h.html#a1582e823b204b085607da45b8c738509", null],
 ["strictly_increasing", "checks_8h.html#adf7e6a25193e1108491056e90951a030", null],
 ["strictly_increasing", "checks_8h.html#a00795366e02c59c0fc5a4982a47a7108", null],
 ["weakly_decreasing", "checks_8h.html#a9ef55b273f5d603669d73063e84a435c", null],
 ["weakly_decreasing", "checks_8h.html#af88f2fc0b462e4d9bf12c1361cd362ec", null],
 ["weakly_increasing", "checks_8h.html#af16aaec72dabb981744e338aef8c9748", null],
 ["weakly_increasing", "checks_8h.html#ad520388adbe4b3719116f02ba3be0145", null]
]
```

Definition at line 1 of file checks\_8h.js.

## 30.182 doc/html/cholesky\_8cpp.js File Reference

### Variables

- var [cholesky\\_8cpp](#)

### 30.182.1 Variable Documentation

#### 30.182.1.1 var cholesky\_8cpp

##### Initial value:

```
=
[
 ["REPORT", "cholesky_8cpp.html#a787099914a94ed31fa544b985b55752f", null],
 ["WANT_MATH", "cholesky_8cpp.html#a8335b327d416f961a5dbaa0e058cc515", null],
 ["Cholesky", "cholesky_8cpp.html#a284464751aa76302c87bb61ea3f6d1f9", null],
 ["Cholesky", "cholesky_8cpp.html#a76f68d776790ec4ed5faf7bc242bb800", null],
 ["square", "cholesky_8cpp.html#ae0c42a7620957500708a95cbb31e4260", null]
]
```

Definition at line 1 of file cholesky\_8cpp.js.

## 30.183 doc/html/class\_control\_word.js File Reference

### Variables

- var [class\\_control\\_word](#)

### 30.183.1 Variable Documentation

#### 30.183.1.1 var class\_control\_word

##### Initial value:

```
=
[
 ["ControlWord", "class_control_word.html#aacdbb089f6862671dale99bda70f4af4", null],
 ["ControlWord", "class_control_word.html#a9e2737a4f4b24d73930088439c6551fc", null],
 ["operator!", "class_control_word.html#aeef9676db08c7fae1934eb3c46a403a39", null],
 ["operator*", "class_control_word.html#af8a40c7f8a1555bd2cf75ec8fe9c8647", null],
 ["operator*=", "class_control_word.html#acc7993ad57360c97fae79feb144d55c9", null],
 ["operator+", "class_control_word.html#a75ce7d9b159be3b28f8084efc8b4f4dc", null],
 ["operator+", "class_control_word.html#a2b1a94955cdd3236371e43cf6fc11372", null],
 ["operator+=", "class_control_word.html#a53f00d494d9ae34d06e03d40432803", null],
 ["operator-", "class_control_word.html#aca3251cbe0c602130e64a1e841e30a4e", null],
 ["operator-=", "class_control_word.html#aaebb50638e89c4b82a786e3da19d990b", null],
 ["operator<=", "class_control_word.html#a550195e839781f72c34aa8db6e12392f", null],
 ["operator>=", "class_control_word.html#ad88a1cee8890c8def3bcb1735ffcaf9", null],
 ["operator^", "class_control_word.html#ae889d5901a335f0c1a8c19b376648a6f", null],
 ["operator~", "class_control_word.html#a8fffd2ba79b2cf045c5a32bb3b3b0bf9a", null],
 ["cw", "class_control_word.html#ae696e3b3c704de2e65c82330c856ff79", null]
]
```

Definition at line 1 of file class\_control\_word.js.

## 30.184 doc/html/class\_curve\_resolution\_sheet.js File Reference

### Variables

- var [class\\_curve\\_resolution\\_sheet](#)

### 30.184.1 Variable Documentation

#### 30.184.1.1 var class\_curve\_resolution\_sheet

##### Initial value:

```
=
[
 ["CurveResolutionSheet", "class_curve_resolution_sheet.html#abc796a029f0fb49851c335326ad945eb", null],
 ["createSheet", "class_curve_resolution_sheet.html#ab222c39ceaeda7a79613e172e4f5e6d6", null],
 ["ok", "class_curve_resolution_sheet.html#af01e3f06bc03483b45a9dea07e61ee71", null],
 ["return_value", "class_curve_resolution_sheet.html#aa6b04a87fd9b92e3ecda823eea6ebb1", null]
]
```

Definition at line 1 of file class\_curve\_resolution\_sheet.js.

## 30.185 doc/html/class\_data\_handler.js File Reference

### Variables

- var [class\\_data\\_handler](#)

### 30.185.1 Variable Documentation

#### 30.185.1.1 var class\_data\_handler

##### Initial value:

```
=
[
 ["DataHandler", "class_data_handler.html#a51d845c1d9bbeef44607ec7592ba3c1d", null],
 ["~DataHandler", "class_data_handler.html#aa20ea75200d943624192976b21f657a3", null],
 ["clear", "class_data_handler.html#a7b19e680cbb9de9fd0089324a021c5f5", null],
 ["create", "class_data_handler.html#ae8ee9c581e71b7e5f9a13a2ecaf6f234", null],
 ["paintable", "class_data_handler.html#ac5537697c71dab750535e2f9f9f62935", null],
 ["propertySheet", "class_data_handler.html#a70d03706137156029dabfa7142dc11c6", null],
 ["tesselator", "class_data_handler.html#ab7caedcbf1e13f7456ce14b84c3b6789", null],
 ["paintable_", "class_data_handler.html#aa1afe2dc6943ccfda36c3bdd26f6d15f", null],
 ["property_sheet_", "class_data_handler.html#a82a24f3c15c88170263d4a64faaf307d", null],
 ["tesselator_", "class_data_handler.html#ac6931f251770a4e9b125d18bb2711092", null]
]
```

Definition at line 1 of file class\_data\_handler.js.

## 30.186 doc/html/class\_data\_handler\_vol\_and\_l\_r.js File Reference

### Variables

- var [class\\_data\\_handler\\_vol\\_and\\_l\\_r](#)

### 30.186.1 Variable Documentation

#### 30.186.1.1 var class\_data\_handler\_vol\_and\_l\_r

##### Initial value:

```
=
[
 ["DataHandlerVolAndLR", "class_data_handler_vol_and_l_r.html#aa9165829b2bcbfd4bbb5c2afbb4ea6c0", null
],
 ["~DataHandlerVolAndLR", "class_data_handler_vol_and_l_r.html#ae9d6524526c45e5a779b6567e5b4ea", null
],
 ["create", "class_data_handler_vol_and_l_r.html#a48c986b0f55f2d0c5cc0eb4ce8e2c7a4", null]
]
```

Definition at line 1 of file class\_data\_handler\_vol\_and\_l\_r.js.

## 30.187 doc/html/class\_default\_data\_handler.js File Reference

### Variables

- var [class\\_default\\_data\\_handler](#)

### 30.187.1 Variable Documentation

#### 30.187.1.1 var class\_default\_data\_handler

##### Initial value:

```
=
[
 ["DefaultDataHandler", "class_default_data_handler.html#a2cc2d1471f7aa6023300736250faf6b4", null],
 ["~DefaultDataHandler", "class_default_data_handler.html#a800919c97fc56c2d08eaa42336a93c54", null],
 ["create", "class_default_data_handler.html#a5af315f8342bbb09ecc0450356737839", null]
]
```

Definition at line 1 of file class\_default\_data\_handler.js.

## 30.188 doc/html/class\_dijkstra.js File Reference

### Variables

- var [class\\_dijkstra](#)

## 30.188.1 Variable Documentation

### 30.188.1.1 var class\_dijkstra

#### Initial value:

```
=
[
 ["Dijkstra", "class_dijkstra.html#ad7c7bb035ef96e4dbd652fe2bd11a2be", null],
 ["~Dijkstra", "class_dijkstra.html#a5f4a99a63b92805a304dea947ac69db5", null],
 ["closestNeighbour", "class_dijkstra.html#a55ebd94cf963d8f693f884b93f3da76f", null],
 ["getDistance", "class_dijkstra.html#ac444cf56e4db2c0cdb8599b1065be584", null],
 ["getDistance", "class_dijkstra.html#afbeafc367fe8e25041c794d807b27575", null],
 ["initialize", "class_dijkstra.html#a0f7f7f26abbe38dc975d1b73da1fd352", null],
 ["insertInCandidates", "class_dijkstra.html#a80acelc298b9754b60468d99392cff64", null],
 ["isFinished", "class_dijkstra.html#a563d8a91c7448efa23f093bce2e937b1", null],
 ["run", "class_dijkstra.html#adb2c14d1e9bd1ac6837d556e93124f8", null],
 ["run", "class_dijkstra.html#ad81d8645d987ca5957e32b39f280b372", null],
 ["run", "class_dijkstra.html#a70daee9f05ba724087abadf7f7ad0e42", null],
 ["setFinished", "class_dijkstra.html#a96173796d7ede8affcdf44a0be23aae3", null],
 ["setGraph", "class_dijkstra.html#a225cd16b0dbe3634e83208fa5b9dbee8", null],
 ["setSource", "class_dijkstra.html#ada95f1751e12ca5084ccd1670d32394d", null],
 ["back_trace_", "class_dijkstra.html#ab9f8f09dfc6b20674b646803a637c258", null],
 ["distances_", "class_dijkstra.html#ab14b9563e6c73cdfd117f7097a19619d", null],
 ["flags_", "class_dijkstra.html#a21e1b614db2ae36d5af65313b0780334", null],
 ["graph_", "class_dijkstra.html#a35ec62ac1573a9a6a74b4b53fdf70dad", null],
 ["large_distance_", "class_dijkstra.html#af7d4f8f644cdf1330bca5ce819fa58a2", null],
 ["queue_", "class_dijkstra.html#af2a9d922323cb2849a2256b53f3df903", null]
]
```

Definition at line 1 of file class\_dijkstra.js.

## 30.189 doc/html/class\_edge\_type.js File Reference

### Variables

- var [class\\_edge\\_type](#)

### 30.189.1 Variable Documentation

#### 30.189.1.1 var class\_edge\_type

#### Initial value:

```
=
[
 ["initEdgeType", "class_edge_type.html#a641b2e8b3a9c3f37cc6069f369fe694b", null],
 ["isVertex", "class_edge_type.html#a9350916f6ca2ba32ab52c284ae619ec2", null],
 ["vertex_", "class_edge_type.html#a90544457b28ef5bc656953195dd129f0", null]
]
```

Definition at line 1 of file class\_edge\_type.js.

## 30.190 doc/html/class\_g\_a\_r\_c\_h11\_\_\_l\_l.js File Reference

### Variables

- var [class\\_g\\_a\\_r\\_c\\_h11\\_\\_\\_l\\_l](#)

### 30.190.1 Variable Documentation

#### 30.190.1.1 var class\_g\_a\_r\_c\_h11\_\_\_l\_l

##### Initial value:

```
=
[
 ["GARCH11_LL", "class_g_a_r_c_h11___l_l.html#aad5b3bde84a4e545d8b3ac3729dc67ad", null],
 ["Derivatives", "class_g_a_r_c_h11___l_l.html#a494bfca46885a5d7f8bee8bade920e59", null],
 ["FI", "class_g_a_r_c_h11___l_l.html#a990b9201af32feeb2677beb0dc3c92a1", null],
 ["IsValid", "class_g_a_r_c_h11___l_l.html#a107b06b171c186cf48584f6a67b90cb3", null],
 ["LogLikelihood", "class_g_a_r_c_h11___l_l.html#a52af6343a9631dce4fe625e80e9b5088", null],
 ["Set", "class_g_a_r_c_h11___l_l.html#aebbd2dd6f57f05ca1403c2a3d238db6d", null]
]
```

Definition at line 1 of file class\_g\_a\_r\_c\_h11\_\_\_l\_l.js.

## 30.191 doc/html/class\_go\_1\_1\_adapt\_curve.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_adapt\\_curve](#)

### 30.191.1 Variable Documentation

#### 30.191.1.1 var class\_go\_1\_1\_adapt\_curve

##### Initial value:

```
=
[
 ["AdaptCurve", "class_go_1_1_adapt_curve.html#a59f26c3f1fceed642efceb138bb79f1e", null],
 ["AdaptCurve", "class_go_1_1_adapt_curve.html#a53e5c367a32d9f393204573ce696f0a2", null],
 ["AdaptCurve", "class_go_1_1_adapt_curve.html#af2cba296962b451ad5176ca9b4c646c5", null],
 ["AdaptCurve", "class_go_1_1_adapt_curve.html#ae4f4202df2fa04ce461e79b819731b84", null],
 ["~AdaptCurve", "class_go_1_1_adapt_curve.html#ab27acbe4726d28c4e383860279fb26a3", null],
 ["AdaptCurve", "class_go_1_1_adapt_curve.html#a055a6c06eee5010316f910a26763bd46", null],
 ["approximate", "class_go_1_1_adapt_curve.html#a93d4c3bbfd8bb8d07ac3e1a6ac9d21bf", null],
 ["getAdaptCurve", "class_go_1_1_adapt_curve.html#a682e70cfb120abb0e1dff82ca6bf68f3", null],
 ["setEndPoints", "class_go_1_1_adapt_curve.html#a344d40537704fbb8528d383ff4ea2bcd", null],
 ["setSmooth", "class_go_1_1_adapt_curve.html#afac2da0e1525cc473326c7a19d68b4ef", null],
 ["unsetSmooth", "class_go_1_1_adapt_curve.html#add23ddbcb56e3ffdeef14f592d8c5f1f", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_adapt\_curve.js.

## 30.192 doc/html/class\_go\_1\_1\_alg\_obj2\_d\_int.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_alg\\_obj2\\_d\\_int](#)

### 30.192.1 Variable Documentation

#### 30.192.1.1 var class\_go\_1\_1\_alg\_obj2\_d\_int

##### Initial value:

```
=
[
 ["AlgObj2DInt", "class_go_1_1_alg_obj2_d_int.html#a41ecad45de4ec9467544468c573a7dd6", null],
 ["AlgObj2DInt", "class_go_1_1_alg_obj2_d_int.html#a6a3f8e74476c271fa65cb8fedea0062f", null],
 ["~AlgObj2DInt", "class_go_1_1_alg_obj2_d_int.html#a65c4d9d9a3aea6323c25d3691a2425ff", null],
 ["numTerms", "class_go_1_1_alg_obj2_d_int.html#a788af448ddf9ab74661aa74917f05e19", null],
 ["term", "class_go_1_1_alg_obj2_d_int.html#aaaaafb3b4c074628a2e5e638e73c45b0c", null],
 ["degree_", "class_go_1_1_alg_obj2_d_int.html#a01db2490cf1855c6303f76b0e6bealle", null],
 ["power_basis_", "class_go_1_1_alg_obj2_d_int.html#a9d0a4185e3ecbfc723d4b8292d2dd52a", null],
 ["terms_", "class_go_1_1_alg_obj2_d_int.html#a1e1990768fc14046b0100f702f0fc269", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_alg\_obj2\_d\_int.js.

## 30.193 doc/html/class\_go\_1\_1\_alg\_obj3\_d\_int.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_alg\\_obj3\\_d\\_int](#)

### 30.193.1 Variable Documentation

#### 30.193.1.1 var class\_go\_1\_1\_alg\_obj3\_d\_int

##### Initial value:

```
=
[
 ["AlgObj3DInt", "class_go_1_1_alg_obj3_d_int.html#a411328d76c197f6d3e1144271b143abe", null],
 ["AlgObj3DInt", "class_go_1_1_alg_obj3_d_int.html#adc15e68c0462a596823e28c84d89a1ae", null],
 ["AlgObj3DInt", "class_go_1_1_alg_obj3_d_int.html#ab8b8b1b212e40fd26042d755f53fb1ab", null],
 ["~AlgObj3DInt", "class_go_1_1_alg_obj3_d_int.html#a19a00c74ba29d3f588290ae19c77c1ba", null],
 ["degree", "class_go_1_1_alg_obj3_d_int.html#a98eb423e117d224879f6eb5d5c019d8e", null],
 ["getImplicit", "class_go_1_1_alg_obj3_d_int.html#ac1d386416f530957080df2da48e2423e", null],
 ["numTerms", "class_go_1_1_alg_obj3_d_int.html#ab4c002603683639c2ee441b09283b599", null],
 ["term", "class_go_1_1_alg_obj3_d_int.html#a36f9991313fdef5a2b777805cff541cf", null],
 ["usingPowerBasis", "class_go_1_1_alg_obj3_d_int.html#ae7e10dc5b80f754207c3044417261c60", null],
 ["bc_", "class_go_1_1_alg_obj3_d_int.html#a4fc32e98d86d35dc5cc4d935871f39e3", null],
 ["degree_", "class_go_1_1_alg_obj3_d_int.html#a0dd25a44d1732d39481bcd80bd9e15b2", null],
 ["implicit_", "class_go_1_1_alg_obj3_d_int.html#a8516851fe71f87e841cfe5fc3cb9ed79", null],
 ["power_basis_", "class_go_1_1_alg_obj3_d_int.html#ac2d65fef799a3b818efa91890b1bcc84", null],
 ["terms_", "class_go_1_1_alg_obj3_d_int.html#ad56baaf737039d1760a78cc87d3232a6", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_alg\_obj3\_d\_int.js.

## 30.194 doc/html/class\_go\_1\_1\_alg\_object\_int.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_alg\\_object\\_int](#)

### 30.194.1 Variable Documentation

#### 30.194.1.1 var class\_go\_1\_1\_alg\_object\_int

##### Initial value:

```
=
[
 ["AlgObjectInt", "class_go_1_1_alg_object_int.html#a4b7e787ad9622ac22a3d508a20dce580", null],
 ["~AlgObjectInt", "class_go_1_1_alg_object_int.html#ae8d2a0ce2d22c39264206bc379c3817a", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_alg\_object\_int.js.

## 30.195 doc/html/class\_go\_1\_1\_approx\_crv\_to\_seqs.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_approx\\_crv\\_to\\_seqs](#)

### 30.195.1 Variable Documentation

#### 30.195.1.1 var class\_go\_1\_1\_approx\_crv\_to\_seqs

##### Initial value:

```
=
[
 ["ApproxCrvToSeqs", "class_go_1_1_approx_crv_to_seqs.html#a94b1145dff622caf5ba6f07ab70f5f9d", null],
 ["ApproxCrvToSeqs", "class_go_1_1_approx_crv_to_seqs.html#a37269ed15b04c386cd52b81ec84fe13e", null],
 ["ApproxCrvToSeqs", "class_go_1_1_approx_crv_to_seqs.html#ac4fe7840d83833c351e46deae60c9abe", null],
 ["~ApproxCrvToSeqs", "class_go_1_1_approx_crv_to_seqs.html#a85f320a7d0ba92e2eff89f9713d66ef", null],
 ["ApproxCrvToSeqs", "class_go_1_1_approx_crv_to_seqs.html#a424ab0ed21332aa0364527f510c9aeb5", null],
 ["getApproxCurves", "class_go_1_1_approx_crv_to_seqs.html#ac38511c7c533bffe4d46abfb64808349", null],
 ["setClApprox", "class_go_1_1_approx_crv_to_seqs.html#a688379ea435e20811eb12428b56ac58d", null],
 ["setEndpoints", "class_go_1_1_approx_crv_to_seqs.html#a39a779ab46160626fadc94a7a5aeeb51", null],
 ["setSmooth", "class_go_1_1_approx_crv_to_seqs.html#a35bec85833f19cb50a34263ee05b4b65", null],
 ["unsetSmooth", "class_go_1_1_approx_crv_to_seqs.html#a54049a70064fe762aa22b740b9aa113b", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_approx\_crv\_to\_seqs.js.

## 30.196 doc/html/class\_go\_1\_1\_approx\_curve.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_approx\\_curve](#)



### 30.196.1 Variable Documentation

#### 30.196.1.1 var class\_go\_1\_1\_approx\_curve

##### Initial value:

```
=
[
 ["ApproxCurve", "class_go_1_1_approx_curve.html#a2b97121a74308760375ebad7de6d33ad", null],
 ["ApproxCurve", "class_go_1_1_approx_curve.html#a6135eed0dac2fde79afd208a67e044c6", null],
 ["ApproxCurve", "class_go_1_1_approx_curve.html#a4353fa67b89c38420f1fe8daef62cb46", null],
 ["~ApproxCurve", "class_go_1_1_approx_curve.html#ad55f51d891e243b0e4b9823alda088ce", null],
 ["ApproxCurve", "class_go_1_1_approx_curve.html#a863ad42d169ffd5c4b5a87916be24a8b", null],
 ["getApproxCurve", "class_go_1_1_approx_curve.html#a5dd813e5bcde851a74b76b6b72c85e3f", null],
 ["setClApprox", "class_go_1_1_approx_curve.html#ad36f56f04bfe61a2e90b709fe840741b", null],
 ["setEndPoints", "class_go_1_1_approx_curve.html#aecf82eb78c617e7fcab8159ecc21e0ee", null],
 ["setSmooth", "class_go_1_1_approx_curve.html#a8e587c29a96f02baf130f78fabfdb6e0", null],
 ["unsetSmooth", "class_go_1_1_approx_curve.html#aca033c663a6ab7a527e4168b0fc416d3", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_approx\_curve.js.

## 30.197 doc/html/class\_go\_1\_1\_approx\_surf.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_approx\\_surf](#)

### 30.197.1 Variable Documentation

#### 30.197.1.1 var class\_go\_1\_1\_approx\_surf

##### Initial value:

```
=
[
 ["ApproxSurf", "class_go_1_1_approx_surf.html#a41a1c2453303fbab79b8bef53c1576a6", null],
 ["ApproxSurf", "class_go_1_1_approx_surf.html#ae046f4d99fff00a02c1707e92b5053dd", null],
 ["~ApproxSurf", "class_go_1_1_approx_surf.html#a5dd3d905523895bf63aa9f9ade89769f", null],
 ["ApproxSurf", "class_go_1_1_approx_surf.html#a7781c1001f19a97499b193907d693b15", null],
 ["edgeFix", "class_go_1_1_approx_surf.html#ab4b9c87258b369bfc3f2be5a5f62f09b", null],
 ["getApproxSurf", "class_go_1_1_approx_surf.html#ae31ab7d1acd7b5aded2c53f996bb528a", null],
 ["getDoRefine", "class_go_1_1_approx_surf.html#a9e272b8f26ba790d9e6ce902fc1eecl", null],
 ["reParam", "class_go_1_1_approx_surf.html#a2410776c1c4f29145e5ba1ae38acd9eb", null],
 ["setClApprox", "class_go_1_1_approx_surf.html#a70d75984a119b781ddf03ed62891017d", null],
 ["setDoRefine", "class_go_1_1_approx_surf.html#a250605b3b40a5d8308349ece57ea5797", null],
 ["setFixBoundary", "class_go_1_1_approx_surf.html#a4891ec1c6ca76a33792bb5bc5398160a", null],
 ["setNormalConditions", "class_go_1_1_approx_surf.html#acc2a7fc44113521b40a7778a59cd52f1", null],
 ["setSmoothingWeight", "class_go_1_1_approx_surf.html#abbd16f25c48e3586e532e6ba3cb1b271", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_approx\_surf.js.

## 30.198 doc/html/class\_go\_1\_1\_array.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_array](#)

### 30.198.1 Variable Documentation

#### 30.198.1.1 var class\_go\_1\_1\_array

Definition at line 1 of file class\_go\_1\_1\_array.js.

## 30.199 doc/html/class\_go\_1\_1\_bary\_coord\_system.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_bary\\_coord\\_system](#)

### 30.199.1 Variable Documentation

#### 30.199.1.1 var class\_go\_1\_1\_bary\_coord\_system

#### Initial value:

```
=
[
 ["BaryCoordSystem", "class_go_1_1_bary_coord_system.html#ad212dfe2c4ae74d37206c179121985e7", null],
 ["BaryCoordSystem", "class_go_1_1_bary_coord_system.html#ae167aelf54c769dc9a3617eb9a40c53", null],
 ["baryToCart", "class_go_1_1_bary_coord_system.html#a4ba4d3d3261342c9e6fad85b26e98fe0", null],
 ["cartToBary", "class_go_1_1_bary_coord_system.html#a4178d4a4b631a598d285a941f2ce67fa", null],
 ["corner", "class_go_1_1_bary_coord_system.html#a64a075601f1119b8d10fdif9d76d14d0", null],
 ["read", "class_go_1_1_bary_coord_system.html#a91eb73692e0ae0c157e5aa749a3352bd", null],
 ["setCorners", "class_go_1_1_bary_coord_system.html#af328a8fe051c0d943038e87a808cf9e1", null],
 ["write", "class_go_1_1_bary_coord_system.html#a1cab0342adaec3d4a49009166490663b", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_bary\_coord\_system.js.

## 30.200 doc/html/class\_go\_1\_1\_bary\_coord\_system\_triangle3\_d.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_bary\\_coord\\_system\\_triangle3\\_d](#)

### 30.200.1 Variable Documentation

#### 30.200.1.1 var class\_go\_1\_1\_bary\_coord\_system\_triangle3\_d

#### Initial value:

```
=
[
 ["BaryCoordSystemTriangle3D", "
class_go_1_1_bary_coord_system_triangle3_d.html#abfb291cb267df0ad9b8412b3da407859", null],
 ["BaryCoordSystemTriangle3D", "
class_go_1_1_bary_coord_system_triangle3_d.html#a88a0f1be74cf6febb5d8badfa771f644", null],
 ["baryToCart", "class_go_1_1_bary_coord_system_triangle3_d.html#a4ab5bfdbd40d2ab7dd2271547cfc7fd1",
null],
 ["cartToBary", "class_go_1_1_bary_coord_system_triangle3_d.html#af18e2f746e4f4df28a0070ba480949f7",
null]
]
```

Definition at line 1 of file class\_go\_1\_1\_bary\_coord\_system\_triangle3\_d.js.

## 30.201 doc/html/class\_go\_1\_1\_bd\_cond\_funcutor.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_bd\\_cond\\_funcutor](#)

#### 30.201.1 Variable Documentation

##### 30.201.1.1 var class\_go\_1\_1\_bd\_cond\_funcutor

###### Initial value:

```
=
[
 ["BdCondFuncutor", "class_go_1_1_bd_cond_funcutor.html#a6643591bd7a19f2bb09efc03979c51cc", null],
 ["~BdCondFuncutor", "class_go_1_1_bd_cond_funcutor.html#a9db56302f598b168bb593fb3f37935db", null],
 ["evaluate", "class_go_1_1_bd_cond_funcutor.html#a3f8723e288cdc98e1f458dcdc3d37ee5", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_bd\_cond\_funcutor.js.

## 30.202 doc/html/class\_go\_1\_1\_bernstein\_multi.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_bernstein\\_multi](#)

#### 30.202.1 Variable Documentation

##### 30.202.1.1 var class\_go\_1\_1\_bernstein\_multi

Definition at line 1 of file class\_go\_1\_1\_bernstein\_multi.js.

## 30.203 doc/html/class\_go\_1\_1\_bernstein\_poly.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_bernstein\\_poly](#)

#### 30.203.1 Variable Documentation

##### 30.203.1.1 var class\_go\_1\_1\_bernstein\_poly

Definition at line 1 of file class\_go\_1\_1\_bernstein\_poly.js.

## 30.204 doc/html/class\_go\_1\_1\_bernstein\_tetrahedral\_poly.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_bernstein\\_tetrahedral\\_poly](#)

### 30.204.1 Variable Documentation

#### 30.204.1.1 var class\_go\_1\_1\_bernstein\_tetrahedral\_poly

##### Initial value:

```
=
[
 ["BernsteinTetrahedralPoly", "
class_go_1_1_bernstein_tetrahedral_poly.html#ab005bc2833b578aaa471c07bc172dc36", null],
 ["BernsteinTetrahedralPoly", "
class_go_1_1_bernstein_tetrahedral_poly.html#ad247772287ecddb047faa036dfe030", null],
 ["BernsteinTetrahedralPoly", "
class_go_1_1_bernstein_tetrahedral_poly.html#a2b12af1efa01954f1f6df3dbc238c986", null],
 ["BernsteinTetrahedralPoly", "
class_go_1_1_bernstein_tetrahedral_poly.html#ad9149fc219fe173125c96bb3e5781caa", null],
 ["blossom", "class_go_1_1_bernstein_tetrahedral_poly.html#a8306205db716601e4efafedb858edb39", null],
 ["degree", "class_go_1_1_bernstein_tetrahedral_poly.html#a4d1c9ae0578d7859d1e2a6bd50b17744", null],
 ["deriv", "class_go_1_1_bernstein_tetrahedral_poly.html#a4d17fb5166380cdd7535a1733844f99a", null],
 ["norm", "class_go_1_1_bernstein_tetrahedral_poly.html#adc109314566e1d364d63b81f3ac6161e", null],
 ["normalize", "class_go_1_1_bernstein_tetrahedral_poly.html#a9e2baed277141f5a913276a8ff10da15", null]
],
 ["operator()", "class_go_1_1_bernstein_tetrahedral_poly.html#a0365e653db7aa14dc30b1aca8cecd5fa", null
],
 ["operator*=", "class_go_1_1_bernstein_tetrahedral_poly.html#a453903158b1725f0c6f365994267d688", null
],
 ["operator*=", "class_go_1_1_bernstein_tetrahedral_poly.html#aaf9d57b373b5f0fcb7786f452cf4b04a", null
],
 ["operator+=", "class_go_1_1_bernstein_tetrahedral_poly.html#abe004a30018c83b578fe09b245eea66b", null
],
 ["operator+=", "class_go_1_1_bernstein_tetrahedral_poly.html#aaa23a6f059b35dec8055c5864013550d", null
],
 ["operator-=", "class_go_1_1_bernstein_tetrahedral_poly.html#a96958fd0e0a6b07bf12eee07c3b16553", null
],
 ["operator-=", "class_go_1_1_bernstein_tetrahedral_poly.html#a28a009b2651a08ff707ac03b014788fb", null
],
 ["operator/=", "class_go_1_1_bernstein_tetrahedral_poly.html#a5d00173b0e92f56ce27af378713b2ea1", null
],
 ["operator[]", "class_go_1_1_bernstein_tetrahedral_poly.html#ab76213ccf45a4c34352c98512b9b8f94", null
],
 ["operator[]", "class_go_1_1_bernstein_tetrahedral_poly.html#a6f4efce3153a33ed22eda0574a25d658", null
],
 ["pickLine", "class_go_1_1_bernstein_tetrahedral_poly.html#ae27bf0913be08acb7d4a461fdf41392f", null],
 ["read", "class_go_1_1_bernstein_tetrahedral_poly.html#af758b9a9c83aff8bd5c5f98906ddfb66", null],
 ["write", "class_go_1_1_bernstein_tetrahedral_poly.html#ada7cd55f1820d75b14415a7f269e8acb", null]
]
]
```

Definition at line 1 of file class\_go\_1\_1\_bernstein\_tetrahedral\_poly.js.

## 30.205 doc/html/class\_go\_1\_1\_bernstein\_triangular\_poly.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_bernstein\\_triangular\\_poly](#)

### 30.205.1 Variable Documentation

#### 30.205.1.1 var class\_go\_1\_1\_bernstein\_triangular\_poly

**Initial value:**

```
=
[
 ["BernsteinTriangularPoly", "
 class_go_1_1_bernstein_triangular_poly.html#a39388bc6bb9f60590dc12a32870dde34", null],
 ["BernsteinTriangularPoly", "
 class_go_1_1_bernstein_triangular_poly.html#a6331bda3ca424f1cc871d419107167a7", null],
 ["BernsteinTriangularPoly", "
 class_go_1_1_bernstein_triangular_poly.html#af7b6eac59ee0a433ed51475b45d4d828", null],
 ["BernsteinTriangularPoly", "
 class_go_1_1_bernstein_triangular_poly.html#a24146637cc3887c6f6044af04344611d", null],
 ["blossom", "class_go_1_1_bernstein_triangular_poly.html#a6fa732f8939598fda5d8d2e648550689", null],
 ["degree", "class_go_1_1_bernstein_triangular_poly.html#a9e55a7537ff94e2537073f4ebeff5601", null],
 ["deriv", "class_go_1_1_bernstein_triangular_poly.html#a7dbba3ad404bdb2de6bd3161491e009e", null],
 ["norm", "class_go_1_1_bernstein_triangular_poly.html#a8353dfcb9eefea481660cf2740aeb940", null],
 ["normalize", "class_go_1_1_bernstein_triangular_poly.html#a2e2d4609cff2f4fc3d31d16442b5ded4", null],
 ["operator()", "class_go_1_1_bernstein_triangular_poly.html#abf8789b1af8d3ca96c169c54d40a2c51", null]
 ,
 ["operator*=", "class_go_1_1_bernstein_triangular_poly.html#a9791d2594a0f484bd0c54776acc6f239", null]
 ,
 ["operator*=", "class_go_1_1_bernstein_triangular_poly.html#a9f772e707567c42315cda96609140849", null]
 ,
 ["operator+=", "class_go_1_1_bernstein_triangular_poly.html#a75f3d6ecde6b2202b50fd9dd6d78f33a", null]
 ,
 ["operator+=", "class_go_1_1_bernstein_triangular_poly.html#a493d810fb68717241b47f31bf8aefef7", null]
 ,
 ["operator-=", "class_go_1_1_bernstein_triangular_poly.html#a429abe1e12cb8a126ab47140f8459820", null]
 ,
 ["operator-=", "class_go_1_1_bernstein_triangular_poly.html#ae9983ce3d05363b8091d770b85644e49", null]
 ,
 ["operator/=", "class_go_1_1_bernstein_triangular_poly.html#aad9bd5d5849408ec0ef88843d7a2ffec", null]
 ,
 ["operator[]", "class_go_1_1_bernstein_triangular_poly.html#a2403c00c44c9afcd0a2d3e95bbe9900a", null]
 ,
 ["operator[]", "class_go_1_1_bernstein_triangular_poly.html#ab3220b27fe1c5c585efd6839a2469ab7", null]
 ,
 ["read", "class_go_1_1_bernstein_triangular_poly.html#addef2d12c934d32868035bbbe005c948", null],
 ["write", "class_go_1_1_bernstein_triangular_poly.html#ae9150fb59d5da6a36c8ab6510e8afa79", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_bernstein\_triangular\_poly.js.

## 30.206 doc/html/class\_go\_1\_1\_bezier\_triangle.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_bezier\\_triangle](#)

### 30.206.1 Variable Documentation

#### 30.206.1.1 var class\_go\_1\_1\_bezier\_triangle

**Initial value:**

```
=
[
 ["BezierTriangle", "class_go_1_1_bezier_triangle.html#ad2750d4bffde460fa881754f001ca916", null],
 ["BezierTriangle", "class_go_1_1_bezier_triangle.html#ac91dbba3fdd4bc13d24fea26abf4e690", null],
 ["eval", "class_go_1_1_bezier_triangle.html#a04e7a7850a4292cd300aac604e17efec", null],
 ["evalderiv", "class_go_1_1_bezier_triangle.html#a476cd8cc51ec39552379566d756d63ed", null],
 ["interpolate", "class_go_1_1_bezier_triangle.html#af40c612f7285415e5c26560374c83d4a", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_bezier\_triangle.js.

## 30.207 doc/html/class\_go\_1\_1\_binomial.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_binomial](#)

### 30.207.1 Variable Documentation

#### 30.207.1.1 var class\_go\_1\_1\_binomial

##### Initial value:

```
=
[
 ["iter", "class_go_1_1_binomial.html#a3fd34a7ed39ffc597f0745a797bc5ee1", null],
 ["Binomial", "class_go_1_1_binomial.html#a43dc66955a04d1f4f0c2ee77a104f75b", null],
 ["Binomial", "class_go_1_1_binomial.html#a57c4eecadc37db79950dcc7d371ac9eb", null],
 ["operator()", "class_go_1_1_binomial.html#ad8bf97eb681e1192050bba8f2d75221c", null],
 ["operator[]", "class_go_1_1_binomial.html#a83a01d6a20068f458723ca748b0388fc", null],
 ["quadrinomial", "class_go_1_1_binomial.html#a06ca56cf6a69e4f16038a06f4f7277d8", null],
 ["trinomial", "class_go_1_1_binomial.html#a02971d15eed7270349ce7e912a5cf15", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_binomial.js.

## 30.208 doc/html/class\_go\_1\_1\_block\_boundary\_condition.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_block\\_boundary\\_condition](#)

### 30.208.1 Variable Documentation

#### 30.208.1.1 var class\_go\_1\_1\_block\_boundary\_condition

##### Initial value:

```
=
[
 ["BlockBoundaryCondition", "
class_go_1_1_block_boundary_condition.html#aafa9deff68d34dbc161df024553660d1c", null],
 ["~BlockBoundaryCondition", "
class_go_1_1_block_boundary_condition.html#ace81f621b075a8f4f84f7b004e3bb985", null],
 ["getBdCoefficients", "class_go_1_1_block_boundary_condition.html#a559abfe01924826eb466d7fb0b733caa",
null],
 ["getBdCoefficients", "class_go_1_1_block_boundary_condition.html#a4cb0d2ca598baae38ca409005fc157a0",
null],
 ["getBdConditionType", "class_go_1_1_block_boundary_condition.html#a0c48e2a8f011cd1160195daa64ad0602",
null],
 ["getCoefficientsEnumeration", "
class_go_1_1_block_boundary_condition.html#a83586b5ee85dd8b50b17c6e63db6e037", null],
 ["getCoefficientsEnumeration", "
class_go_1_1_block_boundary_condition.html#a13b4c1766c3e4157fa8b76b17779a0c7", null],
 ["getSolutionSpaceIdx", "class_go_1_1_block_boundary_condition.html#a5a7dc67deb9972b7db2530fc4e256ff4",
null],
 ["getTolerances", "class_go_1_1_block_boundary_condition.html#af1ed9c0ac379c75d6ee9c6b91bb59c68", null
],
 ["isDirichlet", "class_go_1_1_block_boundary_condition.html#adb8413ec72c0e979c21d7ac0dd9ac20e", null]
,
 ["update", "class_go_1_1_block_boundary_condition.html#a40d988f2e964f750dc64a8c0c70a3cc9", null],
 ["updateBoundaryValue", "class_go_1_1_block_boundary_condition.html#a99eef723b4ec028d9c86c8bb028b6d56",
null],
 ["type_", "class_go_1_1_block_boundary_condition.html#a18931590bb9d12eee532f580fcc284c", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_block\_boundary\_condition.js.

## 30.209 doc/html/class\_go\_1\_1\_block\_point\_bd\_cond.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_block\\_point\\_bd\\_cond](#)

### 30.209.1 Variable Documentation

#### 30.209.1.1 var class\_go\_1\_1\_block\_point\_bd\_cond

##### Initial value:

```
=
[
 ["BlockPointBdCond", "class_go_1_1_block_point_bd_cond.html#a60a47e210e4dffac71d1b55cf25c6d79", null],
 ["~BlockPointBdCond", "class_go_1_1_block_point_bd_cond.html#a1f30225e04abccf12151a2fb6f44416d", null],
 ["getCoefficientsEnumeration", "class_go_1_1_block_point_bd_cond.html#a387cc859d422b66cae4e8d6466a7a881", null],
 ["getConditionValue", "class_go_1_1_block_point_bd_cond.html#a299c70b29f8fc6d53e39a393c667a913", null],
 ["getInterpolationFactors", "class_go_1_1_block_point_bd_cond.html#a44c58e7b4c6aa2543e17060085eb88a0", null],
 ["getSolutionSpaceIdx", "class_go_1_1_block_point_bd_cond.html#a9b6c144b21088a2905e0c75d220728d4", null]
]
```

Definition at line 1 of file `class_go_1_1_block_point_bd_cond.js`.

## 30.210 doc/html/class\_go\_1\_1\_block\_solution.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_block\\_solution](#)

### 30.210.1 Variable Documentation

#### 30.210.1.1 var class\_go\_1\_1\_block\_solution

##### Initial value:

```
=
[
 ["~BlockSolution", "class_go_1_1_block_solution.html#abccda459fc0b334357c04627f4e719bc", null],
 ["asSfSolution", "class_go_1_1_block_solution.html#a8309d187eee54cdb60e437f9b067a6f4", null],
 ["asVolSolution", "class_go_1_1_block_solution.html#aef9e16803912f3c54082d22386cbf03f", null],
 ["basis", "class_go_1_1_block_solution.html#a7e86fd7aa6a2ba59111ca310251f6aff", null],
 ["degree", "class_go_1_1_block_solution.html#a8e601494d515e57129bde0762042b5d7", null],
 ["dimension", "class_go_1_1_block_solution.html#af0d01c6a57dd4280f7c7b50cbad9cf58", null],
 ["distinctKnots", "class_go_1_1_block_solution.html#a01474d8f8abcd6f412d1594e7146062f", null],
 ["erasePreEvaluatedBasisFunctions", "class_go_1_1_block_solution.html#ac61c3598051cdc35f318b5771341de32", null],
 ["getBoundaryCoefficients", "class_go_1_1_block_solution.html#a0f5518b769f1667203c86fc14e91bf57", null],
 ["getBoundaryCoefficients", "class_go_1_1_block_solution.html#ae642cd601e842f80felcf73c18c363f2", null],
 ["getJacobian", "class_go_1_1_block_solution.html#a0d853137bfae9fb9c37f4bdbe44fe6da", null],
]
```

```
[
 ["getMatchingCoefficients", "class_go_1_1_block_solution.html#aa0d2f9019212f644bbcd69e2530662fd", null],
 ["getNmbOfPointBdConditions", "class_go_1_1_block_solution.html#a4eb11c397226646b78c73f389d1ac185", null],
 ["getTolerances", "class_go_1_1_block_solution.html#a648fe2ece018086ce837fd956f64f00c", null],
 ["increaseDegree", "class_go_1_1_block_solution.html#a53f9eb6610d78056d953548779c286eb", null],
 ["insertKnots", "class_go_1_1_block_solution.html#a3d9e252ba38c2f9c858345cc7cf016a7", null],
 ["insertKnots", "class_go_1_1_block_solution.html#ac6bf8f9fa8d490a9a5b5689e0137fdca", null],
 ["knots", "class_go_1_1_block_solution.html#af0bdaab56a91706e89bcafe49ad2ae8f", null],
 ["makeMatchingSplineSpace", "class_go_1_1_block_solution.html#a9fcc41a509b1d30294a6e61beb80099f", null],
 ["matchingSplineSpace", "class_go_1_1_block_solution.html#ad8b7f6875ee5e07a35c57664cedd2df5", null],
 ["nmbCoefs", "class_go_1_1_block_solution.html#a82872c9f524b1283be5e955bd9c65a82", null],
 ["nmbCoefs", "class_go_1_1_block_solution.html#a20ccfda0ce32e93b2e59e48a09eef181", null],
 ["performPreEvaluation", "class_go_1_1_block_solution.html#ald906d9d0c14128d2b1f7bb69ad29b5c", null],
 ["setSolutionCoefficients", "class_go_1_1_block_solution.html#ae7805d78ddc035169b04bed038d34054", null],
 ["updateConditions", "class_go_1_1_block_solution.html#a5ccc5fefafe1de158a301b28e828f773", null],
 ["valuesInGaussPoint", "class_go_1_1_block_solution.html#aa73182487f313b59db56cc407d1bc1c5", null]
]
```

Definition at line 1 of file `class_go_1_1_block_solution.js`.

## 30.211 doc/html/class\_go\_1\_1\_body.js File Reference

### Variables

- var `class_go_1_1_body`

### 30.211.1 Variable Documentation

#### 30.211.1.1 var `class_go_1_1_body`

#### Initial value:

```
=
[
 ["Body", "class_go_1_1_body.html#af98b5efdc28c9ca6ef92cf5aa4e4ce7b", null],
 ["Body", "class_go_1_1_body.html#a4c42b398cbf3acd7f60aa5a9c7ba4294", null],
 ["Body", "class_go_1_1_body.html#a9dfa0a780ba2693bee36ac704ca72450", null],
 ["~Body", "class_go_1_1_body.html#a9be6043b4f7399802d2847162f4f2cab", null],
 ["addBodyPointers", "class_go_1_1_body.html#a2fff22de2dbe4a85153e7c6f4be06406", null],
 ["addshell", "class_go_1_1_body.html#a46133113ce2c6e98963d7ede3d03f9e0", null],
 ["areNeighbours", "class_go_1_1_body.html#a922224cd79d97088407e697945a0ccf9", null],
 ["boundingBox", "class_go_1_1_body.html#a5b884c2792a068ba527d6c9c0f4395b0", null],
 ["eraseBodyAdjacency", "class_go_1_1_body.html#a36f30526618c66e6a0abe03724a635db", null],
 ["getAdjacentBodies", "class_go_1_1_body.html#ad33e3e17e2c7c7093795357f9b988705", null],
 ["getAllShells", "class_go_1_1_body.html#a7412d17026e98794442fb767bb35c7a9", null],
 ["getMaterial", "class_go_1_1_body.html#a90ea918550cfe458e80b35051577ed3e", null],
 ["getOuterShell", "class_go_1_1_body.html#af9d6709445ed276eb2f943c28f5d22bc", null],
 ["getShell", "class_go_1_1_body.html#a88e5994f472e2723e9d837297074632f", null],
 ["getShell", "class_go_1_1_body.html#abafc8090bda7f804e50a0c0236fa6ade", null],
 ["getTolerances", "class_go_1_1_body.html#a982a64e31c403613a4425999a7b4085e", null],
 ["hasMaterialInfo", "class_go_1_1_body.html#ae3b6afc6244e7d1fbd2675b3aa1bda91", null],
 ["isInside", "class_go_1_1_body.html#a3eec909156712b57b8ffecce676b8ee3a", null],
 ["nmbOfFaces", "class_go_1_1_body.html#aa9509060c4878ce1fbc1d2375ef21e9c", null],
 ["nmbOfShells", "class_go_1_1_body.html#ae9863f6474b189bbac7eaae1f35b85a0", null],
 ["setMaterial", "class_go_1_1_body.html#abcac826e4337372f4da084f657ade8f2", null],
 ["vertices", "class_go_1_1_body.html#ab510c765325c26a341abae6e94cddf19", null],
 ["material_id", "class_go_1_1_body.html#a428472bd85631ad2afeb270f55967566", null],
 ["shells_", "class_go_1_1_body.html#a5818332246ccccf5d8e7453afb75584fa", null],
 ["toptol", "class_go_1_1_body.html#acdad160dfffc6b24a7a8382cbe1ee1af", null]
]
```

Definition at line 1 of file `class_go_1_1_body.js`.



## 30.212 doc/html/class\_go\_1\_1\_bounded\_curve.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_bounded\\_curve](#)

### 30.212.1 Variable Documentation

#### 30.212.1.1 var class\_go\_1\_1\_bounded\_curve

Definition at line 1 of file class\_go\_1\_1\_bounded\_curve.js.

## 30.213 doc/html/class\_go\_1\_1\_bounded\_surface.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_bounded\\_surface](#)

### 30.213.1 Variable Documentation

#### 30.213.1.1 var class\_go\_1\_1\_bounded\_surface

Definition at line 1 of file class\_go\_1\_1\_bounded\_surface.js.

## 30.214 doc/html/class\_go\_1\_1\_bounding\_box.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_bounding\\_box](#)

### 30.214.1 Variable Documentation

#### 30.214.1.1 var class\_go\_1\_1\_bounding\_box

#### Initial value:

```
=
[
 ["BoundingBox", "class_go_1_1_bounding_box.html#ae0734e28e9d45e8b273d793d186b8f03", null],
 ["BoundingBox", "class_go_1_1_bounding_box.html#acc9317dd90c8ade7d6f335df02283acb", null],
 ["BoundingBox", "class_go_1_1_bounding_box.html#a0d21780c44c4e1e6b2b16839fd38af98", null],
 ["~BoundingBox", "class_go_1_1_bounding_box.html#aad668154c46aa29195472e95a1e2f9bb", null],
 ["addUnionWith", "class_go_1_1_bounding_box.html#a5c9caf80330aa3c6b114e25d7212d510", null],
 ["addUnionWith", "class_go_1_1_bounding_box.html#a5fc4a3a8769227d3362210dfb3bdb92f", null],
 ["check", "class_go_1_1_bounding_box.html#af9b87cafc77356ef4e7525d27cdfa200", null],
 ["containsBox", "class_go_1_1_bounding_box.html#a8171125fcb4696fc3cca24864c45e914", null],
 ["containsPoint", "class_go_1_1_bounding_box.html#a45052b81883d0f8ec821164866e1dd2b", null],
 ["dimension", "class_go_1_1_bounding_box.html#a623b0b0fb4624d37c3fddddd2392e16d0", null],
 ["getOverlap", "class_go_1_1_bounding_box.html#a83615344c6e97b47d79a79fb1b64beca", null],
 ["high", "class_go_1_1_bounding_box.html#a9ae5809ed615ef58b246e49e93eed4c1", null],
 ["lineIntersect", "class_go_1_1_bounding_box.html#a24f53e3fc42ff8cb66ab9501e4dbf9f0", null],
 ["low", "class_go_1_1_bounding_box.html#a6416bb152b0865a3c993ca3ee21865b7", null],
 ["overlaps", "class_go_1_1_bounding_box.html#a09571a1e82af4a4cc19773fe0d490eff", null],
 ["read", "class_go_1_1_bounding_box.html#a357abf9122958f6d747d55883cf74277", null],
 ["setFromArray", "class_go_1_1_bounding_box.html#a0805a08f0cb2c73debb1f2a893404873", null],
 ["setFromArray", "class_go_1_1_bounding_box.html#a16038b47ca05306e946f083481e6e4cb", null],
 ["setFromPoints", "class_go_1_1_bounding_box.html#a78e1e9b6e49429e96ed14401110d30fb", null],
 ["setFromPoints", "class_go_1_1_bounding_box.html#ae0a8864eaabb6a738f9e7b1c9960ba88", null],
 ["unset", "class_go_1_1_bounding_box.html#acd09101bbf176c80826164092096bdc4", null],
 ["valid", "class_go_1_1_bounding_box.html#a3082546a7b34a3dd3e51b7ae822f74c6", null],
 ["write", "class_go_1_1_bounding_box.html#ac290179e1101df372c184567a2136735", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_bounding\_box.js.

## 30.215 doc/html/class\_go\_1\_1\_bspline\_basis.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_bspline\\_basis](#)

### 30.215.1 Variable Documentation

#### 30.215.1.1 var class\_go\_1\_1\_bspline\_basis

Definition at line 1 of file class\_go\_1\_1\_bspline\_basis.js.

## 30.216 doc/html/class\_go\_1\_1\_c\_p\_uclock.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_c\\_p\\_uclock](#)

### 30.216.1 Variable Documentation

#### 30.216.1.1 var class\_go\_1\_1\_c\_p\_uclock

#### Initial value:

```
=
[
 ["CPUclock", "class_go_1_1_c_p_uclock.html#a1703ea4f729c6386e2621f01c480db32", null],
 ["getInterval", "class_go_1_1_c_p_uclock.html#a16394976d443ede0280d8a0e239f2b45", null],
 ["getTime", "class_go_1_1_c_p_uclock.html#a7de0b3d30890b87605630f907ec9fb92", null],
 ["initTime", "class_go_1_1_c_p_uclock.html#a2e84ee8cdd5acbf8e3de8b60d13e6629", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_c\_p\_uclock.js.

## 30.217 doc/html/class\_go\_1\_1\_cell\_division.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_cell\\_division](#)

### 30.217.1 Variable Documentation

#### 30.217.1.1 var class\_go\_1\_1\_cell\_division

**Initial value:**

```
=
[
 ["CellDivision", "class_go_1_1_cell_division.html#ae3984c674493bac740e0fe531b0faa68", null],
 ["CellDivision", "class_go_1_1_cell_division.html#a34224cf7f5aa7ebb9b506fd0d6cda294", null],
 ["CellDivision", "class_go_1_1_cell_division.html#a3c01ac003b88f67f5ade7361026f5610", null],
 ["CellDivision", "class_go_1_1_cell_division.html#ac4630091ffc59ce65295bfa28eb1d48e", null],
 ["~CellDivision", "class_go_1_1_cell_division.html#a29f003971ccd31f66f21b39bf7181b10", null],
 ["addFace", "class_go_1_1_cell_division.html#aa444ae96eca39c0b288549cf74db581f", null],
 ["big_box", "class_go_1_1_cell_division.html#afc8375bbf68f75b48831ab1be63a0b88", null],
 ["constructCells", "class_go_1_1_cell_division.html#aea6273310eba94d5652e3cea81827552", null],
 ["constructCells", "class_go_1_1_cell_division.html#a08bala93fe1f94213404ee327eb662b4", null],
 ["constructCells", "class_go_1_1_cell_division.html#a3801432599c72318c4938a6003085eb3", null],
 ["constructCells", "class_go_1_1_cell_division.html#a00f69f62a089fd3c3d2940b025e0d736", null],
 ["getCell", "class_go_1_1_cell_division.html#aad1d2c24f0b48e37c5a1ba58d943c5b6", null],
 ["numCells", "class_go_1_1_cell_division.html#a80530b1674b3cadba93246b6ff6579c", null],
 ["removeFace", "class_go_1_1_cell_division.html#a614e7dc461f55e9274dd77207206face", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_cell\_division.js.

## 30.218 doc/html/class\_go\_1\_1\_circle.js File Reference

**Variables**

- var [class\\_go\\_1\\_1\\_circle](#)

### 30.218.1 Variable Documentation

#### 30.218.1.1 var class\_go\_1\_1\_circle

Definition at line 1 of file class\_go\_1\_1\_circle.js.

## 30.219 doc/html/class\_go\_1\_1\_closest\_point\_calculator.js File Reference

**Variables**

- var [class\\_go\\_1\\_1\\_closest\\_point\\_calculator](#)

### 30.219.1 Variable Documentation

#### 30.219.1.1 var class\_go\_1\_1\_closest\_point\_calculator

**Initial value:**

```
=
[
 ["ClosestPointCalculator", "
 class_go_1_1_closest_point_calculator.html#a1a470db9c6214f1489ba25f45e1722ce", null],
 ["compute", "class_go_1_1_closest_point_calculator.html#a0b11536a5c061dd88c31e34589bd736c", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_closest\_point\_calculator.js.

## 30.220 doc/html/class\_go\_1\_1\_complete\_edge\_net.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_complete\\_edge\\_net](#)

### 30.220.1 Variable Documentation

#### 30.220.1.1 var class\_go\_1\_1\_complete\_edge\_net

##### Initial value:

```
=
[
 ["CompleteEdgeNet", "class_go_1_1_complete_edge_net.html#a031ce3f9ef73a370c6dfbeda23b96988", null],
 ["~CompleteEdgeNet", "class_go_1_1_complete_edge_net.html#ac492be28fcb5b757eb2d715142050f21", null],
 ["getMissingEdges", "class_go_1_1_complete_edge_net.html#af3b890805715f009dacac99a488bf8c1", null],
 ["getRegularizedModel", "class_go_1_1_complete_edge_net.html#a5e15f93efe60619ed65507d0931065bb", null],
 ["perform", "class_go_1_1_complete_edge_net.html#abf05b138fa7ad41cdd7bcc3674e2e9b5", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_complete\_edge\_net.js.

## 30.221 doc/html/class\_go\_1\_1\_complexity\_info.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_complexity\\_info](#)

### 30.221.1 Variable Documentation

#### 30.221.1.1 var class\_go\_1\_1\_complexity\_info

##### Initial value:

```
=
[
 ["ComplexityInfo", "class_go_1_1_complexity_info.html#a546896f009ccc6587579780d0ea46d11", null],
 ["~ComplexityInfo", "class_go_1_1_complexity_info.html#afc5aaaa70d3053416053db0c11cc1e5e", null],
 ["getBoxOverlap", "class_go_1_1_complexity_info.html#a92c1d24d585a6eb1f8d75ffc13af649", null],
 ["getConeOverlap", "class_go_1_1_complexity_info.html#aaaff36c97e537bb0fd2781e926af5650e", null],
 ["getNmbIntpts", "class_go_1_1_complexity_info.html#ac2f8f08e99a6bbd33b1e75f5aa79dbb7", null],
 ["getNmbSingpts", "class_go_1_1_complexity_info.html#a56460cacfaa2561db720fc694f853886", null],
 ["isComplex", "class_go_1_1_complexity_info.html#a935f961c24d415a48870ab9092bf72c", null],
 ["setBoxOverlap", "class_go_1_1_complexity_info.html#ab6e28172185f56d7ada4ec5144b7b197", null],
 ["setComplex", "class_go_1_1_complexity_info.html#a276b17d59bf7fa23fd7e1cblleaf6e72b", null],
 ["setConeOverlap", "class_go_1_1_complexity_info.html#aa9669a1e41f8d620972bba723447abd3", null],
 ["setNmbIntpts", "class_go_1_1_complexity_info.html#a398d5a6de6d0feecb6e5449892cbe02f", null],
 ["setNmbSingpts", "class_go_1_1_complexity_info.html#acdeba737674463ea94305240baefb9a", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_complexity\_info.js.

## 30.222 doc/html/class\_go\_1\_1\_composite\_box.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_composite\\_box](#)

#### 30.222.1 Variable Documentation

##### 30.222.1.1 var class\_go\_1\_1\_composite\_box

###### Initial value:

```
=
[
 ["CompositeBox", "class_go_1_1_composite_box.html#a0cf0d75e19e0799e72078877ee6e784d", null],
 ["CompositeBox", "class_go_1_1_composite_box.html#a9c3fc6c4edd357ed588e137e24bd0b45", null],
 ["~CompositeBox", "class_go_1_1_composite_box.html#a48d89324cb37b487064726809444b68a", null],
 ["containsBox", "class_go_1_1_composite_box.html#a901e56f7cfd24d904ef56eb56bda424", null],
 ["containsPoint", "class_go_1_1_composite_box.html#ab4ee797c2c699406b1cb618cd6098786", null],
 ["dimension", "class_go_1_1_composite_box.html#ac3986a965027673dcbdc60fe0d390284", null],
 ["edge", "class_go_1_1_composite_box.html#a5f582420b4561b6bdb20f3c5d64d474e", null],
 ["getOverlap", "class_go_1_1_composite_box.html#a77456613f0380fce3be1e9e7ec8928e2", null],
 ["high", "class_go_1_1_composite_box.html#a148af87030c6be9e045dbfe797ebda57", null],
 ["inner", "class_go_1_1_composite_box.html#a7c1c3449658cbc9d5a8e0ad2007f64fa", null],
 ["low", "class_go_1_1_composite_box.html#a5299b662ecfadf1381a64c6e0d682b4", null],
 ["overlaps", "class_go_1_1_composite_box.html#aba53e7a48998934006c5951635ec4cc5", null],
 ["read", "class_go_1_1_composite_box.html#a963b13d6cb9266e8c9c0c05117869b85", null],
 ["setFromArray", "class_go_1_1_composite_box.html#af23318fa8aff97abd02827a31ca32c0e", null],
 ["setFromPoints", "class_go_1_1_composite_box.html#abf974489a4339bc1c5bf9e7bb0e4283e", null],
 ["write", "class_go_1_1_composite_box.html#abc9e743afa6168a129d765dc1b44014e", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_composite\_box.js.

## 30.223 doc/html/class\_go\_1\_1\_composite\_curve.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_composite\\_curve](#)

#### 30.223.1 Variable Documentation

##### 30.223.1.1 var class\_go\_1\_1\_composite\_curve

Definition at line 1 of file class\_go\_1\_1\_composite\_curve.js.

## 30.224 doc/html/class\_go\_1\_1\_composite\_model.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_composite\\_model](#)

## 30.224.1 Variable Documentation

### 30.224.1.1 var class\_go\_1\_1\_composite\_model

#### Initial value:

```
=
[
 ["CompositeModel", "class_go_1_1_composite_model.html#af17f66cfd0a5a4db92099ad589e02968", null],
 ["~CompositeModel", "class_go_1_1_composite_model.html#abaced49ad71c6a333c166b18f31bd2f6", null],
 ["asSurfaceModel", "class_go_1_1_composite_model.html#acabe5816a92cc859d1ac2fb803124b6c", null],
 ["boundingBox", "class_go_1_1_composite_model.html#a0242bdd0f68b860c74dcb239633d9593", null],
 ["boundingBox", "class_go_1_1_composite_model.html#ae62928c0e2e4d2022c928c4997f4ce9b", null],
 ["boxExtreme", "class_go_1_1_composite_model.html#a1f02915e238a12b98c9ac5238c728143", null],
 ["boxVecDist", "class_go_1_1_composite_model.html#ad53038595bd166d9f84b22482da71162", null],
 ["clone", "class_go_1_1_composite_model.html#a41e07072a1e553d32fe1b369bfbefb8", null],
 ["closestPoint", "class_go_1_1_composite_model.html#ac0ac71ab6cce360976f9a63a65bb4347", null],
 ["curvature", "class_go_1_1_composite_model.html#a4378e0cf48822b06e319632abb0d02a9", null],
 ["evaluate", "class_go_1_1_composite_model.html#aedb3e9516dbc21a1f5e801bf96f2d695", null],
 ["evaluate", "class_go_1_1_composite_model.html#a3d6b55740dc61a6591a0d15bafd56e1a", null],
 ["extremalPoint", "class_go_1_1_composite_model.html#a303f8f928e97078ff106bf57f07342f9", null],
 ["getTolerances", "class_go_1_1_composite_model.html#ae513d50c28ccf26d380b23097958e7e8", null],
 ["intersect", "class_go_1_1_composite_model.html#ad223057621d05e93be29a1543e726968", null],
 ["intersect_plane", "class_go_1_1_composite_model.html#a4985f0641d4a69a36df2c7537f4982c2", null],
 ["isDegenerate", "class_go_1_1_composite_model.html#aed3744a2505c6203380447e312e21bb9", null],
 ["nmbEntities", "class_go_1_1_composite_model.html#a700263c1ca355d07c92fb01b174f1d55", null],
 ["setTolerances", "class_go_1_1_composite_model.html#ad6db5bbd1ea0bce203fa19f4d927b6c2", null],
 ["tessellate", "class_go_1_1_composite_model.html#a55d5534b47d5a0cbabfa530c201adb26", null],
 ["tessellate", "class_go_1_1_composite_model.html#ad5f1cfa14f408e97667346ae312da36e", null],
 ["tessellate", "class_go_1_1_composite_model.html#a1546764e88cba9c13fa2bea819e9fe2c", null],
 ["tessellatedCtrPolygon", "class_go_1_1_composite_model.html#a60e568eb9b8531f2bc5de60acb9d8fd7", null]
,
 ["turn", "class_go_1_1_composite_model.html#a70199a40a9086025bc26b62a3fc25930", null],
 ["turn", "class_go_1_1_composite_model.html#a27bdfc47e9f619b8e3d3279e0ff96f7e", null],
 ["closest_idx", "class_go_1_1_composite_model.html#aedc91ac680300548a1b8bfbea8d3a293", null],
 ["toptol", "class_go_1_1_composite_model.html#a0df2b1e6f0db9214268ee15edafb0968", null]
]

```

Definition at line 1 of file class\_go\_1\_1\_composite\_model.js.

## 30.225 doc/html/class\_go\_1\_1\_composite\_model\_factory.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_composite\\_model\\_factory](#)

## 30.225.1 Variable Documentation

### 30.225.1.1 var class\_go\_1\_1\_composite\_model\_factory

#### Initial value:

```
=
[
 ["CompositeModelFactory", "class_go_1_1_composite_model_factory.html#a3aaa017ed6307c7254588da6af016a88", null],
 ["~CompositeModelFactory", "class_go_1_1_composite_model_factory.html#a879a68b4de429af5a9c2ecf31b661cd0", null],
 ["createCircularArc", "class_go_1_1_composite_model_factory.html#a9b0fe75e31b39562ff27464f60aa4ed6", null],
 ["createEllipticArc", "class_go_1_1_composite_model_factory.html#a545a8424f9ffa04c71c0614bc486f29f", null],
 ["createEmpty", "class_go_1_1_composite_model_factory.html#a9f249bc8fe3290631d3331e5ac9278ff", null],
 ["createFromBox", "class_go_1_1_composite_model_factory.html#a189f25674a6cbe2c9f9ee24136060dec", null]
]

```

```
],
 ["createFromCylinder", "class_go_1_1_composite_model_factory.html#a6cffd625f5fec340d9c294ff9277efaf",
 null],
 ["createFromG2", "class_go_1_1_composite_model_factory.html#a64ded85681130a7de835f7bf04a1a0e4", null]
],
 ["createFromIges", "class_go_1_1_composite_model_factory.html#a1dd4377c13f7feb5432de29f9ad87762", null
],
 ["createFromSis1", "class_go_1_1_composite_model_factory.html#a4fef1e2bed6e94b17fd6a45244874ec9", null
],
 ["createFromSis1", "class_go_1_1_composite_model_factory.html#a5af40ba529889e3b41cef1e1dd9ab0e6", null
],
 ["createFromSphere", "class_go_1_1_composite_model_factory.html#a4c0fel1562cadb2502313b36813c631",
 null],
 ["createFromSphere", "class_go_1_1_composite_model_factory.html#a2b704d2b72f52b9e8cdb04ad1b4c3343",
 null],
 ["createLineSegment", "class_go_1_1_composite_model_factory.html#a8fa82d8594aba36e654f3eb7ffe788b6",
 null],
 ["getModelsFromG2", "class_go_1_1_composite_model_factory.html#a6819151231d96ffb37a53af37b5e1eef",
 null],
 ["getModelsFromIges", "class_go_1_1_composite_model_factory.html#a79de7347a9355c68baldab4a73fff155",
 null],
 ["interpolateCurves", "class_go_1_1_composite_model_factory.html#a593b4d502d5617e8c8468c3b0503ce59",
 null],
 ["interpolateCurves", "class_go_1_1_composite_model_factory.html#a882512419a42ebdfb6aa7d2ebef6c8a1",
 null],
 ["interpolateCurves2", "class_go_1_1_composite_model_factory.html#abba6990c54cb62638e7b2df854e4bfce",
 null],
 ["interpolateCurves2", "class_go_1_1_composite_model_factory.html#ac0426ece72f633af3290417677e20587",
 null]
]
]
```

Definition at line 1 of file class\_go\_1\_1\_composite\_model\_factory.js.

## 30.226 doc/html/class\_go\_1\_1\_composite\_model\_file\_handler.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_composite\\_model\\_file\\_handler](#)

#### 30.226.1 Variable Documentation

##### 30.226.1.1 var class\_go\_1\_1\_composite\_model\_file\_handler

Definition at line 1 of file class\_go\_1\_1\_composite\_model\_file\_handler.js.

## 30.227 doc/html/class\_go\_1\_1\_composite\_surface.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_composite\\_surface](#)

#### 30.227.1 Variable Documentation

##### 30.227.1.1 var class\_go\_1\_1\_composite\_surface

Definition at line 1 of file class\_go\_1\_1\_composite\_surface.js.

## 30.228 doc/html/class\_go\_1\_1\_concrete\_creator.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_concrete\\_creator](#)

### 30.228.1 Variable Documentation

#### 30.228.1.1 var class\_go\_1\_1\_concrete\_creator

#### Initial value:

```
=
[
 ["ConcreteCreator", "class_go_1_1_concrete_creator.html#a936e9ad83889890e7221222b94dde424", null],
 ["create", "class_go_1_1_concrete_creator.html#aac31f03c76a12c9f467bc8f1b59426f3", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_concrete\_creator.js.

## 30.229 doc/html/class\_go\_1\_1\_cone.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_cone](#)

### 30.229.1 Variable Documentation

#### 30.229.1.1 var class\_go\_1\_1\_cone

Definition at line 1 of file class\_go\_1\_1\_cone.js.

## 30.230 doc/html/class\_go\_1\_1\_cone\_volume.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_cone\\_volume](#)



## 30.230.1 Variable Documentation

### 30.230.1.1 var class\_go\_1\_1\_cone\_volume

#### Initial value:

```
=
[
 ["ConeVolume", "class_go_1_1_cone_volume.html#a2b5954de4f3f2d9176bd38655b5c366b", null],
 ["ConeVolume", "class_go_1_1_cone_volume.html#aad5901e2b8208a751cc99400f9683e9c", null],
 ["~ConeVolume", "class_go_1_1_cone_volume.html#a7521e6873042f301b8c289b7134d3e39", null],
 ["boundingBox", "class_go_1_1_cone_volume.html#a5820226622a18e35721d4345bb3b783a", null],
 ["clone", "class_go_1_1_cone_volume.html#a42b1059bcbd0c8a99196f7fda61474d0", null],
 ["closestPoint", "class_go_1_1_cone_volume.html#ab62398029d1c839b1651cf67d5b2a217", null],
 ["dimension", "class_go_1_1_cone_volume.html#ac0e0bd2cb65b7cc24c84ccfa82e85bc1", null],
 ["geometryVolume", "class_go_1_1_cone_volume.html#a44979ecb4ff07ae47f8cab0fa900ae97", null],
 ["getAllBoundarySurfaces", "class_go_1_1_cone_volume.html#ac490d5d0a5e39f716131f09ebc7916a0", null],
 ["instanceType", "class_go_1_1_cone_volume.html#a856d622b8c69c226b45c3811729d5491", null],
 ["nextSegmentVal", "class_go_1_1_cone_volume.html#acb6e3281f5c693cec31dc9fd22d21e2c", null],
 ["parameterSpan", "class_go_1_1_cone_volume.html#a634c93b4e3d7488ab93e1e2d4b7a5453", null],
 ["point", "class_go_1_1_cone_volume.html#ae491fe3f143fad2b2f2e4a4868034bc", null],
 ["point", "class_go_1_1_cone_volume.html#ace9dd15a4afd8be6bd1637ac3d9484b8", null],
 ["read", "class_go_1_1_cone_volume.html#aace95efa8527024bd752680e37e92a9c", null],
 ["reverseParameterDirection", "class_go_1_1_cone_volume.html#a62f4032a05b45bbde40ec78a2c03db84", null],
],
 ["setParameters", "class_go_1_1_cone_volume.html#a3a9076f2baae429c335f1ed05e1d31d1", null],
 ["swapParameterDirection", "class_go_1_1_cone_volume.html#a53ae64faa028f9b5db77388cf0e3cc28", null],
 ["tangentCone", "class_go_1_1_cone_volume.html#a1f7d1a81eae6e98a3422a2320aa932bd", null],
 ["translate", "class_go_1_1_cone_volume.html#a4a8584b5213744c022f37c8602d98711", null],
 ["useCentreDegen", "class_go_1_1_cone_volume.html#a2991ac809f94ab9b98b446f9b208a724", null],
 ["useCornerDegen", "class_go_1_1_cone_volume.html#a49bcd6f881e1afa5153c8d8aa7910a2f", null],
 ["write", "class_go_1_1_cone_volume.html#ad8a963e0086079894fcb89e6d6eb86c7", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_cone\_volume.js.

## 30.231 doc/html/class\_go\_1\_1\_connection\_funcutor.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_connection\\_funcutor](#)

### 30.231.1 Variable Documentation

#### 30.231.1.1 var class\_go\_1\_1\_connection\_funcutor

#### Initial value:

```
=
[
 ["ConnectionFuncutor", "class_go_1_1_connection_funcutor.html#a5e17a025c24007f1a1cb1edcc8c7a307", null],
 ["operator()", "class_go_1_1_connection_funcutor.html#ad3b69e5941c88219636efaacbf619352", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_connection\_funcutor.js.

## 30.232 doc/html/class\_go\_1\_1\_coordinate\_system.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_coordinate\\_system](#)

### 30.232.1 Variable Documentation

#### 30.232.1.1 var class\_go\_1\_1\_coordinate\_system

#### Initial value:

```
=
[
 ["Matrix", "class_go_1_1_coordinate_system.html#a01318b17e30343e7b38c6dd129c5ae57", null],
 ["Vector", "class_go_1_1_coordinate_system.html#a80aab6cc63681fb2713cfb772826025a", null],
 ["CoordinateSystem", "class_go_1_1_coordinate_system.html#ae699f4caf33cb766c550f9818eb8a752", null],
 ["CoordinateSystem", "class_go_1_1_coordinate_system.html#a18bae9c10686eb10a81d512c213a8e43", null],
 ["operator*", "class_go_1_1_coordinate_system.html#a6d2df13a3895240ba29d9fd640c00aea", null],
 ["operator*", "class_go_1_1_coordinate_system.html#ae424bdc31dfc95a8be41120feefc4ee1", null],
 ["rot", "class_go_1_1_coordinate_system.html#a3b0db4d8c6c0a54c0062be5e0290d3d9", null],
 ["rot", "class_go_1_1_coordinate_system.html#ab87a405f0bcbeea108245be504a9089d", null],
 ["tr", "class_go_1_1_coordinate_system.html#a3ed7c4908621c839632a4445b108694e", null],
 ["tr", "class_go_1_1_coordinate_system.html#a96fc5127abeec65820380667fa3ebf55", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_coordinate\_system.js.

## 30.233 doc/html/class\_go\_1\_1\_creator.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_creator](#)

### 30.233.1 Variable Documentation

#### 30.233.1.1 var class\_go\_1\_1\_creator

#### Initial value:

```
=
[
 ["~Creator", "class_go_1_1_creator.html#a8de48cc9ffdab283b7ec4298353599ee", null],
 ["create", "class_go_1_1_creator.html#af22f6695f55c6ef0f002fe9aa781f8c", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_creator.js.

## 30.234 doc/html/class\_go\_1\_1\_cross\_tan\_off\_dist.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_cross\\_tan\\_off\\_dist](#)

### 30.234.1 Variable Documentation

#### 30.234.1.1 var class\_go\_1\_1\_cross\_tan\_off\_dist

##### Initial value:

```
=
[
 ["CrossTanOffDist", "class_go_1_1_cross_tan_off_dist.html#a97eae000ee5999c76763b9a92da85d2e", null],
 ["~CrossTanOffDist", "class_go_1_1_cross_tan_off_dist.html#ac71814d2ee4a5e743f3e0ac2715d17ad", null],
 ["approximationOK", "class_go_1_1_cross_tan_off_dist.html#a4a446ce876c6d3917b362ab28e87c5ec", null],
 ["dim", "class_go_1_1_cross_tan_off_dist.html#aaecb62e918fafc868c512fbc1a63624", null],
 ["end", "class_go_1_1_cross_tan_off_dist.html#ad797f65a0e94e6fb2dc328a2795ff712", null],
 ["eval", "class_go_1_1_cross_tan_off_dist.html#a093f223454d1e0bcae4631d52dca7afe", null],
 ["eval", "class_go_1_1_cross_tan_off_dist.html#a69a4b1d5cca9da187739f607ee4ac765", null],
 ["start", "class_go_1_1_cross_tan_off_dist.html#a89f50dadc9bf0f0456c68929e47b32d4", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_cross\_tan\_off\_dist.js.

## 30.235 doc/html/class\_go\_1\_1\_cross\_tangent\_offset.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_cross\\_tangent\\_offset](#)

### 30.235.1 Variable Documentation

#### 30.235.1.1 var class\_go\_1\_1\_cross\_tangent\_offset

##### Initial value:

```
=
[
 ["CrossTangentOffset", "class_go_1_1_cross_tangent_offset.html#acc9f9197651c21dc5a2514846da2ce92", null],
 ["~CrossTangentOffset", "class_go_1_1_cross_tangent_offset.html#a644a48ebbe193658e2efba294e7b5079", null],
 ["approximationOK", "class_go_1_1_cross_tangent_offset.html#a16ea910d7c4cceb1fa9115d7a4d39659", null],
 ["dim", "class_go_1_1_cross_tangent_offset.html#a96f44ae924935fa5aebcbce207f84f66", null],
 ["end", "class_go_1_1_cross_tangent_offset.html#a804d51d488fdc1db2f7878c127e2297a", null],
 ["eval", "class_go_1_1_cross_tangent_offset.html#abd8823abd7a5dddfb98dda7830f57da5", null],
 ["eval", "class_go_1_1_cross_tangent_offset.html#a3ab9c438427bbee76e368b85988d4145", null],
 ["start", "class_go_1_1_cross_tangent_offset.html#a318c68ba55b0bd3c632076aa9031133d", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_cross\_tangent\_offset.js.

## 30.236 doc/html/class\_go\_1\_1\_crosses\_value.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_crosses\\_value](#)

### 30.236.1 Variable Documentation

#### 30.236.1.1 var class\_go\_1\_1\_crosses\_value

##### Initial value:

```
=
[
 ["argument_type", "class_go_1_1_crosses_value.html#aae9be45a2093dec324dea50127163874", null],
 ["result_type", "class_go_1_1_crosses_value.html#ab2043aeaba5d6bb69f9264dd2facae3b", null],
 ["CrossesValue", "class_go_1_1_crosses_value.html#ad6353deae4d6f6d7dff11d278b9cd7a", null],
 ["operator()", "class_go_1_1_crosses_value.html#aa84ae9ea9111d2e037ef5ded78f0677d", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_crosses\_value.js.

## 30.237 doc/html/class\_go\_1\_1\_curve\_bounded\_domain.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_curve\\_bounded\\_domain](#)

### 30.237.1 Variable Documentation

#### 30.237.1.1 var class\_go\_1\_1\_curve\_bounded\_domain

##### Initial value:

```
=
[
 ["CurveBoundedDomain", "class_go_1_1_curve_bounded_domain.html#a18a47f463be561c4b9159f9a48b75810", null],
 ["CurveBoundedDomain", "class_go_1_1_curve_bounded_domain.html#a64b7a200c859eed5197f0690322577f7", null],
 ["CurveBoundedDomain", "class_go_1_1_curve_bounded_domain.html#af9d5bb9de6ebc645029186e7d60e8639", null],
 ["~CurveBoundedDomain", "class_go_1_1_curve_bounded_domain.html#a8fb6e3de5f72613521288c9156ebd6e", null],
 ["clipWithDomain", "class_go_1_1_curve_bounded_domain.html#a9ebcd31c5c62e1dd7486e49e11ab6df6", null],
 ["closestInDomain", "class_go_1_1_curve_bounded_domain.html#ad7f4566f999a8df8066dc341aded5ddf", null],
 ["closestOnBoundary", "class_go_1_1_curve_bounded_domain.html#a8e5c2a8127ccd5bd9c336e1738c73eaf", null],
 ["containingDomain", "class_go_1_1_curve_bounded_domain.html#ab953d88367589393699deb49b8edbc07", null],
 ["findPcurveInsideSegments", "class_go_1_1_curve_bounded_domain.html#ab756191c1eaab9f8c85e19f9754b7122", null],
 ["findPcurveInsideSegments", "class_go_1_1_curve_bounded_domain.html#a80d23bc9ec923c71f1e66c4300bf798b", null],
 ["getInsideIntervals", "class_go_1_1_curve_bounded_domain.html#a5f7b7785fb897dc16ad33aef7212160c", null],
 ["getInternalPoint", "class_go_1_1_curve_bounded_domain.html#a48bbddab7fed9a716981fd99c02e8e53", null],
 ["isInDomain", "class_go_1_1_curve_bounded_domain.html#aa850241d4083c6dd9e0f25f44e801478", null],
 ["isInDomain2", "class_go_1_1_curve_bounded_domain.html#ad0b5791ca4f4efb2f4ba278a816ac8ed", null],
 ["isOnBoundary", "class_go_1_1_curve_bounded_domain.html#aaa7f183d4288ba4e3681479b3c628bd8", null],
 ["isOnCorner", "class_go_1_1_curve_bounded_domain.html#a6a47ab4ba91107a95bab17d00b2d7685", null],
 ["positionPointInDomain", "class_go_1_1_curve_bounded_domain.html#ac8e0131368dd437a9ea82ee5a704ef16", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_curve\_bounded\_domain.js.

## 30.238 doc/html/class\_go\_1\_1\_curve\_loop.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_curve\\_loop](#)

### 30.238.1 Variable Documentation

#### 30.238.1.1 var class\_go\_1\_1\_curve\_loop

#### Initial value:

```
=
[
 ["CurveLoop", "class_go_1_1_curve_loop.html#a1e1cb95e731e6406e2e85ef158474f74", null],
 ["CurveLoop", "class_go_1_1_curve_loop.html#ade5272d85ed158bc981182df67c4aef4", null],
 ["~CurveLoop", "class_go_1_1_curve_loop.html#a9b6988be13ec186c892a8bed14588ald", null],
 ["begin", "class_go_1_1_curve_loop.html#a0b5be6e67a8d5ee06d3cd7aad64d1ede", null],
 ["begin", "class_go_1_1_curve_loop.html#a27dcb63156cda7fce37fc181e40a9cf4", null],
 ["closestParPoint", "class_go_1_1_curve_loop.html#ab31598dc9197a8b92b87474df3158290", null],
 ["closestPoint", "class_go_1_1_curve_loop.html#a14cb0571736ec8c1740a426c0e53dc23", null],
 ["end", "class_go_1_1_curve_loop.html#a9245355ff03ceed08ead4dfd6484e429", null],
 ["end", "class_go_1_1_curve_loop.html#a6300b8a52488d0dbc9b1d31e7d34able", null],
 ["fixInvalidLoop", "class_go_1_1_curve_loop.html#a4f1db82c3d9aeaac3654d7359a99b0d2", null],
 ["getCorners", "class_go_1_1_curve_loop.html#aba21e7b1abe68382aa7a4365b0a46d65", null],
 ["getCurves", "class_go_1_1_curve_loop.html#a7959761401bcefe5f8b13b03f55d2e82", null],
 ["getMaxCurveDist", "class_go_1_1_curve_loop.html#ae798f4d28424200761a4580538fb24fe", null],
 ["getSmoothCurves", "class_go_1_1_curve_loop.html#aa6c04d6c0a64612db860e66cf457b555", null],
 ["getSpaceEpsilon", "class_go_1_1_curve_loop.html#a5b36e02e53f3d3966331c379aaafc0744", null],
 ["isValid", "class_go_1_1_curve_loop.html#ada94a24a980ee03b6c648b63f4454fee", null],
 ["operator[]", "class_go_1_1_curve_loop.html#af6d96f046a265b3ade9b2248ab9d2616", null],
 ["setCurves", "class_go_1_1_curve_loop.html#a41bd1856152a1c9ae270a8aeb5c6e8d3", null],
 ["setSpaceEpsilon", "class_go_1_1_curve_loop.html#a5971f2d0ed852cd5771fb1c35eb5eab2", null],
 ["simplify", "class_go_1_1_curve_loop.html#a5aelf25efd34a9772c7ealed01f34790", null],
 ["size", "class_go_1_1_curve_loop.html#a29cc84f53283aaee037f169b19f967d9", null],
 ["swap", "class_go_1_1_curve_loop.html#a6e671156a7602ac1cd569251ac74f379", null],
 ["turnOrientation", "class_go_1_1_curve_loop.html#a3595b52d2aebfce9feb2339ac77f6fbb", null]
]
```

Definition at line 1 of file `class_go_1_1_curve_loop.js`.

## 30.239 doc/html/class\_go\_1\_1\_curve\_model.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_curve\\_model](#)

### 30.239.1 Variable Documentation

#### 30.239.1.1 var class\_go\_1\_1\_curve\_model

#### Initial value:

```

=
[
 ["CurveModel", "class_go_1_1_curve_model.html#a28e80476530ffdf76de33d447d385f", null],
 ["~CurveModel", "class_go_1_1_curve_model.html#a06f1cb0cf8090dafc4cf7dc1caf7192a", null],
 ["boundingBox", "class_go_1_1_curve_model.html#a631d5ab539bede1e7db628a82f24b867", null],
 ["boundingBox", "class_go_1_1_curve_model.html#a6291f57225814717ff7ba2a027d77340", null],
 ["clone", "class_go_1_1_curve_model.html#a09a81d7d65beb3ad5c6e4c5b3c4da2d7", null],
 ["closestPoint", "class_go_1_1_curve_model.html#ac66d95156cddfb8bd83bc3ac0329977e", null],
 ["curvature", "class_go_1_1_curve_model.html#a93c57aac70712434cc9b9853795b7320", null],
 ["evaluate", "class_go_1_1_curve_model.html#aec0d0ac671d2e1ad3a3f178c125e8611", null],
 ["evaluate", "class_go_1_1_curve_model.html#ac71fc300c8b0038fe27441bd16c07ffa", null],
 ["extremalPoint", "class_go_1_1_curve_model.html#af4843e0bf615f48d48aba163507a0903", null],
 ["fetchCompositeCurves", "class_go_1_1_curve_model.html#af227f8d4d720f672925567f90eea029", null],
 ["getCurve", "class_go_1_1_curve_model.html#a777dfac8daf0721a5ad5aaf465480a6d", null],
 ["getIndex", "class_go_1_1_curve_model.html#a675ff0222ccd405023bf2350dea0dae", null],
 ["intersect", "class_go_1_1_curve_model.html#a0bc4b7953eb911abc9335dfae22a2e", null],
 ["intersect_plane", "class_go_1_1_curve_model.html#a94c6b368e929c11e40dcde6eab238d0a", null],
 ["isDegenerate", "class_go_1_1_curve_model.html#a4a973653562bc03e0e6e2c13cd875082", null],
 ["nmbEntities", "class_go_1_1_curve_model.html#afb2d4709b0f339aad55d603bef6eaa0e", null],
 ["tessellate", "class_go_1_1_curve_model.html#a5aa6f99880fd5126485b39a59932a30f", null],
 ["tessellate", "class_go_1_1_curve_model.html#a2cc71ae48a6d1024c8779c1fde08eb80", null],
 ["tessellate", "class_go_1_1_curve_model.html#a13361c5bf5d010aa4da56ae5569070e0", null],
 ["tessellatedCtrPolygon", "class_go_1_1_curve_model.html#a0bc9f19a78ec7aaac8b655ea7e25bc7b", null],
 ["turn", "class_go_1_1_curve_model.html#a2487ffc22e3608c85084d013f7268f47", null],
 ["turn", "class_go_1_1_curve_model.html#a3d9c2dfa01cde59f734767c05bc736e0", null]
]

```

Definition at line 1 of file `class_go_1_1_curve_model.js`.

## 30.240 doc/html/class\_go\_1\_1\_curve\_on\_surface.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_curve\\_on\\_surface](#)

#### 30.240.1 Variable Documentation

##### 30.240.1.1 var class\_go\_1\_1\_curve\_on\_surface

Definition at line 1 of file `class_go_1_1_curve_on_surface.js`.

## 30.241 doc/html/class\_go\_1\_1\_curve\_on\_volume.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_curve\\_on\\_volume](#)

#### 30.241.1 Variable Documentation

##### 30.241.1.1 var class\_go\_1\_1\_curve\_on\_volume

Definition at line 1 of file `class_go_1_1_curve_on_volume.js`.

## 30.242 doc/html/class\_go\_1\_1\_curve\_tesselator.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_curve\\_tesselator](#)

#### 30.242.1 Variable Documentation

##### 30.242.1.1 var class\_go\_1\_1\_curve\_tesselator

#### Initial value:

```
=
[
 ["CurveTesselator", "class_go_1_1_curve_tesselator.html#aefe53d0a6b1795aefd01855793a80ffb", null],
 ["~CurveTesselator", "class_go_1_1_curve_tesselator.html#a56a3991ff2a75c1c88ddad4faac8ec5d", null],
 ["changeRes", "class_go_1_1_curve_tesselator.html#a5311aeaeaa8d1c30a583c6b2b53a8624", null],
 ["getMesh", "class_go_1_1_curve_tesselator.html#a4754381346d940ccce5d1dbd1d80fdb8", null],
 ["getRes", "class_go_1_1_curve_tesselator.html#a1bd2bb29c3dbb95c0cbf1f82281eed2a", null],
 ["tesselate", "class_go_1_1_curve_tesselator.html#a2903e0b478f9c9bc52de4b78a2e58554", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_curve\_tesselator.js.

## 30.243 doc/html/class\_go\_1\_1\_cv\_cv\_intersector.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_cv\\_cv\\_intersector](#)

#### 30.243.1 Variable Documentation

##### 30.243.1.1 var class\_go\_1\_1\_cv\_cv\_intersector

#### Initial value:

```
=
[
 ["CvCvIntersector", "class_go_1_1_cv_cv_intersector.html#a537c5e9bc036e51d6cd9ef73868d1866", null],
 ["CvCvIntersector", "class_go_1_1_cv_cv_intersector.html#ae8389b331a736a88309a72d3d17ac79e", null],
 ["~CvCvIntersector", "class_go_1_1_cv_cv_intersector.html#a9b87b3471e95027f9160919f73ffff52", null],
 ["checkCoincidence", "class_go_1_1_cv_cv_intersector.html#ac9aae4b2f7d15af6ead9ba20277e00ce", null],
 ["doSubdivide", "class_go_1_1_cv_cv_intersector.html#adc99533793481a044c9741c7e9eb3169", null],
 ["foundIntersectionNearBoundary", "class_go_1_1_cv_cv_intersector.html#a1ee50a74da971d77a3544c1e9225abc8", null],
 ["linearCase", "class_go_1_1_cv_cv_intersector.html#af985fe190a31248f68f04db1fec0663", null],
 ["lowerOrderIntersector", "class_go_1_1_cv_cv_intersector.html#aa7afb0eb132176cf912444c1c6f4fa87", null],
 ["microCase", "class_go_1_1_cv_cv_intersector.html#af3417373b5669763a1f9c8ec9bb01a0d", null],
 ["numParams", "class_go_1_1_cv_cv_intersector.html#afb42604cac3999a8fb03f93a93b06405", null],
 ["performRotatedBoxTest", "class_go_1_1_cv_cv_intersector.html#a608d160b6dd891589dfdc64dc9bc9059", null],
 ["repairIntersections", "class_go_1_1_cv_cv_intersector.html#a8a9f48935289182621b7b8d68451bf92", null],
 ["simpleCase2", "class_go_1_1_cv_cv_intersector.html#aab2bbbed701c08c25d6a5b66f795ee803", null],
 ["updateIntersections", "class_go_1_1_cv_cv_intersector.html#a48103c3ad37f68052730b66d9c5ca7d4", null],
 ["writeOut", "class_go_1_1_cv_cv_intersector.html#a4100e3e3e0b6dc0de319965182234bcd", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_cv\_cv\_intersector.js.

## 30.244 doc/html/class\_go\_1\_1\_cv\_pt\_intersector.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_cv\\_pt\\_intersector](#)

#### 30.244.1 Variable Documentation

##### 30.244.1.1 var class\_go\_1\_1\_cv\_pt\_intersector

#### Initial value:

```
=
[
 ["CvPtIntersector", "class_go_1_1_cv_pt_intersector.html#afeedc2980f108a85ccae64927e647a8", null],
 ["CvPtIntersector", "class_go_1_1_cv_pt_intersector.html#a2c2d2af25116cb8064076dc17a797c19", null],
 ["~CvPtIntersector", "class_go_1_1_cv_pt_intersector.html#aef366d280b059044a6eae987957f4c3c", null],
 ["checkCoincidence", "class_go_1_1_cv_pt_intersector.html#acc052cc4ca301d2d32c995c3df6fa7cf", null],
 ["doSubdivide", "class_go_1_1_cv_pt_intersector.html#a7513f37903ddcf8cdce38d442340b70e", null],
 ["linearCase", "class_go_1_1_cv_pt_intersector.html#afb4468ca7e9c7a779afdf08670fd6ae4", null],
 ["lowerOrderIntersector", "class_go_1_1_cv_pt_intersector.html#a430ee1375f35b0fb3f7affd0267e87bd",
 null],
 ["microCase", "class_go_1_1_cv_pt_intersector.html#a14aa72005ed4f9500c4006a646d2c958", null],
 ["numParams", "class_go_1_1_cv_pt_intersector.html#aa370d43e0769f4364cdce0522ac42710", null],
 ["repairIntersections", "class_go_1_1_cv_pt_intersector.html#a5edd96a2492f5b3753a35021bdaf1db6", null
],
 ["updateIntersections", "class_go_1_1_cv_pt_intersector.html#a877f4b46c1493d3cb63a4234fa86390c", null
]
]
```

Definition at line 1 of file class\_go\_1\_1\_cv\_pt\_intersector.js.

## 30.245 doc/html/class\_go\_1\_1\_cylinder.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_cylinder](#)

#### 30.245.1 Variable Documentation

##### 30.245.1.1 var class\_go\_1\_1\_cylinder

Definition at line 1 of file class\_go\_1\_1\_cylinder.js.

## 30.246 doc/html/class\_go\_1\_1\_cylinder\_int.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_cylinder\\_int](#)



## 30.246.1 Variable Documentation

### 30.246.1.1 var class\_go\_1\_1\_cylinder\_int

#### Initial value:

```
=
[
 ["CylinderInt", "class_go_1_1_cylinder_int.html#a2578453e0bbb9b5981ed8520be74eef4", null],
 ["CylinderInt", "class_go_1_1_cylinder_int.html#a9067b1a615e16beb0781fd491f2393f0", null],
 ["~CylinderInt", "class_go_1_1_cylinder_int.html#abd97bcc1dbc2d672df0938af7bc80f9b", null],
 ["ax_dir", "class_go_1_1_cylinder_int.html#ab5b9d1555a5cd5ff0fb313dc1318a20e", null],
 ["ax_pt", "class_go_1_1_cylinder_int.html#ae4baecc0dcf30c7a05f330363e59d0e5", null],
 ["radius", "class_go_1_1_cylinder_int.html#a4493b91edd175e4166b7ab07a9b35c2c", null],
 ["read", "class_go_1_1_cylinder_int.html#ab9916e69cd910d06ad5f05b9f780c8fe", null],
 ["surface", "class_go_1_1_cylinder_int.html#abe689862760d5baabdd7b0af13165e90", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_cylinder\_int.js.

## 30.247 doc/html/class\_go\_1\_1\_cylinder\_volume.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_cylinder\\_volume](#)

## 30.247.1 Variable Documentation

### 30.247.1.1 var class\_go\_1\_1\_cylinder\_volume

#### Initial value:

```
=
[
 ["CylinderVolume", "class_go_1_1_cylinder_volume.html#a33ba420957082fdb531a2fff19a610d6", null],
 ["CylinderVolume", "class_go_1_1_cylinder_volume.html#a067ee920433858475d5d9f144be8564b", null],
 ["~CylinderVolume", "class_go_1_1_cylinder_volume.html#a9943701c31b3abe8e66c06e379be6074", null],
 ["boundingBox", "class_go_1_1_cylinder_volume.html#a4dd92d7237465874199b360c3b3b4542", null],
 ["clone", "class_go_1_1_cylinder_volume.html#a9a95da3d1968325a413adb492dc7c451", null],
 ["closestPoint", "class_go_1_1_cylinder_volume.html#a53a9894772b555d76e84e903735f40d4", null],
 ["dimension", "class_go_1_1_cylinder_volume.html#a1ff46706ca7cbd86498bab4fca4d7fac", null],
 ["geometryVolume", "class_go_1_1_cylinder_volume.html#aa0056b200b910e3f3d3ffde922c99c5f", null],
 ["getAllBoundarySurfaces", "class_go_1_1_cylinder_volume.html#ae1659ebfb6a33d06b9ad472a6eb13d6f", null],
],
 ["instanceType", "class_go_1_1_cylinder_volume.html#a4932b362009cb7e47701268da3b2afa2", null],
 ["nextSegmentVal", "class_go_1_1_cylinder_volume.html#a0c272d275e627cfc00a631069af5808d", null],
 ["parameterSpan", "class_go_1_1_cylinder_volume.html#ab424d00c5300c4fd5e058a9406a4ff18", null],
 ["point", "class_go_1_1_cylinder_volume.html#a5354dee5ba81a64f6c2dbfa3478e60c8", null],
 ["point", "class_go_1_1_cylinder_volume.html#a5d4d6c291f3ee93eb3a51a77a054f4ed", null],
 ["read", "class_go_1_1_cylinder_volume.html#a065b3bab47de82c643de7ec5cb0f9f23", null],
 ["reverseParameterDirection", "class_go_1_1_cylinder_volume.html#aa7f2f6000ad778b67288ebb5c045cf0e", null],
],
 ["setParameters", "class_go_1_1_cylinder_volume.html#ad5b19fba2fe405f52eb17b5b69ac6fda", null],
 ["swapParameterDirection", "class_go_1_1_cylinder_volume.html#ab225baab08fd5da7c676ed5fa00c0d36", null],
],
 ["tangentCone", "class_go_1_1_cylinder_volume.html#a4d5146512b6324e7616b54c3c7fb3a51", null],
 ["translate", "class_go_1_1_cylinder_volume.html#a95f2bf0d0c0951293774486e4878d596", null],
 ["useCentreDegen", "class_go_1_1_cylinder_volume.html#af7b3a1c185cbecl07127341c236207c", null],
 ["useCornerDegen", "class_go_1_1_cylinder_volume.html#a1cbc47a1c07e0a0dcf6290c4d01b4f6d", null],
 ["write", "class_go_1_1_cylinder_volume.html#a44d169907ccb3ff1a30daadd7bdc61bf", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_cylinder\_volume.js.

## 30.248 doc/html/class\_go\_1\_1\_degenerated\_intersection\_curve.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_degenerated\\_intersection\\_curve](#)

### 30.248.1 Variable Documentation

#### 30.248.1.1 var class\_go\_1\_1\_degenerated\_intersection\_curve

##### Initial value:

```
=
[
 ["~DegeneratedIntersectionCurve", "
 class_go_1_1_degenerated_intersection_curve.html#a6c3f612a168385ed8de3211b1c67ccf4", null],
 ["evaluateAt", "class_go_1_1_degenerated_intersection_curve.html#a85b03ada4cb505fc3ae343a022e43573",
 null],
 ["getCurve", "class_go_1_1_degenerated_intersection_curve.html#a1e17134aeb92ef825f51baf5c460c8dd",
 null],
 ["getParamCurve", "class_go_1_1_degenerated_intersection_curve.html#ac69c51f4f48de3c148a9f5e18cd038f9",
 null],
 ["getParamSpan", "class_go_1_1_degenerated_intersection_curve.html#a545c99900936cb26034eaa6e17c9c19e",
 null],
 ["isDegenerated", "class_go_1_1_degenerated_intersection_curve.html#a65df4f9997b7f8a90b7539af3f7e204e",
 null],
 ["isIsocurve", "class_go_1_1_degenerated_intersection_curve.html#a55bd0adb170110339946500b1272a64b",
 null],
 ["refine", "class_go_1_1_degenerated_intersection_curve.html#ace1dcc6493ec18d1a85e0be2e8bbd619", null
],
 ["constructIntersectionCurve", "
 class_go_1_1_degenerated_intersection_curve.html#a6ab34095fc2901f18aff0d440f5f9ea5", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_degenerated\_intersection\_curve.js.

## 30.249 doc/html/class\_go\_1\_1\_direction\_cone.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_direction\\_cone](#)

### 30.249.1 Variable Documentation

#### 30.249.1.1 var class\_go\_1\_1\_direction\_cone

##### Initial value:

```
=
[
 ["DirectionCone", "class_go_1_1_direction_cone.html#a610b024d1fe8db299d840c78697988ed", null],
 ["DirectionCone", "class_go_1_1_direction_cone.html#af7fa23ab65c4febb3f5d9796f8c6fbdc", null],
 ["DirectionCone", "class_go_1_1_direction_cone.html#a0da440ae62eb38b9a8c2a6b656f8ae32", null],
 ["DirectionCone", "class_go_1_1_direction_cone.html#af0862e064737f5a30a88465d436e812c", null],
 ["addUnionWith", "class_go_1_1_direction_cone.html#a7ff8462326b11e73e10401677ec2f48b", null],
 ["addUnionWith", "class_go_1_1_direction_cone.html#a4c66b5915142fb2b49fdf5f26cea9f8b", null],
 ["angle", "class_go_1_1_direction_cone.html#af8706494f1e2f20e92d8ee953f043c94", null],
 ["centre", "class_go_1_1_direction_cone.html#a5dfalee0bb2cf83fed1e9a3a79bade30", null],
 ["containsDirection", "class_go_1_1_direction_cone.html#a10e1875d1032c9adf51fca48ea239d46", null],
 ["dimension", "class_go_1_1_direction_cone.html#a88869693ca6f3a7767ffe00939809bfe", null],
 ["greaterThanPi", "class_go_1_1_direction_cone.html#a3cfd7cfd74b9fe7aebec24f6d7caf640", null],
 ["overlaps", "class_go_1_1_direction_cone.html#a5011100ad94dc603a30164e8babf73ed", null],
 ["perpendicularOverlaps", "class_go_1_1_direction_cone.html#ad28cb1673e8f7e6ed38e6a5090e01e37", null]
],
 ["read", "class_go_1_1_direction_cone.html#a41d9c682879fec0ab8efae60c8b999e7", null],
 ["setFromArray", "class_go_1_1_direction_cone.html#a0acae3ce71fe94aadfeb7a9b7ee24e9b", null],
 ["write", "class_go_1_1_direction_cone.html#aa60780e28b9bec47b945b0bf5b1e8764", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_direction\_cone.js.

## 30.250 doc/html/class\_go\_1\_1\_disc.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_disc](#)

### 30.250.1 Variable Documentation

#### 30.250.1.1 var class\_go\_1\_1\_disc

Definition at line 1 of file class\_go\_1\_1\_disc.js.

## 30.251 doc/html/class\_go\_1\_1\_domain.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_domain](#)

### 30.251.1 Variable Documentation

#### 30.251.1.1 var class\_go\_1\_1\_domain

#### Initial value:

```
=
[
 ["~Domain", "class_go_1_1_domain.html#a8e725b87dcbbc5c3bcd8bf147546a2c4", null],
 ["closestInDomain", "class_go_1_1_domain.html#a96484652ffda6725dc28d26754757f46", null],
 ["closestOnBoundary", "class_go_1_1_domain.html#a5df4d48b552ead2c20d954ec76f5000f", null],
 ["isInDomain", "class_go_1_1_domain.html#ablable5716b47d764695791c6eadc318", null],
 ["isInDomain2", "class_go_1_1_domain.html#a8d5e1e76d48da07550d51425e4c5d7d4", null],
 ["isOnBoundary", "class_go_1_1_domain.html#a6591a84eae85b0a7d6b4b57260cb9c06", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_domain.js.

## 30.252 doc/html/class\_go\_1\_1\_edge\_vertex.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_edge\\_vertex](#)

### 30.252.1 Variable Documentation

#### 30.252.1.1 var class\_go\_1\_1\_edge\_vertex

Initial value:

```
=
[
 ["EdgeVertex", "class_go_1_1_edge_vertex.html#a12c9ba460662887ecd93ae0a8a60b2d9", null],
 ["EdgeVertex", "class_go_1_1_edge_vertex.html#a742a60072c0d2338bd8bd7b9d169a577", null],
 ["~EdgeVertex", "class_go_1_1_edge_vertex.html#a50b46f7d36e192c95cf471d0f549752c", null],
 ["addEdge", "class_go_1_1_edge_vertex.html#a148b9ff26d95cc718caefb77fa78037d", null],
 ["addEdgeVertex", "class_go_1_1_edge_vertex.html#aea7c9b67efb2a7cccb2714bdc268041f", null],
 ["allEdges", "class_go_1_1_edge_vertex.html#acd05c9fe40639b5829ff35687da8fdac", null],
 ["allEdges", "class_go_1_1_edge_vertex.html#a778cdeb70763a00ac3b30f5fd769803d", null],
 ["averageSplineEdges", "class_go_1_1_edge_vertex.html#a2d6a90905aab47cd126a75ae5096b1cb", null],
 ["checkRadialEdgeTopology", "class_go_1_1_edge_vertex.html#adc277eff97b07706cb1212cdf2d66f5f", null],
 ["disconnectTwin", "class_go_1_1_edge_vertex.html#a793cf21c09317c9da8f4f9eb4805544f", null],
 ["getAdjacentBodies", "class_go_1_1_edge_vertex.html#a8ae041883b11b40a5ddfa1130a976286", null],
 ["getAdjacentFaces", "class_go_1_1_edge_vertex.html#a76e3a610619272724db3485d4dc180e1", null],
 ["getAdjacentFaces", "class_go_1_1_edge_vertex.html#a014ce2f236b5a245431f7bb547b2c225", null],
 ["getEdge", "class_go_1_1_edge_vertex.html#ac364b5174e462487bd2cbec27d4342d3", null],
 ["hasEdge", "class_go_1_1_edge_vertex.html#ab67e7ccd3f8b03b9265a424e7ee05561", null],
 ["hasEdgeSingle", "class_go_1_1_edge_vertex.html#a6b6633146bc41d7701d35a9ec889b985", null],
 ["nmbUniqueEdges", "class_go_1_1_edge_vertex.html#aab9a1fcb19bd978a92baf20f534e4935", null],
 ["nmbUniqueEdges", "class_go_1_1_edge_vertex.html#a4880f8730b184e40dd0ea1497bd71b71", null],
 ["organizeTwins", "class_go_1_1_edge_vertex.html#a620f8ad91fb83627c90a5ebd964045b8", null],
 ["removeEdge", "class_go_1_1_edge_vertex.html#a0f3b6dcded230c892b134abc40995ba1", null],
 ["reOrganize", "class_go_1_1_edge_vertex.html#ab8de2fc21bf706398617009b1b0b7585", null],
 ["splitAtVertex", "class_go_1_1_edge_vertex.html#aab28625c4d539800b84bbdb9a89cf9f", null],
 ["uniqueEdges", "class_go_1_1_edge_vertex.html#a07c73b1706f3636174d8f830aa50668d", null],
 ["uniqueEdges", "class_go_1_1_edge_vertex.html#a3e2f850162d60e61d0bedbb65e55503a", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_edge\_vertex.js.

### 30.253 doc/html/class\_go\_1\_1\_element2\_d.js File Reference

Variables

- var [class\\_go\\_1\\_1\\_element2\\_d](#)

#### 30.253.1 Variable Documentation

##### 30.253.1.1 var class\_go\_1\_1\_element2\_d

Definition at line 1 of file class\_go\_1\_1\_element2\_d.js.

### 30.254 doc/html/class\_go\_1\_1\_elementary\_curve.js File Reference

Variables

- var [class\\_go\\_1\\_1\\_elementary\\_curve](#)

### 30.254.1 Variable Documentation

#### 30.254.1.1 var class\_go\_1\_1\_elementary\_curve

##### Initial value:

```
=
[
 ["ElementaryCurve", "class_go_1_1_elementary_curve.html#a6c9c6c1e099007d2ef132a36ec255f63", null],
 ["~ElementaryCurve", "class_go_1_1_elementary_curve.html#a1d398c75da253b0c7cd7946e4511a14b", null],
 ["clone", "class_go_1_1_elementary_curve.html#ae5f7a018119b2c65b50331c62f5eaff1", null],
 ["createSplineCurve", "class_go_1_1_elementary_curve.html#a2ec7658b6ee1ebfce72563fdf348bc0a", null],
 ["getReversedParameter", "class_go_1_1_elementary_curve.html#a8cc3324b114ae56a769fbd92e5271a29", null],
],
 ["isReversed", "class_go_1_1_elementary_curve.html#a64be203c5c90fdefb79c055ea2bb616f", null],
 ["reverseParameterDirection", "class_go_1_1_elementary_curve.html#a36a6e21faf644b7567279e8a7d7b1299", null],
],
 ["setParamBounds", "class_go_1_1_elementary_curve.html#afc982c46ae92e088302b2cb0a0fb0ed4", null],
 ["subCurve", "class_go_1_1_elementary_curve.html#a8c71872f62c5427499ed1ed9aaf4b643", null],
 ["swapParameters2D", "class_go_1_1_elementary_curve.html#a0e09e54b118b23e9ee176fc7aef59c6a", null],
 ["translateCurve", "class_go_1_1_elementary_curve.html#aac91b68dc5a447a1124a914cea30e2b4", null],
 ["isReversed_", "class_go_1_1_elementary_curve.html#a4c5b3bf2b92fc76dc7b2752079a064c5", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_elementary\_curve.js.

## 30.255 doc/html/class\_go\_1\_1\_elementary\_surface.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_elementary\\_surface](#)

### 30.255.1 Variable Documentation

#### 30.255.1.1 var class\_go\_1\_1\_elementary\_surface

##### Initial value:

```
=
[
 ["ElementarySurface", "class_go_1_1_elementary_surface.html#a71a1228cd825f443855f5895d45ac4ff", null],
 ["~ElementarySurface", "class_go_1_1_elementary_surface.html#a58a9ae8c3839a441e59f05ecda95d30e", null],
],
 ["area", "class_go_1_1_elementary_surface.html#ab20ceab1a6924addf76e0d04850de566", null],
 ["asSplineSurface", "class_go_1_1_elementary_surface.html#a9e75a55c8d72e608fbd6296c0b9cd774", null],
 ["clone", "class_go_1_1_elementary_surface.html#a6db0440ad98003fb958ed880a8fca0cf", null],
 ["closestInDomain", "class_go_1_1_elementary_surface.html#a634760ecaeb71c9887d54f8200776833", null],
 ["containingDomain", "class_go_1_1_elementary_surface.html#a4ca4805ddddd25393e8bb5eda817b3d8", null],
 ["createSplineSurface", "class_go_1_1_elementary_surface.html#aa73a66ee60fc2c33c6dc5217f8d6f207", null],
],
 ["geometrySurface", "class_go_1_1_elementary_surface.html#a6504be7f468129a6dda4c055a5973aff", null],
 ["getCornerPoints", "class_go_1_1_elementary_surface.html#a0bdb7c1e01378de25162da6e65c86adc", null],
 ["getElementaryParamCurve", "class_go_1_1_elementary_surface.html#ad7236175c38cc23115f4f8fe62ccd824", null],
],
 ["getOrientedParameters", "class_go_1_1_elementary_surface.html#ac0aa83e480d77aa16e51cd0fd5dfc287", null],
],
 ["inDomain", "class_go_1_1_elementary_surface.html#ab7a1df7c8f47b20c86338fd5a03a4942", null],
 ["inDomain2", "class_go_1_1_elementary_surface.html#afdaa731dfaf4019e1df8597b5cc4b4ea", null],
 ["isBounded", "class_go_1_1_elementary_surface.html#a36db1675585d324c61e9697aac76a0e4", null],
 ["isClosed", "class_go_1_1_elementary_surface.html#a432f5674498353c1cc4dd591a2d14e2", null],
 ["isSwapped", "class_go_1_1_elementary_surface.html#aaa6168898c2ae401885d572593036b1a", null],
 ["onBoundary", "class_go_1_1_elementary_surface.html#a5361dlb1a53647579c1a8ede335045d8", null],
 ["outerBoundaryLoop", "class_go_1_1_elementary_surface.html#af29090b6cbf2c2b5b2455e9ed65a3b13", null]
]
```

```
[
 ,
 ["reverseParameterDirection", "class_go_1_1_elementary_surface.html#a580b9242c586c5c8f6a99dc5237a57b2"
 , null],
 ["setParameterBounds", "class_go_1_1_elementary_surface.html#a5addf83a86dc0264784997f6a1c148e9", null
],
 ["swapParameterDirection", "class_go_1_1_elementary_surface.html#a24fd226efala448c4375555ec3c9c937",
 null],
 ["turnOrientation", "class_go_1_1_elementary_surface.html#ab7a7d157d35de73437a1cc3e0a51e9bc", null],
 ["isSwapped", "class_go_1_1_elementary_surface.html#a5567042280cff31567eb5868910eb883", null]
]
```

Definition at line 1 of file `class_go_1_1_elementary_surface.js`.

## 30.256 doc/html/class\_go\_1\_1\_elementary\_volume.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_elementary\\_volume](#)

#### 30.256.1 Variable Documentation

##### 30.256.1.1 var class\_go\_1\_1\_elementary\_volume

###### Initial value:

```
=
[
 ["~ElementaryVolume", "class_go_1_1_elementary_volume.html#aace856a29a9e915070e648d142c8f3b4", null],
 ["clone", "class_go_1_1_elementary_volume.html#a5242e3de8b56bf6ca52fa0e3908b4d2d", null],
 ["geometryVolume", "class_go_1_1_elementary_volume.html#ab5dda4f80a667b9414d3477a10911da0", null]
]
```

Definition at line 1 of file `class_go_1_1_elementary_volume.js`.

## 30.257 doc/html/class\_go\_1\_1\_ellipse.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_ellipse](#)

#### 30.257.1 Variable Documentation

##### 30.257.1.1 var class\_go\_1\_1\_ellipse

Definition at line 1 of file `class_go_1_1_ellipse.js`.

## 30.258 doc/html/class\_go\_1\_1\_eval\_curve.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_eval\\_curve](#)

### 30.258.1 Variable Documentation

#### 30.258.1.1 var class\_go\_1\_1\_eval\_curve

##### Initial value:

```
=
[
 ["~EvalCurve", "class_go_1_1_eval_curve.html#aeba525b92071e851085947debd51aaa3", null],
 ["approximationOK", "class_go_1_1_eval_curve.html#a762cfdc5c5a07f6a98ddcb72319e2cef", null],
 ["dim", "class_go_1_1_eval_curve.html#a31a0268f4843c33a21cf88b18e0fb97e", null],
 ["end", "class_go_1_1_eval_curve.html#a537c4baaf5cf5b54ce682d1c21f7e3d6", null],
 ["eval", "class_go_1_1_eval_curve.html#a005dde8b68603c09495f227c816db32e", null],
 ["eval", "class_go_1_1_eval_curve.html#aee94d28a2a2be53264fed1694208648a", null],
 ["start", "class_go_1_1_eval_curve.html#a9af98ce2c8de8dc7364d727daf974236", null],
 ["write", "class_go_1_1_eval_curve.html#a4ff86322f9dc39ffc4620ee477a2ce1e", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_eval\_curve.js.

## 30.259 doc/html/class\_go\_1\_1\_eval\_curve\_set.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_eval\\_curve\\_set](#)

### 30.259.1 Variable Documentation

#### 30.259.1.1 var class\_go\_1\_1\_eval\_curve\_set

##### Initial value:

```
=
[
 ["~EvalCurveSet", "class_go_1_1_eval_curve_set.html#aaeled637c545c93a5be946bc166e51e2", null],
 ["approximationOK", "class_go_1_1_eval_curve_set.html#af4968a577be7184ba447ce777206b97a", null],
 ["dim", "class_go_1_1_eval_curve_set.html#a80c4eae4de2d67081f89f10a8cdeb4f0", null],
 ["end", "class_go_1_1_eval_curve_set.html#ab898fdd26d67c986bdb4f1d5b2e753e3", null],
 ["eval", "class_go_1_1_eval_curve_set.html#a6cad51d0922a7dedac3a31b2026ab0a4", null],
 ["eval", "class_go_1_1_eval_curve_set.html#af94dfe476d9a31a9283263e1c7eb8bb3", null],
 ["nmbCvs", "class_go_1_1_eval_curve_set.html#aaaf41679ad78e081e33c346ca15c3e3dc", null],
 ["resetErr", "class_go_1_1_eval_curve_set.html#ad980bd9d596a83f4946e570088207019", null],
 ["start", "class_go_1_1_eval_curve_set.html#add92cc9caecf70771a9851c3e6efee07", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_eval\_curve\_set.js.

## 30.260 doc/html/class\_go\_1\_1\_eval\_functor\_curve.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_eval\\_functor\\_curve](#)

### 30.260.1 Variable Documentation

#### 30.260.1.1 var class\_go\_1\_1\_eval\_functor\_curve

##### Initial value:

```
=
[
 ["EvalFunctorCurve", "class_go_1_1_eval_functor_curve.html#ab63841530bb9fe53f152f9b7ef0f4a30", null],
 ["~EvalFunctorCurve", "class_go_1_1_eval_functor_curve.html#a3b184f0c9b44be70f7f258e2f9b4f3bc", null]
 ,
 ["approximationOK", "class_go_1_1_eval_functor_curve.html#ac34c98eecb84f37f1bd0e6226356ef6a", null],
 ["dim", "class_go_1_1_eval_functor_curve.html#a912b89fbc70a53f66322b50937106b66", null],
 ["end", "class_go_1_1_eval_functor_curve.html#aba56f588c2338c61ce35c079047911cf", null],
 ["eval", "class_go_1_1_eval_functor_curve.html#adc7d79d2ad002096776641323d421fc0", null],
 ["eval", "class_go_1_1_eval_functor_curve.html#ad765b1fdca8d47430a90b05d08358c42", null],
 ["start", "class_go_1_1_eval_functor_curve.html#abal21fb3ed8a35d0ad9d5f3aae10c1cf", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_eval\_functor\_curve.js.

## 30.261 doc/html/class\_go\_1\_1\_eval\_functor\_surface.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_eval\\_functor\\_surface](#)

### 30.261.1 Variable Documentation

#### 30.261.1.1 var class\_go\_1\_1\_eval\_functor\_surface

##### Initial value:

```
=
[
 ["EvalFunctorSurface", "class_go_1_1_eval_functor_surface.html#acc474678964ce779ea9b608d32b1a6b2",
 null],
 ["~EvalFunctorSurface", "class_go_1_1_eval_functor_surface.html#a49102c1eb44d51ec302cc4856fc7efc3",
 null],
 ["approximationOK", "class_go_1_1_eval_functor_surface.html#a3f1f67e6402acc424125d7b9e0acd3e4", null]
 ,
 ["dim", "class_go_1_1_eval_functor_surface.html#aa3fd1667b9a884d89ca4da4ff67f1734", null],
 ["end_u", "class_go_1_1_eval_functor_surface.html#a92b4a3dae1a78e53951b3a652de3c062", null],
 ["end_v", "class_go_1_1_eval_functor_surface.html#ac50aeec2db8861e7396f7c19323eb366", null],
 ["eval", "class_go_1_1_eval_functor_surface.html#a4bba1fd389a8a21408c76279077d8c98", null],
 ["eval", "class_go_1_1_eval_functor_surface.html#a52a9888c724819231d867b5f8ecc9edb", null],
 ["start_u", "class_go_1_1_eval_functor_surface.html#ae9becd7292db4913e793eb111be918c2", null],
 ["start_v", "class_go_1_1_eval_functor_surface.html#a4c98990f6b8aac5fd8cac7187c30d076", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_eval\_functor\_surface.js.



## 30.262 doc/html/class\_go\_1\_1\_eval\_offset\_surface.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_eval\\_offset\\_surface](#)

### 30.262.1 Variable Documentation

#### 30.262.1.1 var class\_go\_1\_1\_eval\_offset\_surface

##### Initial value:

```
=
[
 ["EvalOffsetSurface", "class_go_1_1_eval_offset_surface.html#afe1b3f4bf307f0614888f73ef8a6eb5c", null],
 ["~EvalOffsetSurface", "class_go_1_1_eval_offset_surface.html#a9d6f5e55968022cc86d7b9811e9b0198", null],
 ["approximationOK", "class_go_1_1_eval_offset_surface.html#af0570735361d62cb20e915d11a30d05f", null],
 ["dim", "class_go_1_1_eval_offset_surface.html#a35881f83eb4f2e557e694aa2a174c5cc", null],
 ["end_u", "class_go_1_1_eval_offset_surface.html#a0d4549a358b573313f7c8354b457a5ec", null],
 ["end_v", "class_go_1_1_eval_offset_surface.html#a798642a259a4d33b0cd73fb8479ab262", null],
 ["eval", "class_go_1_1_eval_offset_surface.html#alf13de4389d651e8293d209931ac84d5", null],
 ["eval", "class_go_1_1_eval_offset_surface.html#af5b3364c29d6874afa06ed665788c448", null],
 ["getProjKinkCurves", "class_go_1_1_eval_offset_surface.html#afc7bddd486c0ad22e5cedef03a6dc877", null],
 ["gridKinks", "class_go_1_1_eval_offset_surface.html#a765e5a3801de2111f1cafb8c5c9f310e", null],
 ["gridSelfIntersections", "class_go_1_1_eval_offset_surface.html#aae3c5f3f06e4a75eaf2fbbce6cfbe6c5", null],
 ["start_u", "class_go_1_1_eval_offset_surface.html#aa2d8866fab10e8936ec210fc2ca91759", null],
 ["start_v", "class_go_1_1_eval_offset_surface.html#aale6ab3ed0b12f1f1318b97eb82ffbee", null]
]
```

Definition at line 1 of file [class\\_go\\_1\\_1\\_eval\\_offset\\_surface.js](#).

## 30.263 doc/html/class\_go\_1\_1\_eval\_param\_curve.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_eval\\_param\\_curve](#)

### 30.263.1 Variable Documentation

#### 30.263.1.1 var class\_go\_1\_1\_eval\_param\_curve

##### Initial value:

```
=
[
 ["EvalParamCurve", "class_go_1_1_eval_param_curve.html#a40e626ea4d004d43579185cb55ae6cc0", null],
 ["~EvalParamCurve", "class_go_1_1_eval_param_curve.html#a66eb2f1e5afb01ae14b0155fa2bf8621", null],
 ["approximationOK", "class_go_1_1_eval_param_curve.html#a7add6299daa6277d3797ad9e0381687", null],
 ["dim", "class_go_1_1_eval_param_curve.html#a0cd1e0a0e2fa20e34e2afb73f78256f7", null],
 ["end", "class_go_1_1_eval_param_curve.html#a4c302f47bb29a2b73b9559a4ffe684cc", null],
 ["eval", "class_go_1_1_eval_param_curve.html#a765082ce27f3a3ac77f9dbdal1001cf1", null],
 ["eval", "class_go_1_1_eval_param_curve.html#ae053eebea18f9dbe4a3289a7aeb2563d", null],
 ["start", "class_go_1_1_eval_param_curve.html#aa3d62769a1980cb3524a2359e83c66ab", null],
 ["write", "class_go_1_1_eval_param_curve.html#a1e4877d32b0ce7cac403583c1710511c", null]
]
```

Definition at line 1 of file [class\\_go\\_1\\_1\\_eval\\_param\\_curve.js](#).

## 30.264 doc/html/class\_go\_1\_1\_eval\_surface.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_eval\\_surface](#)

### 30.264.1 Variable Documentation

#### 30.264.1.1 var class\_go\_1\_1\_eval\_surface

#### Initial value:

```
=
[
 ["~EvalSurface", "class_go_1_1_eval_surface.html#aaf6a7b7a118319c4835eaf188f26f012", null],
 ["approximationOK", "class_go_1_1_eval_surface.html#a4cee285c65f02c95138c61649366196f", null],
 ["dim", "class_go_1_1_eval_surface.html#a348640a6de454d8ba92c934601831b3d", null],
 ["end_u", "class_go_1_1_eval_surface.html#af5e34529731512a96116ebc9b7dee51f", null],
 ["end_v", "class_go_1_1_eval_surface.html#a50cbf234743c7e59484ac72f92966a3e", null],
 ["eval", "class_go_1_1_eval_surface.html#a2048831767bc076026fb999eb2f1036c", null],
 ["eval", "class_go_1_1_eval_surface.html#ac30c4ea3682ad3433ed510bebf64c968", null],
 ["start_u", "class_go_1_1_eval_surface.html#a05cabd37b0bb19aba04aafee43364e74", null],
 ["start_v", "class_go_1_1_eval_surface.html#a700225bd935f675e29e18f34d7ff3fd4", null],
 ["write", "class_go_1_1_eval_surface.html#a04f0ae2e984ad26c9fea650a5f0f54f5", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_eval\_surface.js.

## 30.265 doc/html/class\_go\_1\_1\_face\_adjacency.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_face\\_adjacency](#)

### 30.265.1 Variable Documentation

#### 30.265.1.1 var class\_go\_1\_1\_face\_adjacency

#### Initial value:

```
=
[
 ["FaceAdjacency", "class_go_1_1_face_adjacency.html#a3301eab6895130fcaa1924fff513361e", null],
 ["FaceAdjacency", "class_go_1_1_face_adjacency.html#a586af6c2a4877a6fccadc369b1807d37", null],
 ["~FaceAdjacency", "class_go_1_1_face_adjacency.html#a42506d48ad19e20f7a2eedc8a4564169", null],
 ["computeAdjacency", "class_go_1_1_face_adjacency.html#a5cc8e257771caa53d08c8f4b041d8373", null],
 ["computeAdjacency", "class_go_1_1_face_adjacency.html#aa85085bf8fc96f491ebf4160ce10a72f", null],
 ["computeFaceAdjacency", "class_go_1_1_face_adjacency.html#ad70260b669fa2935e9651420cf08ee55", null],
 ["computeFaceAdjacency", "class_go_1_1_face_adjacency.html#aed61a81bf5156b00b245e9ed45ef4f6f", null],
 ["computeFaceAdjacency", "class_go_1_1_face_adjacency.html#ab274fcf2ec5007ef799285d6bf78363b", null],
 ["computeFaceAdjacency", "class_go_1_1_face_adjacency.html#ad207eela2fd98af726de34604e341c2", null],
 ["connectTwins", "class_go_1_1_face_adjacency.html#a1bb423a1be8c29ce621a163f35273b38", null],
 ["getTolerances", "class_go_1_1_face_adjacency.html#a9c319c64873d68c4dac0742c2027d52b", null],
 ["releaseFaceAdjacency", "class_go_1_1_face_adjacency.html#a51273eb10f58cd29aa2ae323a8abdd0a", null],
 ["setConnectivity", "class_go_1_1_face_adjacency.html#a1681edc44704cbb9e8d3544c5b08ccda", null],
 ["setConnectivity", "class_go_1_1_face_adjacency.html#acf4bc974120a1aecec2e71a20ccb9e58", null],
 ["setTolerances", "class_go_1_1_face_adjacency.html#a6a20573f260f2c40023c9a2e48813bd7", null],
 ["updateConnectivity", "class_go_1_1_face_adjacency.html#a077c01c491735a39ebc0ee0edff04e31", null],
 ["new_edges", "class_go_1_1_face_adjacency.html#a47a4a49406d103caaf027dbb5fd3a0e9", null],
 ["tol", "class_go_1_1_face_adjacency.html#a9af2ef45a2ee40db27c4f2078d07ad69", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_face\_adjacency.js.

## 30.266 doc/html/class\_go\_1\_1\_face\_connectivity\_utils.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_face\\_connectivity\\_utils](#)

#### 30.266.1 Variable Documentation

##### 30.266.1.1 var class\_go\_1\_1\_face\_connectivity\_utils

###### Initial value:

```
=
[
 ["BoundaryLoops", "class_go_1_1_face_connectivity_utils.html#a6fc13b0597434c4e46b8d94d7f4fe506", null],
 ["BoundaryLoops", "class_go_1_1_face_connectivity_utils.html#aa84040cbf1fc0e2e6456d6dc70ccb490", null],
 ["cornersAndKinks", "class_go_1_1_face_connectivity_utils.html#af2d3bef904caac42c6ea7c43242c334a", null],
 ["disjointObjects", "class_go_1_1_face_connectivity_utils.html#ad8f359dfd963dd282a616afa46298173", null],
 ["edgesBoundingFace", "class_go_1_1_face_connectivity_utils.html#ad76e67d53a3aa54f6072500ac182a1f4", null],
 ["smoothEdges", "class_go_1_1_face_connectivity_utils.html#ab02ab238985992fe96cdc155808e1279", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_face\_connectivity\_utils.js.

## 30.267 doc/html/class\_go\_1\_1\_face\_set\_quality.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_face\\_set\\_quality](#)

#### 30.267.1 Variable Documentation

##### 30.267.1.1 var class\_go\_1\_1\_face\_set\_quality

Definition at line 1 of file class\_go\_1\_1\_face\_set\_quality.js.

## 30.268 doc/html/class\_go\_1\_1\_face\_set\_repair.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_face\\_set\\_repair](#)

### 30.268.1 Variable Documentation

#### 30.268.1.1 var class\_go\_1\_1\_face\_set\_repair

##### Initial value:

```
=
[
 ["FaceSetRepair", "class_go_1_1_face_set_repair.html#a4ab24096dd61caa312afdd87dbad917c", null],
 ["FaceSetRepair", "class_go_1_1_face_set_repair.html#af9ale4cdc65ea8fd3b6189f38bbca57c", null],
 ["~FaceSetRepair", "class_go_1_1_face_set_repair.html#ac439daafe811750cd54658ec73ecceb2", null],
 ["consistentFaceNormal", "class_go_1_1_face_set_repair.html#a1c80999e1664e237219be4ec79483370", null]
 ,
 ["getAssociatedSfModel", "class_go_1_1_face_set_repair.html#a66e6740743c2b29466c632927e01ffdc", null]
 ,
 ["identicalAndEmbeddedFaces", "class_go_1_1_face_set_repair.html#ab30952f5cfc5281414f1c90af73de25e",
 null],
 ["identicalVertices", "class_go_1_1_face_set_repair.html#a06b2af4b37d3fe6dd6c26fca17d150e3", null],
 ["mendEdgeDistance", "class_go_1_1_face_set_repair.html#a523caabe3b52dd462e86d683cf60229b", null],
 ["mendGaps", "class_go_1_1_face_set_repair.html#a44d229273c3088685a73148080fab442", null],
 ["optimizeVertexPosition", "class_go_1_1_face_set_repair.html#aaa6f724b25062d2e404b45186f9a672b", null
]
]
```

Definition at line 1 of file class\_go\_1\_1\_face\_set\_repair.js.

## 30.269 doc/html/class\_go\_1\_1\_factory.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_factory](#)

### 30.269.1 Variable Documentation

#### 30.269.1.1 var class\_go\_1\_1\_factory

##### Initial value:

```
=
[
 ["~Factory", "class_go_1_1_factory.html#a248ab10834992ea327357bd218617bd3", null],
 ["registerClass", "class_go_1_1_factory.html#a53348d9768f5fc831ab399667f57a75a", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_factory.js.

## 30.270 doc/html/class\_go\_1\_1\_fun2\_fun.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_fun2\\_fun](#)

### 30.270.1 Variable Documentation

#### 30.270.1.1 var class\_go\_1\_1\_fun2\_fun

##### Initial value:

```
=
[
 ["Fun2Fun", "class_go_1_1_fun2_fun.html#a9a72fe3655b2819d0ef0cde391a57bd7", null],
 ["maxPar", "class_go_1_1_fun2_fun.html#ab229b3fe6ea09e812ac03bdcdb3e9373", null],
 ["minPar", "class_go_1_1_fun2_fun.html#a8f83e8c2be2c112b0f360717ceb60994", null],
 ["operator()", "class_go_1_1_fun2_fun.html#a45600cb0eada16c9ef7d2e05a9e05ebd", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_fun2\_fun.js.

## 30.271 doc/html/class\_go\_1\_1\_function\_minimizer.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_function\\_minimizer](#)

### 30.271.1 Variable Documentation

#### 30.271.1.1 var class\_go\_1\_1\_function\_minimizer

##### Initial value:

```
=
[
 ["FunctionMinimizer", "class_go_1_1_function_minimizer.html#af01b5848d83482b770ffe42f3de5251a", null],
 ["~FunctionMinimizer", "class_go_1_1_function_minimizer.html#a27572672ed10f56598cabaee915d2a2a", null],
 ["atMax", "class_go_1_1_function_minimizer.html#a47cf89619002eed0b9ae0f32069c7275", null],
 ["atMin", "class_go_1_1_function_minimizer.html#a2d287703fcbb4dfbfff1fcc6ac2a0536e", null],
 ["fval", "class_go_1_1_function_minimizer.html#af7fd5958d66da72008d8a37497b25073", null],
 ["fval", "class_go_1_1_function_minimizer.html#a0f69b49805f71c1cd9e9d901b3006fc3", null],
 ["getPar", "class_go_1_1_function_minimizer.html#a6199eb10189b03c08469ea6b68b8d65f", null],
 ["getPar", "class_go_1_1_function_minimizer.html#a884069b2aecb10ff0c478ab84f7db26e", null],
 ["grad", "class_go_1_1_function_minimizer.html#a17ec134a193d56dbe43f56403481dbeb", null],
 ["grad", "class_go_1_1_function_minimizer.html#a58b443fe5a9bb188aa6f72243a765111", null],
 ["linminBrent", "class_go_1_1_function_minimizer.html#a171d9cbe54a0ee0dfb8587f1f6ff03c", null],
 ["minimize", "class_go_1_1_function_minimizer.html#a2bfa66cb87c1e1b7095ebaea699003a4", null],
 ["moveUV", "class_go_1_1_function_minimizer.html#a15bd0257636d3d0429547be0fed98387", null],
 ["numPars", "class_go_1_1_function_minimizer.html#a85887c9e0d47259574b25076b4213fa9", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_function\_minimizer.js.

## 30.272 doc/html/class\_go\_1\_1\_general\_mesh.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_general\\_mesh](#)

### 30.272.1 Variable Documentation

#### 30.272.1.1 var class\_go\_1\_1\_general\_mesh

##### Initial value:

```
=
[
 ["GeneralMesh", "class_go_1_1_general_mesh.html#a09725cc3c16ce78026e542ce4fcd0d79", null],
 ["~GeneralMesh", "class_go_1_1_general_mesh.html#a5c1f1add8aa787766728dce46b6f8ada", null],
 ["asGenericTriMesh", "class_go_1_1_general_mesh.html#a57d8daf7adf72646ffe95677b3fa533", null],
 ["asLineStrip", "class_go_1_1_general_mesh.html#a6b130a6a905fc74ccc7f8f27f504e6dd", null],
 ["asRegularMesh", "class_go_1_1_general_mesh.html#a36ee20c757b6b61368fe3aa557c4bb86", null],
 ["atBoundary", "class_go_1_1_general_mesh.html#a548ca69e695da9a39f4734b078d8db0a", null],
 ["numTriangles", "class_go_1_1_general_mesh.html#a6033cece0be278ca8f70a96226fe80b9", null],
 ["numVertices", "class_go_1_1_general_mesh.html#a55f396aefbe644cb958eaa0e25883d58", null],
 ["paramArray", "class_go_1_1_general_mesh.html#af0436cd3df3c3575c61887daf9b5df0c", null],
 ["translate", "class_go_1_1_general_mesh.html#ae531ce5bb0c394c963f68f7ce7a337b5", null],
 ["triangleIndexArray", "class_go_1_1_general_mesh.html#a7a0c390d462c3b227ca824b5fa2cc3a7", null],
 ["vertexArray", "class_go_1_1_general_mesh.html#ae6a2f8a755ce92ced7301dceb5f7e4a8", null],
 ["vert_translation", "class_go_1_1_general_mesh.html#a1acf92dd244f23728a0a59d9e8517ed8", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_general\_mesh.js.

## 30.273 doc/html/class\_go\_1\_1\_generic\_tri\_mesh.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_generic\\_tri\\_mesh](#)

### 30.273.1 Variable Documentation

#### 30.273.1.1 var class\_go\_1\_1\_generic\_tri\_mesh

##### Initial value:

```
=
[
 ["GenericTriMesh", "class_go_1_1_generic_tri_mesh.html#aal6f22a46fe984eac8b8ae82d54d5bc7", null],
 ["~GenericTriMesh", "class_go_1_1_generic_tri_mesh.html#aa6982bb365a9b5ccaf1d2a32c640554", null],
 ["asGenericTriMesh", "class_go_1_1_generic_tri_mesh.html#a9f92374a31266813e906fa882fd1f4cc", null],
 ["atBoundary", "class_go_1_1_generic_tri_mesh.html#a687e79c84b8bb334e8f8c90cf59b430c", null],
 ["boundaryArray", "class_go_1_1_generic_tri_mesh.html#a137746e4f59c4c239cb3995fc9ae8289", null],
 ["normalArray", "class_go_1_1_generic_tri_mesh.html#ae916f28083e9041c9bdd9179298fbec4", null],
 ["numTriangles", "class_go_1_1_generic_tri_mesh.html#ab3155db37ee55273675d4a58efe28c91", null],
 ["numVertices", "class_go_1_1_generic_tri_mesh.html#aceaf753237940bcd388266f34f6c7290", null],
 ["paramArray", "class_go_1_1_generic_tri_mesh.html#ac01004f7b44d8cad36cb19bdb4e6f0a7", null],
 ["resize", "class_go_1_1_generic_tri_mesh.html#alb4a4d46c0ab288f094f02b5356a2e18", null],
 ["texcoordArray", "class_go_1_1_generic_tri_mesh.html#ad92c094ce3e6c0def5bb36a7ff5d215e", null],
 ["triangleIndexArray", "class_go_1_1_generic_tri_mesh.html#afdc4359022e38272b171b34f8d1185bc", null],
 ["useNormals", "class_go_1_1_generic_tri_mesh.html#a3de2c07766a4122114170a33b62c3fb3", null],
 ["useTexCoords", "class_go_1_1_generic_tri_mesh.html#ac9d07cc916b40d1c5893a42500cc0fc5", null],
 ["vertexArray", "class_go_1_1_generic_tri_mesh.html#a84039be881913d8e0248be14fd32dbdd", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_generic\_tri\_mesh.js.

## 30.274 doc/html/class\_go\_1\_1\_geo\_tol.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_geo\\_tol](#)

### 30.274.1 Variable Documentation

#### 30.274.1.1 var class\_go\_1\_1\_geo\_tol

##### Initial value:

```
=
[
 ["GeoTol", "class_go_1_1_geo_tol.html#a41f90517a5761fac882359aa9714a785", null],
 ["GeoTol", "class_go_1_1_geo_tol.html#a58a54012ff9cd1d7b2cbd6cccb8f5d62", null],
 ["~GeoTol", "class_go_1_1_geo_tol.html#add1cee7409c22a23865c7cad4ce0775f", null],
 ["getAngleTol", "class_go_1_1_geo_tol.html#a324a71f9156baf7cb20c2c7aa6c77deb", null],
 ["getEpsBracket", "class_go_1_1_geo_tol.html#a5c471b28a9d28557e204ef0ee9069a93", null],
 ["getEpsge", "class_go_1_1_geo_tol.html#a8b7bc072822915300a661c40f9e5f52e", null],
 ["getMaxEpsge", "class_go_1_1_geo_tol.html#a749f343f73a1a3efc6c8d6f71611543d", null],
 ["getMinEpsge", "class_go_1_1_geo_tol.html#a02c05193604c1a9cae7d9b3b54fc93fe", null],
 ["getNumericalTol", "class_go_1_1_geo_tol.html#a2471d316627662dddal6aa5ec6ad1273", null],
 ["getRefAng", "class_go_1_1_geo_tol.html#ab6748beb1d2e6f74521fe664537dccb1", null],
 ["getRelParRes", "class_go_1_1_geo_tol.html#a83d8f119ecb58d913ceb1f58a293f7bd", null],
 ["read", "class_go_1_1_geo_tol.html#a9273319186522ead8f717dae87ae9b1f", null],
 ["write", "class_go_1_1_geo_tol.html#a4442a98b53da78dc97f9d82c27e8fce1", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_geo\_tol.js.

## 30.275 doc/html/class\_go\_1\_1\_geom\_object.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_geom\\_object](#)

### 30.275.1 Variable Documentation

#### 30.275.1.1 var class\_go\_1\_1\_geom\_object

##### Initial value:

```
=
[
 ["~GeomObject", "class_go_1_1_geom_object.html#a2e0a76f903c4b3958b548730902d1646", null],
 ["boundingBox", "class_go_1_1_geom_object.html#a067f1ee296da03387b3675d89977d2ce", null],
 ["clone", "class_go_1_1_geom_object.html#a30ae24ea6a80ee95ad7ea028d02d27a0", null],
 ["dimension", "class_go_1_1_geom_object.html#a00210cfb960925b27a272c6b24e71c06", null],
 ["instanceType", "class_go_1_1_geom_object.html#a76a0b2f3b42d1ee4273240e1295b72b5", null],
 ["writeStandardHeader", "class_go_1_1_geom_object.html#a4859ea7a2cc668eb91d7d8f26c214ba5", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_geom\_object.js.

## 30.276 doc/html/class\_go\_1\_1\_geom\_object\_int.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_geom\\_object\\_int](#)

#### 30.276.1 Variable Documentation

##### 30.276.1.1 var class\_go\_1\_1\_geom\_object\_int

###### Initial value:

```
=
[
 ["~GeomObjectInt", "class_go_1_1_geom_object_int.html#a5980a9a4fba51bac14701fbfb5a9e34b", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_geom\_object\_int.js.

## 30.277 doc/html/class\_go\_1\_1\_go\_tools.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_go\\_tools](#)

#### 30.277.1 Variable Documentation

##### 30.277.1.1 var class\_go\_1\_1\_go\_tools

###### Initial value:

```
=
[
 ["GoTools", "class_go_1_1_go_tools.html#ae87697caa683027a61775292e25dab50", null],
 ["~GoTools", "class_go_1_1_go_tools.html#af2b31efe6166a1059d637b87490d408c", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_go\_tools.js.

## 30.278 doc/html/class\_go\_1\_1\_gpu\_matrix.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_gpu\\_matrix](#)



### 30.278.1 Variable Documentation

#### 30.278.1.1 var class\_go\_1\_1\_gpu\_matrix

##### Initial value:

```
=
[
 ["GpuMatrix", "class_go_1_1_gpu_matrix.html#a45b14719d2e445da324b8bbd95620114", null],
 ["GpuMatrix", "class_go_1_1_gpu_matrix.html#a6725ea84822469b2243ec8a3a739ae8c", null],
 ["GpuMatrix", "class_go_1_1_gpu_matrix.html#a2f56334696195b5c6db0dfb26d0c03d4", null],
 ["~GpuMatrix", "class_go_1_1_gpu_matrix.html#adc2729c001118fae54ef0a40d2395c33", null],
 ["download", "class_go_1_1_gpu_matrix.html#a79a89591dbe8955ba9f8ca33ab937665", null],
 ["getCols", "class_go_1_1_gpu_matrix.html#a89254656639c7c57d40c954b53093ab2", null],
 ["getRows", "class_go_1_1_gpu_matrix.html#ae4f2528cedcb24aee0536967888debde", null],
 ["upload", "class_go_1_1_gpu_matrix.html#aafe7bfb9f3e83a5a77203ac66cddfca", null],
 ["operator<<", "class_go_1_1_gpu_matrix.html#a2b093b490ce041a8c938a54a461f75a5", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_gpu\_matrix.js.

## 30.279 doc/html/class\_go\_1\_1\_hermitte\_app\_c.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_hermitte\\_app\\_c](#)

### 30.279.1 Variable Documentation

#### 30.279.1.1 var class\_go\_1\_1\_hermitte\_app\_c

##### Initial value:

```
=
[
 ["HermiteAppC", "class_go_1_1_hermitte_app_c.html#a0943f80d1aa006f7a6e714878f4e77d4", null],
 ["HermiteAppC", "class_go_1_1_hermitte_app_c.html#a49691528c5f2540d6354336548d65338", null],
 ["~HermiteAppC", "class_go_1_1_hermitte_app_c.html#aa8228556786e4929fe1771f565e2709f", null],
 ["getCurve", "class_go_1_1_hermitte_app_c.html#aaef51b0f3e322a3020410a2802d8072f", null],
 ["refineApproximation", "class_go_1_1_hermitte_app_c.html#af84bc9bd237fc875db9d2c0a57fd350e", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_hermitte\_app\_c.js.

## 30.280 doc/html/class\_go\_1\_1\_hermitte\_app\_s.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_hermitte\\_app\\_s](#)

### 30.280.1 Variable Documentation

#### 30.280.1.1 var class\_go\_1\_1\_hermite\_app\_s

##### Initial value:

```
=
[
 ["HermiteAppS", "class_go_1_1_hermite_app_s.html#a2ff07b2356d1a56e949e78c4065872dc", null],
 ["HermiteAppS", "class_go_1_1_hermite_app_s.html#acled3a2e63aace0271734d411b783975", null],
 ["~HermiteAppS", "class_go_1_1_hermite_app_s.html#a051342929f58766beba7535e30854676", null],
 ["getCurves", "class_go_1_1_hermite_app_s.html#a9d86f1ca7c3af1f68fc5057a24e7605a", null],
 ["refineApproximation", "class_go_1_1_hermite_app_s.html#a7b39c26e87f1cbbcf4ec8ca4b7d9173d", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_hermite\_app\_s.js.

### 30.281 doc/html/class\_go\_1\_1\_hermite\_appr\_eval\_surf.js File Reference

#### Variables

- var [class\\_go\\_1\\_1\\_hermite\\_appr\\_eval\\_surf](#)

### 30.281.1 Variable Documentation

#### 30.281.1.1 var class\_go\_1\_1\_hermite\_appr\_eval\_surf

##### Initial value:

```
=
[
 ["HermiteApprEvalSurf", "class_go_1_1_hermite_appr_eval_surf.html#a1a8384dd2d0b16693bc5ceec64e24545", null],
 ["HermiteApprEvalSurf", "class_go_1_1_hermite_appr_eval_surf.html#a543290f3c8b594c33f798fdb2dc61829", null],
 ["~HermiteApprEvalSurf", "class_go_1_1_hermite_appr_eval_surf.html#a5fc02763c166960341d1e972f7be54cd", null],
 ["getGrid", "class_go_1_1_hermite_appr_eval_surf.html#a58fc0b4472305da14ec6c5f53d309a77", null],
 ["getSurface", "class_go_1_1_hermite_appr_eval_surf.html#a5503754fab767b035fe6560761db5e8b", null],
 ["refineApproximation", "class_go_1_1_hermite_appr_eval_surf.html#a45b27206ac7510f9f7a1543a3619a981", null],
 ["removeGridLines", "class_go_1_1_hermite_appr_eval_surf.html#a488bfcc03194718559b1804f9caa6161", null],
 ["setNoSplit", "class_go_1_1_hermite_appr_eval_surf.html#aa2026be935bbdf883acd7b68576d1914", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_hermite\_appr\_eval\_surf.js.

### 30.282 doc/html/class\_go\_1\_1\_hermite\_grid1\_d.js File Reference

#### Variables

- var [class\\_go\\_1\\_1\\_hermite\\_grid1\\_d](#)

### 30.282.1 Variable Documentation

#### 30.282.1.1 var class\_go\_1\_1\_hermite\_grid1\_d

##### Initial value:

```
=
[
 ["HermiteGrid1D", "class_go_1_1_hermite_grid1_d.html#a1b81f053ba34e2a982a6f718d68a433c", null],
 ["HermiteGrid1D", "class_go_1_1_hermite_grid1_d.html#aa3329e478d6cd783b77b509c9fa227ea", null],
 ["~HermiteGrid1D", "class_go_1_1_hermite_grid1_d.html#a934186e38205a15deb1359ab19a913e1", null],
 ["addKnot", "class_go_1_1_hermite_grid1_d.html#af02c064ffac2e277322f461bfac3a47e", null],
 ["dim", "class_go_1_1_hermite_grid1_d.html#a31f7a904e168d258f5918f3b47424632", null],
 ["getData", "class_go_1_1_hermite_grid1_d.html#abcb3e9670ba08213a5a43e613a4f1db8", null],
 ["getKnots", "class_go_1_1_hermite_grid1_d.html#a3b8c8c081bf266a096c292fe7ce25b34", null],
 ["getSegment", "class_go_1_1_hermite_grid1_d.html#a6506abbf1ad2ae7ded6eda6cb24611f9", null],
 ["size", "class_go_1_1_hermite_grid1_d.html#ad698a4b9b46029bd50974dcd9c5e1bf0", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_hermite\_grid1\_d.js.

## 30.283 doc/html/class\_go\_1\_1\_hermite\_grid1\_d\_multi.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_hermite\\_grid1\\_d\\_multi](#)

### 30.283.1 Variable Documentation

#### 30.283.1.1 var class\_go\_1\_1\_hermite\_grid1\_d\_multi

##### Initial value:

```
=
[
 ["HermiteGrid1DMulti", "class_go_1_1_hermite_grid1_d_multi.html#ae7c75ec1562eeb19d988721db3f1da0f", null],
 ["HermiteGrid1DMulti", "class_go_1_1_hermite_grid1_d_multi.html#af1c804b7313f736d1cf6a2fabfc459b5", null],
 ["~HermiteGrid1DMulti", "class_go_1_1_hermite_grid1_d_multi.html#ae00f2b7b0c4a8da2ccce89e5e92ac4a2", null],
 ["addKnot", "class_go_1_1_hermite_grid1_d_multi.html#a54769997303b03a47d40876aec0c5399", null],
 ["dims", "class_go_1_1_hermite_grid1_d_multi.html#a5b754461948e41d978f654cc790a0b12", null],
 ["getData", "class_go_1_1_hermite_grid1_d_multi.html#abe6459e25f1b19d220886ab4b9f98f8e", null],
 ["getKnots", "class_go_1_1_hermite_grid1_d_multi.html#aab9a3975f3402005d4d4b6a71bf4ad35", null],
 ["getSegment", "class_go_1_1_hermite_grid1_d_multi.html#adb5eb3b602206625e441c81fec10f2c8", null],
 ["size", "class_go_1_1_hermite_grid1_d_multi.html#a6ee2725031e371d94e071b8b1cc4e8f0", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_hermite\_grid1\_d\_multi.js.

## 30.284 doc/html/class\_go\_1\_1\_hermite\_grid2\_d.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_hermite\\_grid2\\_d](#)

### 30.284.1 Variable Documentation

#### 30.284.1.1 var class\_go\_1\_1\_hermite\_grid2\_d

##### Initial value:

```
=
[
 ["HermiteGrid2D", "class_go_1_1_hermite_grid2_d.html#ae81d862f9800ddc6e47bdf14f2326b6c", null],
 ["HermiteGrid2D", "class_go_1_1_hermite_grid2_d.html#a91d5e279416a29d2c5ac7f38a6b3765a", null],
 ["~HermiteGrid2D", "class_go_1_1_hermite_grid2_d.html#a35254c89198d2a9349d47b951e54d6db", null],
 ["addKnot", "class_go_1_1_hermite_grid2_d.html#a566111484d7c08960b0801993f53ccd0", null],
 ["dim", "class_go_1_1_hermite_grid2_d.html#a3f76477434404555905d872e8d9404ce", null],
 ["getData", "class_go_1_1_hermite_grid2_d.html#ad282e73379d3d17d1aa31bffd6b61b1cc", null],
 ["getKnots", "class_go_1_1_hermite_grid2_d.html#ac0d38cf0efe2dcf73f907cf46716f939", null],
 ["getNoSplitStatus", "class_go_1_1_hermite_grid2_d.html#addea8cf9b57ffd49a1a197de1833eee6", null],
 ["getSegment", "class_go_1_1_hermite_grid2_d.html#aa9d9011b0d98df4730d03ec86e36ae7c", null],
 ["removeGridLines", "class_go_1_1_hermite_grid2_d.html#a4c276b7cbebadce38d320b5a9b182ald", null],
 ["setNoSplitStatus", "class_go_1_1_hermite_grid2_d.html#aef1b566e91c2e332eb14c49792b9e144", null],
 ["size1", "class_go_1_1_hermite_grid2_d.html#a2dc8d7cba101cef564d6e485c13ad51b", null],
 ["size2", "class_go_1_1_hermite_grid2_d.html#a67525fbddcd890f0ae16ecd8eddd629", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_hermite\_grid2\_d.js.

### 30.285 doc/html/class\_go\_1\_1\_hermite\_interpolator.js File Reference

#### Variables

- var [class\\_go\\_1\\_1\\_hermite\\_interpolator](#)

### 30.285.1 Variable Documentation

#### 30.285.1.1 var class\_go\_1\_1\_hermite\_interpolator

##### Initial value:

```
=
[
 ["HermiteInterpolator", "class_go_1_1_hermite_interpolator.html#a7e25bb3a71ad4970945158567091185a", null],
 ["~HermiteInterpolator", "class_go_1_1_hermite_interpolator.html#ae63830bfb3e2f612ce81c58de90a9bd2", null],
 ["basis", "class_go_1_1_hermite_interpolator.html#afc7bd50c2ce97845859c0d709f9cf8a52", null],
 ["interpolate", "class_go_1_1_hermite_interpolator.html#a68e50f552daa807516e02308fd7d5d39", null],
 ["interpolate", "class_go_1_1_hermite_interpolator.html#a2187f35b4e95e6017fc8e2499d4681cf", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_hermite\_interpolator.js.

### 30.286 doc/html/class\_go\_1\_1\_hyperbola.js File Reference

#### Variables

- var [class\\_go\\_1\\_1\\_hyperbola](#)

### 30.286.1 Variable Documentation

#### 30.286.1.1 var class\_go\_1\_1\_hyperbola

Definition at line 1 of file class\_go\_1\_1\_hyperbola.js.

## 30.287 doc/html/class\_go\_1\_1\_i\_g\_e\_sconverter.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_i\\_g\\_e\\_sconverter](#)

### 30.287.1 Variable Documentation

#### 30.287.1.1 var class\_go\_1\_1\_i\_g\_e\_sconverter

#### Initial value:

```
=
[
 ["IGESconverter", "class_go_1_1_i_g_e_sconverter.html#abc89d7444873a3bf95cb251f48488080", null],
 ["IGESconverter", "class_go_1_1_i_g_e_sconverter.html#a9cc2a58937a1e2f744dc8ebf86e0de6d", null],
 ["~IGESconverter", "class_go_1_1_i_g_e_sconverter.html#af250f2764974dbf68ce3208clad13ee", null],
 ["addGeom", "class_go_1_1_i_g_e_sconverter.html#a6f6556ad80f83e2186437d4d8e0d9136", null],
 ["getColour", "class_go_1_1_i_g_e_sconverter.html#ad4093f5ada9d0ceb642a73b8f201d55f", null],
 ["getGoGeom", "class_go_1_1_i_g_e_sconverter.html#a1078c220e3f7a12d09ae711e64567dba", null],
 ["getGroup", "class_go_1_1_i_g_e_sconverter.html#a49c0a7c5a7b5a09b7ca3c92f4bedb967", null],
 ["header", "class_go_1_1_i_g_e_sconverter.html#a13e6f5f9c53cc67d6eb7f74a27d89e99", null],
 ["minResolution", "class_go_1_1_i_g_e_sconverter.html#a617cd9a7fc03caa270590dc950cce4a5", null],
 ["num_geom", "class_go_1_1_i_g_e_sconverter.html#a3d9a5e1044f2d08e14d97cbdfc8bf6e3", null],
 ["num_group", "class_go_1_1_i_g_e_sconverter.html#a52a55d0882142aa3fad6637464ce1d83", null],
 ["readdisp", "class_go_1_1_i_g_e_sconverter.html#ae3780587f267dbe1829ffeb6a28858e6", null],
 ["readgo", "class_go_1_1_i_g_e_sconverter.html#a9efb92459fed2d88e231d3031cf74390", null],
 ["readIGES", "class_go_1_1_i_g_e_sconverter.html#a558b37c0b9a7850a9ad016c031e95988", null],
 ["readsislcrvs", "class_go_1_1_i_g_e_sconverter.html#aa6c8a2a81c9a91ef646e3626b5bd5a5f", null],
 ["readsislrfs", "class_go_1_1_i_g_e_sconverter.html#ab95f5e0eda003d0d8f7ee4f99d356d48", null],
 ["writedis", "class_go_1_1_i_g_e_sconverter.html#aa48025e09158d3df2aebdd188c1dab05", null],
 ["writego", "class_go_1_1_i_g_e_sconverter.html#a5f25c2b363c2ee650deea96950eab445", null],
 ["writeIGES", "class_go_1_1_i_g_e_sconverter.html#a12blab902e4cbb47fd1c10a324fb5fcb", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_i\_g\_e\_sconverter.js.

## 30.288 doc/html/class\_go\_1\_1\_identity.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_identity](#)

### 30.288.1 Variable Documentation

#### 30.288.1.1 var class\_go\_1\_1\_identity

##### Initial value:

```
=
[
 ["identicalCvs", "class_go_1_1_identity.html#acf3fc01c2e37aa833e398ad4d81d9fee", null],
 ["identicalSfs", "class_go_1_1_identity.html#a14ba65d23ce10a0689f5f353aff2d25a", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_identity.js.

## 30.289 doc/html/class\_go\_1\_1\_implicitize\_curve\_algo.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_implicitize\\_curve\\_algo](#)

### 30.289.1 Variable Documentation

#### 30.289.1.1 var class\_go\_1\_1\_implicitize\_curve\_algo

##### Initial value:

```
=
[
 ["ImplicitizeCurveAlgo", "class_go_1_1_implicitize_curve_algo.html#a8ba7d946e84aa0f3c2fd1cba6369bc84",
 null],
 ["ImplicitizeCurveAlgo", "class_go_1_1_implicitize_curve_algo.html#a64bc2aa9582de885cff5b3596d89d332",
 null],
 ["ImplicitizeCurveAlgo", "class_go_1_1_implicitize_curve_algo.html#aacc0a11e92c206f7db6f731885f6ec43",
 null],
 ["getResultData", "class_go_1_1_implicitize_curve_algo.html#a39d80d4ab3aa45910413b025a1915e38", null]
,
 ["perform", "class_go_1_1_implicitize_curve_algo.html#a6f12e7225bed0c7b2929a0154b958d63", null],
 ["setDegree", "class_go_1_1_implicitize_curve_algo.html#a61f6d229c42e3697581bfb3a20d85641", null],
 ["setTolerance", "class_go_1_1_implicitize_curve_algo.html#aa3f875e818a5e92ab1dc430a967a0504", null],
 ["useSplineCurve", "class_go_1_1_implicitize_curve_algo.html#a939b612b7484c57f9b16c5968d14c9cc", null
]
]
```

Definition at line 1 of file class\_go\_1\_1\_implicitize\_curve\_algo.js.

## 30.290 doc/html/class\_go\_1\_1\_implicitize\_curve\_and\_vector\_algo.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_implicitize\\_curve\\_and\\_vector\\_algo](#)

### 30.290.1 Variable Documentation

#### 30.290.1.1 var class\_go\_1\_1\_implicitize\_curve\_and\_vector\_algo

##### Initial value:

```
=
[
 ["ImplicitizeCurveAndVectorAlgo", "
 class_go_1_1_implicitize_curve_and_vector_algo.html#a2835321bb2f08b93348677e7de9462d7", null],
 ["ImplicitizeCurveAndVectorAlgo", "
 class_go_1_1_implicitize_curve_and_vector_algo.html#ac714e00d39489b629f567c74076e9f62", null],
 ["ImplicitizeCurveAndVectorAlgo", "
 class_go_1_1_implicitize_curve_and_vector_algo.html#a4c10042b074532089af07aaa090b4eb1", null],
 ["getResultData", "
 class_go_1_1_implicitize_curve_and_vector_algo.html#a123db1e99d3ab524fd4b8452ba831b8b", null],
 ["perform", "class_go_1_1_implicitize_curve_and_vector_algo.html#a6e679ce6cf7377743e56f55a829dfea2",
 null],
 ["setDegree", "class_go_1_1_implicitize_curve_and_vector_algo.html#a01b6d54ae7bdf9c71dd3ab989c1a7487",
 null],
 ["setTolerance", "
 class_go_1_1_implicitize_curve_and_vector_algo.html#a821e18cb61f8fbe530409275c9a0c985", null],
 ["useSplineCurve", "
 class_go_1_1_implicitize_curve_and_vector_algo.html#aeb94fd01cadbfbedb760eee2f6daea73", null],
 ["useVector", "class_go_1_1_implicitize_curve_and_vector_algo.html#a96f5c125df5dd849bdf081cfde23af96",
 null]
]
```

Definition at line 1 of file class\_go\_1\_1\_implicitize\_curve\_and\_vector\_algo.js.

## 30.291 doc/html/class\_go\_1\_1\_implicitize\_point\_cloud\_algo.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_implicitize\\_point\\_cloud\\_algo](#)

### 30.291.1 Variable Documentation

#### 30.291.1.1 var class\_go\_1\_1\_implicitize\_point\_cloud\_algo

##### Initial value:

```
=
[
 ["ImplicitizePointCloudAlgo", "
 class_go_1_1_implicitize_point_cloud_algo.html#a56817f5ef2277c80ffaafc1d7a9ebba", null],
 ["ImplicitizePointCloudAlgo", "
 class_go_1_1_implicitize_point_cloud_algo.html#a2033dedcaeeb3c2ff8122bd71763cb8d", null],
 ["ImplicitizePointCloudAlgo", "
 class_go_1_1_implicitize_point_cloud_algo.html#a5d83ca3dda5e07902d984fe3f96e892", null],
 ["getResultData", "class_go_1_1_implicitize_point_cloud_algo.html#ab26d12a65128964e9bca2ba889020272",
 null],
 ["perform", "class_go_1_1_implicitize_point_cloud_algo.html#a13a2323f9e7364577c6129fcb09f80ef", null]
 ,
 ["setDegree", "class_go_1_1_implicitize_point_cloud_algo.html#adb74703b42a2db51f58ae2ee0ff11979", null
],
 ["setTolerance", "class_go_1_1_implicitize_point_cloud_algo.html#a8f71deb3bb6843f905b0f7c6a09875f2",
 null],
 ["usePointCloud", "class_go_1_1_implicitize_point_cloud_algo.html#aec01db50bf187f939a3b0b33dfceb5aa",
 null]
]
```

Definition at line 1 of file class\_go\_1\_1\_implicitize\_point\_cloud\_algo.js.

## 30.292 doc/html/class\_go\_1\_1\_implicitize\_surface\_algo.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_implicitize\\_surface\\_algo](#)

### 30.292.1 Variable Documentation

#### 30.292.1.1 var class\_go\_1\_1\_implicitize\_surface\_algo

##### Initial value:

```
=
[
 ["ImplicitizeSurfaceAlgo", "
 class_go_1_1_implicitize_surface_algo.html#ac8cfe4021e223dca749e76ce3d23e681", null],
 ["ImplicitizeSurfaceAlgo", "
 class_go_1_1_implicitize_surface_algo.html#a8ab33df8d56a57d9e16cc89194a6ffb0", null],
 ["ImplicitizeSurfaceAlgo", "
 class_go_1_1_implicitize_surface_algo.html#a5257feced044874f73c4c6dbcc6c2e61", null],
 ["getResultData", "class_go_1_1_implicitize_surface_algo.html#aae034eac7b892ee40184c58f92cbd646", null
],
 ["perform", "class_go_1_1_implicitize_surface_algo.html#a55760971a434c9855fdc2d11d40093c7", null],
 ["setDegree", "class_go_1_1_implicitize_surface_algo.html#ac28c46855c2ca5dcfc2196cfcaae4399", null],
 ["setTolerance", "class_go_1_1_implicitize_surface_algo.html#ac6bc70f82ced59b2038835e22aec0c0", null
],
 ["useSplineSurface", "class_go_1_1_implicitize_surface_algo.html#a0ae851ad58fce208d8f9e868a5b758c7",
 null]
]
```

Definition at line 1 of file class\_go\_1\_1\_implicitize\_surface\_algo.js.

## 30.293 doc/html/class\_go\_1\_1\_index\_mesh2\_d\_iterator.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_index\\_mesh2\\_d\\_iterator](#)

### 30.293.1 Variable Documentation

#### 30.293.1.1 var class\_go\_1\_1\_index\_mesh2\_d\_iterator

##### Initial value:

```
=
[
 ["IndexMesh2DIterator", "class_go_1_1_index_mesh2_d_iterator.html#a489ffd145fab0714f8e19c6db2051adf",
 null],
 ["operator!=", "class_go_1_1_index_mesh2_d_iterator.html#ac141669bbe4cee43488d112d8a6a0a14", null],
 ["operator*", "class_go_1_1_index_mesh2_d_iterator.html#aad063a5ce63ee01ce6c0695a96a87f2b", null],
 ["operator++", "class_go_1_1_index_mesh2_d_iterator.html#a75ff1f9dfd67889a0f0128cb95db10a", null],
 ["operator==", "class_go_1_1_index_mesh2_d_iterator.html#a6b74f5c71ddfff353ee5d46ff525d7bc", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_index\_mesh2\_d\_iterator.js.



## 30.294 doc/html/class\_go\_1\_1\_int\_crv\_evaluator.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_int\\_crv\\_evaluator](#)

### 30.294.1 Variable Documentation

#### 30.294.1.1 var class\_go\_1\_1\_int\_crv\_evaluator

##### Initial value:

```
=
[
 ["IntCrvEvaluator", "class_go_1_1_int_crv_evaluator.html#a588d5572ae712c10fb0619c330eb42f3", null],
 ["~IntCrvEvaluator", "class_go_1_1_int_crv_evaluator.html#a1311da99f6ae0f828ce9f9acd7010f23", null],
 ["approximationOK", "class_go_1_1_int_crv_evaluator.html#ab173e8c1a5e26341deadfb30c8d28473", null],
 ["dim", "class_go_1_1_int_crv_evaluator.html#a55988451a46e7982b16d33e976917e68", null],
 ["end", "class_go_1_1_int_crv_evaluator.html#a26b27e0e2dd99a85d0d91a74f7e9ee2b", null],
 ["eval", "class_go_1_1_int_crv_evaluator.html#ab9f4d31119d434a7399b4f57a99c7efb", null],
 ["eval", "class_go_1_1_int_crv_evaluator.html#a212ac4e6d0896e12fef2d5a0bd895e58", null],
 ["getMaxErr", "class_go_1_1_int_crv_evaluator.html#a84caa845e65e381a79545a200b52dc3f", null],
 ["nmbCvs", "class_go_1_1_int_crv_evaluator.html#a2a6d139b257cbab81385d1a5b3f03905", null],
 ["resetErr", "class_go_1_1_int_crv_evaluator.html#a151e46048397db954352f388c409c2c3", null],
 ["start", "class_go_1_1_int_crv_evaluator.html#ad5622093eda08e7c27d4a06e5d9e0060", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_int\_crv\_evaluator.js.

## 30.295 doc/html/class\_go\_1\_1\_int\_results\_comp\_cv.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_int\\_results\\_comp\\_cv](#)

### 30.295.1 Variable Documentation

#### 30.295.1.1 var class\_go\_1\_1\_int\_results\_comp\_cv

##### Initial value:

```
=
[
 ["IntResultsCompCv", "class_go_1_1_int_results_comp_cv.html#a39913161fc82b56def0dff1dfc8ac964", null],
 ["IntResultsCompCv", "class_go_1_1_int_results_comp_cv.html#adb349e45f1f3d0db949dc41544595b42", null],
 ["~IntResultsCompCv", "class_go_1_1_int_results_comp_cv.html#a15116e6ea54db12ad8d530ac633a4f78", null],
 ["addIntCv", "class_go_1_1_int_results_comp_cv.html#ae2899f24ed3eafee66fd43e9932a7b0e", null],
 ["addIntPt", "class_go_1_1_int_results_comp_cv.html#ae57b06c0659eeb57546103c2e49581fa", null],
 ["getIntersectionCurves", "class_go_1_1_int_results_comp_cv.html#ab5a011b67270f12265151fc393caad07", null],
 ["getIntersectionPoints", "class_go_1_1_int_results_comp_cv.html#a413551fbc611a02b8eeb65ba79015256", null],
 ["hasIntCurves", "class_go_1_1_int_results_comp_cv.html#a1a87b6dfd2ae43340f88b25321b515c1", null],
 ["hasIntersections", "class_go_1_1_int_results_comp_cv.html#a6e3f3295623b4f774c81746ee70be724", null],
 ["hasIntPoints", "class_go_1_1_int_results_comp_cv.html#a553c3b81a105302a5f99539c7b806ddd", null],
 ["nmbCurveSegments", "class_go_1_1_int_results_comp_cv.html#aa1ab06a91745daa8bfc8fd368e1de952", null],
 ["nmbIntPoints", "class_go_1_1_int_results_comp_cv.html#add3a6f4ed07c05dc81b96eef002d00a2", null],
 ["tesselate", "class_go_1_1_int_results_comp_cv.html#a0a2c6122f9a871bf7b9666d2e0ad2a2f", null],
 ["tesselate", "class_go_1_1_int_results_comp_cv.html#aaa85f9ea710d6c6700b49456212c73ba", null],
 ["tesselate", "class_go_1_1_int_results_comp_cv.html#a679437896e0247ae64a4375234f1e270", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_int\_results\_comp\_cv.js.

## 30.296 doc/html/class\_go\_1\_1\_int\_results\_model.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_int\\_results\\_model](#)

### 30.296.1 Variable Documentation

#### 30.296.1.1 var class\_go\_1\_1\_int\_results\_model

##### Initial value:

```
=
[
 ["IntResultsModel", "class_go_1_1_int_results_model.html#a37c7c5856b4bcbdc4c768280a89332fa", null],
 ["~IntResultsModel", "class_go_1_1_int_results_model.html#a2c1bb1fbabe1e108981ac0991c7c4409", null],
 ["addLineInfo", "class_go_1_1_int_results_model.html#a3794036a737801309c23c848582e48bd", null],
 ["addPlaneInfo", "class_go_1_1_int_results_model.html#a45214769a80e4576533438d217bb4428", null],
 ["hasIntCurves", "class_go_1_1_int_results_model.html#a187610d37b96d7f4aa1e21e494270dc6", null],
 ["hasIntersections", "class_go_1_1_int_results_model.html#a4d12d17e152778ee104ce8647a54ec1e", null],
 ["hasIntPoints", "class_go_1_1_int_results_model.html#a7b3d18f6047b9702f10f3b01c8970606", null],
 ["nmbCurveSegments", "class_go_1_1_int_results_model.html#a784cbb4e684109f3d3890b4eee84ff08", null],
 ["nmbIntPoints", "class_go_1_1_int_results_model.html#abca77fdcbf99084d17de8e1d8dcb90ec", null],
 ["tessellate", "class_go_1_1_int_results_model.html#af758fa08ald9a2e291102486e4029128", null],
 ["tessellate", "class_go_1_1_int_results_model.html#a5cba20336f32b65cfd840edbf8f46bbd", null],
 ["tessellate", "class_go_1_1_int_results_model.html#aa65af6225021c30ea430fc9287191cd9", null],
 ["line_", "class_go_1_1_int_results_model.html#a26a8c35b714e8b4d2197d3733bcd1b40", null],
 ["numpar_", "class_go_1_1_int_results_model.html#a5e5b3f81b17ef87819a0cd45a308ddf6", null],
 ["plane_", "class_go_1_1_int_results_model.html#a6d576fb9cf1d034f020d2ccc660d0ab0", null],
 ["type_", "class_go_1_1_int_results_model.html#a5590accd90a55e8c52874c67ae025e7d", null]
]
```

Definition at line 1 of file `class_go_1_1_int_results_model.js`.

## 30.297 doc/html/class\_go\_1\_1\_int\_results\_sf\_model.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_int\\_results\\_sf\\_model](#)

### 30.297.1 Variable Documentation

#### 30.297.1.1 var class\_go\_1\_1\_int\_results\_sf\_model

##### Initial value:

```

=
[
 ["IntResultsSfModel", "class_go_1_1_int_results_sf_model.html#a28ca82d75321afc3924e9a324126b16c", null],
 ["IntResultsSfModel", "class_go_1_1_int_results_sf_model.html#a5f7779331a8c02e73cf5ea82db399a8d", null],
 ["~IntResultsSfModel", "class_go_1_1_int_results_sf_model.html#a28e255c2d40fcf43a6dd9a414adb912f", null],
 ["addIntCvs", "class_go_1_1_int_results_sf_model.html#aa2895b90ffe93590a1540bbe2bbcef40", null],
 ["addIntPts", "class_go_1_1_int_results_sf_model.html#adf7cfa67e69fcc8e43790705fd2a11d2", null],
 ["getIntersectionCurves", "class_go_1_1_int_results_sf_model.html#a60ffd1eb09ed472a55fd4d782aadf016", null],
 ["getIntersectionPoints", "class_go_1_1_int_results_sf_model.html#abc0327970bcf4a9205cf18ffb29e9992", null],
 ["hasIntCurves", "class_go_1_1_int_results_sf_model.html#af2438d92c1844f33c2dc5479d0e0a126", null],
 ["hasIntersections", "class_go_1_1_int_results_sf_model.html#a22560da1c5a2ba7ed7a0819d6b84c7c2", null],
 ["hasIntPoints", "class_go_1_1_int_results_sf_model.html#a09a75034c35c6698915e9ca7f9b6f26e", null],
 ["nmbCurveSegments", "class_go_1_1_int_results_sf_model.html#ab0b2785827237422cf3a2236b9c22e0b", null],
 ["nmbIntPoints", "class_go_1_1_int_results_sf_model.html#a7ff0fecdb5e074a38d23f9488811c", null],
 ["tessellate", "class_go_1_1_int_results_sf_model.html#ab7833b7cdf1ab93b3635e2d68cea77e4", null],
 ["tessellate", "class_go_1_1_int_results_sf_model.html#a0482f11c47cb1d5aa53dd87f7fa59f5", null],
 ["tessellate", "class_go_1_1_int_results_sf_model.html#a9f75cafb391badc9153f52c01393e207", null]
]

```

Definition at line 1 of file `class_go_1_1_int_results_sf_model.js`.

## 30.298 doc/html/class\_go\_1\_1\_interpolated\_intersection\_curve.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_interpolated\\_intersection\\_curve](#)

### 30.298.1 Variable Documentation

#### 30.298.1.1 var class\_go\_1\_1\_interpolated\_intersection\_curve

#### Initial value:

```

=
[
 ["~InterpolatedIntersectionCurve", "class_go_1_1_interpolated_intersection_curve.html#a2cc666f8a99ef65f09ed7ec33d83edd", null],
 ["evaluateAt", "class_go_1_1_interpolated_intersection_curve.html#a55691f39dd225caf3c4f2ecafea9c488", null],
 ["getCurve", "class_go_1_1_interpolated_intersection_curve.html#a2298291895656223adb4885e21962ddc", null],
 ["getGuidePointTangent", "class_go_1_1_interpolated_intersection_curve.html#a54741b7a4c648ec5cdb38dba9f879fc4", null],
 ["getParamCurve", "class_go_1_1_interpolated_intersection_curve.html#ad2e38f7ec50c9e6b142599ef4af57d87", null],
 ["getParamSpan", "class_go_1_1_interpolated_intersection_curve.html#ab75e62cd368335a219751766521c2752", null],
 ["isDegenerated", "class_go_1_1_interpolated_intersection_curve.html#acb486e3188e2799ec14dd3d394ae7f51", null],
 ["isIsocurve", "class_go_1_1_interpolated_intersection_curve.html#a375594c872bd6bb5396b58a277a3779c", null],
 ["read", "class_go_1_1_interpolated_intersection_curve.html#a91bc2564d6bc1f840f451e9850dcae5f", null],
 ["refine", "class_go_1_1_interpolated_intersection_curve.html#a0def052e3c76d761f6803f6d29a36d77", null],
 ["write", "class_go_1_1_interpolated_intersection_curve.html#aa6f5f6040f556d1b2c85846ae264c15b", null],
 ["constructIntersectionCurve", "class_go_1_1_interpolated_intersection_curve.html#a6ab34095fc2901f18aff0d440f5f9ea5", null]
]

```

Definition at line 1 of file `class_go_1_1_interpolated_intersection_curve.js`.

## 30.299 doc/html/class\_go\_1\_1\_interpolator.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_interpolator](#)

### 30.299.1 Variable Documentation

#### 30.299.1.1 var class\_go\_1\_1\_interpolator

#### Initial value:

```
=
[
 ["~-Interpolator", "class_go_1_1_interpolator.html#a89e82a127c1680a5f65c7ecb1a2e989f", null],
 ["basis", "class_go_1_1_interpolator.html#af3b4041503a093bc17a8dbf7d62cd49d", null],
 ["interpolate", "class_go_1_1_interpolator.html#aa30a5694c065bf3664df3f269a16de77", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_interpolator.js.

## 30.300 doc/html/class\_go\_1\_1\_intersection\_curve.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_intersection\\_curve](#)

### 30.300.1 Variable Documentation

#### 30.300.1.1 var class\_go\_1\_1\_intersection\_curve

#### Initial value:

```
=
[
 ["~-IntersectionCurve", "class_go_1_1_intersection_curve.html#a6cf2e4736ade62dbb3985ef072d034e0", null],
 ["IntersectionCurve", "class_go_1_1_intersection_curve.html#ac4003ff43d0327c953305800ac2364cf", null],
 ["evaluateAt", "class_go_1_1_intersection_curve.html#a08b1e3c0a8b0a4c54351d79e035cc56a", null],
 ["getCurve", "class_go_1_1_intersection_curve.html#a0fa20bdfc6319f0baac26ce841b8b1bc", null],
 ["getGuidePoint", "class_go_1_1_intersection_curve.html#af9939e163fe8bf63c937b44df9432133", null],
 ["getGuidePointTangent", "class_go_1_1_intersection_curve.html#a88760b129ab2bb3ca6a7176e544339c5", null],
 ["getParamCurve", "class_go_1_1_intersection_curve.html#aa2c34260ae6d57e4099c51231e769f87", null],
 ["getParamSpan", "class_go_1_1_intersection_curve.html#acbale0e9e4fc27bfa2f2c5aa8a9db7cf", null],
 ["isDegenerated", "class_go_1_1_intersection_curve.html#a3601834a09fa3ca28a806c6890352ca1", null],
 ["isIsocurve", "class_go_1_1_intersection_curve.html#ac9ff615827c0995f089f7f6c8c687fd6", null],
 ["numGuidePoints", "class_go_1_1_intersection_curve.html#a54f8c14777db7e2cf1ea606ae9c90479", null],
 ["refine", "class_go_1_1_intersection_curve.html#aab981c74101ef930bdaa69659c56aae6", null],
 ["writeIPointsToStream", "class_go_1_1_intersection_curve.html#a56df8bd8dc13845e89a18014952bf106", null],
 ["constructIntersectionCurve", "class_go_1_1_intersection_curve.html#a6ab34095fc2901f18aff0d440f5f9ea5", null],
 ["ipoints_", "class_go_1_1_intersection_curve.html#a68cfad916eee474a4a67c5fe59d960c5", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_intersection\_curve.js.

## 30.301 doc/html/class\_go\_1\_1\_intersection\_link.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_intersection\\_link](#)

### 30.301.1 Variable Documentation

#### 30.301.1.1 var class\_go\_1\_1\_intersection\_link

##### Initial value:

```
=
[
 ["IntersectionLink", "class_go_1_1_intersection_link.html#a477cd7b24eff533968ee2bf98b835e21", null],
 ["copyMetaInformation", "class_go_1_1_intersection_link.html#a2bb0c1fe4e2d007c6dbe90fe52b6818b", null],
 ["crosses", "class_go_1_1_intersection_link.html#a67f77e2cc8b5e6a4aedcfd2863526ccd", null],
 ["delimitsPAC", "class_go_1_1_intersection_link.html#a2a3288dad241e87d64ab03bc4645cafe", null],
 ["dumpToStream", "class_go_1_1_intersection_link.html#afba207d4b816da43af23c7e7200130ac", null],
 ["getIntersectionPoints", "class_go_1_1_intersection_link.html#abb105a7042d65505d4fe6313d45aec59", null],
 ["getIntersectionPoints", "class_go_1_1_intersection_link.html#ae10db109ef2c456e73489b4fee5eb61b", null],
 ["getOtherPoint", "class_go_1_1_intersection_link.html#a3f76e5433759be357816d4b748600b34", null],
 ["isAttachedTo", "class_go_1_1_intersection_link.html#ad68a81877afad6ce1937beb724ce6f71", null],
 ["isIsoparametric", "class_go_1_1_intersection_link.html#a7657127d5b019eba307308b8289ac570", null],
 ["isIsoparametricIn", "class_go_1_1_intersection_link.html#a1113623834f5105dbdb86e0a3341a84a", null],
 ["linkType", "class_go_1_1_intersection_link.html#a0a766df4c33506fd0e0ef77dc3dc1154", null],
 ["linkType", "class_go_1_1_intersection_link.html#af16fc0f41eb6ac32d77f3bfbfc9a76e0", null],
 ["numParams", "class_go_1_1_intersection_link.html#a0f9b865218db7ac600972f4e44df4d66", null],
 ["setIsoparametricIn", "class_go_1_1_intersection_link.html#a8f3ff44ecd585946b0c45bd8813dbf7e", null],
 ["setPAC", "class_go_1_1_intersection_link.html#a28e5aa3a8a57c2a9dcec078703b09dde", null],
 ["writeInfo", "class_go_1_1_intersection_link.html#ad85f22a9a2be902612a768e8425fc671", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_intersection\_link.js.

## 30.302 doc/html/class\_go\_1\_1\_intersection\_point.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_intersection\\_point](#)

### 30.302.1 Variable Documentation

#### 30.302.1.1 var class\_go\_1\_1\_intersection\_point

Definition at line 1 of file class\_go\_1\_1\_intersection\_point.js.

## 30.303 doc/html/class\_go\_1\_1\_intersection\_pool.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_intersection\\_pool](#)

### 30.303.1 Variable Documentation

#### 30.303.1.1 var class\_go\_1\_1\_intersection\_pool

Definition at line 1 of file class\_go\_1\_1\_intersection\_pool.js.

## 30.304 doc/html/class\_go\_1\_1\_intersector.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_intersector](#)

### 30.304.1 Variable Documentation

#### 30.304.1.1 var class\_go\_1\_1\_intersector

Definition at line 1 of file class\_go\_1\_1\_intersector.js.

## 30.305 doc/html/class\_go\_1\_1\_intersector2\_obj.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_intersector2\\_obj](#)

### 30.305.1 Variable Documentation

#### 30.305.1.1 var class\_go\_1\_1\_intersector2\_obj

Definition at line 1 of file class\_go\_1\_1\_intersector2\_obj.js.

## 30.306 doc/html/class\_go\_1\_1\_intersector\_alg\_par.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_intersector\\_alg\\_par](#)

### 30.306.1 Variable Documentation

#### 30.306.1.1 var class\_go\_1\_1\_intersector\_alg\_par

##### Initial value:

```
=
[
 ["IntersectorAlgPar", "class_go_1_1_intersector_alg_par.html#a1c694796c371a332dd93b5eb6f53dbcd", null],
 ["~IntersectorAlgPar", "class_go_1_1_intersector_alg_par.html#ac45bb8bc7ea8c2b3ed54fa9a43a171d9", null],
 ["checkCoincidence", "class_go_1_1_intersector_alg_par.html#a66286ac52cb3eb4425cd0317d4220119", null],
 ["complexityReduced", "class_go_1_1_intersector_alg_par.html#a6704561a8cf298374b282b1c11d26237", null],
 ["compute", "class_go_1_1_intersector_alg_par.html#a48b07877d418bce39c4c358ea484b46e", null],
 ["doSubdivide", "class_go_1_1_intersector_alg_par.html#aa68c2f708e7e5a9f9b0fad0bd034ebfc", null],
 ["getBoundaryIntersections", "class_go_1_1_intersector_alg_par.html#abde4c8ecb977e3eaebbcd8b8e5011d1a", null],
 ["getResult", "class_go_1_1_intersector_alg_par.html#ad2a2470a8b2f294bd6fed0ed6c8baea7", null],
 ["handleComplexity", "class_go_1_1_intersector_alg_par.html#aa0a325a32e1aebc7d037d289b7392a5f", null],
 ["isLinear", "class_go_1_1_intersector_alg_par.html#a0cb169819cf12698621460ef2de17f90", null],
 ["linearCase", "class_go_1_1_intersector_alg_par.html#ad329ae8fd44ff253f65a550e6d98e789", null],
 ["microCase", "class_go_1_1_intersector_alg_par.html#a45f7221f062bfd84d22cf8602d7815e3", null],
 ["numParams", "class_go_1_1_intersector_alg_par.html#a37538dfac8aae3bf8b80d8029c06fbf9", null],
 ["performInterception", "class_go_1_1_intersector_alg_par.html#a9f3d35e28e826f010b8ace562b46867e", null],
 ["printObjs", "class_go_1_1_intersector_alg_par.html#aa5794587a6cfed1116cdad7f3d11ebb8", null],
 ["printDebugInfo", "class_go_1_1_intersector_alg_par.html#ab0dece88519c5dabde40bb1f0d2c1758", null],
 ["repairIntersections", "class_go_1_1_intersector_alg_par.html#a876949b1be857c253173f43981bf2da7", null],
 ["simpleCase", "class_go_1_1_intersector_alg_par.html#aa7cb69632605ec30f8599635426a77c4", null],
 ["updateIntersections", "class_go_1_1_intersector_alg_par.html#aeac71c7d5eae6e528d0dff692dae85c4", null],
 ["algobj_int_", "class_go_1_1_intersector_alg_par.html#a6e26ce4d645ba59581bcfd4863522bb8", null],
 ["int_func_const_", "class_go_1_1_intersector_alg_par.html#a410ff9bdc7efb7c7dcc1fa83d0497019", null],
 ["param_int_", "class_go_1_1_intersector_alg_par.html#ab5851b92d89b38e33ccd25df1e396b9e", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_intersector\_alg\_par.js.

## 30.307 doc/html/class\_go\_1\_1\_intersector\_func\_const.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_intersector\\_func\\_const](#)

### 30.307.1 Variable Documentation

#### 30.307.1.1 var class\_go\_1\_1\_intersector\_func\_const

##### Initial value:

```

=
[
 ["IntersectorFuncConst", "class_go_1_1_intersector_func_const.html#a6bcb2cc97acf2965e4cfad9e5d28e22c",
 null],
 ["~IntersectorFuncConst", "class_go_1_1_intersector_func_const.html#a983904e7810e64e408440ee24f855eab",
 null],
 ["checkCoincidence", "class_go_1_1_intersector_func_const.html#af5c7947ff3c6a1a080a0aee6e3d044e7",
 null],
 ["complexityReduced", "class_go_1_1_intersector_func_const.html#af9008c446f653e3e40a69ecf7c87ae9c",
 null],
 ["doSubdivide", "class_go_1_1_intersector_func_const.html#a57925a75587862d58ff47c983db6858a", null],
 ["getBoundaryIntersections", "
class_go_1_1_intersector_func_const.html#ab1655bc8826029c5ee7b027f1a5fe65a", null],
 ["handleComplexity", "class_go_1_1_intersector_func_const.html#ad3578e18407c67f5104c659f1c2599b1",
 null],
 ["isLinear", "class_go_1_1_intersector_func_const.html#ab486f36f5ec691973af45a53249e7b72", null],
 ["linearCase", "class_go_1_1_intersector_func_const.html#a95cc7ee76c02164a95d1d6117ec82196", null],
 ["lowerOrderIntersector", "class_go_1_1_intersector_func_const.html#a78ee9f26cfec9eb60915138474000d99",
 null],
 ["microCase", "class_go_1_1_intersector_func_const.html#a64378e6a401f57776a81e7bc26d8e70f", null],
 ["performInterception", "class_go_1_1_intersector_func_const.html#a1180d08f6237927a83a9dffda02726",
 null],
 ["print_objs", "class_go_1_1_intersector_func_const.html#ac18633acaa75673422e4c6ebc685404f", null],
 ["printDebugInfo", "class_go_1_1_intersector_func_const.html#a44f2d14e061331d73c796feal7f6e9be", null
],
 ["simpleCase", "class_go_1_1_intersector_func_const.html#ac40bcda7d02c8f607a36c3beabbcc1bf", null],
 ["updateIntersections", "class_go_1_1_intersector_func_const.html#abb3abcdd46d34046f81562e3ce140c58",
 null],
 ["IntersectorAlgPar", "class_go_1_1_intersector_func_const.html#ae0bce107ca8435c4f869edeb948b496cf",
 null],
 ["C_", "class_go_1_1_intersector_func_const.html#a6f6bea7870b300865b13e632ba7ff999", null],
 ["func_int_", "class_go_1_1_intersector_func_const.html#abc8f68ae80af7bc1a741a24c728d61aa", null]
]

```

Definition at line 1 of file `class_go_1_1_intersector_func_const.js`.

## 30.308 doc/html/class\_go\_1\_1\_isogeometric\_block.js File Reference

### Variables

- var `class_go_1_1_isogeometric_block`

### 30.308.1 Variable Documentation

#### 30.308.1.1 var `class_go_1_1_isogeometric_block`

#### Initial value:

```

=
[
 ["IsogeometricBlock", "class_go_1_1_isogeometric_block.html#adbaef92b058186f56820f17ea7baf993", null]
 ,
 ["~IsogeometricBlock", "class_go_1_1_isogeometric_block.html#ae9dba3fa0afac1125a11216937c980a", null
],
 ["asIsogeometricSfBlock", "class_go_1_1_isogeometric_block.html#a89ceba2d4508f702d5b8102a9d0e327e",
 null],
 ["asIsogeometricVolBlock", "class_go_1_1_isogeometric_block.html#a2956de5b336cf9c1cf00178642942ea3",
 null],
 ["basis", "class_go_1_1_isogeometric_block.html#a5108d3b186ad71c55650b67d9f1c52b3", null],
 ["degree", "class_go_1_1_isogeometric_block.html#a142fc052d8bfe32d3739857a298d70aa", null],
 ["distinctKnots", "class_go_1_1_isogeometric_block.html#aafdc258372ef2ca1e6ce84a063b39dff", null],
 ["erasePreEvaluatedBasisFunctions", "
class_go_1_1_isogeometric_block.html#abfae372b8a17aa83ae93896b60c9a66a", null],
 ["getNeighbour", "class_go_1_1_isogeometric_block.html#ac64b26be4b6943fe4c1a4edcf08c40c1", null],
 ["getNmbOfBoundaryConditions", "class_go_1_1_isogeometric_block.html#a45e7fc4d5f482308fa1e88cb9105b09c",
 null],
 ["getNmbOfPointBdConditions", "class_go_1_1_isogeometric_block.html#a53f20a3c1bc3ad279009d3ac00bde336",
 null],
 ["getTolerances", "class_go_1_1_isogeometric_block.html#a5befb6b5755ad2f6ae346c419461714b", null],
 ["increaseGeometryDegree", "class_go_1_1_isogeometric_block.html#aac6ae87fb733aeebfd3966e0d84f969a",
 null]
]

```



```

 null],
 ["isNeighbour", "class_go_1_1_isogeometric_block.html#ad8162b7aabb61ec205481cab51e924e4", null],
 ["knots", "class_go_1_1_isogeometric_block.html#ad515166c60c7e702dc9f27d48f7122d3", null],
 ["model", "class_go_1_1_isogeometric_block.html#a3291268349ffe201233515dd98180e51", null],
 ["nmbCoefs", "class_go_1_1_isogeometric_block.html#a1df8869cd9bf7fc7b81817efd6a383d5", null],
 ["nmbCoefs", "class_go_1_1_isogeometric_block.html#afec6077376d14cc322481b69544ce4ac", null],
 ["nmbOfNeighbours", "class_go_1_1_isogeometric_block.html#aba8aadfa2b253123d0843d9ddcaebadb", null],
 ["refineGeometry", "class_go_1_1_isogeometric_block.html#a183d6da50bc059de6c2e9ad9d450c2b2", null],
 ["refineGeometry", "class_go_1_1_isogeometric_block.html#a91d31036cf25a770ef78958f943a20e9", null]
]

```

Definition at line 1 of file class\_go\_1\_1\_isogeometric\_block.js.

## 30.309 doc/html/class\_go\_1\_1\_isogeometric\_model.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_isogeometric\\_model](#)

### 30.309.1 Variable Documentation

#### 30.309.1.1 var class\_go\_1\_1\_isogeometric\_model

#### Initial value:

```

=
[
 ["IsogeometricModel", "class_go_1_1_isogeometric_model.html#a7713fd03f049fe5869c2c7f7aed83e44", null],
 ["~IsogeometricModel", "class_go_1_1_isogeometric_model.html#ac3b0754afc9a05cba86424d484354c40", null],
 ["getNmbOfBoundaries", "class_go_1_1_isogeometric_model.html#a3a93321f31a8fd0bf4e417f9a37d17c5", null],
 ["getTolerances", "class_go_1_1_isogeometric_model.html#ac0446a93c3803b137c6b5a9a2e68e27b", null],
 ["setMinimumDegree", "class_go_1_1_isogeometric_model.html#a88b11d5e9d7207ca6d648f4cc90263e7", null],
 ["setTolerances", "class_go_1_1_isogeometric_model.html#aa1ccd0d24f6689b4454bfdd8dbbce279", null],
 ["updateSolutionSplineSpace", "class_go_1_1_isogeometric_model.html#ae66cb05951237f68a9d8d25c044d7c04", null],
 ["updateSolutionSplineSpace", "class_go_1_1_isogeometric_model.html#ae66cb05951237f68a9d8d25c044d7c04", null]
]

```

Definition at line 1 of file class\_go\_1\_1\_isogeometric\_model.js.

## 30.310 doc/html/class\_go\_1\_1\_isogeometric\_sf\_block.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_isogeometric\\_sf\\_block](#)

### 30.310.1 Variable Documentation

#### 30.310.1.1 var class\_go\_1\_1\_isogeometric\_sf\_block

Definition at line 1 of file class\_go\_1\_1\_isogeometric\_sf\_block.js.

## 30.311 doc/html/class\_go\_1\_1\_isogeometric\_sf\_model.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_isogeometric\\_sf\\_model](#)

### 30.311.1 Variable Documentation

#### 30.311.1.1 var class\_go\_1\_1\_isogeometric\_sf\_model

#### Initial value:

```
=
[
 ["IsogeometricSfModel", "class_go_1_1_isogeometric_sf_model.html#a616e969de7d558342e6efc6d7af6d9ba",
 null],
 ["~IsogeometricSfModel", "class_go_1_1_isogeometric_sf_model.html#a931c2cabd748d485b690c56acf67ecb8",
 null],
 ["addBoundaryCond", "class_go_1_1_isogeometric_sf_model.html#a01182f12f04c8e064649fd24ee0a7fe0", null
],
 ["addDirichletPointBdCond", "class_go_1_1_isogeometric_sf_model.html#ab5c10a272b7025c1684da9a78b5c10ab",
 null],
 ["getBoundary", "class_go_1_1_isogeometric_sf_model.html#a06136e015cb7cf3ce30f44457006e930", null],
 ["getIsogeometricBlocks", "class_go_1_1_isogeometric_sf_model.html#a3b8282ac229e6260944e9d459c51f538",
 null],
 ["getNmbOfBoundaries", "class_go_1_1_isogeometric_sf_model.html#a12df67ab6291c252cdd9f2d5d0c7e259",
 null],
 ["getOuterBoundary", "class_go_1_1_isogeometric_sf_model.html#af2b2fea00fe988f1336a4b38efa615a8", null
],
 ["setMinimumDegree", "class_go_1_1_isogeometric_sf_model.html#a4867f7b27e486728802bc52782ff6af2", null
],
 ["updateSolutionSplineSpace", "
 class_go_1_1_isogeometric_sf_model.html#a6986f65d9c64d1ae1da7e63288623970", null],
 ["updateSolutionSplineSpace", "
 class_go_1_1_isogeometric_sf_model.html#abab24442415ceebc7b5450aec123557", null]
]
```

Definition at line 1 of file `class_go_1_1_isogeometric_sf_model.js`.

## 30.312 doc/html/class\_go\_1\_1\_isogeometric\_vol\_block.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_isogeometric\\_vol\\_block](#)

### 30.312.1 Variable Documentation

#### 30.312.1.1 var class\_go\_1\_1\_isogeometric\_vol\_block

Definition at line 1 of file `class_go_1_1_isogeometric_vol_block.js`.

## 30.313 doc/html/class\_go\_1\_1\_isogeometric\_vol\_model.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_isogeometric\\_vol\\_model](#)

### 30.313.1 Variable Documentation

#### 30.313.1.1 var class\_go\_1\_1\_isogeometric\_vol\_model

##### Initial value:

```
=
[
 ["IsogeometricVolModel", "class_go_1_1_isogeometric_vol_model.html#a191bda162849259a19a2645439413c53",
 null],
 ["~IsogeometricVolModel", "class_go_1_1_isogeometric_vol_model.html#ac2c540f181b13a49282c8f692ce36598",
 null],
 ["addBoundaryCond", "class_go_1_1_isogeometric_vol_model.html#a03f5d3640ce444afdd95cbfc827d67fe", null
],
 ["addDirichletPointBdCond", "
class_go_1_1_isogeometric_vol_model.html#af652b4cc83bd08a7c59f632cd2a54e63", null],
 ["getBoundary", "class_go_1_1_isogeometric_vol_model.html#aec6dbe3d391c933a01dc859f4d40753c", null],
 ["getIsogeometricBlocks", "class_go_1_1_isogeometric_vol_model.html#a8083fd8d35010c10fda6e86fca59408c",
 null],
 ["getNmbOfBoundaries", "class_go_1_1_isogeometric_vol_model.html#ad2a0e7d195efa4f550c6b79a7f6822bb",
 null],
 ["getOuterBoundary", "class_go_1_1_isogeometric_vol_model.html#ad671f34d6688a4896f9d3146599cfec7",
 null],
 ["setMinimumDegree", "class_go_1_1_isogeometric_vol_model.html#a9a325bc4fc0f75f22dcf2cced31f2978",
 null],
 ["updateSolutionSplineSpace", "
class_go_1_1_isogeometric_vol_model.html#a56a7137a0a3ed9ca016d5c61654dfd16", null],
 ["updateSolutionSplineSpace", "
class_go_1_1_isogeometric_vol_model.html#addf842e26fccc1d8d294471685ea3d51", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_isogeometric\_vol\_model.js.

## 30.314 doc/html/class\_go\_1\_1\_isoparametric\_intersection\_curve.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_isoparametric\\_intersection\\_curve](#)

### 30.314.1 Variable Documentation

#### 30.314.1.1 var class\_go\_1\_1\_isoparametric\_intersection\_curve

##### Initial value:

```
=
[
 ["~IsoparametricIntersectionCurve", "
class_go_1_1_isoparametric_intersection_curve.html#a33076578d4f57e3a0adaf23633a1ba9b", null],
 ["IsoparametricIntersectionCurve", "
class_go_1_1_isoparametric_intersection_curve.html#a2be3f33607399b9123ef4791a308efa5", null],
 ["evaluateAt", "class_go_1_1_isoparametric_intersection_curve.html#ab8ffc72daf63c3fc8dc438786123d8e7",
 null],
 ["getCurve", "class_go_1_1_isoparametric_intersection_curve.html#aa65d454702cf7db78921d2c242e13f7a",
 null],
 ["getParamCurve", "
class_go_1_1_isoparametric_intersection_curve.html#a99ee627cd4f96e3a974a956a1d8222cf", null],
 ["getParamSpan", "class_go_1_1_isoparametric_intersection_curve.html#a0bd9ce871cc54e8848eb1506f796fa34",
 null],
 ["isDegenerated", "
class_go_1_1_isoparametric_intersection_curve.html#aabbc125bd7e2cb9d891114019a2011e", null],
 ["isIsocurve", "class_go_1_1_isoparametric_intersection_curve.html#a63dbd4d644632fd93f60d7322f93e851",
 null],
 ["precalculate_iso_curves", "

```

```

 class_go_1_1_isoparametric_intersection_curve.html#a04435fb820979d102fd49d877f3139e7", null],
 ["refine", "class_go_1_1_isoparametric_intersection_curve.html#adf83f654ee1cb30db922a470eb6fad81",
 null],
 ["constructIntersectionCurve", "
 class_go_1_1_isoparametric_intersection_curve.html#a6ab34095fc2901f18aff0d440f5f9ea5", null],
 ["isopar_geom_curve", "
 class_go_1_1_isoparametric_intersection_curve.html#aeff1c36f6ae6a949a3225bf9a182b1b5", null],
 ["isopar_param_curve_1", "
 class_go_1_1_isoparametric_intersection_curve.html#a6c228da0c9c5d9bed3789867be0b6a90", null],
 ["isopar_param_curve_2", "
 class_go_1_1_isoparametric_intersection_curve.html#a5063222f0b360b3fed58a3822fbdc959", null],
 ["temp", "class_go_1_1_isoparametric_intersection_curve.html#a93ea81ff005415650192405c4a3748f1", null
]
]

```

Definition at line 1 of file `class_go_1_1_isoparametric_intersection_curve.js`.

## 30.315 doc/html/class\_go\_1\_1\_l\_r\_b\_spline2\_d.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_l\\_r\\_b\\_spline2\\_d](#)

#### 30.315.1 Variable Documentation

##### 30.315.1.1 var class\_go\_1\_1\_l\_r\_b\_spline2\_d

Definition at line 1 of file `class_go_1_1_l_r_b_spline2_d.js`.

## 30.316 doc/html/class\_go\_1\_1\_l\_r\_spline\_eval\_grid.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_l\\_r\\_spline\\_eval\\_grid](#)

#### 30.316.1 Variable Documentation

##### 30.316.1.1 var class\_go\_1\_1\_l\_r\_spline\_eval\_grid

#### Initial value:

```

=
[
 ["param_float_type", "class_go_1_1_l_r_spline_eval_grid.html#a054a8a1a64ac866831717daf34aef80c", null
],
 ["LRSplineEvalGrid", "class_go_1_1_l_r_spline_eval_grid.html#a8cff465e2f97c8520954aa02c18bbd5b", null
],
 ["LRSplineEvalGrid", "class_go_1_1_l_r_spline_eval_grid.html#a5d1a06e6aa41caf486ed2deaed5acbe4", null
],
 ["computeBezCoefs", "class_go_1_1_l_r_spline_eval_grid.html#a12b44833237e752c5243763a284ead9b", null]
 ,
 ["dim", "class_go_1_1_l_r_spline_eval_grid.html#af31ce0f73a48c18deald2d4244022094", null],
 ["elements_begin", "class_go_1_1_l_r_spline_eval_grid.html#a09c1a952450107a7e610b803041d7d62", null],
 ["elements_end", "class_go_1_1_l_r_spline_eval_grid.html#ab4e82a60c327eed27dec27ce64647f77", null],
 ["evaluate", "class_go_1_1_l_r_spline_eval_grid.html#ad1191746b99b23ce99b488de54624bd4", null],
 ["evaluateGrid", "class_go_1_1_l_r_spline_eval_grid.html#a8f7e20b354263965d8f204df7af3d206", null],
 ["high", "class_go_1_1_l_r_spline_eval_grid.html#a15472d1bb9ee98179e8c889ddd580437", null],
 ["low", "class_go_1_1_l_r_spline_eval_grid.html#ad8ad2ec1e1a411bbfe600979dc5db1b0", null],
 ["numElements", "class_go_1_1_l_r_spline_eval_grid.html#a19bcc6cdfa3e2fa346722bbe0fe98adf", null],
 ["orderU", "class_go_1_1_l_r_spline_eval_grid.html#aa3476c89350806a253ca21c0bae55f77", null],
 ["orderV", "class_go_1_1_l_r_spline_eval_grid.html#aa3bfa78a3d1eeb6cc4d7a02752680180", null],
 ["origDom", "class_go_1_1_l_r_spline_eval_grid.html#a7b3b06e76e9922cf6e41856535011534", null],
 ["testCoefComputation", "class_go_1_1_l_r_spline_eval_grid.html#aa54fecef0eef9f16c192793911cad182",
 null]
]

```

Definition at line 1 of file `class_go_1_1_l_r_spline_eval_grid.js`.

## 30.317 doc/html/class\_go\_1\_1\_l\_r\_spline\_surface.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_l\\_r\\_spline\\_surface](#)

### 30.317.1 Variable Documentation

#### 30.317.1.1 var class\_go\_1\_1\_l\_r\_spline\_surface

Definition at line 1 of file class\_go\_1\_1\_l\_r\_spline\_surface.js.

## 30.318 doc/html/class\_go\_1\_1\_l\_r\_surf\_approx.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_l\\_r\\_surf\\_approx](#)

### 30.318.1 Variable Documentation

#### 30.318.1.1 var class\_go\_1\_1\_l\_r\_surf\_approx

#### Initial value:

```
=
[
 ["LRSurfApprox", "class_go_1_1_l_r_surf_approx.html#a0e06814b34380b34a63ab1675a94630a", null],
 ["LRSurfApprox", "class_go_1_1_l_r_surf_approx.html#a83105920d7bf2b295a5266936ede7478", null],
 ["LRSurfApprox", "class_go_1_1_l_r_surf_approx.html#ad2a82ddd7bd1d5d7e0dbe26315ebded7", null],
 ["LRSurfApprox", "class_go_1_1_l_r_surf_approx.html#a5b82491974b46c97a9a41784f71b8cce", null],
 ["LRSurfApprox", "class_go_1_1_l_r_surf_approx.html#a7993228606deaac67888bbf2f68525fc", null],
 ["LRSurfApprox", "class_go_1_1_l_r_surf_approx.html#a6428c335796f1ae92fc4b47ba3e5c53f", null],
 ["~LRSurfApprox", "class_go_1_1_l_r_surf_approx.html#a5f1611b8f2fc051a6ab77c6578fc626b", null],
 ["addLowerConstraint", "class_go_1_1_l_r_surf_approx.html#a3cb5cccb10272bb19de0ca6cc40d7117", null],
 ["addUpperConstraint", "class_go_1_1_l_r_surf_approx.html#af85623897f46f555f2db433a472127ac", null],
 ["edgeFix", "class_go_1_1_l_r_surf_approx.html#a42a280081704a6d9c86491186fb24918", null],
 ["getApproxSurf", "class_go_1_1_l_r_surf_approx.html#ac020c8533e8a4afd38e020bdc4ff79de", null],
 ["setFixBoundary", "class_go_1_1_l_r_surf_approx.html#a04275515ac612e8b540bd3cad9c46443", null],
 ["setFixCorner", "class_go_1_1_l_r_surf_approx.html#a6ef4fa9bf486f9693e28f2db0b4bf4a9", null],
 ["setGridInfo", "class_go_1_1_l_r_surf_approx.html#a12118907424e0c3329691e2121b57814", null],
 ["setInitMBA", "class_go_1_1_l_r_surf_approx.html#a276e342a75da7d0f13aa18d1d302d64b", null],
 ["setLocalConstraint", "class_go_1_1_l_r_surf_approx.html#ad3d4f8821662e034cac32fb831e62901", null],
 ["setMakeGhostPoints", "class_go_1_1_l_r_surf_approx.html#ab37a0d43758446b5ed55a29c7f048763", null],
 ["setMinimumElementSize", "class_go_1_1_l_r_surf_approx.html#a89e9bc64a9979d7a51c49d16e1916852", null],
],
 ["setSmoothBoundary", "class_go_1_1_l_r_surf_approx.html#a035f8fe9c6ea3cdc270560f9dadcc375", null],
 ["setSmoothingWeight", "class_go_1_1_l_r_surf_approx.html#a6c58d5d5a79e64a0c0118051efe27a26", null],
 ["setSwitchToMBA", "class_go_1_1_l_r_surf_approx.html#a34710961e043c9f9ce69603ef405c1be", null],
 ["setTurn3D", "class_go_1_1_l_r_surf_approx.html#ab9c044bde47d60ed4ad67ab564e2c202", null],
 ["setUseMBA", "class_go_1_1_l_r_surf_approx.html#a5b8d1edbc06cddb7dc4d2aaa050c4e1", null],
 ["setVerbose", "class_go_1_1_l_r_surf_approx.html#a3e39c57d2d6762a710d17a971fc93c1f", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_l\_r\_surf\_approx.js.

## 30.319 doc/html/class\_go\_1\_1\_l\_r\_surf\_smooth\_l\_s.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_l\\_r\\_surf\\_smooth\\_l\\_s](#)

### 30.319.1 Variable Documentation

#### 30.319.1.1 var class\_go\_1\_1\_l\_r\_surf\_smooth\_l\_s

#### Initial value:

```
=
[
 ["LRSurfSmoothLS", "class_go_1_1_l_r_surf_smooth_l_s.html#a8acba4937645e7d7c619ce67ae71c5b8", null],
 ["LRSurfSmoothLS", "class_go_1_1_l_r_surf_smooth_l_s.html#ad3d1d6e980d0818742352b8719702ac8", null],
 ["~LRSurfSmoothLS", "class_go_1_1_l_r_surf_smooth_l_s.html#a7f78193fffe738e7136727d606d0ba90", null],
 ["addDataPoints", "class_go_1_1_l_r_surf_smooth_l_s.html#a02b7663ebae4b911bdde4f5815a72c13", null],
 ["equationSolve", "class_go_1_1_l_r_surf_smooth_l_s.html#a0640b72ec11008abc0bfd8ec302ab098", null],
 ["hasDataPoints", "class_go_1_1_l_r_surf_smooth_l_s.html#a0d01b246abdd6bf76dda9943a43c171f", null],
 ["setInitSf", "class_go_1_1_l_r_surf_smooth_l_s.html#aeafa36938b4c58251e3a0417a8bc36432", null],
 ["setLeastSquares", "class_go_1_1_l_r_surf_smooth_l_s.html#a630e6d60764fclaed9cddcd7cd5925c9e", null],
 ["setLeastSquares", "class_go_1_1_l_r_surf_smooth_l_s.html#ac1016970a65dc1a22f7024ce4a7c498d", null],
 ["setLeastSquares_omp", "class_go_1_1_l_r_surf_smooth_l_s.html#a1ba981415be945d8fe6d642819dffc15",
 null],
 ["setOptimize", "class_go_1_1_l_r_surf_smooth_l_s.html#a50b78986ea49afb693b745771cd58f53", null],
 ["smoothBoundary", "class_go_1_1_l_r_surf_smooth_l_s.html#ab2c5a16e1a90e18f01a0a7c0288da9f1", null],
 ["updateLocals", "class_go_1_1_l_r_surf_smooth_l_s.html#ae2f65b8eba961085a9e7343edc3d39ca", null]
]
```

Definition at line 1 of file `class_go_1_1_l_r_surf_smooth_l_s.js`.

## 30.320 doc/html/class\_go\_1\_1\_lift\_curve.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_lift\\_curve](#)

### 30.320.1 Variable Documentation

#### 30.320.1.1 var class\_go\_1\_1\_lift\_curve

#### Initial value:

```
=
[
 ["LiftCurve", "class_go_1_1_lift_curve.html#a655cb07426e304f64987a8a87dcd761a", null],
 ["~LiftCurve", "class_go_1_1_lift_curve.html#aalab368f807769e4528ad58e02b1c079", null],
 ["approximationOK", "class_go_1_1_lift_curve.html#a1bdb9dd6b1efe58b1278b5e3f0f54ff2", null],
 ["dim", "class_go_1_1_lift_curve.html#a1c9f085116ace80e3e112a55044739c0", null],
 ["end", "class_go_1_1_lift_curve.html#aab7ea38541b9c66e122406afa6e70dd2", null],
 ["eval", "class_go_1_1_lift_curve.html#ae512f443f85b1704a433e11bb8782af5", null],
 ["eval", "class_go_1_1_lift_curve.html#a2cb2e79f40e4d61f22303b5bcf2aeba3", null],
 ["start", "class_go_1_1_lift_curve.html#a3c30dFd4d4ae55fb36ec055f57db28eb", null]
]
```

Definition at line 1 of file `class_go_1_1_lift_curve.js`.

## 30.321 doc/html/class\_go\_1\_1\_line.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_line](#)

#### 30.321.1 Variable Documentation

##### 30.321.1.1 var class\_go\_1\_1\_line

Definition at line 1 of file class\_go\_1\_1\_line.js.

## 30.322 doc/html/class\_go\_1\_1\_line2\_d\_int.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_line2\\_d\\_int](#)

#### 30.322.1 Variable Documentation

##### 30.322.1.1 var class\_go\_1\_1\_line2\_d\_int

#### Initial value:

```
=
[
 ["Line2DInt", "class_go_1_1_line2_d_int.html#a952e7cb1b60293e602ca08379dc1ed40", null],
 ["Line2DInt", "class_go_1_1_line2_d_int.html#afd526f4b651e589a5de6bcfa49f6454d", null],
 ["~Line2DInt", "class_go_1_1_line2_d_int.html#a19e49dabf8e66c9906f116e15244e7bd", null],
 ["a", "class_go_1_1_line2_d_int.html#abcacfea3993471707231506965ed7e10", null],
 ["b", "class_go_1_1_line2_d_int.html#a1bf14760b8c49e8b71cd24fa7a38cd35", null],
 ["c", "class_go_1_1_line2_d_int.html#a267566d852799214325a5702eb5cef12", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_line2\_d\_int.js.

## 30.323 doc/html/class\_go\_1\_1\_line\_cloud.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_line\\_cloud](#)

### 30.323.1 Variable Documentation

#### 30.323.1.1 var class\_go\_1\_1\_line\_cloud

##### Initial value:

```
=
[
 ["LineCloud", "class_go_1_1_line_cloud.html#a71f37875ee1d59cd8d1157481b97cbac", null],
 ["LineCloud", "class_go_1_1_line_cloud.html#a24910b33a551d9edbad30a93adb79ac0", null],
 ["~LineCloud", "class_go_1_1_line_cloud.html#a23e2c04166b62866378f350e0da0ddf4", null],
 ["boundingBox", "class_go_1_1_line_cloud.html#af24b05564a5513fbaa3ec0e298328672", null],
 ["clone", "class_go_1_1_line_cloud.html#a5b7d1724236c19e42ec4e18fa8e5f922", null],
 ["dimension", "class_go_1_1_line_cloud.html#a504af7fef8693fdfa12ba0a5b7f35f3", null],
 ["instanceType", "class_go_1_1_line_cloud.html#a07c8f5ac835090a871029a2a5094136e", null],
 ["numLines", "class_go_1_1_line_cloud.html#abe9b6c2ebb9d26c8aadd0dec85ef4d81", null],
 ["point", "class_go_1_1_line_cloud.html#a1067891005886183a1f1662814e798e0", null],
 ["point", "class_go_1_1_line_cloud.html#a76a4ad5e381de0e8ba9a444909a164cc", null],
 ["rawData", "class_go_1_1_line_cloud.html#a3430f9f2b895c270e16190bdcc361a60", null],
 ["read", "class_go_1_1_line_cloud.html#a3f9b45790a01bc9a29ed9f3c3bd460eb", null],
 ["setCloud", "class_go_1_1_line_cloud.html#abbc686694baae61de1ab32a6f54d1693", null],
 ["write", "class_go_1_1_line_cloud.html#ad0108951f753e802c0deb441b83ec4b6", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_line\_cloud.js.

### 30.324 doc/html/class\_go\_1\_1\_line\_cloud\_tesselator.js File Reference

#### Variables

- var [class\\_go\\_1\\_1\\_line\\_cloud\\_tesselator](#)

### 30.324.1 Variable Documentation

#### 30.324.1.1 var class\_go\_1\_1\_line\_cloud\_tesselator

##### Initial value:

```
=
[
 ["LineCloudTesselator", "class_go_1_1_line_cloud_tesselator.html#a8dbce8787df6e91c135995703e7206af",
 null],
 ["~LineCloudTesselator", "class_go_1_1_line_cloud_tesselator.html#af3c1b9270dca9d1cbe6f7a0bfb1b6acd6",
 null],
 ["getRenderCloud", "class_go_1_1_line_cloud_tesselator.html#ae71455a7bc76b3e98bdba4de80fdc34b", null]
 ,
 ["getScale", "class_go_1_1_line_cloud_tesselator.html#ad663f41ec157353e7d588b2e552bed04", null],
 ["setScale", "class_go_1_1_line_cloud_tesselator.html#affe864cbece5d76e738c7043150abfaa", null],
 ["tesselate", "class_go_1_1_line_cloud_tesselator.html#a741142d2fb183073f39999edbd0987490", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_line\_cloud\_tesselator.js.

### 30.325 doc/html/class\_go\_1\_1\_line\_strip.js File Reference

#### Variables

- var [class\\_go\\_1\\_1\\_line\\_strip](#)



### 30.325.1 Variable Documentation

#### 30.325.1.1 var class\_go\_1\_1\_line\_strip

##### Initial value:

```
=
[
 ["LineStrip", "class_go_1_1_line_strip.html#a17ebfbc9a353bae4e9a2ce47b9806fd7", null],
 ["~LineStrip", "class_go_1_1_line_strip.html#a9cd2ed1410d2eb1b8140f3dc114fb499", null],
 ["asLineStrip", "class_go_1_1_line_strip.html#a0f8f068735d230132a601688f235cdd1", null],
 ["atBoundary", "class_go_1_1_line_strip.html#a34168c4b91f16e62867b03cbd7d9eaa5", null],
 ["numVertices", "class_go_1_1_line_strip.html#afba3b41d7d374de4a19dad3095684501", null],
 ["paramArray", "class_go_1_1_line_strip.html#af618b12bf6e99f89ac7460ae714a03e8", null],
 ["resize", "class_go_1_1_line_strip.html#a00a995b0ae74ff6b5fd7d04e991c87cd", null],
 ["stripArray", "class_go_1_1_line_strip.html#aa79b95ca91e96e2af3b9d3aacc7aee73", null],
 ["triangleIndexArray", "class_go_1_1_line_strip.html#af375f512ceb6aa626654cab1dfde79b6", null],
 ["vertexArray", "class_go_1_1_line_strip.html#a62aada6f75be07b856cb62d6247ab342", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_line\_strip.js.

## 30.326 doc/html/class\_go\_1\_1\_locked\_dir\_dist\_func.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_locked\\_dir\\_dist\\_func](#)

### 30.326.1 Variable Documentation

#### 30.326.1.1 var class\_go\_1\_1\_locked\_dir\_dist\_func

##### Initial value:

```
=
[
 ["LockedDirDistFunc", "class_go_1_1_locked_dir_dist_func.html#a041d8942dad387effdafb1f17920db8d", null],
 ["grad", "class_go_1_1_locked_dir_dist_func.html#ac2be05d7ef6c39967e72873a66c88442", null],
 ["maxPar", "class_go_1_1_locked_dir_dist_func.html#a53c05142ec881ffc80ff87c95db653d5", null],
 ["minPar", "class_go_1_1_locked_dir_dist_func.html#a384ecae731f9ca22fd6c773fc89d760f", null],
 ["operator()", "class_go_1_1_locked_dir_dist_func.html#a18bf8be5221b7cd88436347483f400c0", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_locked\_dir\_dist\_func.js.

## 30.327 doc/html/class\_go\_1\_1\_loop.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_loop](#)

### 30.327.1 Variable Documentation

#### 30.327.1.1 var class\_go\_1\_1\_loop

Definition at line 1 of file class\_go\_1\_1\_loop.js.

## 30.328 doc/html/class\_go\_1\_1\_march\_point.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_march\\_point](#)

### 30.328.1 Variable Documentation

#### 30.328.1.1 var class\_go\_1\_1\_march\_point

#### Initial value:

```
=
[
 ["MarchPoint", "class_go_1_1_march_point.html#a4cac7ff78f8455fbc03868a58d87fc8d", null],
 ["operator<", "class_go_1_1_march_point.html#ac78dc3d4fb54fecc91dc403cbabe29f4", null],
 ["cos_ang", "class_go_1_1_march_point.html#ae3e55443888a52d3ca2e4f4f52809641", null],
 ["dist", "class_go_1_1_march_point.html#a2ba6c91c4d397e6691978cdb43bf18d4", null],
 ["par", "class_go_1_1_march_point.html#a115fd23876bf53f582bac27500ca685d", null],
 ["par2", "class_go_1_1_march_point.html#a9b9a7eaa9e47863860e1ca7448a9c83c", null],
 ["pt", "class_go_1_1_march_point.html#a7c18a4cc30c19ed9440f704f172b5e60", null],
 ["status", "class_go_1_1_march_point.html#a61d65a560e49c61e7716af58c112b352", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_march\_point.js.

## 30.329 doc/html/class\_go\_1\_1\_matrix\_x\_d.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_matrix\\_x\\_d](#)

### 30.329.1 Variable Documentation

#### 30.329.1.1 var class\_go\_1\_1\_matrix\_x\_d

Initial value:

```
=
[
 ["MatrixXD", "class_go_1_1_matrix_x_d.html#abc36c9d0d865b01e4be5f7a5e3048acd", null],
 ["~MatrixXD", "class_go_1_1_matrix_x_d.html#a7d2e4060a4e8d4c306ad02910258a3fa", null],
 ["det", "class_go_1_1_matrix_x_d.html#af852c8d82a01fcc655b9d207f33b6298", null],
 ["frobeniusNorm", "class_go_1_1_matrix_x_d.html#a0aead8317fd39e96c220c6f6e01b4f30", null],
 ["get", "class_go_1_1_matrix_x_d.html#a6fcc68410e3e2145dd8bfd876c6f3812", null],
 ["identity", "class_go_1_1_matrix_x_d.html#a69ef9806148e42baf80b86cf813022b5", null],
 ["mult", "class_go_1_1_matrix_x_d.html#a5a3aef5518e8457f5096da4901edea21", null],
 ["operator()", "class_go_1_1_matrix_x_d.html#a21212214d40879c4497a9d31fd566975", null],
 ["operator()", "class_go_1_1_matrix_x_d.html#a41b31b7a1f978d57bb629b43c3f1df87", null],
 ["operator*", "class_go_1_1_matrix_x_d.html#ac1b00aebfa90c1bdf7eed2b2f85b2cd6", null],
 ["operator*", "class_go_1_1_matrix_x_d.html#a1009b735295cec2dadd0523e4e95d103", null],
 ["operator*", "class_go_1_1_matrix_x_d.html#ab0c202c6b895826501fb6c655c1daf7f", null],
 ["operator*=", "class_go_1_1_matrix_x_d.html#acce83f7be83a54d903d49e525b5436d", null],
 ["operator*=", "class_go_1_1_matrix_x_d.html#a0174e525d484ee78c6b1b12bfbd6ab22", null],
 ["operator+", "class_go_1_1_matrix_x_d.html#ade3dfae0fa095ecf94c4ed0b24a7a0d2", null],
 ["operator+=", "class_go_1_1_matrix_x_d.html#a729c48f6895d51b60e375f11fd5cf833", null],
 ["operator-", "class_go_1_1_matrix_x_d.html#a44145ce1b97d63fa93c4188df54d14f0", null],
 ["setToRotation", "class_go_1_1_matrix_x_d.html#a6aelbd453269fc64eb6a341d27f9154a", null],
 ["setToRotation", "class_go_1_1_matrix_x_d.html#a88a8319ccec5f230019a38fdacb289e", null],
 ["setToRotation", "class_go_1_1_matrix_x_d.html#a406d3734ccc71684b136192dafd87205", null],
 ["setToRotation", "class_go_1_1_matrix_x_d.html#aab37d12a89916cc58ac7ff4efff76675", null],
 ["setToRotation", "class_go_1_1_matrix_x_d.html#aa803df5eb9aa5f2128c6e7e5acb07664", null],
 ["submatrix", "class_go_1_1_matrix_x_d.html#aa88a11a8b68a10040d2c845346dd8b0b", null],
 ["trace", "class_go_1_1_matrix_x_d.html#a052cc3bbeledfb81cf346208e469c346", null],
 ["transpose", "class_go_1_1_matrix_x_d.html#a4aa305ce84a54651878dfalc5e00157b", null],
 ["zero", "class_go_1_1_matrix_x_d.html#ac360b2abe7e401969db7516ce266e416", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_matrix\_x\_d.js.

## 30.330 doc/html/class\_go\_1\_1\_mesh2\_d.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_mesh2\\_d](#)

### 30.330.1 Variable Documentation

#### 30.330.1.1 var class\_go\_1\_1\_mesh2\_d

Definition at line 1 of file class\_go\_1\_1\_mesh2\_d.js.

## 30.331 doc/html/class\_go\_1\_1\_mesh2\_d\_iterator.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_mesh2\\_d\\_iterator](#)

### 30.331.1 Variable Documentation

#### 30.331.1.1 var class\_go\_1\_1\_mesh2\_d\_iterator

##### Initial value:

```
=
[
 ["Mesh2DIterator", "class_go_1_1_mesh2_d_iterator.html#ab203e32e6e6725d88e6b9ec6526da685", null],
 ["Mesh2DIterator", "class_go_1_1_mesh2_d_iterator.html#a0a999cdceeeaf922e9312e84b7025eb6", null],
 ["operator!=", "class_go_1_1_mesh2_d_iterator.html#ae168f09a65470e294e89eab7c935d8a6", null],
 ["operator*", "class_go_1_1_mesh2_d_iterator.html#ac6dbcb4e7100d2bc7a40017fc01b0ca1", null],
 ["operator++", "class_go_1_1_mesh2_d_iterator.html#a597307e4043a6b0091f7dcc65ad99382", null],
 ["operator==", "class_go_1_1_mesh2_d_iterator.html#a06395033f9730e9aa15cb9673e106999", null],
 ["swap", "class_go_1_1_mesh2_d_iterator.html#a47f82c40e166cb3eb24b97521e86546b", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_mesh2\_d\_iterator.js.

## 30.332 doc/html/class\_go\_1\_1\_model\_quality.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_model\\_quality](#)

### 30.332.1 Variable Documentation

#### 30.332.1.1 var class\_go\_1\_1\_model\_quality

Definition at line 1 of file class\_go\_1\_1\_model\_quality.js.

## 30.333 doc/html/class\_go\_1\_1\_model\_repair.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_model\\_repair](#)

### 30.333.1 Variable Documentation

#### 30.333.1.1 var class\_go\_1\_1\_model\_repair

##### Initial value:

```
=
[
 ["ModelRepair", "class_go_1_1_model_repair.html#ac1efabc94d69a90145525c8ec16322b3", null],
 ["ModelRepair", "class_go_1_1_model_repair.html#a4c080a1d32f0b35eff8c9caal32ccc3b", null],
 ["consistentFaceNormal", "class_go_1_1_model_repair.html#ab3446f078e05ba1336b932032c5f0b81", null],
 ["identicalAndEmbeddedFaces", "class_go_1_1_model_repair.html#a966573f66568b95f898e4e2d6bd0517f", null],
 ["identicalVertices", "class_go_1_1_model_repair.html#acb862290dbbc7a5a0045e771baa03c53", null],
 ["mendEdgeDistance", "class_go_1_1_model_repair.html#a9f10ebaa64e0b173656592e3752a7247", null],
 ["mendGaps", "class_go_1_1_model_repair.html#ac2b2377fb248f25c694845d80875fd43", null],
 ["optimizeVertexPosition", "class_go_1_1_model_repair.html#a7033e321f3a14b4ab46587f12b28124a", null],
 ["results_", "class_go_1_1_model_repair.html#a4e0074ebcc8f570fa3b15ae1b7c53560", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_model\_repair.js.

## 30.334 doc/html/class\_go\_1\_1\_non\_evaluable\_intersection\_curve.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_non\\_evaluable\\_intersection\\_curve](#)

#### 30.334.1 Variable Documentation

##### 30.334.1.1 var class\_go\_1\_1\_non\_evaluable\_intersection\_curve

###### Initial value:

```
=
[
 ["~NonEvaluableIntersectionCurve", "
 class_go_1_1_non_evaluable_intersection_curve.html#a1336c1259406236ab0fbbbfd9a14e85a", null],
 ["evaluateAt", "class_go_1_1_non_evaluable_intersection_curve.html#a38301c7f2cc913a2de3b509258ad92dc",
 null],
 ["getCurve", "class_go_1_1_non_evaluable_intersection_curve.html#a52d5d35b2ab2cd538f17e8f5d381cb68",
 null],
 ["getParamCurve", "
 class_go_1_1_non_evaluable_intersection_curve.html#a94806b4652e34382d51f80cc15a00f23", null],
 ["getParamSpan", "class_go_1_1_non_evaluable_intersection_curve.html#a54c1038d6f86981191ee5db7bf0f525c",
 null],
 ["isDegenerated", "
 class_go_1_1_non_evaluable_intersection_curve.html#a157e2231b3942a2d85f4480216c478fd", null],
 ["isIsocurve", "class_go_1_1_non_evaluable_intersection_curve.html#a2f78f13b63e1d0b02467d421887e44a0",
 null],
 ["refine", "class_go_1_1_non_evaluable_intersection_curve.html#ad98f3ef398ef8f75f6e46063d93a7df1",
 null],
 ["constructIntersectionCurve", "
 class_go_1_1_non_evaluable_intersection_curve.html#a6ab34095fc2901f18aff0d440f5f9ea5", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_non\_evaluable\_intersection\_curve.js.

## 30.335 doc/html/class\_go\_1\_1\_noop\_tesselator.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_noop\\_tesselator](#)

#### 30.335.1 Variable Documentation

##### 30.335.1.1 var class\_go\_1\_1\_noop\_tesselator

###### Initial value:

```
=
[
 ["~NoopTesselator", "class_go_1_1_noop_tesselator.html#a6aee770a3a1923b56476499935bcbb7e", null],
 ["tesselate", "class_go_1_1_noop_tesselator.html#a098bad44d4975404a9ecbf097a895423", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_noop\_tesselator.js.

## 30.336 doc/html/class\_go\_1\_1\_object\_header.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_object\\_header](#)

### 30.336.1 Variable Documentation

#### 30.336.1.1 var class\_go\_1\_1\_object\_header

#### Initial value:

```
=
[
 ["ObjectHeader", "class_go_1_1_object_header.html#a9e3681d49e50d583c190d28e8463577a", null],
 ["ObjectHeader", "class_go_1_1_object_header.html#ad0ab83c08e83f08abc15c7e6bd7a3d53", null],
 ["ObjectHeader", "class_go_1_1_object_header.html#a061a7c83b6929bcc4ba64377213066f1", null],
 ["~ObjectHeader", "class_go_1_1_object_header.html#a3a9c9eb50ff26128020aba008648f603", null],
 ["auxdata", "class_go_1_1_object_header.html#a9feaa9009aeceeb738a4078889945994", null],
 ["auxdataSize", "class_go_1_1_object_header.html#aa17b0d9026afbb13ef8a33942afc7518", null],
 ["classType", "class_go_1_1_object_header.html#aa57081ec3377d55feeb6b6403ddeb484", null],
 ["majorVersion", "class_go_1_1_object_header.html#aea923ba58e068268585cc313e2a8dcd7", null],
 ["minorVersion", "class_go_1_1_object_header.html#ab0f37b183fab7d773722584c52553871", null],
 ["read", "class_go_1_1_object_header.html#a40795d9e4fa51548dfe133befd6085ac", null],
 ["write", "class_go_1_1_object_header.html#a9c9e45ce64df26134637633bc2c5cc8c", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_object\_header.js.

## 30.337 doc/html/class\_go\_1\_1\_par0\_func\_intersector.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_par0\\_func\\_intersector](#)

### 30.337.1 Variable Documentation

#### 30.337.1.1 var class\_go\_1\_1\_par0\_func\_intersector

#### Initial value:

```
=
[
 ["Par0FuncIntersector", "class_go_1_1_par0_func_intersector.html#a25ad497265e8d5c922ea8a7355518d1c", null],
 ["~Par0FuncIntersector", "class_go_1_1_par0_func_intersector.html#a12827957c6bca3572887835fdb426695", null],
 ["checkCoincidence", "class_go_1_1_par0_func_intersector.html#a158ccd51ee0f49c97e4bc5a165a8198d", null],
 ["doSubdivide", "class_go_1_1_par0_func_intersector.html#acea16fb2e5679e359b6e3c5ec8fcb95e", null],
 ["lowerOrderIntersector", "class_go_1_1_par0_func_intersector.html#ab5d84c1b2360f24b675710dadf27a034", null],
 ["microCase", "class_go_1_1_par0_func_intersector.html#a281a2f47287385dc505a9c6835f5ebb9", null],
 ["numParams", "class_go_1_1_par0_func_intersector.html#a0067a654e9a3737e3dad07fd84c0cbfb", null],
 ["repairIntersections", "class_go_1_1_par0_func_intersector.html#ab5a6c10d8b9ac2a977c970535e54caf", null],
 ["updateIntersections", "class_go_1_1_par0_func_intersector.html#aed50a6673c585cc8a4e30ba80d637d6e", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_par0\_func\_intersector.js.

## 30.338 doc/html/class\_go\_1\_1\_par1\_func\_intersector.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_par1\\_func\\_intersector](#)

### 30.338.1 Variable Documentation

#### 30.338.1.1 var class\_go\_1\_1\_par1\_func\_intersector

##### Initial value:

```
=
[
 ["Par1FuncIntersector", "class_go_1_1_par1_func_intersector.html#a1814cfd747af220df82cc9311525a699",
 null],
 ["~Par1FuncIntersector", "class_go_1_1_par1_func_intersector.html#a4446d9a0bd19b1df7cb61afe75420d39",
 null],
 ["checkCoincidence", "class_go_1_1_par1_func_intersector.html#a175c2e760c41b924e57a39b88057b7c4", null
],
 ["doSubdivide", "class_go_1_1_par1_func_intersector.html#adfcaf89a824ff6dbdb1954b0a83d8279", null],
 ["lowerOrderIntersector", "class_go_1_1_par1_func_intersector.html#a0ceda62694bf2ddca4dae43b779058ec",
 null],
 ["microCase", "class_go_1_1_par1_func_intersector.html#aa0115ece8e0c51b8cccb315a7dc2bb1a", null],
 ["numParams", "class_go_1_1_par1_func_intersector.html#a9e8fcl9e2502b5647ee5f00b6ff31b0", null],
 ["repairIntersections", "class_go_1_1_par1_func_intersector.html#a002d05b76452464e0badb00106729352",
 null],
 ["updateIntersections", "class_go_1_1_par1_func_intersector.html#ab711341365c1b93dfa690f3581a388d1",
 null]
]
```

Definition at line 1 of file class\_go\_1\_1\_par1\_func\_intersector.js.

## 30.339 doc/html/class\_go\_1\_1\_par2\_func\_intersector.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_par2\\_func\\_intersector](#)

### 30.339.1 Variable Documentation

#### 30.339.1.1 var class\_go\_1\_1\_par2\_func\_intersector

##### Initial value:

```
=
[
 ["Par2FuncIntersector", "class_go_1_1_par2_func_intersector.html#a595ba972baf1bec027ff0c1b0ba5af1f",
 null],
 ["~Par2FuncIntersector", "class_go_1_1_par2_func_intersector.html#a352be801cd61742f4282cdf08b3e4ae8",
 null],
 ["canConnect", "class_go_1_1_par2_func_intersector.html#ae3ae4e205f71319fcdccf332927c86aa", null],
 ["checkCoincidence", "class_go_1_1_par2_func_intersector.html#a729a8656409adb0165db1380b7f178e2", null
],
 ["connectDirected", "class_go_1_1_par2_func_intersector.html#a23896c07fbc87b20a161a7e1735f793c", null
],
 ["doSubdivide", "class_go_1_1_par2_func_intersector.html#a11a42067174f7e8710a84cbc689a0b97", null],
 ["isConnected", "class_go_1_1_par2_func_intersector.html#a3868472fa9ff67a24de9663fe4da554d", null],
 ["isConnected", "class_go_1_1_par2_func_intersector.html#a4d4f49287dfcd25c2ed048cdc025f893", null],
 ["lowerOrderIntersector", "class_go_1_1_par2_func_intersector.html#acca35ce72841e66325b368e851a21c96",
 null],
 ["microCase", "class_go_1_1_par2_func_intersector.html#a2741576ed17c7faa2b6f48fd0b02e9fe", null],
 ["numParams", "class_go_1_1_par2_func_intersector.html#aff355284c180fbc0911003a0f1dc4b86", null],
 ["repairIntersections", "class_go_1_1_par2_func_intersector.html#aaebda7a2875ce831b9d47dcce7cf915a",
 null],
 ["updateIntersections", "class_go_1_1_par2_func_intersector.html#a3cbd4e1b888316afd1d15c07bfd6b306",
 null]
]
```

Definition at line 1 of file class\_go\_1\_1\_par2\_func\_intersector.js.

## 30.340 doc/html/class\_go\_1\_1\_parabola.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_parabola](#)

### 30.340.1 Variable Documentation

#### 30.340.1.1 var class\_go\_1\_1\_parabola

Definition at line 1 of file class\_go\_1\_1\_parabola.js.

## 30.341 doc/html/class\_go\_1\_1\_parallelepiped.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_parallelepiped](#)

### 30.341.1 Variable Documentation

#### 30.341.1.1 var class\_go\_1\_1\_parallelepiped

#### Initial value:

```
=
[
 ["Parallelepiped", "class_go_1_1_parallelepiped.html#a462264b88bc3fb72501ff635588f2d6", null],
 ["Parallelepiped", "class_go_1_1_parallelepiped.html#ab1b4f71b71ade8f4abc822cc78fcbd17", null],
 ["~Parallelepiped", "class_go_1_1_parallelepiped.html#a38c875eed4adee6d21c94111ae36a358", null],
 ["boundingBox", "class_go_1_1_parallelepiped.html#a902d894e6fc161ff0d283e837114c49b", null],
 ["clone", "class_go_1_1_parallelepiped.html#a3fc00924fc7e6dd2f39988dc261fb066", null],
 ["closestPoint", "class_go_1_1_parallelepiped.html#a2c2e1c088ea9235d5bcd32e6a1c06330", null],
 ["dimension", "class_go_1_1_parallelepiped.html#a094028bb7467bde915db6f2c0c62fa75", null],
 ["geometryVolume", "class_go_1_1_parallelepiped.html#a77725fc481eca9862548c45f1c4b4817", null],
 ["getAllBoundarySurfaces", "class_go_1_1_parallelepiped.html#a79fb25f5d453fd59e7ef8752f263f501", null],
],
 ["instanceType", "class_go_1_1_parallelepiped.html#a387a7463e29baaa90f94db11e520903b", null],
 ["nextSegmentVal", "class_go_1_1_parallelepiped.html#afab08aea8607a06b73c4e21abb5492a0", null],
 ["parameterSpan", "class_go_1_1_parallelepiped.html#afeb5a372f8f5cb2c8aacd21618571901", null],
 ["point", "class_go_1_1_parallelepiped.html#a1c862db9370a8e3238bab13a02935d28", null],
 ["point", "class_go_1_1_parallelepiped.html#a17879120de2721d210991fa03ce8fd46", null],
 ["read", "class_go_1_1_parallelepiped.html#ada24cffb4496199778e5371de67f716a", null],
 ["reverseParameterDirection", "class_go_1_1_parallelepiped.html#a2046346c0585a4da905f930430a05dd3", null],
 ["swapParameterDirection", "class_go_1_1_parallelepiped.html#aeb1c073f8739cc5674285ab56f66d217", null],
],
 ["tangentCone", "class_go_1_1_parallelepiped.html#a65b0a65033c8c65f6a2884fe2e6a11ca", null],
 ["translate", "class_go_1_1_parallelepiped.html#ab065301718efca14404a9a5419e6d53f", null],
 ["write", "class_go_1_1_parallelepiped.html#ada46eff243691531b4b0885515be7e1b", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_parallelepiped.js.



## 30.342 doc/html/class\_go\_1\_1\_param0\_function\_int.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_param0\\_function\\_int](#)

#### 30.342.1 Variable Documentation

##### 30.342.1.1 var class\_go\_1\_1\_param0\_function\_int

Definition at line 1 of file class\_go\_1\_1\_param0\_function\_int.js.

## 30.343 doc/html/class\_go\_1\_1\_param1\_function\_int.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_param1\\_function\\_int](#)

#### 30.343.1 Variable Documentation

##### 30.343.1.1 var class\_go\_1\_1\_param1\_function\_int

Definition at line 1 of file class\_go\_1\_1\_param1\_function\_int.js.

## 30.344 doc/html/class\_go\_1\_1\_param2\_function\_int.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_param2\\_function\\_int](#)

#### 30.344.1 Variable Documentation

##### 30.344.1.1 var class\_go\_1\_1\_param2\_function\_int

Definition at line 1 of file class\_go\_1\_1\_param2\_function\_int.js.

## 30.345 doc/html/class\_go\_1\_1\_param\_curve.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_param\\_curve](#)

### 30.345.1 Variable Documentation

#### 30.345.1.1 var class\_go\_1\_1\_param\_curve

Definition at line 1 of file class\_go\_1\_1\_param\_curve.js.

## 30.346 doc/html/class\_go\_1\_1\_param\_curve\_int.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_param\\_curve\\_int](#)

### 30.346.1 Variable Documentation

#### 30.346.1.1 var class\_go\_1\_1\_param\_curve\_int

Definition at line 1 of file class\_go\_1\_1\_param\_curve\_int.js.

## 30.347 doc/html/class\_go\_1\_1\_param\_function\_int.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_param\\_function\\_int](#)

### 30.347.1 Variable Documentation

#### 30.347.1.1 var class\_go\_1\_1\_param\_function\_int

#### Initial value:

```
=
[
 ["~ParamFunctionInt", "class_go_1_1_param_function_int.html#a8a214e379c2dcccblb4d5e49df049ea7", null],
 ["getBoundaryObjects", "class_go_1_1_param_function_int.html#abeb07b127558af03aa02e7c50f56ade9", null],
 ["getMesh", "class_go_1_1_param_function_int.html#a1fe259c2e850acea2b32a660ed15a801", null],
 ["getMeshSize", "class_go_1_1_param_function_int.html#ab2c20f161e853bd264125b603e1d2d32", null],
 ["getParam0FunctionInt", "class_go_1_1_param_function_int.html#a13566aed323615d740614e5691d9f500", null],
 ["getParam1FunctionInt", "class_go_1_1_param_function_int.html#a45343f900136c3b7c0442e795c1fb9b0", null],
 ["getParam2FunctionInt", "class_go_1_1_param_function_int.html#a396b2813d2462c5d8e31769a50625c3c", null],
 ["monotone", "class_go_1_1_param_function_int.html#a0746e0714a26dcba78d567f34d71b9bb", null],
 ["paramFromMesh", "class_go_1_1_param_function_int.html#a67db1e8d424105c98d76ef893f40873d", null],
 ["subdivide", "class_go_1_1_param_function_int.html#aa430f18ef349897dbae9317100ae6d74", null],
 ["boundary_obj", "class_go_1_1_param_function_int.html#aeecef00947ffe0462d0356b21a3612227", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_param\_function\_int.js.

## 30.348 doc/html/class\_go\_1\_1\_param\_geom\_int.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_param\\_geom\\_int](#)

### 30.348.1 Variable Documentation

#### 30.348.1.1 var class\_go\_1\_1\_param\_geom\_int

##### Initial value:

```
=
[
 ["ParamGeomInt", "class_go_1_1_param_geom_int.html#a3e1e109e9f9ceb3e76d7b98a1cfdbbb4", null],
 ["~ParamGeomInt", "class_go_1_1_param_geom_int.html#ad9ee33938a318097a524213425ea9ac6", null],
 ["checkPeriodicity", "class_go_1_1_param_geom_int.html#a12d8e812fddc8b5872ff50fa2786e19e", null],
 ["compositeBox", "class_go_1_1_param_geom_int.html#a1deded580c47f7e36facd6b2722c90af", null],
 ["coneLargerThanPi", "class_go_1_1_param_geom_int.html#a72c4e7f6c41b7acacd8fc6677890e745", null],
 ["dimension", "class_go_1_1_param_geom_int.html#a12a5ea9d43671f311c69528d9285caf2", null],
 ["directionCone", "class_go_1_1_param_geom_int.html#a4392fb3e2447dc9a2b012f431d6ec14a", null],
 ["getBoundaryObject", "class_go_1_1_param_geom_int.html#a9b02408318205978eb4d6e1804a12989", null],
 ["getBoundaryObjects", "class_go_1_1_param_geom_int.html#a4ae4e9d29ea2e57238115b060fcca15d", null],
 ["getMesh", "class_go_1_1_param_geom_int.html#aa156d7e529cf37cbce8c8dd4cc856b25", null],
 ["getMeshSize", "class_go_1_1_param_geom_int.html#a6af02806353f736d262b2d29eeb83ac0", null],
 ["getOptimizedConeAngle", "class_go_1_1_param_geom_int.html#a5d0a7f71e603911c0a507c8e6643adcd", null]
],
[
 ["isDegenerate", "class_go_1_1_param_geom_int.html#a816a1737227d8e5e9cd56943730955b2", null],
 ["isDegenerate", "class_go_1_1_param_geom_int.html#a73ce9db41e9770d9be8580b51925dc9d", null],
 ["isLinear", "class_go_1_1_param_geom_int.html#ac9ee62ab3226e99bbf57826042e2d68c", null],
 ["isSpline", "class_go_1_1_param_geom_int.html#afb4151c2b0c41239914fe1b340e29c83", null],
 ["nmbBdObj", "class_go_1_1_param_geom_int.html#aeb78548d665d498273bdce9edafaa7c5", null],
 ["paramFromMesh", "class_go_1_1_param_geom_int.html#a25d1bb04c5d76b86846a52a99e7ec32d", null],
 ["reducedDirectionCone", "class_go_1_1_param_geom_int.html#a59f088e8e6dbdc0890ac3d7a0219bb49", null],
 ["subdivide", "class_go_1_1_param_geom_int.html#afbc32b467e418ddd4de459957b50e16e", null],
 ["boundary_obj_", "class_go_1_1_param_geom_int.html#aaebc3813fbc16f661a664f79dd13df90", null]
]
```

Definition at line 1 of file [class\\_go\\_1\\_1\\_param\\_geom\\_int.js](#).

## 30.349 doc/html/class\_go\_1\_1\_param\_object\_int.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_param\\_object\\_int](#)

### 30.349.1 Variable Documentation

#### 30.349.1.1 var class\_go\_1\_1\_param\_object\_int

Definition at line 1 of file [class\\_go\\_1\\_1\\_param\\_object\\_int.js](#).

## 30.350 doc/html/class\_go\_1\_1\_param\_point\_int.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_param\\_point\\_int](#)

### 30.350.1 Variable Documentation

#### 30.350.1.1 var class\_go\_1\_1\_param\_point\_int

Definition at line 1 of file class\_go\_1\_1\_param\_point\_int.js.

## 30.351 doc/html/class\_go\_1\_1\_param\_surface.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_param\\_surface](#)

### 30.351.1 Variable Documentation

#### 30.351.1.1 var class\_go\_1\_1\_param\_surface

Definition at line 1 of file class\_go\_1\_1\_param\_surface.js.

## 30.352 doc/html/class\_go\_1\_1\_param\_surface\_int.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_param\\_surface\\_int](#)

### 30.352.1 Variable Documentation

#### 30.352.1.1 var class\_go\_1\_1\_param\_surface\_int

Definition at line 1 of file class\_go\_1\_1\_param\_surface\_int.js.

## 30.353 doc/html/class\_go\_1\_1\_param\_volume.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_param\\_volume](#)

### 30.353.1 Variable Documentation

#### 30.353.1.1 var class\_go\_1\_1\_param\_volume

##### Initial value:

```
=
[
 ["~ParamVolume", "class_go_1_1_param_volume.html#a4ea39ed9d48ccaf6360bb893db0ddd15", null],
 ["asSplineVolume", "class_go_1_1_param_volume.html#ab5cb2687d903913bb4059448d38b807a", null],
 ["clone", "class_go_1_1_param_volume.html#aide81ce34caefb7e5fb10c917e209b25", null],
 ["closestPoint", "class_go_1_1_param_volume.html#aa63791c277eadd80f922c180d4f1327f", null],
 ["constParamSurface", "class_go_1_1_param_volume.html#acb411c77d591b150fb4324acdeabfe1e", null],
 ["getAllBoundarySurfaces", "class_go_1_1_param_volume.html#a05827157124165e1a563d25fe6cc2d32", null],
 ["isSpline", "class_go_1_1_param_volume.html#a6bfb414644129e1637a2c21bb9560946", null],
 ["nextSegmentVal", "class_go_1_1_param_volume.html#aeee67285d1ced1e0d5b589d0a555f5ff", null],
 ["parameterSpan", "class_go_1_1_param_volume.html#a5ca5317d0c2b6b337a59057f38f603d6", null],
 ["point", "class_go_1_1_param_volume.html#a20df50b9fd19d4b1033dc6d11826b96c", null],
 ["point", "class_go_1_1_param_volume.html#a58500ca3de9bc1ba3d810badleaf5df7", null],
 ["reverseParameterDirection", "class_go_1_1_param_volume.html#a5bca97a9401b74ca989e2e67f89ee6fe", null],
],
 ["swapParameterDirection", "class_go_1_1_param_volume.html#ab898ec524b6fabc57b49aea98b8d0724", null],
 ["tangentCone", "class_go_1_1_param_volume.html#af5b51bc450da49f76b73f038618a57b6", null],
 ["translate", "class_go_1_1_param_volume.html#a0aea3fc27f39a5f4805589a7b8744ec8", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_param\_volume.js.

## 30.354 doc/html/class\_go\_1\_1\_parametric\_surface\_tessellator.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_parametric\\_surface\\_tessellator](#)

### 30.354.1 Variable Documentation

#### 30.354.1.1 var class\_go\_1\_1\_parametric\_surface\_tessellator

##### Initial value:

```
=
[
 ["ParametricSurfaceTessellator", "class_go_1_1_parametric_surface_tessellator.html#a77549810bc3c8c1ce9f06345f46f0fde", null],
 ["~ParametricSurfaceTessellator", "class_go_1_1_parametric_surface_tessellator.html#a4c6fef013ecb6c00b14d59801aad9a58", null],
 ["changeRes", "class_go_1_1_parametric_surface_tessellator.html#ac6cf0e50d6c2db47e3c0fd35d29a4d4b", null],
 ["getMesh", "class_go_1_1_parametric_surface_tessellator.html#aae43406f79ade6e3ce59bc63bf086398", null],
 ["getRes", "class_go_1_1_parametric_surface_tessellator.html#a4296acdedb75f67f557ea1238680b399", null],
 ["tessellate", "class_go_1_1_parametric_surface_tessellator.html#af0a157fa8039d15ed49d25eccf1d6d8b", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_parametric\_surface\_tessellator.js.

## 30.355 doc/html/class\_go\_1\_1\_plane.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_plane](#)

### 30.355.1 Variable Documentation

#### 30.355.1.1 var class\_go\_1\_1\_plane

Definition at line 1 of file class\_go\_1\_1\_plane.js.

## 30.356 doc/html/class\_go\_1\_1\_plane\_int.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_plane\\_int](#)

### 30.356.1 Variable Documentation

#### 30.356.1.1 var class\_go\_1\_1\_plane\_int

#### Initial value:

```
=
[
 ["PlaneInt", "class_go_1_1_plane_int.html#ace6d9c215dbaac46b7e0cc2c7840a03c", null],
 ["PlaneInt", "class_go_1_1_plane_int.html#a9a3a7f5fcaaf1ef41b8868973358f807", null],
 ["PlaneInt", "class_go_1_1_plane_int.html#ad30a3bef23db55954c556ac2ab4e166a", null],
 ["~PlaneInt", "class_go_1_1_plane_int.html#a1cb315c135f20f3d5da0daf0bee52126", null],
 ["a", "class_go_1_1_plane_int.html#a75c4540ff637b7a3f2a33182e81fef33", null],
 ["b", "class_go_1_1_plane_int.html#aa61eda35ce195ebbb59321069d0b20f1", null],
 ["c", "class_go_1_1_plane_int.html#a4f9d4c6288fb0ee151a466648816fca1", null],
 ["d", "class_go_1_1_plane_int.html#adbef9642db16186c9411dce8c3fd7c92", null],
 ["read", "class_go_1_1_plane_int.html#affbfeab5d401db79d65ef540646358ce", null],
 ["surface", "class_go_1_1_plane_int.html#ae3448a0854290ca30e8255d0caf35c01", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_plane\_int.js.

## 30.357 doc/html/class\_go\_1\_1\_point.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_point](#)

### 30.357.1 Variable Documentation

#### 30.357.1.1 var class\_go\_1\_1\_point

Definition at line 1 of file class\_go\_1\_1\_point.js.

## 30.358 doc/html/class\_go\_1\_1\_point\_cloud.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_point\\_cloud](#)

### 30.358.1 Variable Documentation

#### 30.358.1.1 var class\_go\_1\_1\_point\_cloud

#### Initial value:

```
=
[
 ["PointCloud", "class_go_1_1_point_cloud.html#a7490ab07b2c7cf4310cce602a7ab9986", null],
 ["PointCloud", "class_go_1_1_point_cloud.html#a8f97cb147f7ebee848cdbcabbd1b2a9", null],
 ["PointCloud", "class_go_1_1_point_cloud.html#a0f7871311a54860fc2034cf0e04e0b9f", null],
 ["~PointCloud", "class_go_1_1_point_cloud.html#ae5ea6d7258c0de75830c052d71560749", null],
 ["boundingBox", "class_go_1_1_point_cloud.html#a2dd12e28999339fclc26e7f2746cfcc7", null],
 ["clone", "class_go_1_1_point_cloud.html#ae5ca0b7dfbb382a470f6320850fb937d", null],
 ["dimension", "class_go_1_1_point_cloud.html#ab41af5a432bbcaf4bd0e33fc7e40ecee", null],
 ["instanceType", "class_go_1_1_point_cloud.html#a03360c883d516a0d92126fd43e57ab79", null],
 ["numPoints", "class_go_1_1_point_cloud.html#a2f7f157ac7a85a17988a929d9c6bf46e", null],
 ["point", "class_go_1_1_point_cloud.html#ad7ac0d21c2c40d989392581665d9de18", null],
 ["point", "class_go_1_1_point_cloud.html#ac3a2b633ec9d4e1bb2046efafdf58c5c", null],
 ["pointVector", "class_go_1_1_point_cloud.html#a41f059d1c220fe273bb9c56904fec441", null],
 ["rawData", "class_go_1_1_point_cloud.html#aeb184284d03948126f32911f9bc18300", null],
 ["rawData", "class_go_1_1_point_cloud.html#a8d46240635ff22b7c3317c8a220a8eee", null],
 ["read", "class_go_1_1_point_cloud.html#afc7e14c4ab1b7b37ec56314124a28d18", null],
 ["rotate", "class_go_1_1_point_cloud.html#a452e3de6386a0028ce7cc8ed4be1d8f7", null],
 ["translate", "class_go_1_1_point_cloud.html#afcd16132768c8ca62f09e23ed8f6538d", null],
 ["write", "class_go_1_1_point_cloud.html#ad89fec212c58f01559025679842c0c1f", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_point\_cloud.js.

## 30.359 doc/html/class\_go\_1\_1\_point\_on\_curve.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_point\\_on\\_curve](#)

### 30.359.1 Variable Documentation

#### 30.359.1.1 var class\_go\_1\_1\_point\_on\_curve

##### Initial value:

```
=
[
 ["PointOnCurve", "class_go_1_1_point_on_curve.html#af7dc3923025103fe26069d15c6d79323", null],
 ["PointOnCurve", "class_go_1_1_point_on_curve.html#a707f6c1f8eb557cbf82bf024549357ca", null],
 ["PointOnCurve", "class_go_1_1_point_on_curve.html#ad7f9d6420bc461a316d17f5246d12318", null],
 ["~PointOnCurve", "class_go_1_1_point_on_curve.html#a9d4a33cf169f40a915691bfae715b5ff", null],
 ["evaluate", "class_go_1_1_point_on_curve.html#af6e6545ee40518b31c4965960beb6458", null],
 ["getCurve", "class_go_1_1_point_on_curve.html#a62985eeecb070c3a07358411c8746be3", null],
 ["getPar", "class_go_1_1_point_on_curve.html#ad690d68c8b17aef2f7cf91cf1cb3bde7", null],
 ["getPos", "class_go_1_1_point_on_curve.html#ac6254eed9d25e8467a0b0438411deda8", null],
 ["setParInterval", "class_go_1_1_point_on_curve.html#aec36cbd9c3c7255624655661d3fbc286", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_point\_on\_curve.js.

## 30.360 doc/html/class\_go\_1\_1\_point\_on\_edge.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_point\\_on\\_edge](#)

### 30.360.1 Variable Documentation

#### 30.360.1.1 var class\_go\_1\_1\_point\_on\_edge

##### Initial value:

```
=
[
 ["PointOnEdge", "class_go_1_1_point_on_edge.html#a5e5f979e946582190563dec5095b2eab", null],
 ["~PointOnEdge", "class_go_1_1_point_on_edge.html#a3e5ceaf951feebf66b5a8e1f2c48b0df", null],
 ["edge", "class_go_1_1_point_on_edge.html#aaae9baee52e5453776627caa2fa9f37c", null],
 ["par", "class_go_1_1_point_on_edge.html#a7adaa2b8c7e136e1f073073ed9a0522d", null],
 ["position", "class_go_1_1_point_on_edge.html#aa17d25f9201f436aa7b23478f4dacedf", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_point\_on\_edge.js.

## 30.361 doc/html/class\_go\_1\_1\_point\_sequence.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_point\\_sequence](#)



### 30.361.1 Variable Documentation

#### 30.361.1.1 var class\_go\_1\_1\_point\_sequence

##### Initial value:

```
=
[
 ["PointSequence", "class_go_1_1_point_sequence.html#a6ede86745290d981b66486135dbd3726", null],
 ["PointSequence", "class_go_1_1_point_sequence.html#a814deda46911d830408d171c2d6ba7ee", null],
 ["PointSequence", "class_go_1_1_point_sequence.html#a5451cae773d6f7b8a3b09ad6fc2bbf44", null],
 ["PointSequence", "class_go_1_1_point_sequence.html#ad03a7c46eda2eefcd84a95138c8360a5", null],
 ["~PointSequence", "class_go_1_1_point_sequence.html#a1b6de6c0745f63fb78e18de7d9ac49d8", null],
 ["coefs_begin", "class_go_1_1_point_sequence.html#aa10cbbc7ab05c9a6bfd342c3f7107f9f", null],
 ["coefs_begin", "class_go_1_1_point_sequence.html#af8497ebb399a76f8f0e7dd75c729a34c", null],
 ["coefs_end", "class_go_1_1_point_sequence.html#aa5464ecec7c7152430533b333723d558", null],
 ["coefs_end", "class_go_1_1_point_sequence.html#aa383d5038fecac721e34b4b133c4ee01", null],
 ["dimension", "class_go_1_1_point_sequence.html#aaed3864489fec99e60c824e5a8abab7b", null],
 ["grid_dimension", "class_go_1_1_point_sequence.html#af79419e2681dfa632d2a624d3e94512b", null],
 ["grid_length", "class_go_1_1_point_sequence.html#ad33f7300513ca8053eacce4c50430cf7", null],
 ["type", "class_go_1_1_point_sequence.html#a340ee55ee4cb9794d34caf81378e55d4", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_point\_sequence.js.

## 30.362 doc/html/class\_go\_1\_1\_project\_curve.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_project\\_curve](#)

### 30.362.1 Variable Documentation

#### 30.362.1.1 var class\_go\_1\_1\_project\_curve

##### Initial value:

```
=
[
 ["ProjectCurve", "class_go_1_1_project_curve.html#a903a63cbfc5d148e426569683cd35c9a", null],
 ["~ProjectCurve", "class_go_1_1_project_curve.html#aee78cf0ca3ae9e2d100d3f105c07b1ec", null],
 ["approximationOK", "class_go_1_1_project_curve.html#aa3dac6e8cd4eff8508c08069fb8f3593", null],
 ["dim", "class_go_1_1_project_curve.html#ac7df656a336fc7b892b78abfe6bfb2f0", null],
 ["end", "class_go_1_1_project_curve.html#a0ecec1dae182c841b612368fdff62d60d", null],
 ["eval", "class_go_1_1_project_curve.html#af98blced4a661a5876e60db1d659f4fa", null],
 ["eval", "class_go_1_1_project_curve.html#a96186d4cf98720c6777bfd7686f27f49", null],
 ["eval", "class_go_1_1_project_curve.html#aa04bfd8be7761c82b715e59ab2537bc3", null],
 ["start", "class_go_1_1_project_curve.html#ad717dabd8cfd3e33c8a8f2ec2c15d108", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_project\_curve.js.

## 30.363 doc/html/class\_go\_1\_1\_project\_curve\_and\_cross\_tan.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_project\\_curve\\_and\\_cross\\_tan](#)

### 30.363.1 Variable Documentation

#### 30.363.1.1 var class\_go\_1\_1\_project\_curve\_and\_cross\_tan

##### Initial value:

```
=
[
 ["ProjectCurveAndCrossTan", "
 class_go_1_1_project_curve_and_cross_tan.html#aa98cefc1fc3a7db17ca6bf8adf1cc2cb", null],
 ["~ProjectCurveAndCrossTan", "
 class_go_1_1_project_curve_and_cross_tan.html#ab15663097429c73fd5aabc8bf688dd0c", null],
 ["approximationOK", "class_go_1_1_project_curve_and_cross_tan.html#a8e4d5da040b79e7292393b61e96e807e",
 null],
 ["dim", "class_go_1_1_project_curve_and_cross_tan.html#a97b9499363e6ee7ec50c328b9a4c3be7", null],
 ["end", "class_go_1_1_project_curve_and_cross_tan.html#a2b9c6580329f03cd67d13be3fc1c6bfb", null],
 ["eval", "class_go_1_1_project_curve_and_cross_tan.html#a2887b1adafc03d983eb8405ea079f828", null],
 ["eval", "class_go_1_1_project_curve_and_cross_tan.html#a30550274a1876d054692881e34d62f74", null],
 ["nmbCvs", "class_go_1_1_project_curve_and_cross_tan.html#a30550274a1876d054692881e34d62f74", null],
 ["start", "class_go_1_1_project_curve_and_cross_tan.html#a208c12f85f1e92823c65e59a8b8ce593", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_project\_curve\_and\_cross\_tan.js.

## 30.364 doc/html/class\_go\_1\_1\_project\_intersection\_curve.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_project\\_intersection\\_curve](#)

### 30.364.1 Variable Documentation

#### 30.364.1.1 var class\_go\_1\_1\_project\_intersection\_curve

##### Initial value:

```
=
[
 ["ProjectIntersectionCurve", "
 class_go_1_1_project_intersection_curve.html#a94cbcd76f44f8128cf07e807fb51e5aa", null],
 ["~ProjectIntersectionCurve", "
 class_go_1_1_project_intersection_curve.html#a54b56fd7d6b0b35a7b91058e249443f9", null],
 ["approximationOK", "class_go_1_1_project_intersection_curve.html#a98edd82310fd00192191691dd4015fb8",
 null],
 ["dim", "class_go_1_1_project_intersection_curve.html#ad1d0f0e7b7ffd406d0f341c2e7caca5f", null],
 ["end", "class_go_1_1_project_intersection_curve.html#a0b38e17e16d84dda80bf971151f30fc6", null],
 ["eval", "class_go_1_1_project_intersection_curve.html#a6c51a345c6bc7eef9ad51c6a6af1a6a7", null],
 ["eval", "class_go_1_1_project_intersection_curve.html#ae83c55aba4c48f4f9e67ee10a469eebd", null],
 ["start", "class_go_1_1_project_intersection_curve.html#a513617c2b52868c77eddb71a2e57571b", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_project\_intersection\_curve.js.

## 30.365 doc/html/class\_go\_1\_1\_pt\_pt\_intersector.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_pt\\_pt\\_intersector](#)

### 30.365.1 Variable Documentation

#### 30.365.1.1 var class\_go\_1\_1\_pt\_pt\_intersector

##### Initial value:

```
=
[
 ["PtPtIntersector", "class_go_1_1_pt_pt_intersector.html#a8abebb35c917366d6bbaf41829b286ea", null],
 ["PtPtIntersector", "class_go_1_1_pt_pt_intersector.html#a25f10f779cccf595aea7571b90967b0c", null],
 ["~PtPtIntersector", "class_go_1_1_pt_pt_intersector.html#a79c8f5cee79db3b1ecf43f8107b79c55", null],
 ["checkCoincidence", "class_go_1_1_pt_pt_intersector.html#ab52e5b1f4165d2039a897d8cbcf156a9", null],
 ["doSubdivide", "class_go_1_1_pt_pt_intersector.html#a320d02f07500b9860d90f1d1c2f5d7cb", null],
 ["linearCase", "class_go_1_1_pt_pt_intersector.html#a0dd0c8e19b541c4062ba5ccd5f73ce76", null],
 ["lowerOrderIntersector", "class_go_1_1_pt_pt_intersector.html#a6944d80db8438f36edef75789e19e720",
 null],
 ["microCase", "class_go_1_1_pt_pt_intersector.html#afdd93b4cc9a0eaaaf17125d6a3497da15", null],
 ["numParams", "class_go_1_1_pt_pt_intersector.html#a40f4c4e8b3303e805b27410c06ba5933", null],
 ["repairIntersections", "class_go_1_1_pt_pt_intersector.html#a2f0b05861cfb6be0e2897cd2f28b0893", null
],
 ["updateIntersections", "class_go_1_1_pt_pt_intersector.html#a8bd72351be94e541948b869caca74671", null
]
]
```

Definition at line 1 of file class\_go\_1\_1\_pt\_pt\_intersector.js.

## 30.366 doc/html/class\_go\_1\_1\_quad\_mesh.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_quad\\_mesh](#)

### 30.366.1 Variable Documentation

#### 30.366.1.1 var class\_go\_1\_1\_quad\_mesh

##### Initial value:

```
=
[
 ["QuadMesh", "class_go_1_1_quad_mesh.html#a42a89aad9e6131930a1a030446d8cbd4", null],
 ["atBoundary", "class_go_1_1_quad_mesh.html#ad422630ca2b536ac79ce4f4b30df1017", null],
 ["normalArray", "class_go_1_1_quad_mesh.html#a530f168f59723189046abf2c8a314ed1", null],
 ["numQuads", "class_go_1_1_quad_mesh.html#ad1cc15c4ab4222a6d92291c7d20b9e1a", null],
 ["numTriangles", "class_go_1_1_quad_mesh.html#aac1def69f7fb92e427c3374a612407d8", null],
 ["numVertices", "class_go_1_1_quad_mesh.html#aad6b01f5c482f7130a3169e11a1119ec", null],
 ["paramArray", "class_go_1_1_quad_mesh.html#a62a86c6ad3deb2b5d69535e57ce268b2", null],
 ["quadIndexArray", "class_go_1_1_quad_mesh.html#ab0e105fbfa07d09fc909edcfa843a601", null],
 ["resize", "class_go_1_1_quad_mesh.html#abec0e7c5c07b8c3f3e9a2e64d5dc8825", null],
 ["texcoordArray", "class_go_1_1_quad_mesh.html#a8f4773fb463b4596728438fe326ce9c2", null],
 ["triangleIndexArray", "class_go_1_1_quad_mesh.html#a51a80ee53f7f8c9177e59396088cec5a", null],
 ["useNormals", "class_go_1_1_quad_mesh.html#a1c6c4ccf80cfaa4f74032526e854f3e7", null],
 ["useTexCoords", "class_go_1_1_quad_mesh.html#a444e1bc577f7f56e98c2d1a0e922deea", null],
 ["vertexArray", "class_go_1_1_quad_mesh.html#a3af97e3ee79b09302191ff51f01b6f6a", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_quad\_mesh.js.

## 30.367 doc/html/class\_go\_1\_1\_quality\_results.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_quality\\_results](#)

### 30.367.1 Variable Documentation

#### 30.367.1.1 var class\_go\_1\_1\_quality\_results

##### Initial value:

```
=
[
 ["~QualityResults", "class_go_1_1_quality_results.html#a5e06951b5eeb29104be4a68f080ec03c", null],
 ["FaceSetQuality", "class_go_1_1_quality_results.html#ae820d513170db3b6b7ce7887b8a0d4dc", null],
 ["FaceSetRepair", "class_go_1_1_quality_results.html#a18e8ba52e738ee5a12e03dab6f012c27", null],
 ["ModelQuality", "class_go_1_1_quality_results.html#a8b28ece74f17b35d5d93ee5b4e8bb3a1", null],
 ["ModelRepair", "class_go_1_1_quality_results.html#adbeec2dc762f5b47140d310dbba07cf8", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_quality\_results.js.

## 30.368 doc/html/class\_go\_1\_1\_rational.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_rational](#)

### 30.368.1 Variable Documentation

#### 30.368.1.1 var class\_go\_1\_1\_rational

##### Initial value:

```
=
[
 ["Rational", "class_go_1_1_rational.html#a986b1722c95bc708600cfc0114e0451a", null],
 ["Rational", "class_go_1_1_rational.html#acadf8909244c77f7bb16036cd69d0700", null],
 ["Rational", "class_go_1_1_rational.html#a2bd8648917321cbfa7bb8b04279b582a", null],
 ["operator!=", "class_go_1_1_rational.html#ab4ae93c96df696d22d2e6c6340508c33", null],
 ["operator*=", "class_go_1_1_rational.html#ad4340450c2910a6fed4bab7a3e571e33", null],
 ["operator+=", "class_go_1_1_rational.html#a8748a34a9c3d6c49b2ee9f1d56424684", null],
 ["operator-", "class_go_1_1_rational.html#a1859a8d2e77f0fc87402311299436117", null],
 ["operator--", "class_go_1_1_rational.html#a587873f26e68502c1451badf78fc84d6", null],
 ["operator/=", "class_go_1_1_rational.html#a8663ea66582a6031b6e04323316c8a9e", null],
 ["operator==", "class_go_1_1_rational.html#a64fc86530525676d751aefb80677e73b", null],
 ["simplify", "class_go_1_1_rational.html#a856e2194491fe5c02d399ef93ed74fb4", null],
 ["write", "class_go_1_1_rational.html#ac7152300ba2fd30d30012851ae15feff", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_rational.js.

## 30.369 doc/html/class\_go\_1\_1\_rect\_domain.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_rect\\_domain](#)

### 30.369.1 Variable Documentation

#### 30.369.1.1 var class\_go\_1\_1\_rect\_domain

##### Initial value:

```
=
[
 ["RectDomain", "class_go_1_1_rect_domain.html#a6139e6ec57ff66593293bc88ce6f9e07", null],
 ["RectDomain", "class_go_1_1_rect_domain.html#a06296bd084d733d10574aeaab9bbe6d0", null],
 ["~RectDomain", "class_go_1_1_rect_domain.html#a70400b3359fb09ba4d499b7bfeaffe25", null],
 ["addUnionWith", "class_go_1_1_rect_domain.html#a7eef80b8d800ef0c13526c7b0548d877", null],
 ["closestInDomain", "class_go_1_1_rect_domain.html#abe899d3313ccc0d7d66ba21b542c3c20", null],
 ["closestOnBoundary", "class_go_1_1_rect_domain.html#a160cc78d4037a479ce7025397dff06a9", null],
 ["diagLength", "class_go_1_1_rect_domain.html#a6945d4911e2f94263b5b0207e2ccad3e", null],
 ["intersectWith", "class_go_1_1_rect_domain.html#a75d70b5c718c4c7e322cbaa715040401", null],
 ["isInDomain", "class_go_1_1_rect_domain.html#a4b6fd9c3cabebb20053f4a5294d3645c", null],
 ["isInDomain2", "class_go_1_1_rect_domain.html#ad0913c6796862086c00ec3f886fffb3c5", null],
 ["isOnBoundary", "class_go_1_1_rect_domain.html#a6cbe39ae203ae9544d1de840a69d4b39", null],
 ["isOnCorner", "class_go_1_1_rect_domain.html#a006f5cfb2f639ff1002cb26ea48823d7", null],
 ["lowerLeft", "class_go_1_1_rect_domain.html#aa39c894a00fd949d85ae545da89feb7e", null],
 ["umax", "class_go_1_1_rect_domain.html#a262526c21dfellaaafa83df300bf7cd0", null],
 ["umin", "class_go_1_1_rect_domain.html#af2bc2771491bc66906aceae6b077c08d", null],
 ["upperRight", "class_go_1_1_rect_domain.html#a85738771848251c582c3f45fcdc73b4f", null],
 ["vmax", "class_go_1_1_rect_domain.html#a1098315f51edb78f585dfa554ec0ef82", null],
 ["vmin", "class_go_1_1_rect_domain.html#ac0f08bf7180b9bdcfafe21c9ec19b9cc", null],
 ["whichBoundary", "class_go_1_1_rect_domain.html#a846d3147d576b8c7b1ddad3a5d282ee7", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_rect\_domain.js.

## 30.370 doc/html/class\_go\_1\_1\_rect\_grid.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_rect\\_grid](#)

### 30.370.1 Variable Documentation

#### 30.370.1.1 var class\_go\_1\_1\_rect\_grid

##### Initial value:

```
=
[
 ["SwapGrid", "class_go_1_1_rect_grid.html#a405bf6317ae2a11be80be4f42c2cc892", null],
 ["RectGrid", "class_go_1_1_rect_grid.html#a706aca3d44753123da4927551af3426b", null],
 ["~RectGrid", "class_go_1_1_rect_grid.html#aa5a6481237c7dd736a85778e178eb6c0", null],
 ["boundingBox", "class_go_1_1_rect_grid.html#a4047b6a21de4b11d61b55858c5c5a62", null],
 ["clone", "class_go_1_1_rect_grid.html#a1alad87fbda4442c4d4afb3925be4710", null],
 ["dimension", "class_go_1_1_rect_grid.html#a7e7c780dbc4b78c1400b23627a4c1d9c", null],
 ["instanceType", "class_go_1_1_rect_grid.html#a1d2eaff39f755862c57e1a6499c8cc48", null],
 ["numCoefs_u", "class_go_1_1_rect_grid.html#af20d6c6ee42c7b4380b8d1830a6f4acd", null],
 ["numCoefs_v", "class_go_1_1_rect_grid.html#a138dfe11ad3e11e78791c481af903e00", null],
 ["rawData", "class_go_1_1_rect_grid.html#a2228c05541484aeec6f376769db48a3f", null],
 ["rawData", "class_go_1_1_rect_grid.html#a8838e86fa3c332093ed997b3233b7102", null],
 ["read", "class_go_1_1_rect_grid.html#ab674ee685f79f10cbf7513391c7b138f", null],
 ["setGrid", "class_go_1_1_rect_grid.html#a25aa21a9281f7f4abd0f8ed49e66a4b6", null],
 ["swapDirections", "class_go_1_1_rect_grid.html#a3990ed52f8ced0fbae066db4bf47c5af", null],
 ["write", "class_go_1_1_rect_grid.html#aaff9c01ccc35a6db444c68e8c62e31335", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_rect\_grid.js.

## 30.371 doc/html/class\_go\_1\_1\_rect\_grid\_tesselator.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_rect\\_grid\\_tesselator](#)

#### 30.371.1 Variable Documentation

##### 30.371.1.1 var class\_go\_1\_1\_rect\_grid\_tesselator

#### Initial value:

```
=
[
 ["RectGridTesselator", "class_go_1_1_rect_grid_tesselator.html#a9df6f4cf94ed15445f949330c1324b91",
 null],
 ["~RectGridTesselator", "class_go_1_1_rect_grid_tesselator.html#a940255dc109fdc0b63e9be1b0599eb82",
 null],
 ["getMesh", "class_go_1_1_rect_grid_tesselator.html#a5386d70b1a291ebf8c4b217fc43883b4", null],
 ["tesselate", "class_go_1_1_rect_grid_tesselator.html#a37c7dac1717250aae0fe930161e4406b", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_rect\_grid\_tesselator.js.

## 30.372 doc/html/class\_go\_1\_1\_rectangular\_surface\_tesselator.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_rectangular\\_surface\\_tesselator](#)

#### 30.372.1 Variable Documentation

##### 30.372.1.1 var class\_go\_1\_1\_rectangular\_surface\_tesselator

#### Initial value:

```
=
[
 ["RectangularSurfaceTesselator", "
class_go_1_1_rectangular_surface_tesselator.html#a94f824a5aa465420949b0ec0ea023ec3", null],
 ["~RectangularSurfaceTesselator", "
class_go_1_1_rectangular_surface_tesselator.html#a653dcbcd23292b65b7bd762cc581338a", null],
 ["changeIsolineNumRes", "
class_go_1_1_rectangular_surface_tesselator.html#af26e32ee9754f890c6623043a5a73944", null],
 ["changeIsolines", "class_go_1_1_rectangular_surface_tesselator.html#a19df054bd56bfff01332bb12564b6cfce",
 null],
 ["changeRes", "class_go_1_1_rectangular_surface_tesselator.html#a0b0146a3cc6bf0f683811b1c2414d49e",
 null],
 ["getIsolineNumRes", "
class_go_1_1_rectangular_surface_tesselator.html#aff81e68e4dcf66dcfe2e6441dec7aee6", null],
 ["getIsolines", "class_go_1_1_rectangular_surface_tesselator.html#a54cb336dcd511cddfe642c061b91fc53",
 null],
 ["getIsolineStrips", "
class_go_1_1_rectangular_surface_tesselator.html#ae6753b2a326964bed1da65145579851b", null],
 ["getMesh", "class_go_1_1_rectangular_surface_tesselator.html#aa52bea5d343ea5c478adbb7423fc6421", null
],
 ["getRes", "class_go_1_1_rectangular_surface_tesselator.html#afd3e3c52a9a795e7158ca34e12c0d972", null
],
 ["tesselate", "class_go_1_1_rectangular_surface_tesselator.html#a98bc65dc65f847c17602033717c41843",
 null]
]
```

Definition at line 1 of file class\_go\_1\_1\_rectangular\_surface\_tesselator.js.

## 30.373 doc/html/class\_go\_1\_1\_rectangular\_volume\_tesselator.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_rectangular\\_volume\\_tesselator](#)

#### 30.373.1 Variable Documentation

##### 30.373.1.1 var class\_go\_1\_1\_rectangular\_volume\_tesselator

###### Initial value:

```
=
[
 ["RectangularVolumeTesselator", "
 class_go_1_1_rectangular_volume_tesselator.html#a8ca37a7589470343564d4b0e1ab70d3a", null],
 ["~RectangularVolumeTesselator", "
 class_go_1_1_rectangular_volume_tesselator.html#a2f2d395814ce7e30b16c9f1d9445a194", null],
 ["changeRes", "class_go_1_1_rectangular_volume_tesselator.html#aec3b68a4550e588adc68030b0cb9d3ea",
 null],
 ["getMesh", "class_go_1_1_rectangular_volume_tesselator.html#a0e95ba4459220e4b64bb0a365d44194c", null
],
 ["getRes", "class_go_1_1_rectangular_volume_tesselator.html#ab6b4b2ae2d4fe17754d89d5e058f712a", null]
 ,
 ["tesselate", "class_go_1_1_rectangular_volume_tesselator.html#a81d00516682f244ebac9097b6de6da78",
 null]
]
```

Definition at line 1 of file class\_go\_1\_1\_rectangular\_volume\_tesselator.js.

## 30.374 doc/html/class\_go\_1\_1\_registrator.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_registrator](#)

#### 30.374.1 Variable Documentation

##### 30.374.1.1 var class\_go\_1\_1\_registrator

###### Initial value:

```
=
[
 ["Registrator", "class_go_1_1_registrator.html#a4a0166f70ae0c394255192a1de75cdcd", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_registrator.js.

## 30.375 doc/html/class\_go\_1\_1\_regular\_mesh.js File Reference

### Variables

- var `class_go_1_1_regular_mesh`

### 30.375.1 Variable Documentation

#### 30.375.1.1 var `class_go_1_1_regular_mesh`

##### Initial value:

```
=
[
 ["RegularMesh", "class_go_1_1_regular_mesh.html#a29769a85f7b602a60a0f40cec517f2", null],
 ["~RegularMesh", "class_go_1_1_regular_mesh.html#a6e7f58ac32c6dede6759e8bf0cddca82", null],
 ["asRegularMesh", "class_go_1_1_regular_mesh.html#ad7b90ae67f047691a9bb14caa942c7e4", null],
 ["atBoundary", "class_go_1_1_regular_mesh.html#a2f536b939f395cf4ca82dcc52009a83d", null],
 ["normalArray", "class_go_1_1_regular_mesh.html#a94757a0a4f89d6f175fce85322c2336e", null],
 ["numStrips", "class_go_1_1_regular_mesh.html#a90e12891714f58f59ba7277f97e49f97", null],
 ["numTriangles", "class_go_1_1_regular_mesh.html#ade5afca2996ffc7c37c97f8d54320015", null],
 ["numVertices", "class_go_1_1_regular_mesh.html#ale3b557e2f657967e3def1ebee433779", null],
 ["paramArray", "class_go_1_1_regular_mesh.html#ae09bbe206c2c880bf58368a2ab4d48f6", null],
 ["resize", "class_go_1_1_regular_mesh.html#a708b79258bc5bbb1fd2bfff72da9e35e", null],
 ["stripArray", "class_go_1_1_regular_mesh.html#afe793fae50dc92ea2827ef43d8f9b248", null],
 ["stripLength", "class_go_1_1_regular_mesh.html#a31db9c56238e8ec9d1a0e743ee78232e", null],
 ["texcoordArray", "class_go_1_1_regular_mesh.html#a4f7d4bcb69cbe7ebefa3ca9e20212535", null],
 ["translate", "class_go_1_1_regular_mesh.html#ac7c87ccafd92b52a264ca0e42b54494d", null],
 ["triangleArray", "class_go_1_1_regular_mesh.html#a21a2c136bca8cfff0423630ae15aa622d", null],
 ["triangleIndexArray", "class_go_1_1_regular_mesh.html#a2a712930e8f589c6f476ef44fef45eb4", null],
 ["useNormals", "class_go_1_1_regular_mesh.html#a7b4fb9e0919ab6d13b0f4ea7c79a7a22", null],
 ["useTexCoords", "class_go_1_1_regular_mesh.html#a9ce0af7c87c09f192914683a83052c93", null],
 ["vertexArray", "class_go_1_1_regular_mesh.html#a8864170fb6e43b991e6d7aa824a59eec", null]
]
```

Definition at line 1 of file `class_go_1_1_regular_mesh.js`.

## 30.376 doc/html/class\_go\_1\_1\_regular\_vol\_mesh.js File Reference

### Variables

- var `class_go_1_1_regular_vol_mesh`

### 30.376.1 Variable Documentation

#### 30.376.1.1 var `class_go_1_1_regular_vol_mesh`

##### Initial value:

```
=
[
 ["RegularVolMesh", "class_go_1_1_regular_vol_mesh.html#ac4fc411b5c481015ac0689c83ee9989e", null],
 ["~RegularVolMesh", "class_go_1_1_regular_vol_mesh.html#a7372faed2d9a370c63c7493aa98dfa49", null],
 ["asRegularMesh", "class_go_1_1_regular_vol_mesh.html#a6fac7e08e05112d16e4bb6e5e3212017", null],
 ["atBoundary", "class_go_1_1_regular_vol_mesh.html#a323aac367245aba58261f0eec3749a67", null],
 ["normalArray", "class_go_1_1_regular_vol_mesh.html#aa033c470d67c77ffa9ff3029fdbd283d", null],
 ["numStrips", "class_go_1_1_regular_vol_mesh.html#ab329df8832a21b5ad327e99839e9e71d", null],
 ["numTriangles", "class_go_1_1_regular_vol_mesh.html#a30e4455a9c235be9dd1a22da1750c304", null],
 ["numVertices", "class_go_1_1_regular_vol_mesh.html#aac90df3b33cb53c7bc82565d0d6e3e5b", null],
 ["paramArray", "class_go_1_1_regular_vol_mesh.html#a011e0502fb2df42b1702174f9432794c", null],
 ["resize", "class_go_1_1_regular_vol_mesh.html#a4edc77e881efd887150ff09572e8c7b4", null],
 ["stripArray", "class_go_1_1_regular_vol_mesh.html#ac3ac9119971a83aa32d23e5d3d4195dd", null],
 ["stripLength", "class_go_1_1_regular_vol_mesh.html#ad960f397db6b7c381abb637eb15dfad9", null],
 ["texcoordArray", "class_go_1_1_regular_vol_mesh.html#aa00172245efedb41eb2654d5a39a50e2", null],
 ["triangleArray", "class_go_1_1_regular_vol_mesh.html#af5396255502298b2bb3a7b175254d2da", null],
 ["triangleIndexArray", "class_go_1_1_regular_vol_mesh.html#a2e6e9892f7b90e98a932a97999cb09363", null],
 ["useNormals", "class_go_1_1_regular_vol_mesh.html#aa181fa9878a84e2922b3920dddf7da0f", null],
 ["useTexCoords", "class_go_1_1_regular_vol_mesh.html#a1c2bd4d2707c711bdbc3ef8217abd72f", null],
 ["vertexArray", "class_go_1_1_regular_vol_mesh.html#aa4alae38f84a2def4df60ade47676a39", null]
]
```

Definition at line 1 of file `class_go_1_1_regular_vol_mesh.js`.



## 30.377 doc/html/class\_go\_1\_1\_regularize\_face.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_regularize\\_face](#)

### 30.377.1 Variable Documentation

#### 30.377.1.1 var class\_go\_1\_1\_regularize\_face

##### Initial value:

```
=
[
 ["RegularizeFace", "class_go_1_1_regularize_face.html#ae66c33b8e763792881e3fe8d48d49bc6", null],
 ["RegularizeFace", "class_go_1_1_regularize_face.html#ae4eda51c1c8af89aedcf3a14f1b984a9", null],
 ["RegularizeFace", "class_go_1_1_regularize_face.html#a8930b47f709f146c62e441863074b175", null],
 ["~RegularizeFace", "class_go_1_1_regularize_face.html#ac717a1845370d38f03cfbdf7bbfc427", null],
 ["classifyVertices", "class_go_1_1_regularize_face.html#a33df81a44d92b45ceba2e2bd16133db2", null],
 ["fetchVxPntCorr", "class_go_1_1_regularize_face.html#a919c13841fade539b38a4de946ba5de8", null],
 ["getRegularFaces", "class_go_1_1_regularize_face.html#a20ab6b81d7092740b17edef7edd9fdcd", null],
 ["getSeamJointInfo", "class_go_1_1_regularize_face.html#aaf4ebal6bbd976d0b2aee107129aee67", null],
 ["setAxis", "class_go_1_1_regularize_face.html#ae59fb2b79693d9a47906ccfe66f44e9b", null],
 ["setCandSplit", "class_go_1_1_regularize_face.html#a72d4abae8e9aabe7aee610e67187bce9", null],
 ["setDivideInT", "class_go_1_1_regularize_face.html#a92a9f45e89d93c3bd816a8aba6f53ddb", null],
 ["setNonTjointFaces", "class_go_1_1_regularize_face.html#a5e3aa322a925409e1f5967d82988c16c", null],
 ["setSplitMode", "class_go_1_1_regularize_face.html#a5f4880db4aac0cc6d1f5adcf39071de6", null],
 ["unsetAxis", "class_go_1_1_regularize_face.html#a2cd655766d474e9acadf7800b38b6139", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_regularize\_face.js.

## 30.378 doc/html/class\_go\_1\_1\_regularize\_face\_set.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_regularize\\_face\\_set](#)

### 30.378.1 Variable Documentation

#### 30.378.1.1 var class\_go\_1\_1\_regularize\_face\_set

##### Initial value:

```
=
[
 ["RegularizeFaceSet", "class_go_1_1_regularize_face_set.html#a2f2f0ecc6bb750aff0491f64231e1de4", null],
 ["RegularizeFaceSet", "class_go_1_1_regularize_face_set.html#af04cbe86d84deccb8046bebf8a684e8c", null],
 ["RegularizeFaceSet", "class_go_1_1_regularize_face_set.html#ad9edfadd6cccc6d4743e3323f838f2d4", null],
 ["~RegularizeFaceSet", "class_go_1_1_regularize_face_set.html#a8cab4384c3a972a4277d8019e1ed6c7e", null],
 ["fetchVxPntCorr", "class_go_1_1_regularize_face_set.html#a8fd526b05f5e833e036b35954b16042b", null],
 ["getModifiedAdjacentModels", "class_go_1_1_regularize_face_set.html#a34990507c71ab0eda950a7ac7d5f7262", null],
 ["getRegularFaces", "class_go_1_1_regularize_face_set.html#ac57ef67ddab543bacf1d58b326034a5d", null],
 ["getRegularModel", "class_go_1_1_regularize_face_set.html#a6252578be66c666df2c91cf9cc87f9fc", null],
 ["setFaceCorrespondance", "class_go_1_1_regularize_face_set.html#a60fd9cdce49edale5a4cccc02b2aa7656", null],
 ["setSplitMode", "class_go_1_1_regularize_face_set.html#aa1561739318f51557d51bb5544864310", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_regularize\_face\_set.js.

## 30.379 doc/html/class\_go\_1\_1\_rotated\_box.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_rotated\\_box](#)

### 30.379.1 Variable Documentation

#### 30.379.1.1 var class\_go\_1\_1\_rotated\_box

#### Initial value:

```
=
[
 ["RotatedBox", "class_go_1_1_rotated_box.html#a6e66e8aa97df21c0f3b9713f2282ec04", null],
 ["RotatedBox", "class_go_1_1_rotated_box.html#af1a80a629a9a03f8a49dc69e87f05fce", null],
 ["~RotatedBox", "class_go_1_1_rotated_box.html#a202785ee700306fdd294c3346e854846", null],
 ["box", "class_go_1_1_rotated_box.html#a595c578a5270ed58a88a7786ed9bbbea", null],
 ["containsBox", "class_go_1_1_rotated_box.html#ad4081a3019f80f5b974a40021689280b", null],
 ["containsPoint", "class_go_1_1_rotated_box.html#af492b49fc4134d967968a580b5130076", null],
 ["dimension", "class_go_1_1_rotated_box.html#aaaf893877961275937968119db5e1c98d", null],
 ["high", "class_go_1_1_rotated_box.html#ada40640cc038e93a5247edb0bad109cc", null],
 ["high_rot", "class_go_1_1_rotated_box.html#a1bd0b0f72dc928e337646752d5152e90", null],
 ["low", "class_go_1_1_rotated_box.html#a693566afb0380c8cf682c2d005a1a3a4", null],
 ["low_rot", "class_go_1_1_rotated_box.html#a001cfbfde2371156ffa216b2d018bfe2", null],
 ["overlaps", "class_go_1_1_rotated_box.html#a5ccb8f04bc93e7d8e6d7d30b68ad6e2b", null],
 ["setFromArray", "class_go_1_1_rotated_box.html#a3dc5a303224f779564e9a4c861b37827", null],
 ["setFromPoints", "class_go_1_1_rotated_box.html#a4d42e0ef9d11ae23de0d593a33c2ceb8", null]
]
```

Definition at line 1 of file `class_go_1_1_rotated_box.js`.

## 30.380 doc/html/class\_go\_1\_1\_scratch\_vect.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_scratch\\_vect](#)

### 30.380.1 Variable Documentation

#### 30.380.1.1 var class\_go\_1\_1\_scratch\_vect

#### Initial value:

```
=
[
 ["const_iterator", "class_go_1_1_scratch_vect.html#a3ec1bec2fc393949be4a21c24d98f87a", null],
 ["iterator", "class_go_1_1_scratch_vect.html#a32fb8c0c9542cc419f37d1d88d921fd1", null],
 ["ScratchVect", "class_go_1_1_scratch_vect.html#a6ce2b342b4fef39b03401b6bc1446fd8", null],
 ["ScratchVect", "class_go_1_1_scratch_vect.html#a3a7e8c7efc23c5b2f2f97ac6256c1bca", null],
 ["ScratchVect", "class_go_1_1_scratch_vect.html#a71c323d17436f29df665b72cc83053d3", null],
 ["ScratchVect", "class_go_1_1_scratch_vect.html#abb5789f4f2b661add44ecdb88756b1ef", null],
 ["ScratchVect", "class_go_1_1_scratch_vect.html#a4f2b1691b9acf034e049c057ecddc072", null],
 ["ScratchVect", "class_go_1_1_scratch_vect.html#a67c0c1ca3eb678e8bdd40f20cd93721a", null],
 ["assign", "class_go_1_1_scratch_vect.html#a31b2194dac225c9dee3ae40c4fd4db40", null],
 ["begin", "class_go_1_1_scratch_vect.html#a8da0cecb4d2706137378d32cf66bb47b", null],
 ["begin", "class_go_1_1_scratch_vect.html#a64a482100e4a62901119fec2a83f288b", null],
 ["clear", "class_go_1_1_scratch_vect.html#abd402d4e3e7377bab1ba270c98565492", null],
 ["end", "class_go_1_1_scratch_vect.html#a826bdc8790ec02ba7ccef221c7316747", null],
 ["end", "class_go_1_1_scratch_vect.html#a8e920737b60f7e7ab11293b0a0ada95c", null],
 ["operator=", "class_go_1_1_scratch_vect.html#ac26468529663df92e39b7c17d9a3a67e", null],
 ["operator[]", "class_go_1_1_scratch_vect.html#ab36a541205e92d2a565b362dd79c423b", null],
 ["operator[]", "class_go_1_1_scratch_vect.html#a991556544538c31eb8510d2f33b2190f", null],
 ["push_back", "class_go_1_1_scratch_vect.html#a6635d6ad2fec63775b0aff5225967236", null],
 ["resize", "class_go_1_1_scratch_vect.html#aa531f2726f96d8a3d3daba65a583a17d", null],
 ["size", "class_go_1_1_scratch_vect.html#a06699229f96c54b0a507f58273d794d0", null]
]
```

Definition at line 1 of file `class_go_1_1_scratch_vect.js`.

## 30.381 doc/html/class\_go\_1\_1\_sf\_boundary\_condition.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_sf\\_boundary\\_condition](#)

#### 30.381.1 Variable Documentation

##### 30.381.1.1 var class\_go\_1\_1\_sf\_boundary\_condition

#### Initial value:

```
=
[
 ["SfBoundaryCondition", "class_go_1_1_sf_boundary_condition.html#alc041de3bfffec3d96cd7f9d040e599fe",
 null],
 ["SfBoundaryCondition", "class_go_1_1_sf_boundary_condition.html#aec91fcel684b4a81dfdfa7846c9cabe6f",
 null],
 ["~SfBoundaryCondition", "class_go_1_1_sf_boundary_condition.html#a268d6191f998901f5dd99602bc689e9b",
 null],
 ["edgeNumber", "class_go_1_1_sf_boundary_condition.html#a4a6db47690ed1a613d229433524e61fb", null],
 ["getBasisFunctions", "class_go_1_1_sf_boundary_condition.html#ae4102fc1e7a23a7734b8b7e9289cb8b1",
 null],
 ["getBdCoefficients", "class_go_1_1_sf_boundary_condition.html#a2779e540b7012c00a91f93dcb11a12c5",
 null],
 ["getBdCoefficients", "class_go_1_1_sf_boundary_condition.html#a5d9fb20a43e34fdf923da0bed7d70fcf",
 null],
 ["getCoefficientsEnumeration", "
 class_go_1_1_sf_boundary_condition.html#a9464332ed2e576b13115a2898ccf5751", null],
 ["getCoefficientsEnumeration", "
 class_go_1_1_sf_boundary_condition.html#a498aff2c5e5fa0c3fbe664e7fcd48abf", null],
 ["getSplineApproximation", "class_go_1_1_sf_boundary_condition.html#aafdfa9becaee39d468649bd4a62a8cd0",
 null],
 ["getTolerances", "class_go_1_1_sf_boundary_condition.html#afb48a2afb30f74ba9e1721897f789a29", null],
 ["update", "class_go_1_1_sf_boundary_condition.html#a2e1c4b265662294ed2c345e93d40e48d", null],
 ["updateBoundaryValue", "class_go_1_1_sf_boundary_condition.html#adaf032fe7da6a50fdf3fb9b837a8e7b5",
 null]
]
```

Definition at line 1 of file class\_go\_1\_1\_sf\_boundary\_condition.js.

## 30.382 doc/html/class\_go\_1\_1\_sf\_cv\_intersector.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_sf\\_cv\\_intersector](#)

#### 30.382.1 Variable Documentation

##### 30.382.1.1 var class\_go\_1\_1\_sf\_cv\_intersector

#### Initial value:

```

=
[
 ["SfCvIntersector", "class_go_1_1_sf_cv_intersector.html#a7cbd7559c549dcf97a3f558368dc4156", null],
 ["SfCvIntersector", "class_go_1_1_sf_cv_intersector.html#aa4d5f1d7975a52e38eab1a7e964e281b", null],
 ["~SfCvIntersector", "class_go_1_1_sf_cv_intersector.html#a4d1e54a1a2664035b7930cebb0730338", null],
 ["checkCoincidence", "class_go_1_1_sf_cv_intersector.html#ae6a99e5274dlbba3886c286b3e6e42f0", null],
 ["doSubdivide", "class_go_1_1_sf_cv_intersector.html#adb8c5a7d56bf130367ec54dcb6d17638", null],
 ["foundIntersectionNearBoundary", "
 class_go_1_1_sf_cv_intersector.html#a94902bbebc892d94038b3b26ac721048", null],
 ["linearCase", "class_go_1_1_sf_cv_intersector.html#a23a2ed2ada6f80a6ad6b98c21656cddb", null],
 ["lowerOrderIntersector", "class_go_1_1_sf_cv_intersector.html#aa41c798641134369a99cb72d03be514b",
 null],
 ["microCase", "class_go_1_1_sf_cv_intersector.html#a2150d138f6c474f492a2c64ca70d95f6", null],
 ["numParams", "class_go_1_1_sf_cv_intersector.html#aac37e15b03fb3fac620c8e7cba8a4109", null],
 ["performRotatedBoxTest", "class_go_1_1_sf_cv_intersector.html#a6bc43c7a6e5ec19e935a1495384cbf40",
 null],
 ["postIterate", "class_go_1_1_sf_cv_intersector.html#a16ea99f766b4a831ff889462de90f8a5", null],
 ["postIterateBd", "class_go_1_1_sf_cv_intersector.html#a00dlb0b929be7995656dc5ac93cab6dd", null],
 ["repairIntersections", "class_go_1_1_sf_cv_intersector.html#a920d1a5009fa813fbb1c232149cdfc4c", null
],
 ["simpleCase", "class_go_1_1_sf_cv_intersector.html#acb048f5e9313fdbb4925baa7e2f83ed2", null],
 ["simpleCase2", "class_go_1_1_sf_cv_intersector.html#a99af53068b40d2836e9711ce08afcc0c", null],
 ["updateIntersections", "class_go_1_1_sf_cv_intersector.html#a5319f2c616b9e9ef80928c10bba09850", null
]
]

```

Definition at line 1 of file `class_go_1_1_sf_cv_intersector.js`.

## 30.383 doc/html/class\_go\_1\_1\_sf\_point\_bd\_cond.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_sf\\_point\\_bd\\_cond](#)

### 30.383.1 Variable Documentation

#### 30.383.1.1 var class\_go\_1\_1\_sf\_point\_bd\_cond

#### Initial value:

```

=
[
 ["SfPointBdCond", "class_go_1_1_sf_point_bd_cond.html#a1c6300a8c267ef1e993605e5dfb7799e", null],
 ["~SfPointBdCond", "class_go_1_1_sf_point_bd_cond.html#ac62802ef7a9d6ba59c3b3d7da19a58fd", null],
 ["edgeNumber", "class_go_1_1_sf_point_bd_cond.html#a96f0401fc151e72922b8c54866da9375", null],
 ["getCoefficientsEnumeration", "class_go_1_1_sf_point_bd_cond.html#a8336434dd73f3b3d3c8d775c73140c8a",
 null],
 ["getConditionValue", "class_go_1_1_sf_point_bd_cond.html#ae00d934a9b19dd1947408c669e2bc657", null],
 ["getInterpolationFactors", "class_go_1_1_sf_point_bd_cond.html#ac6f9125824a4518fccdb2bf5d898898b",
 null],
 ["getParam", "class_go_1_1_sf_point_bd_cond.html#afe3046bb2d3451ae74a021fbf680207d", null]
]

```

Definition at line 1 of file `class_go_1_1_sf_point_bd_cond.js`.

## 30.384 doc/html/class\_go\_1\_1\_sf\_pt\_intersector.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_sf\\_pt\\_intersector](#)

### 30.384.1 Variable Documentation

#### 30.384.1.1 var class\_go\_1\_1\_sf\_pt\_intersector

##### Initial value:

```
=
[
 ["SfPtIntersector", "class_go_1_1_sf_pt_intersector.html#a6725e40907178685dbdcdf7ad42b0176", null],
 ["~SfPtIntersector", "class_go_1_1_sf_pt_intersector.html#a06f1038df8f0d3de9dfe394fcf4be8d8", null],
 ["checkCoincidence", "class_go_1_1_sf_pt_intersector.html#a2d28043817a9593aba93a145f9433cc2", null],
 ["doSubdivide", "class_go_1_1_sf_pt_intersector.html#a4d6f8b65b3c65a99003a857b3eec7fa0", null],
 ["linearCase", "class_go_1_1_sf_pt_intersector.html#a7293a5a1f82fa696a9792395delb5fb3", null],
 ["lowerOrderIntersector", "class_go_1_1_sf_pt_intersector.html#ab45e535c287a829b4e97b8b14a8a9312",
 null],
 ["microCase", "class_go_1_1_sf_pt_intersector.html#a3ffe21eebdb2fce42761a435b20b5aba", null],
 ["numParams", "class_go_1_1_sf_pt_intersector.html#afe092761f4dala0d27219d81a44ee0f4", null],
 ["performRotatedBoxTest", "class_go_1_1_sf_pt_intersector.html#a25c6c68449fd891dda8087055b85fd72",
 null],
 ["repairIntersections", "class_go_1_1_sf_pt_intersector.html#adc8345bfad76daef30b266e35084c32d", null
],
 ["updateIntersections", "class_go_1_1_sf_pt_intersector.html#a01bb670cd8e24286a9ed82efbc63a525", null
]
]
```

Definition at line 1 of file class\_go\_1\_1\_sf\_pt\_intersector.js.

## 30.385 doc/html/class\_go\_1\_1\_sf\_self\_intersector.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_sf\\_self\\_intersector](#)

### 30.385.1 Variable Documentation

#### 30.385.1.1 var class\_go\_1\_1\_sf\_self\_intersector

##### Initial value:

```
=
[
 ["SfSelfIntersector", "class_go_1_1_sf_self_intersector.html#a292603cfa3655e634ce2cdf8cada8bab", null
],
 ["SfSelfIntersector", "class_go_1_1_sf_self_intersector.html#af93f199faa8152d934c83dc4ebf108d5", null
],
 ["~SfSelfIntersector", "class_go_1_1_sf_self_intersector.html#a8c26fed06a4be0d1b4bc3615a58633bc", null
],
 ["addComplexDomain", "class_go_1_1_sf_self_intersector.html#a2607f572f485a4eb9fa9303d9a62e6da", null]
 ,
 ["checkCoincidence", "class_go_1_1_sf_self_intersector.html#ab26ef10d07b7fa0d67207872ca61e3ac", null]
 ,
 ["complexityReduced", "class_go_1_1_sf_self_intersector.html#a85f403cf97c1a8b8302fa747b25f71bf", null
],
 ["compute", "class_go_1_1_sf_self_intersector.html#a960f75b41c6b74c3bdc7f648e5c4b4fe", null],
 ["computeG1", "class_go_1_1_sf_self_intersector.html#a858ae1504bd971f811fb753b44f3d58b", null],
 ["doSubdivide", "class_go_1_1_sf_self_intersector.html#ac19c3db16d630fadde0dd9a2c5a163cf", null],
 ["getBoundaryIntersections", "class_go_1_1_sf_self_intersector.html#a8f85396926b50fce8576df5db562c9bd"
 , null],
 ["getNmbComplexDomain", "class_go_1_1_sf_self_intersector.html#acd33e130f0cd93696e1a1403921bbc88",
 null],
 ["getNonSelfintersecting", "class_go_1_1_sf_self_intersector.html#aa938d6e5d5a8dca7480f27d976a60358",
 null],
 ["handleComplexity", "class_go_1_1_sf_self_intersector.html#a10116a1fb0208e84f11096c6b620f4d3", null]
 ,
]
```

```
["isLinear", "class_go_1_1_sf_self_intersector.html#afb7774dd883c4ef72df3e90d88099a9f", null],
["isSelfIntersection", "class_go_1_1_sf_self_intersector.html#a0b8581f4298ef335817492d26b3d6bdb", null
],
["linearCase", "class_go_1_1_sf_self_intersector.html#a8837d1aabd305fde4e348cf69ea3ddb", null],
["microCase", "class_go_1_1_sf_self_intersector.html#a68c79e3acda0e303715607c82341f6f7", null],
["numParams", "class_go_1_1_sf_self_intersector.html#a473fd548a0e9b7e5fed70033fd048ba7", null],
["performInterception", "class_go_1_1_sf_self_intersector.html#ab5eb703e184e265bb0403a1cf388efe3",
null],
["print_objs", "class_go_1_1_sf_self_intersector.html#a1da8df452bde7d8b7c75f01b9dd5f371", null],
["printDebugInfo", "class_go_1_1_sf_self_intersector.html#a3d6fa6e97cd99b62b3a640584f78a38d", null],
["repairIntersections", "class_go_1_1_sf_self_intersector.html#a9ddbbaa701f5f07b6548cbe4ad9562f77",
null],
["setMaxRec", "class_go_1_1_sf_self_intersector.html#a380314647ff6707a6d4bc8b47a5731f8", null],
["simpleCase", "class_go_1_1_sf_self_intersector.html#a7728685ee6435b631a0a18f32614e51c", null],
["updateIntersections", "class_go_1_1_sf_self_intersector.html#ae57a60093ced3c944dd989cf295955e4",
null]
]
```

Definition at line 1 of file `class_go_1_1_sf_self_intersector.js`.

## 30.386 doc/html/class\_go\_1\_1\_sf\_sf\_intersector.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_sf\\_sf\\_intersector](#)

#### 30.386.1 Variable Documentation

##### 30.386.1.1 var class\_go\_1\_1\_sf\_sf\_intersector

Definition at line 1 of file `class_go_1_1_sf_sf_intersector.js`.

## 30.387 doc/html/class\_go\_1\_1\_sf\_solution.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_sf\\_solution](#)

#### 30.387.1 Variable Documentation

##### 30.387.1.1 var class\_go\_1\_1\_sf\_solution

Definition at line 1 of file `class_go_1_1_sf_solution.js`.

## 30.388 doc/html/class\_go\_1\_1\_singularity\_info.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_singularity\\_info](#)

### 30.388.1 Variable Documentation

#### 30.388.1.1 var class\_go\_1\_1\_singularity\_info

##### Initial value:

```
=
[
 ["SingularityInfo", "class_go_1_1_singularity_info.html#a9f59b8ecc0f46dcaa8f92f2dd9a496fb", null],
 ["SingularityInfo", "class_go_1_1_singularity_info.html#a5ac06c970d1fa621f89f621fd7ea9b8d", null],
 ["SingularityInfo", "class_go_1_1_singularity_info.html#a7186becaa454a4e63d10dfc09b656cae", null],
 ["addIterationCount", "class_go_1_1_singularity_info.html#a9d97fbed7caf7832d2052aa6890084af", null],
 ["addSimpleCount", "class_go_1_1_singularity_info.html#a00e6fa1617e6737f8339d0b2188603d2", null],
 ["addSingularPoint", "class_go_1_1_singularity_info.html#ac620460320aadd62d78d6ca148037cf3", null],
 ["cleanUp", "class_go_1_1_singularity_info.html#a9469003aba26ccecc140cbd686fbfeff", null],
 ["getHighPriSing", "class_go_1_1_singularity_info.html#a4ba9c54f65cfa62649356c94edc44053", null],
 ["getHighPriSing", "class_go_1_1_singularity_info.html#a78e8baed1532507502a8830e5685d28c", null],
 ["getNmbPoints", "class_go_1_1_singularity_info.html#ac96ed326e4d316ccb6fcedbd426a2b51", null],
 ["getParam", "class_go_1_1_singularity_info.html#a54fb1464f75d0015f25608050197a5d0", null],
 ["getPoint", "class_go_1_1_singularity_info.html#alc0dad3c017e250db65608b68180aefd", null],
 ["hasHighPriSing", "class_go_1_1_singularity_info.html#a65a274b95eeb3096cbc51e66bcc5e817", null],
 ["hasPoint", "class_go_1_1_singularity_info.html#a96c00aa4f624a8cbf211f8d89fb02edb", null],
 ["iterationDone", "class_go_1_1_singularity_info.html#acb6bbfc680a27d94c7f35e74179093ff", null],
 ["iterationSucceed", "class_go_1_1_singularity_info.html#aab4c59c2815c462f7cb638cb7daac11a", null],
 ["nmbSimple1", "class_go_1_1_singularity_info.html#a10bd21a92cdd61718650f03e77ceb8ad", null],
 ["nmbSimple2", "class_go_1_1_singularity_info.html#ad918021f37a1f0c960e816f55f443fa6", null],
 ["setHighPriSing", "class_go_1_1_singularity_info.html#a0f2381178f0bc334f9e75208e06cfe12", null],
 ["setHighPriSingType", "class_go_1_1_singularity_info.html#aclae9e7f6860f0d5b6d37bc2d07dc194", null],
 ["setSingularPoint", "class_go_1_1_singularity_info.html#ad57cff2b86dc536b5f6bf5d3d7fa9ce5", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_singularity\_info.js.

## 30.389 doc/html/class\_go\_1\_1\_smooth\_curve.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_smooth\\_curve](#)

### 30.389.1 Variable Documentation

#### 30.389.1.1 var class\_go\_1\_1\_smooth\_curve

##### Initial value:

```
=
[
 ["SmoothCurve", "class_go_1_1_smooth_curve.html#a997891fc729c75c15e7d51fb696f7e", null],
 ["~SmoothCurve", "class_go_1_1_smooth_curve.html#af462faecc2dec14fe83a50e8ecaa7f6", null],
 ["attach", "class_go_1_1_smooth_curve.html#af8827ed574dd9b6396c81c2686e7621f", null],
 ["equationSolve", "class_go_1_1_smooth_curve.html#a9ec9db6e387593e2a2c1357bb21a0d1f", null],
 ["setLeastSquares", "class_go_1_1_smooth_curve.html#a4315d10efe2bfff4647aaca76368ac360", null],
 ["setOptim", "class_go_1_1_smooth_curve.html#a6a0621a4371cc0a8204f793395029981", null],
 ["setPeriodicity", "class_go_1_1_smooth_curve.html#a9af91497da714d4c282f3fd662f7c623", null],
 ["setSideConstraints", "class_go_1_1_smooth_curve.html#a8712b606e347145b7e1b0d7aa23c9d13", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_smooth\_curve.js.

## 30.390 doc/html/class\_go\_1\_1\_smooth\_curve\_set.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_smooth\\_curve\\_set](#)

### 30.390.1 Variable Documentation

#### 30.390.1.1 var class\_go\_1\_1\_smooth\_curve\_set

##### Initial value:

```
=
[
 ["SmoothCurveSet", "class_go_1_1_smooth_curve_set.html#ab1c06166e4ff97daad32cc9bd5aaab4a", null],
 ["~SmoothCurveSet", "class_go_1_1_smooth_curve_set.html#aa800dae0c9d242c0b20297614cc51935", null],
 ["attach", "class_go_1_1_smooth_curve_set.html#a3e52c06a3d2731e5e74cec22b8b7a105", null],
 ["equationSolve", "class_go_1_1_smooth_curve_set.html#af0dc5d9c6876b8ad40a7e501856028cd", null],
 ["setApproxOrig", "class_go_1_1_smooth_curve_set.html#a3100515e453d2583013952882fc3f3ae", null],
 ["setApproxSideConstraints", "class_go_1_1_smooth_curve_set.html#a219153549040f251397b83070de38d1f",
 null],
 ["setCvSetConstraints", "class_go_1_1_smooth_curve_set.html#a80a2995a97446fab7b5688e12ed5a55", null]
 ,
 ["setInterpolationConditions", "class_go_1_1_smooth_curve_set.html#adcbb3725d37a4baca1e8aae285b62407",
 null],
 ["setLeastSquares", "class_go_1_1_smooth_curve_set.html#ac547146ba45d1708ccb203c71d3dc2b2", null],
 ["setOptimize", "class_go_1_1_smooth_curve_set.html#a87a858d3333ea8ce25bc78bb390cfcba", null],
 ["setOrthCond", "class_go_1_1_smooth_curve_set.html#a96c9878fc673ae367d5f6bc9b8276e85", null],
 ["setSideConstraints", "class_go_1_1_smooth_curve_set.html#a1a90097414632550addc5a8726264bcb", null]
]
```

Definition at line 1 of file `class_go_1_1_smooth_curve_set.js`.

## 30.391 doc/html/class\_go\_1\_1\_smooth\_surf.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_smooth\\_surf](#)

### 30.391.1 Variable Documentation

#### 30.391.1.1 var class\_go\_1\_1\_smooth\_surf

Definition at line 1 of file `class_go_1_1_smooth_surf.js`.

## 30.392 doc/html/class\_go\_1\_1\_smooth\_surf\_set.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_smooth\\_surf\\_set](#)



## 30.392.1 Variable Documentation

### 30.392.1.1 var class\_go\_1\_1\_smooth\_surf\_set

#### Initial value:

```
=
[
 ["SmoothSurfSet", "class_go_1_1_smooth_surf_set.html#adb08dff7bccd990edleb86a61e493d7c", null],
 ["SmoothSurfSet", "class_go_1_1_smooth_surf_set.html#a56a91451a7169ae7d482d77bd5c6f122", null],
 ["~SmoothSurfSet", "class_go_1_1_smooth_surf_set.html#a6dbdecfb93357dfffa803363e6df733d1", null],
 ["approxOrig", "class_go_1_1_smooth_surf_set.html#a0ec9042766eed55423b805d4c057f47a", null],
 ["attach", "class_go_1_1_smooth_surf_set.html#a8e2c19721cef02da423a6f73118db412", null],
 ["equationSolve", "class_go_1_1_smooth_surf_set.html#a1a6aee6aeb5d6dcc04fa8922b34aa611", null],
 ["getBasis", "class_go_1_1_smooth_surf_set.html#a2b762aa31a7566841e8833622f17bf46", null],
 ["setApproxSideConstraints", "class_go_1_1_smooth_surf_set.html#aa558e24ac46b2f9cef5c5eb495f4cf22",
 null],
 ["setLeastSquares", "class_go_1_1_smooth_surf_set.html#aec7cae351a4dcd6998f4b076d4847b75", null],
 ["setNormalCond", "class_go_1_1_smooth_surf_set.html#a57998100f559697262419a2d61aa724b", null],
 ["setOptimize", "class_go_1_1_smooth_surf_set.html#a21fd1b98cea633463871448d2c4bb6ce", null],
 ["setSideConstraints", "class_go_1_1_smooth_surf_set.html#ad7d25eed9ca1638678b7bd7436454d3", null],
 ["coef_array_", "class_go_1_1_smooth_surf_set.html#a502c7f2a3b6baaf699faa76628b78fd4", null],
 ["coef_known_", "class_go_1_1_smooth_surf_set.html#a513e1c358f983184c48594b199214431", null],
 ["copy_coefs_", "class_go_1_1_smooth_surf_set.html#a0a8877e05ec6b51709dbd3a1d1089aee", null],
 ["gmat_", "class_go_1_1_smooth_surf_set.html#ad1e20276ff25cfde6a017b252adf9a0d", null],
 ["gright_", "class_go_1_1_smooth_surf_set.html#abb8d17ab393b7f693037d20085d694cd", null],
 ["ider_", "class_go_1_1_smooth_surf_set.html#a9cd4f276e32e6b8a8af0f717f1d73ecc", null],
 ["idim1_", "class_go_1_1_smooth_surf_set.html#a6bc8ed2b7ed1be18453f2a33078ec191", null],
 ["idim_", "class_go_1_1_smooth_surf_set.html#ad0418ef35691b72afb35924f95b211cc", null],
 ["kdim_", "class_go_1_1_smooth_surf_set.html#a16650b6975a4aaf03bf49ccd70a973e1", null],
 ["kncond_", "class_go_1_1_smooth_surf_set.html#aa530cc21c18294d89efb3e986b5f1ad3", null],
 ["knconstraint_", "class_go_1_1_smooth_surf_set.html#ab4b15fa0722035377b4651a00d147e5f", null],
 ["pivot_", "class_go_1_1_smooth_surf_set.html#a6bc1e1268bfde5066fb14ad898ecfa5", null],
 ["srfs_", "class_go_1_1_smooth_surf_set.html#a8f87da302f483b02568e3eda83659f96", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_smooth\_surf\_set.js.

## 30.393 doc/html/class\_go\_1\_1\_smooth\_transition.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_smooth\\_transition](#)

## 30.393.1 Variable Documentation

### 30.393.1.1 var class\_go\_1\_1\_smooth\_transition

#### Initial value:

```
=
[
 ["SmoothTransition", "class_go_1_1_smooth_transition.html#ac9d86a12ebaf5d3ca69cdc0c6597e40b", null],
 ["~SmoothTransition", "class_go_1_1_smooth_transition.html#a05192f1a013464642634c41f237b6ab1", null],
 ["approximationOK", "class_go_1_1_smooth_transition.html#a2da0b7a28bbbaae2d1718bfbf9ea1b054", null],
 ["dim", "class_go_1_1_smooth_transition.html#a8340dbd93a462b989b31cb09d7ab1287", null],
 ["end", "class_go_1_1_smooth_transition.html#ac5f27b776ad38a26b224ec780b59f00d", null],
 ["eval", "class_go_1_1_smooth_transition.html#a9dec775aeea43960e2f31389536c020d", null],
 ["eval", "class_go_1_1_smooth_transition.html#a3014fd12933c038db7eb267fbed90b50", null],
 ["nmbCvs", "class_go_1_1_smooth_transition.html#a1f4a21968fb3891636dfbce556dd0889", null],
 ["start", "class_go_1_1_smooth_transition.html#aaeaa6284f240bf81a519ce02a117dc10", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_smooth\_transition.js.

## 30.394 doc/html/class\_go\_1\_1\_smooth\_volume.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_smooth\\_volume](#)

### 30.394.1 Variable Documentation

#### 30.394.1.1 var class\_go\_1\_1\_smooth\_volume

#### Initial value:

```
=
[
 ["SmoothVolume", "class_go_1_1_smooth_volume.html#ab36c61f3ee1035f33e73f33950fd901c", null],
 ["SmoothVolume", "class_go_1_1_smooth_volume.html#a323d38d3a85eb6ba7dbf4f6f5162c7e0", null],
 ["~SmoothVolume", "class_go_1_1_smooth_volume.html#a4fd31d89cef318c5e2ab15ca9f7f361d", null],
 ["attach", "class_go_1_1_smooth_volume.html#ae184d6e53a42c286ec7c8e657decbe71", null],
 ["equationSolve", "class_go_1_1_smooth_volume.html#ac673b1170a398dae4375eeee9a79b0e9", null],
 ["reset", "class_go_1_1_smooth_volume.html#ad4d7ccb13cee80f696a0e13b2f6e71e0", null],
 ["setLeastSquares", "class_go_1_1_smooth_volume.html#a23f0583c789f13276efcea7fe4ba29ff", null],
 ["setOptimize", "class_go_1_1_smooth_volume.html#a46f42b0dc67d40329c991e2e77dbbc5c", null],
 ["setPeriodicity", "class_go_1_1_smooth_volume.html#a03b396alf59e001d04d0ed05dc26b2fc", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_smooth\_volume.js.

## 30.395 doc/html/class\_go\_1\_1\_solve\_b\_c\_g.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_solve\\_b\\_c\\_g](#)

### 30.395.1 Variable Documentation

#### 30.395.1.1 var class\_go\_1\_1\_solve\_b\_c\_g

#### Initial value:

```
=
[
 ["SolveBCG", "class_go_1_1_solve_b_c_g.html#a95775031e0ba5483999cb9232ffdc26", null],
 ["~SolveBCG", "class_go_1_1_solve_b_c_g.html#a6a286f7629218dcafd7405d0550a6d5f", null],
 ["precond", "class_go_1_1_solve_b_c_g.html#aab460752091a97b5e71ee1b1f347a17f", null],
 ["solve", "class_go_1_1_solve_b_c_g.html#afd42fc309a7bc7a560d2d8a81b7122e3", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_solve\_b\_c\_g.js.

## 30.396 doc/html/class\_go\_1\_1\_solve\_c\_g.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_solve\\_c\\_g](#)

### 30.396.1 Variable Documentation

#### 30.396.1.1 var class\_go\_1\_1\_solve\_c\_g

##### Initial value:

```
=
[
 ["SolveCG", "class_go_1_1_solve_c_g.html#afb6a00ea7eaba2769d150cd6a2cee41", null],
 ["~SolveCG", "class_go_1_1_solve_c_g.html#a252ba2c8835b8bb120108e0aled9dcbe", null],
 ["attachMatrix", "class_go_1_1_solve_c_g.html#a38cc2291f0077b8a4450256c4a2583fb", null],
 ["forwBack", "class_go_1_1_solve_c_g.html#ab6a1730362dbd3d7fe17e0be0222876b", null],
 ["getIndex", "class_go_1_1_solve_c_g.html#a9407314f95f0213a2f2d83a743197789", null],
 ["matrixProduct", "class_go_1_1_solve_c_g.html#a5f9f78040e15ebb974d1f5634589e54d", null],
 ["precondRILU", "class_go_1_1_solve_c_g.html#a219af16e5a129711e165e45151db4947", null],
 ["printPrecond", "class_go_1_1_solve_c_g.html#ada542105c2a69c44a542729aa5b3b64d", null],
 ["setMaxIterations", "class_go_1_1_solve_c_g.html#ac713f643a8bf5856af40e6779e8e7a85", null],
 ["setTolerance", "class_go_1_1_solve_c_g.html#a9e5aff10de6c1125c4de120aecf6ea3b", null],
 ["solve", "class_go_1_1_solve_c_g.html#a7d29fb9ae4fbae0c9ae50d8aed46ea0b", null],
 ["solveRILU", "class_go_1_1_solve_c_g.html#ace35787c89741236cef838423d3c022c", null],
 ["solveStd", "class_go_1_1_solve_c_g.html#a36f1fe62558ae9b76ca764caf28a75b5", null],
 ["transposedMatrixProduct", "class_go_1_1_solve_c_g.html#a5eed0d88e65581da9f97632998084e97", null],
 ["A", "class_go_1_1_solve_c_g.html#a3de1a62612cb883f2f7af53f13057fd7", null],
 ["diagonal_", "class_go_1_1_solve_c_g.html#ac8ae0582fc46b18fcbba1257abdc3f37", null],
 ["diagset_", "class_go_1_1_solve_c_g.html#afbd7021a161b7500c76bd32d8dd8e7ec", null],
 ["irow_", "class_go_1_1_solve_c_g.html#a8331b9396a655ff1ffcc5770b48665eb", null],
 ["jcol_", "class_go_1_1_solve_c_g.html#a5718e8d392c16f9adfc2b1c2147b8c8f", null],
 ["M_", "class_go_1_1_solve_c_g.html#ac8ef49ab38d13d40f19874a80534d645", null],
 ["max_iterations_", "class_go_1_1_solve_c_g.html#a8d84c3337b466ea8a2935b3a63fa95f7", null],
 ["nn_", "class_go_1_1_solve_c_g.html#af771970e012471d59d9158e0138cc943", null],
 ["np_", "class_go_1_1_solve_c_g.html#af30c8e220d9e4c7285e446d71161e6ff", null],
 ["omega_", "class_go_1_1_solve_c_g.html#a895bee6e49c4b8bfefdde37741693433", null],
 ["tolerance_", "class_go_1_1_solve_c_g.html#a285f20e1f80b1c792914b35ed926e464", null]
]
```

Definition at line 1 of file [class\\_go\\_1\\_1\\_solve\\_c\\_g.js](#).

## 30.397 doc/html/class\_go\_1\_1\_solve\_c\_g\_c\_o.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_solve\\_c\\_g\\_c\\_o](#)

### 30.397.1 Variable Documentation

#### 30.397.1.1 var class\_go\_1\_1\_solve\_c\_g\_c\_o

##### Initial value:

```
=
[
 ["SolveCGCO", "class_go_1_1_solve_c_g_c_o.html#a2a2fb754327fd23316c826fb7cd9bce0", null],
 ["~SolveCGCO", "class_go_1_1_solve_c_g_c_o.html#a1200ae5180d1bb549d3a4dc46108e79e", null],
 ["precondRILU", "class_go_1_1_solve_c_g_c_o.html#a11595655ce535ab77b037c70370e07e0", null]
]
```

Definition at line 1 of file [class\\_go\\_1\\_1\\_solve\\_c\\_g\\_c\\_o.js](#).

## 30.398 doc/html/class\_go\_1\_1\_space\_int\_crv.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_space\\_int\\_crv](#)

### 30.398.1 Variable Documentation

#### 30.398.1.1 var class\_go\_1\_1\_space\_int\_crv

#### Initial value:

```
=
[
 ["SpaceIntCrv", "class_go_1_1_space_int_crv.html#aac5ae41720a39f76fae66def168e0f34", null],
 ["~SpaceIntCrv", "class_go_1_1_space_int_crv.html#a3b1f2c843cbb4819924654a30acd953b", null],
 ["approximationOK", "class_go_1_1_space_int_crv.html#a5378ef678a717768958de4c974d200f0", null],
 ["dim", "class_go_1_1_space_int_crv.html#ad0f940e4e1ef71cc46cb8f575dd29173", null],
 ["end", "class_go_1_1_space_int_crv.html#afd98ec991b60b68feb1ec9f79ba30dcc", null],
 ["eval", "class_go_1_1_space_int_crv.html#ab9a922c9e907889610ba88f507340ed3", null],
 ["eval", "class_go_1_1_space_int_crv.html#a52581fc9d01325d625eff7e90fd5ad23", null],
 ["start", "class_go_1_1_space_int_crv.html#a76ac6fd8de4800f53e2e29670e56737f", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_space\_int\_crv.js.

## 30.399 doc/html/class\_go\_1\_1\_sphere.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_sphere](#)

### 30.399.1 Variable Documentation

#### 30.399.1.1 var class\_go\_1\_1\_sphere

Definition at line 1 of file class\_go\_1\_1\_sphere.js.

## 30.400 doc/html/class\_go\_1\_1\_sphere\_int.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_sphere\\_int](#)

### 30.400.1 Variable Documentation

#### 30.400.1.1 var class\_go\_1\_1\_sphere\_int

##### Initial value:

```
=
[
 ["SphereInt", "class_go_1_1_sphere_int.html#ae7cc09fbdbe552545e302b764aa89c28", null],
 ["SphereInt", "class_go_1_1_sphere_int.html#ad0240982e2c5359a8cdaaeeb6148e874", null],
 ["~SphereInt", "class_go_1_1_sphere_int.html#a482fe2bb316a6f3acc0b5aaa58e00c28", null],
 ["center", "class_go_1_1_sphere_int.html#aa9bd1fdc153147b5811610bffe8af4b", null],
 ["radius", "class_go_1_1_sphere_int.html#abb8450f4ca8990149d73bleea2f3015d", null],
 ["read", "class_go_1_1_sphere_int.html#a0bfbff9d7fd3639303e1deb0fda46d57", null],
 ["surface", "class_go_1_1_sphere_int.html#a95e1198a35b5a37f2859f7b57b8561eb", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_sphere\_int.js.

## 30.401 doc/html/class\_go\_1\_1\_sphere\_volume.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_sphere\\_volume](#)

### 30.401.1 Variable Documentation

#### 30.401.1.1 var class\_go\_1\_1\_sphere\_volume

##### Initial value:

```
=
[
 ["SphereVolume", "class_go_1_1_sphere_volume.html#a9dc16295cadf204488037a6150eff3d9", null],
 ["SphereVolume", "class_go_1_1_sphere_volume.html#ab90843ec3286e9bcb9ddc7370ecf8d22", null],
 ["~SphereVolume", "class_go_1_1_sphere_volume.html#a640957dd5250d634ab62b26ffab9f215", null],
 ["boundingBox", "class_go_1_1_sphere_volume.html#a270f903cd3b8e089d3a675fd6f49a13f", null],
 ["clone", "class_go_1_1_sphere_volume.html#ad45ef3f270b63649b5845db77ef359c2", null],
 ["closestPoint", "class_go_1_1_sphere_volume.html#ab74c0aeaa8fc5058b027ae5988546b4f", null],
 ["dimension", "class_go_1_1_sphere_volume.html#acc766755a92f0aa8bc9775391fd96236", null],
 ["geometryVolume", "class_go_1_1_sphere_volume.html#a0b1bdd00aab5d0aca43f23703c421c2f", null],
 ["getAllBoundarySurfaces", "class_go_1_1_sphere_volume.html#a892d1585351de32b9821b007f430adb1", null]
],
[
 ["instanceType", "class_go_1_1_sphere_volume.html#a4aa5be5278afe92f64363a2f4f5ba06a", null],
 ["nextSegmentVal", "class_go_1_1_sphere_volume.html#aef2daa2809a336204eb23e0fa7eec994", null],
 ["parameterSpan", "class_go_1_1_sphere_volume.html#a55310ecd2f752023b347cccd32490240", null],
 ["point", "class_go_1_1_sphere_volume.html#a80097f92d21d95203b08963d77e19ddf", null],
 ["point", "class_go_1_1_sphere_volume.html#a386ad949073f0d00007d37a8a207aa14", null],
 ["read", "class_go_1_1_sphere_volume.html#a5c808dfd3cdab6319034a20cdc0b286b", null],
 ["reverseParameterDirection", "class_go_1_1_sphere_volume.html#a9370080898f867930b91f1f7d2be420e", null],
 ["swapParameterDirection", "class_go_1_1_sphere_volume.html#adc04ac96070f95360e8fc43f98491562", null]
],
[
 ["tangentCone", "class_go_1_1_sphere_volume.html#a6f5a787cfa1abfd98de924e4dc36f262", null],
 ["translate", "class_go_1_1_sphere_volume.html#a6d0fe28f10cb283757ba6fda1e29e6d6", null],
 ["write", "class_go_1_1_sphere_volume.html#af39ffdf622b868ef6cc6d7915b28bade", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_sphere\_volume.js.

## 30.402 doc/html/class\_go\_1\_1\_spline1\_function\_int.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_spline1\\_function\\_int](#)

### 30.402.1 Variable Documentation

#### 30.402.1.1 var class\_go\_1\_1\_spline1\_function\_int

#### Initial value:

```
=
[
 ["Spline1FunctionInt", "class_go_1_1_spline1_function_int.html#a2c6d8568c778780eaa59b52c993984eb",
 null],
 ["Spline1FunctionInt", "class_go_1_1_spline1_function_int.html#a5fed7ec6319a7066360ef881c20d2ced",
 null],
 ["~Spline1FunctionInt", "class_go_1_1_spline1_function_int.html#a39a1398bcb5c34569b2a1f6eaf9c9873",
 null],
 ["getCriticalValsAndKnots", "class_go_1_1_spline1_function_int.html#af1c3bd9b79e881b04bd750edfb63cff4",
 null],
 ["getInnerKnotVals", "class_go_1_1_spline1_function_int.html#ab3b8861e222f919911bb8cb4ca63dce5", null
],
 ["getMesh", "class_go_1_1_spline1_function_int.html#afabb39444f6cf87ab6b8f99a7148f7e4", null],
 ["getMeshSize", "class_go_1_1_spline1_function_int.html#a5bedbf83800128fa9b3bdcaecf4868c3", null],
 ["hasCriticalValsOrKnots", "class_go_1_1_spline1_function_int.html#aa9c57b16305da4da62cda9719346defc",
 null],
 ["hasInnerKnots", "class_go_1_1_spline1_function_int.html#a4d9497d0e8cf61630cdf06ee5cb5c81f", null],
 ["knotIntervalFuzzy", "class_go_1_1_spline1_function_int.html#ae54cff5b82d26a4171300d8fbb3de2d8", null
],
 ["makeIntFunction", "class_go_1_1_spline1_function_int.html#af16f6194cdf0ce72d9ae949035610c99", null]
 ,
 ["monotone", "class_go_1_1_spline1_function_int.html#ad050f3ae8c56910a289edb8803a2f67b", null],
 ["nextSegmentVal", "class_go_1_1_spline1_function_int.html#a3b8fba9596b7a15cbcc111842a8f18eec", null],
 ["paramFromMesh", "class_go_1_1_spline1_function_int.html#ae4fdec921bf54d84215f37f429a4cf8e", null],
 ["spcv_", "class_go_1_1_spline1_function_int.html#ad6c174a54dea53f965539d968cc4bf41", null]
]
```

Definition at line 1 of file `class_go_1_1_spline1_function_int.js`.

## 30.403 doc/html/class\_go\_1\_1\_spline2\_function\_int.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_spline2\\_function\\_int](#)

### 30.403.1 Variable Documentation

#### 30.403.1.1 var class\_go\_1\_1\_spline2\_function\_int

#### Initial value:

```
=
[
 ["Spline2FunctionInt", "class_go_1_1_spline2_function_int.html#a4d564c97b16b1d64d7559c034be3669e",
 null],
 ["Spline2FunctionInt", "class_go_1_1_spline2_function_int.html#aba434cf4df79a3bc591e92bef0f7018b",
 null],
 ["~Spline2FunctionInt", "class_go_1_1_spline2_function_int.html#a4bba6d6032510758566a41f8f18a36d5",
 null],
 ["createGradSurface", "class_go_1_1_spline2_function_int.html#adb0cc518cd429dc5cdd18087a2ae3ce6", null
],
 ["endParam", "class_go_1_1_spline2_function_int.html#a5c49dc316446c1522954af89cc16cdc8", null],
 ["getBoundaryObjects", "class_go_1_1_spline2_function_int.html#a246f53dce46d0fe36de82587cbcf82b",
 null],
 ["getCriticalValsAndKnots", "class_go_1_1_spline2_function_int.html#a7a82482309720707001c78e86dc6c18b",
 null],
 ["getInnerKnotVals", "class_go_1_1_spline2_function_int.html#a418ef2e18dc5f1564alcfd136b22831", null
],
 ["hasCriticalValsOrKnots", "class_go_1_1_spline2_function_int.html#af76103f813d05c0f6a8fddfbbab62f8ec",
 null],
 ["hasInnerKnots", "class_go_1_1_spline2_function_int.html#a7d1177a5519455fdbb360963715b58e9", null],
 ["knotIntervalFuzzy", "class_go_1_1_spline2_function_int.html#a7229f63cd797e93f4aed7b4f3f8650ba", null
],
 ["makeIntFunction", "class_go_1_1_spline2_function_int.html#ac3c433d64c98fa115ab59934fd35d886", null
],
 ["monotone", "class_go_1_1_spline2_function_int.html#a97f28f20e0f89e748ec2141838eccfle", null],
 ["nextSegmentVal", "class_go_1_1_spline2_function_int.html#ab5e6f71b156eea6e5b04a5d54e58daec", null],
 ["startParam", "class_go_1_1_spline2_function_int.html#aff91e09e7bf96b3c430071483dc14a72", null],
 ["surface3D", "class_go_1_1_spline2_function_int.html#a4996265954c03b57d1ba94bec7f75f95", null],
 ["spsf_", "class_go_1_1_spline2_function_int.html#a495b8863489c0c6cf28eb861263f8281", null]
]

```

Definition at line 1 of file class\_go\_1\_1\_spline2\_function\_int.js.

## 30.404 doc/html/class\_go\_1\_1\_spline\_approximator.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_spline\\_approximator](#)

### 30.404.1 Variable Documentation

#### 30.404.1.1 var class\_go\_1\_1\_spline\_approximator

##### Initial value:

```
=
[
 ["SplineApproximator", "class_go_1_1_spline_approximator.html#a8b17f58e863857c7b4e6f9d754558fc4", null
],
 ["~SplineApproximator", "class_go_1_1_spline_approximator.html#a997b02a8ab659082a8f1a9081a6a0536",
 null],
 ["basis", "class_go_1_1_spline_approximator.html#a13933317f34367974941c838d3d4ed5c", null],
 ["interpolate", "class_go_1_1_spline_approximator.html#a7de87b26c84b508582a797f534dc3a7b", null],
 ["setNumCoefs", "class_go_1_1_spline_approximator.html#afla5920f8f77fa41473903e6841413c6", null],
 ["setSplineSpace", "class_go_1_1_spline_approximator.html#a5647a74af450dc2f60719334d73ae2ba", null]
]

```

Definition at line 1 of file class\_go\_1\_1\_spline\_approximator.js.

## 30.405 doc/html/class\_go\_1\_1\_spline\_curve.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_spline\\_curve](#)

### 30.405.1 Variable Documentation

#### 30.405.1.1 var class\_go\_1\_1\_spline\_curve

Definition at line 1 of file class\_go\_1\_1\_spline\_curve.js.

## 30.406 doc/html/class\_go\_1\_1\_spline\_curve\_int.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_spline\\_curve\\_int](#)

### 30.406.1 Variable Documentation

#### 30.406.1.1 var class\_go\_1\_1\_spline\_curve\_int

#### Initial value:

```
=
[
 ["SplineCurveInt", "class_go_1_1_spline_curve_int.html#ad51a929635f99ec6829f5436780635bd", null],
 ["SplineCurveInt", "class_go_1_1_spline_curve_int.html#aa8087368a9c9584162e8379f0aba222b", null],
 ["~SplineCurveInt", "class_go_1_1_spline_curve_int.html#a49f3bad30d5918d43efc581daf5547cf", null],
 ["checkPeriodicity", "class_go_1_1_spline_curve_int.html#a0676afa57028113c50e7b468d3171a8e", null],
 ["getCriticalValsAndKnots", "class_go_1_1_spline_curve_int.html#a1046378d116335428f89c43757924720",
 null],
 ["getInnerKnotVals", "class_go_1_1_spline_curve_int.html#a1a3de5f76a1802397d9fc49324bd29c2", null],
 ["getMesh", "class_go_1_1_spline_curve_int.html#a18b45016c1317321050d9405b5062449", null],
 ["getMeshSize", "class_go_1_1_spline_curve_int.html#a67760ffce1c4ab5a496615f86793d57f", null],
 ["getOptimizedConeAngle", "class_go_1_1_spline_curve_int.html#af39040af24e52ba2220c8f84d4c7623e", null
],
 ["getRotatedBox", "class_go_1_1_spline_curve_int.html#a67f76c52a91887640ad572bbcb99ec31", null],
 ["getSpline", "class_go_1_1_spline_curve_int.html#ad5054a07cc39131012f96516854a6c0d", null],
 ["hasCriticalValsOrKnots", "class_go_1_1_spline_curve_int.html#ad6a95462ef0764b34de570d3f96aeb06",
 null],
 ["hasInnerKnots", "class_go_1_1_spline_curve_int.html#af0f24e3fb1ed7abba5522a130a268ff1", null],
 ["isSpline", "class_go_1_1_spline_curve_int.html#a4aaefbb06506ac85d3433657f4c0d788", null],
 ["knotIntervalFuzzy", "class_go_1_1_spline_curve_int.html#aaacc36c235ff72d3989818aa3d99b024", null],
 ["makeIntObject", "class_go_1_1_spline_curve_int.html#ac9f02f98f6700b360043e574b3330827", null],
 ["nextSegmentVal", "class_go_1_1_spline_curve_int.html#a246edc5a71c414dd042c412aaf11cca8", null],
 ["paramFromMesh", "class_go_1_1_spline_curve_int.html#ac20cafe73fc3cc741579753e4fa47748", null],
 ["spcv_", "class_go_1_1_spline_curve_int.html#a2b22620d48228bf1a41565f0220c9584", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_spline\_curve\_int.js.

## 30.407 doc/html/class\_go\_1\_1\_spline\_interpolator.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_spline\\_interpolator](#)



## 30.407.1 Variable Documentation

### 30.407.1.1 var class\_go\_1\_1\_spline\_interpolator

#### Initial value:

```
=
[
 ["CondType", "class_go_1_1_spline_interpolator.html#aacf4f80d98ea6b554729125fafd24a78", [
 ["None", "
class_go_1_1_spline_interpolator.html#aacf4f80d98ea6b554729125fafd24a78a02bef9ce6e415b73a3cd9fe89aace722", null],
 ["Hermite", "
class_go_1_1_spline_interpolator.html#aacf4f80d98ea6b554729125fafd24a78a51b46c3d3f007d4d2ef458f51d85610c", null],
 ["Natural", "
class_go_1_1_spline_interpolator.html#aacf4f80d98ea6b554729125fafd24a78a97ae16f658015873c02850158cd655fd", null],
 ["Free", "
class_go_1_1_spline_interpolator.html#aacf4f80d98ea6b554729125fafd24a78abfde7d7c13ab6f25fd3025e73385e147", null],
 ["NaturalAtStart", "
class_go_1_1_spline_interpolator.html#aacf4f80d98ea6b554729125fafd24a78a5b6b1f58b2d5ddefab7785b4e637e213", null],
 ["NaturalAtEnd", "
class_go_1_1_spline_interpolator.html#aacf4f80d98ea6b554729125fafd24a78a94a08e9e644774498ad7e091d8535415", null]
]],
 ["SplineInterpolator", "class_go_1_1_spline_interpolator.html#af479204f61f6910c9a1d6497c7963280", null
],
 ["~SplineInterpolator", "class_go_1_1_spline_interpolator.html#a35a552cf7bd9b6d2cb25cb96c61bc90d",
 null],
 ["basis", "class_go_1_1_spline_interpolator.html#a440b6c23ced112a1df18c75287379ba1", null],
 ["getCondType", "class_go_1_1_spline_interpolator.html#a0d18b08c5d7b321c274f11ec29428f18", null],
 ["interpolate", "class_go_1_1_spline_interpolator.html#a719a56b88881dcaeeeeeec2c177b9526", null],
 ["interpolate", "class_go_1_1_spline_interpolator.html#ac3c845e4babcd0b547bbb129b1fb4f58", null],
 ["interpolate", "class_go_1_1_spline_interpolator.html#a2cbea93b5c2bbdb028d96aba219bc6f9", null],
 ["makeBasis", "class_go_1_1_spline_interpolator.html#abd5d723fd648cbcb546a7a40930e98c1", null],
 ["setBasis", "class_go_1_1_spline_interpolator.html#a0a6bbc78efd2a6d0976cc7395dafaef0", null],
 ["setEndTangents", "class_go_1_1_spline_interpolator.html#a53357524e94a651351d5b35684595452", null],
 ["setFreeConditions", "class_go_1_1_spline_interpolator.html#a38f8655b7996cfc6326e788cc5991b43", null
],
 ["setHermiteConditions", "class_go_1_1_spline_interpolator.html#a7963acba60b6c8d09b2423af655191d1",
 null],
 ["setNaturalConditions", "class_go_1_1_spline_interpolator.html#a2b0f0350ea2ddc78d099b095a01551ba",
 null],
 ["setNaturalEndCondition", "class_go_1_1_spline_interpolator.html#a79393970684c01eb300c005ee5b831ac",
 null],
 ["setNaturalStartCondition", "class_go_1_1_spline_interpolator.html#ada472681b18c81f58888d7abf68439e0",
 null]
]
]
```

Definition at line 1 of file class\_go\_1\_1\_spline\_interpolator.js.

## 30.408 doc/html/class\_go\_1\_1\_spline\_surface.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_spline\\_surface](#)

## 30.408.1 Variable Documentation

### 30.408.1.1 var class\_go\_1\_1\_spline\_surface

Definition at line 1 of file class\_go\_1\_1\_spline\_surface.js.

## 30.409 doc/html/class\_go\_1\_1\_spline\_surface\_int.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_spline\\_surface\\_int](#)

#### 30.409.1 Variable Documentation

##### 30.409.1.1 var class\_go\_1\_1\_spline\_surface\_int

Definition at line 1 of file class\_go\_1\_1\_spline\_surface\_int.js.

## 30.410 doc/html/class\_go\_1\_1\_spline\_volume.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_spline\\_volume](#)

#### 30.410.1 Variable Documentation

##### 30.410.1.1 var class\_go\_1\_1\_spline\_volume

Definition at line 1 of file class\_go\_1\_1\_spline\_volume.js.

## 30.411 doc/html/class\_go\_1\_1\_streamable.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_streamable](#)

#### 30.411.1 Variable Documentation

##### 30.411.1.1 var class\_go\_1\_1\_streamable

#### Initial value:

```
=
[
 ["EOFException", "class_go_1_1_streamable_1_1_eof_exception.html", null],
 ["~Streamable", "class_go_1_1_streamable.html#aaf0c80b774e961da637d83049423e231", null],
 ["read", "class_go_1_1_streamable.html#a22b855f93460bfe8942090502a15fb01", null],
 ["write", "class_go_1_1_streamable.html#ae8180cc0ef0185251251e1e48714ad79", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_streamable.js.

## 30.412 doc/html/class\_go\_1\_1\_surface\_assembly.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_surface\\_assembly](#)

### 30.412.1 Variable Documentation

#### 30.412.1.1 var class\_go\_1\_1\_surface\_assembly

#### Initial value:

```
=
[
 ["SurfaceAssembly", "class_go_1_1_surface_assembly.html#afd932cf53c41869a0ca1c762b026eee1", null],
 ["~SurfaceAssembly", "class_go_1_1_surface_assembly.html#a529d17a8f3e4a81be3289c3044f39c99", null],
 ["doTouch", "class_go_1_1_surface_assembly.html#ac6b9d0ec9a3b08740309b87dc74f3375", null],
 ["getNextAssembly", "class_go_1_1_surface_assembly.html#a4782048bda2526dc64ee0195759748f4", null],
 ["getNextSubSurface", "class_go_1_1_surface_assembly.html#a70ea748bb136dc2d63fa745a599fc17b", null],
 ["getNmbSubSurface", "class_go_1_1_surface_assembly.html#aacf40d4c4a29105fd4086760ee09b963", null],
 ["getSubSurfaceIndex", "class_go_1_1_surface_assembly.html#a265282a66502d77dd1075d89bf652f29", null],
 ["getUdiv", "class_go_1_1_surface_assembly.html#af832fa5d003e154e0fc57005e8bfa5c8", null],
 ["getVdiv", "class_go_1_1_surface_assembly.html#a7cebaf387a7619b4388b6031b5ab3ee", null],
 ["isInFirstAssembly", "class_go_1_1_surface_assembly.html#ad6d88d0730cbedc343dd67bc23c0e804", null],
 ["isInPrevAssembly", "class_go_1_1_surface_assembly.html#a19025a8731f8318d667a8896b5616861", null],
 ["resetAssemblyIndex", "class_go_1_1_surface_assembly.html#a16f355a2cb9cd9f591212d6421efb953", null],
 ["resetSubIndex", "class_go_1_1_surface_assembly.html#a7828ecf6cc02b5685265fef693044d02", null],
 ["subSfNeighbour", "class_go_1_1_surface_assembly.html#aad350becfdfla86b37825e7a372057fd", null],
 ["touchAtSingularity", "class_go_1_1_surface_assembly.html#a7100fd21e0ab170f696bbee50e03892e", null]
]
```

Definition at line 1 of file `class_go_1_1_surface_assembly.js`.

## 30.413 doc/html/class\_go\_1\_1\_surface\_model.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_surface\\_model](#)

### 30.413.1 Variable Documentation

#### 30.413.1.1 var class\_go\_1\_1\_surface\_model

Definition at line 1 of file `class_go_1_1_surface_model.js`.

## 30.414 doc/html/class\_go\_1\_1\_surface\_of\_linear\_extrusion.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_surface\\_of\\_linear\\_extrusion](#)

### 30.414.1 Variable Documentation

#### 30.414.1.1 var class\_go\_1\_1\_surface\_of\_linear\_extrusion

Definition at line 1 of file class\_go\_1\_1\_surface\_of\_linear\_extrusion.js.

## 30.415 doc/html/class\_go\_1\_1\_surface\_of\_revolution.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_surface\\_of\\_revolution](#)

### 30.415.1 Variable Documentation

#### 30.415.1.1 var class\_go\_1\_1\_surface\_of\_revolution

Definition at line 1 of file class\_go\_1\_1\_surface\_of\_revolution.js.

## 30.416 doc/html/class\_go\_1\_1\_surface\_on\_volume.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_surface\\_on\\_volume](#)

### 30.416.1 Variable Documentation

#### 30.416.1.1 var class\_go\_1\_1\_surface\_on\_volume

Definition at line 1 of file class\_go\_1\_1\_surface\_on\_volume.js.

## 30.417 doc/html/class\_go\_1\_1\_sweep\_surface\_creator.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_sweep\\_surface\\_creator](#)

### 30.417.1 Variable Documentation

#### 30.417.1.1 var class\_go\_1\_1\_sweep\_surface\_creator

**Initial value:**

```
=
[
 ["~SweepSurfaceCreator", "class_go_1_1_sweep_surface_creator.html#a15a2b5747aa42c90f1efe1ddaf289c30",
 null]
]
```

Definition at line 1 of file class\_go\_1\_1\_sweep\_surface\_creator.js.

## 30.418 doc/html/class\_go\_1\_1\_sweep\_volume\_creator.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_sweep\\_volume\\_creator](#)

### 30.418.1 Variable Documentation

#### 30.418.1.1 var class\_go\_1\_1\_sweep\_volume\_creator

**Initial value:**

```
=
[
 ["~SweepVolumeCreator", "class_go_1_1_sweep_volume_creator.html#ab6401ce96389698269ab22c171b84075",
 null]
]
```

Definition at line 1 of file class\_go\_1\_1\_sweep\_volume\_creator.js.

## 30.419 doc/html/class\_go\_1\_1\_tesselator.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_tesselator](#)

### 30.419.1 Variable Documentation

#### 30.419.1.1 var class\_go\_1\_1\_tesselator

**Initial value:**

```
=
[
 ["~Tesselator", "class_go_1_1_tesselator.html#a2390ddf5ca3451622eb5dd23eda35b91", null],
 ["tesselate", "class_go_1_1_tesselator.html#ac990c28ea9a36e37b26edd30cc523ac3", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_tesselator.js.

## 30.420 doc/html/class\_go\_1\_1\_test\_in\_domain.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_test\\_in\\_domain](#)

### 30.420.1 Variable Documentation

#### 30.420.1.1 var class\_go\_1\_1\_test\_in\_domain

##### Initial value:

```
=
[
 ["argument_type", "class_go_1_1_test_in_domain.html#a5190b2f25aa4dd7d98dffa5ffc306291", null],
 ["result_type", "class_go_1_1_test_in_domain.html#ae3accb3874ee41024e38e02d09e63a90", null],
 ["TestInDomain", "class_go_1_1_test_in_domain.html#a0884e5a897b9217af52967dc2713a52c", null],
 ["operator()", "class_go_1_1_test_in_domain.html#a85187fdd79b35e3d79884b45cfabbc74", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_test\_in\_domain.js.

## 30.421 doc/html/class\_go\_1\_1\_torus.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_torus](#)

### 30.421.1 Variable Documentation

#### 30.421.1.1 var class\_go\_1\_1\_torus

Definition at line 1 of file class\_go\_1\_1\_torus.js.

## 30.422 doc/html/class\_go\_1\_1\_torus\_volume.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_torus\\_volume](#)

### 30.422.1 Variable Documentation

#### 30.422.1.1 var class\_go\_1\_1\_torus\_volume

##### Initial value:

```
=
[
 ["TorusVolume", "class_go_1_1_torus_volume.html#a0c3dc4910d48a2a9d1450b19be226d7c", null],
 ["TorusVolume", "class_go_1_1_torus_volume.html#a516fdfbcade6037b9879f657f9ace662", null],
 ["~TorusVolume", "class_go_1_1_torus_volume.html#ab336a8d113ebd8f59f782b036f98d364", null],
 ["boundingBox", "class_go_1_1_torus_volume.html#a4335419037adac3514e85a55ebe0b109", null],
 ["clone", "class_go_1_1_torus_volume.html#a1fff48a493a6c5996a7520ef218f010c0", null],
 ["closestPoint", "class_go_1_1_torus_volume.html#af674b7b73d3ed5094f40036a652a7962", null],
 ["dimension", "class_go_1_1_torus_volume.html#a4fd9bd42fc92d18308f835492dca6dde", null],
 ["geometryVolume", "class_go_1_1_torus_volume.html#aeb2024c1909972c22fde6a19bd4b2b7d", null],
 ["getAllBoundarySurfaces", "class_go_1_1_torus_volume.html#a10fcdd38505a34d65d94fb2b21c88342", null],
 ["instanceType", "class_go_1_1_torus_volume.html#a9ca9cdb4f81f3d50ed4ea8706ffc7dd0", null],
 ["nextSegmentVal", "class_go_1_1_torus_volume.html#af861ea80b6f06756411cb7f30c663de4", null],
 ["parameterSpan", "class_go_1_1_torus_volume.html#a6018ef9de8b522522f4df5dce4c05155", null],
 ["point", "class_go_1_1_torus_volume.html#a74901d29fd70d4753974c6839465f77a", null],
 ["point", "class_go_1_1_torus_volume.html#a0ad5f99df06ee7ae8ceb2ea8a8a73348", null],
 ["read", "class_go_1_1_torus_volume.html#a1257580f5c26ae9ee6378065bbcf9bb3", null],
 ["reverseParameterDirection", "class_go_1_1_torus_volume.html#aea36af2167236eb7c01a433a8b7151db", null],
],
 ["setParameters", "class_go_1_1_torus_volume.html#a210544afaf286644f7e98f5b3c0b918b", null],
 ["swapParameterDirection", "class_go_1_1_torus_volume.html#a601ce6f28a1f1738903c02e741d0ff79", null],
 ["tangentCone", "class_go_1_1_torus_volume.html#a4e5d384ca79038f368e5f447959407c2", null],
 ["translate", "class_go_1_1_torus_volume.html#a28984c3d0545a533f1990db25dd95d35", null],
 ["useCentreDegen", "class_go_1_1_torus_volume.html#a7c6fc62e10fc1f6e1b90b93e51698893", null],
 ["useCornerDegen", "class_go_1_1_torus_volume.html#a51c25ad52acb084648ea406a4bef2d23", null],
 ["write", "class_go_1_1_torus_volume.html#a6add375e29268f3fcfa51fb5335401b4", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_torus\_volume.js.

## 30.423 doc/html/class\_go\_1\_1\_triangle.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_triangle](#)

### 30.423.1 Variable Documentation

#### 30.423.1.1 var class\_go\_1\_1\_triangle

##### Initial value:

```
=
[
 ["Triangle", "class_go_1_1_triangle.html#af127a9641996c2230f101a3e5406c645", null],
 ["index", "class_go_1_1_triangle.html#a96b16c3ada91986b2c220b65d348cd33", null],
 ["setIndex", "class_go_1_1_triangle.html#ae108e05d561ebfa913a755e43dc6197f", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_triangle.js.

## 30.424 doc/html/class\_go\_1\_1\_trim\_curve.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_trim\\_curve](#)

### 30.424.1 Variable Documentation

#### 30.424.1.1 var class\_go\_1\_1\_trim\_curve

#### Initial value:

```
=
[
 ["TrimCurve", "class_go_1_1_trim_curve.html#a6525211e9b874c3d27187e06d27365ab", null],
 ["TrimCurve", "class_go_1_1_trim_curve.html#ab22e895bf04a4aa16cb0c6c1f1881fc3", null],
 ["TrimCurve", "class_go_1_1_trim_curve.html#a9bb20e93689b7a98cf88acbd3292c2c", null],
 ["~TrimCurve", "class_go_1_1_trim_curve.html#ad8584545d4d86980244eac16ba7f8cf2", null],
 ["approximationOK", "class_go_1_1_trim_curve.html#a8acd45f1699c5d978926b12c53f4e936", null],
 ["dim", "class_go_1_1_trim_curve.html#a7bc5bdebd06185cdfbaa6f84dc9d9776", null],
 ["end", "class_go_1_1_trim_curve.html#a3d558100c2f6cbe660b3dc91e0f17532", null],
 ["eval", "class_go_1_1_trim_curve.html#a383077ca958c414fc6c39ecde49df68f", null],
 ["eval", "class_go_1_1_trim_curve.html#a94b8b25ba0369b8bf01a5e18caafa7da", null],
 ["nmbCvs", "class_go_1_1_trim_curve.html#a529938e5cfff3d1bf3e863a80fe8ae36", null],
 ["start", "class_go_1_1_trim_curve.html#a7f1bb7e3edd36d68ffa4a26d1280ac19", null]
]
```

Definition at line 1 of file `class_go_1_1_trim_curve.js`.

## 30.425 doc/html/class\_go\_1\_1\_vertex.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_vertex](#)

### 30.425.1 Variable Documentation

#### 30.425.1.1 var class\_go\_1\_1\_vertex

Definition at line 1 of file `class_go_1_1_vertex.js`.

## 30.426 doc/html/class\_go\_1\_1\_vol\_boundary\_condition.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_vol\\_boundary\\_condition](#)



### 30.426.1 Variable Documentation

#### 30.426.1.1 var class\_go\_1\_1\_vol\_boundary\_condition

##### Initial value:

```
=
[
 ["VolBoundaryCondition", "class_go_1_1_vol_boundary_condition.html#a985022cb5c4455ded9c807b97572a1e7",
 null],
 ["VolBoundaryCondition", "class_go_1_1_vol_boundary_condition.html#a35a8cc1d5cbdc30b2325dd9454b4b5ab",
 null],
 ["~VolBoundaryCondition", "class_go_1_1_vol_boundary_condition.html#ad4457d87135143681c4f35d78553130f",
 null],
 ["faceNumber", "class_go_1_1_vol_boundary_condition.html#a917407b5d8f423b026002d6b0fldacc1", null],
 ["getBasisFunctions", "class_go_1_1_vol_boundary_condition.html#a1cbd6ba58dcc3613431f89c1f405374b",
 null],
 ["getBdCoefficients", "class_go_1_1_vol_boundary_condition.html#a8f27e48926dca66230faca4a63a8b71a",
 null],
 ["getBdCoefficients", "class_go_1_1_vol_boundary_condition.html#aad93e802e1e2c8c984b3a9fe6db27dbc",
 null],
 ["getCoefficientsEnumeration", "
 class_go_1_1_vol_boundary_condition.html#a6ce6cfa45df2f48eae4b6213fa4f0e4", null],
 ["getCoefficientsEnumeration", "
 class_go_1_1_vol_boundary_condition.html#a3df5d53a4de859552e8ab183c978e367", null],
 ["getSplineApproximation", "class_go_1_1_vol_boundary_condition.html#a5fb8fdb7a405791fcea72c2ee9cff42",
 null],
 ["getTolerances", "class_go_1_1_vol_boundary_condition.html#a0c2d2f72c5e91742072b3bd4f3ab08ca", null],
 ["update", "class_go_1_1_vol_boundary_condition.html#a26620f80a54a7abc88f5aab52436162d", null],
 ["updateBoundaryValue", "class_go_1_1_vol_boundary_condition.html#ad3a26eef46fb942f2e3974d49b545f09",
 null]
]
```

Definition at line 1 of file class\_go\_1\_1\_vol\_boundary\_condition.js.

## 30.427 doc/html/class\_go\_1\_1\_vol\_point\_bd\_cond.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_vol\\_point\\_bd\\_cond](#)

### 30.427.1 Variable Documentation

#### 30.427.1.1 var class\_go\_1\_1\_vol\_point\_bd\_cond

##### Initial value:

```
=
[
 ["VolPointBdCond", "class_go_1_1_vol_point_bd_cond.html#a64429f5f92a4ed25aee5a5fee6e6d348", null],
 ["~VolPointBdCond", "class_go_1_1_vol_point_bd_cond.html#a77171dab2e277ecaa9756a4fe274116d", null],
 ["faceNumber", "class_go_1_1_vol_point_bd_cond.html#a64e1b684ea2f63c8eeb42a1e6219cd38", null],
 ["getCoefficientsEnumeration", "class_go_1_1_vol_point_bd_cond.html#a38b405221f1771b32af415f34062a354",
 null],
 ["getConditionValue", "class_go_1_1_vol_point_bd_cond.html#a702d0d011f48f4c14892a5f062dd25d8", null],
 ["getInterpolationFactors", "class_go_1_1_vol_point_bd_cond.html#af9b62460eb7dec2b616586c741d70497",
 null],
 ["getParam", "class_go_1_1_vol_point_bd_cond.html#a19a06569354848e104c9b9eaa6e6a36", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_vol\_point\_bd\_cond.js.

## 30.428 doc/html/class\_go\_1\_1\_vol\_solution.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_vol\\_solution](#)

### 30.428.1 Variable Documentation

#### 30.428.1.1 var class\_go\_1\_1\_vol\_solution

Definition at line 1 of file class\_go\_1\_1\_vol\_solution.js.

## 30.429 doc/html/class\_go\_1\_1\_volume\_adjacency.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_volume\\_adjacency](#)

### 30.429.1 Variable Documentation

#### 30.429.1.1 var class\_go\_1\_1\_volume\_adjacency

#### Initial value:

```
=
[
 ["VolumeAdjacency", "class_go_1_1_volume_adjacency.html#a5444a3b2c457925ca3c6b5dcc8fc5681", null],
 ["~VolumeAdjacency", "class_go_1_1_volume_adjacency.html#a196d094e8f0fc59610180e957b19f7c7", null],
 ["setAdjacency", "class_go_1_1_volume_adjacency.html#aaa0ec7940d4alb262311c2b0686b4f29", null],
 ["setAdjacency", "class_go_1_1_volume_adjacency.html#a309fdde674bf6c8b2cac5da3961ad760", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_volume\_adjacency.js.

## 30.430 doc/html/class\_go\_1\_1\_volume\_model.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_volume\\_model](#)

### 30.430.1 Variable Documentation

#### 30.430.1.1 var class\_go\_1\_1\_volume\_model

Definition at line 1 of file class\_go\_1\_1\_volume\_model.js.

## 30.431 doc/html/class\_go\_1\_1\_volume\_model\_file\_handler.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_volume\\_model\\_file\\_handler](#)

### 30.431.1 Variable Documentation

#### 30.431.1.1 var class\_go\_1\_1\_volume\_model\_file\_handler

##### Initial value:

```
=
[
 ["VolumeModelFileHandler", "
 class_go_1_1_volume_model_file_handler.html#ad7e0d3efd94523e835406a97beb7d059", null],
 ["~VolumeModelFileHandler", "
 class_go_1_1_volume_model_file_handler.html#a1b249da05dedaff81762fa07b937aa67", null],
 ["readVolume", "class_go_1_1_volume_model_file_handler.html#a2cb244f3583e6a6a52ba0613653c3a2e", null]
,
 ["readVolumeModel", "class_go_1_1_volume_model_file_handler.html#a4b009237ca68adb7d5a853d15a44b52a",
 null],
 ["writeVolume", "class_go_1_1_volume_model_file_handler.html#a33ec81b81c6c421f7a3b8fb46386a431", null
],
 ["writeVolumeModel", "class_go_1_1_volume_model_file_handler.html#a95961adb981b37907e23673cb18505a0",
 null]
]
```

Definition at line 1 of file class\_go\_1\_1\_volume\_model\_file\_handler.js.

## 30.432 doc/html/class\_go\_1\_1\_volume\_parameter\_curve.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_volume\\_parameter\\_curve](#)

### 30.432.1 Variable Documentation

#### 30.432.1.1 var class\_go\_1\_1\_volume\_parameter\_curve

##### Initial value:

```
=
[
 ["VolumeParameterCurve", "class_go_1_1_volume_parameter_curve.html#a9c53fbd8c50484783d122f3041788c4",
 null],
 ["VolumeParameterCurve", "class_go_1_1_volume_parameter_curve.html#adaeb5c4b719b82cf31accbdb89840cb9",
 null],
 ["~VolumeParameterCurve", "class_go_1_1_volume_parameter_curve.html#a003c2f1384c02c34dc11f7453ecea143",
 null],
 ["approximationOK", "class_go_1_1_volume_parameter_curve.html#a94ed3214219c50ccb994be11319cf7df", null
],
 ["dim", "class_go_1_1_volume_parameter_curve.html#a791188c37d4dda0e212f5e1ee5d2bac3", null],
 ["end", "class_go_1_1_volume_parameter_curve.html#a87a3c5acb7a0484c6dc29f640a28a836", null],
 ["eval", "class_go_1_1_volume_parameter_curve.html#a9fe5deedf6c920a026949df25a804eeb", null],
 ["eval", "class_go_1_1_volume_parameter_curve.html#a7a1967e05c656a34b29c9e455c6985d3", null],
 ["start", "class_go_1_1_volume_parameter_curve.html#a2eb4e73a4ae17c7a0a57e33c797f0960", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_volume\_parameter\_curve.js.

## 30.433 doc/html/class\_go\_1\_1\_volume\_space\_curve.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_volume\\_space\\_curve](#)

### 30.433.1 Variable Documentation

#### 30.433.1.1 var class\_go\_1\_1\_volume\_space\_curve

##### Initial value:

```
=
[
 ["VolumeSpaceCurve", "class_go_1_1_volume_space_curve.html#a95eae74b6d93c2e4ae5ff3cb675813f4", null],
 ["~VolumeSpaceCurve", "class_go_1_1_volume_space_curve.html#adbce44f47e7d50e7b77b6ecd59749afe", null]
,
 ["approximationOK", "class_go_1_1_volume_space_curve.html#a27dccb1129ce3c426679b09eb8903420", null],
 ["dim", "class_go_1_1_volume_space_curve.html#a9d0c85b04c21ae090dcabc0d4e617e8b", null],
 ["end", "class_go_1_1_volume_space_curve.html#a74c52038b1d9a4989ee16e7306ea3484", null],
 ["eval", "class_go_1_1_volume_space_curve.html#a9c680f8db2892c4000e199d3591c4ec6", null],
 ["eval", "class_go_1_1_volume_space_curve.html#adf40f504a4ffcd83d09f88dd023deb69", null],
 ["start", "class_go_1_1_volume_space_curve.html#a82e4dd31f04cfeeb2944f2d38bfab9c9", null]
]
```

Definition at line 1 of file class\_go\_1\_1\_volume\_space\_curve.js.

## 30.434 doc/html/class\_go\_1\_1\_box\_structuring\_1\_1\_bounding\_box\_structure.js File Reference

### Variables

- var [class\\_go\\_1\\_1\\_box\\_structuring\\_1\\_1\\_bounding\\_box\\_structure](#)

### 30.434.1 Variable Documentation

#### 30.434.1.1 var class\_go\_1\_1\_box\_structuring\_1\_1\_bounding\_box\_structure

##### Initial value:

```
=
[
 ["addBox", "
class_go_1_1_box_structuring_1_1_bounding_box_structure.html#aa6e1a511a5f48c80dd1042ba69bb68ff", null],
 ["addSurface", "
class_go_1_1_box_structuring_1_1_bounding_box_structure.html#a0e5b6293f56db1709b1ef89c7311a1bb", null],
 ["big_vox_low", "
class_go_1_1_box_structuring_1_1_bounding_box_structure.html#a736464d67dfef8fd9282e106cf343e70", null],
 ["boxes_in_voxel", "
class_go_1_1_box_structuring_1_1_bounding_box_structure.html#a202bbc9eabebc042a3fda903381058f0", null],
 ["BuildVoxelStructure", "
class_go_1_1_box_structuring_1_1_bounding_box_structure.html#ab4ffd31e7ff43f6fbc0070e5911d0fab", null],
 ["closestPoint", "
class_go_1_1_box_structuring_1_1_bounding_box_structure.html#a9d3483bf9db84395d7b0243f1a83f1d8", null],
 ["getBox", "
class_go_1_1_box_structuring_1_1_bounding_box_structure.html#a76008f7678515e6b849abdc5ab263814", null],
 ["getSurface", "

```

```

class_go_1_lbox_structuring_1_1_bounding_box_structure.html#a0c63afb6c73a475dbad5ba9c7f67d294", null],
["n_boxes", "
class_go_1_lbox_structuring_1_1_bounding_box_structure.html#a60971e92b7351db884394bb924002f99", null],
["n_surfaces", "
class_go_1_lbox_structuring_1_1_bounding_box_structure.html#aa51b875860291f92adb86b4034bdef4e", null],
["n Voxels_x", "
class_go_1_lbox_structuring_1_1_bounding_box_structure.html#af796d20691409c02d3d3831106917bba", null],
["n Voxels_y", "
class_go_1_lbox_structuring_1_1_bounding_box_structure.html#a70d433f73cb7086fd851e190a3ede696", null],
["n Voxels_z", "
class_go_1_lbox_structuring_1_1_bounding_box_structure.html#a6e6bec0d1412870fabcf13a14c244e3d", null],
["setSurfaceCopies", "
class_go_1_lbox_structuring_1_1_bounding_box_structure.html#aac1768031db6b3832c58e4aff59640ae", null],
["voxel_length", "
class_go_1_lbox_structuring_1_1_bounding_box_structure.html#a650cf3d4173e47c2eb0caa919471a7a2", null]
]

```

Definition at line 1 of file class\_go\_1\_1box\_structuring\_1\_1\_bounding\_box\_structure.js.

## 30.435 doc/html/class\_go\_1\_1box\_structuring\_1\_1\_sub\_surface\_bounding\_box.js File Reference

### Variables

- var [class\\_go\\_1\\_1box\\_structuring\\_1\\_1\\_sub\\_surface\\_bounding\\_box](#)

### 30.435.1 Variable Documentation

#### 30.435.1.1 var class\_go\_1\_1box\_structuring\_1\_1\_sub\_surface\_bounding\_box

#### Initial value:

```

=
[
["SubSurfaceBoundingBox", "
class_go_1_lbox_structuring_1_1_sub_surface_bounding_box.html#a028931b8aef4c065a3af24bba3c9c6d7", null],
["add_polygon_corners", "
class_go_1_lbox_structuring_1_1_sub_surface_bounding_box.html#a3561f48b916a553d92531b1fe6ec1bd8", null],
["box", "
class_go_1_lbox_structuring_1_1_sub_surface_bounding_box.html#a1523a25a1aeea8ae277807aa680ce357", null],
["has_polygon", "
class_go_1_lbox_structuring_1_1_sub_surface_bounding_box.html#a80a5139fc405eae3fe76b423b6df55ec", null],
["inside", "
class_go_1_lbox_structuring_1_1_sub_surface_bounding_box.html#ab382401be6394619919c2b1eb46a3849", null],
["inside", "
class_go_1_lbox_structuring_1_1_sub_surface_bounding_box.html#a38c7a0bb3c94ecedd32bbcf9bb023c5f", null],
["par_domain", "
class_go_1_lbox_structuring_1_1_sub_surface_bounding_box.html#a45940dcad0e6712eb009e8079494c5ae", null],
["pos_u", "
class_go_1_lbox_structuring_1_1_sub_surface_bounding_box.html#a04c502630091ab5ce64eb27dffabcca9", null],
["pos_v", "
class_go_1_lbox_structuring_1_1_sub_surface_bounding_box.html#abcd64cb19ec756fecf9b212cdd1a7776", null],
["remove_polygon", "
class_go_1_lbox_structuring_1_1_sub_surface_bounding_box.html#a6f1290e112d8f28fce991ed6c1c3638", null],
["setInside", "
class_go_1_lbox_structuring_1_1_sub_surface_bounding_box.html#a067587c6ca6ecb9e3399438d96eda11b", null],
["size_polygon", "
class_go_1_lbox_structuring_1_1_sub_surface_bounding_box.html#a4914de8c686a57577f57fc4eed7762cc", null],
["surface_data", "
class_go_1_lbox_structuring_1_1_sub_surface_bounding_box.html#add41a4bbfaf7be7e4ac1ce36f30f9246", null]
]

```

Definition at line 1 of file class\_go\_1\_1box\_structuring\_1\_1\_sub\_surface\_bounding\_box.js.

## 30.436 doc/html/class\_go\_1\_lbox\_structuring\_1\_1\_surface\_data.js File Reference

### Variables

- var [class\\_go\\_1\\_lbox\\_structuring\\_1\\_1\\_surface\\_data](#)

### 30.436.1 Variable Documentation

#### 30.436.1.1 var class\_go\_1\_lbox\_structuring\_1\_1\_surface\_data

##### Initial value:

```
=
[
 ["SurfaceData", "class_go_1_lbox_structuring_1_1_surface_data.html#ac3260cf68ee70164ae4d70e6e8d63e5d",
 null],
 ["add_inside_point", "
class_go_1_lbox_structuring_1_1_surface_data.html#a8c85c14e584efc5f4367ed68c01f2dd4", null],
 ["index", "class_go_1_lbox_structuring_1_1_surface_data.html#adb3818aaf057e43c17aef99f9056945e", null
],
 ["inside_points", "class_go_1_lbox_structuring_1_1_surface_data.html#a305cfe037489488770c8b5ded9de7bf8
", null],
 ["segs_u", "class_go_1_lbox_structuring_1_1_surface_data.html#a14685aae32717091917327b622f1afde", null
],
 ["segs_v", "class_go_1_lbox_structuring_1_1_surface_data.html#a47316eeba2205c1ee16dbdbae91a8be2", null
],
 ["setIndex", "class_go_1_lbox_structuring_1_1_surface_data.html#a0082256cb9d872df530a36ed2ee2c949",
 null],
 ["setSegments", "class_go_1_lbox_structuring_1_1_surface_data.html#a9f84328f4883871ea62da278bfb7ce6",
 null],
 ["setSurfaceCopies", "
class_go_1_lbox_structuring_1_1_surface_data.html#af30alc4ee89f538b46ba55c42c943538", null],
 ["surface", "class_go_1_lbox_structuring_1_1_surface_data.html#a82208ca531f95f68c442e2fa22d0d41d",
 null]
]
```

Definition at line 1 of file [class\\_go\\_1\\_lbox\\_structuring\\_1\\_1\\_surface\\_data.js](#).

## 30.437 doc/html/class\_go\_1\_lft\_cell.js File Reference

### Variables

- var [class\\_go\\_1\\_lft\\_cell](#)

### 30.437.1 Variable Documentation

#### 30.437.1.1 var class\_go\_1\_lft\_cell

##### Initial value:

```
=
[
 ["ftCell", "class_go_1_lft_cell.html#aa92b81a003e69f3c2b7712abc39dcc13", null],
 ["~ftCell", "class_go_1_lft_cell.html#a1133f48e5add15a8503563c34cb5251d", null],
 ["addFace", "class_go_1_lft_cell.html#a9a5e3adbd11434a6ad181ba660be853c", null],
 ["addFaceBox", "class_go_1_lft_cell.html#abe64e97467a88d5c2827abf4e178e9e", null],
 ["box", "class_go_1_lft_cell.html#afe2c069e75af417181523d1d6c7454a8", null],
 ["face", "class_go_1_lft_cell.html#aa121981ee3b13d136e7d29b0ee4bcd77", null],
 ["faceBox", "class_go_1_lft_cell.html#a5e5ae761c933390e66693af8ae239122", null],
 ["num_faces", "class_go_1_lft_cell.html#a2612f2186d11c244866b9e01032f49ba", null],
 ["removeFace", "class_go_1_lft_cell.html#a485d9f75df9713040406521a722351e7", null],
 ["setBox", "class_go_1_lft_cell.html#a4dda9139b4201bfeae98f593ba51d8ef", null],
 ["box_", "class_go_1_lft_cell.html#adcd3dc1436e37ba68b9a56bb9faf24aa", null],
 ["face_boxes_", "class_go_1_lft_cell.html#a9947e2f9b0e6a893dc537205c65c8442", null],
 ["faces_", "class_go_1_lft_cell.html#a1818339a8ddd910cf35a2953c311b4af", null]
]
```

Definition at line 1 of file [class\\_go\\_1\\_lft\\_cell.js](#).

## 30.438 doc/html/class\_go\_1\_1ft\_cell\_info.js File Reference

### Variables

- var [class\\_go\\_1\\_1ft\\_cell\\_info](#)

#### 30.438.1 Variable Documentation

##### 30.438.1.1 var class\_go\_1\_1ft\_cell\_info

#### Initial value:

```
=
[
 ["ftCellInfo", "class_go_1_1ft_cell_info.html#ac201d70737750e45dcaea2e5c692f543", null],
 ["ftCellInfo", "class_go_1_1ft_cell_info.html#aed8671b7fe5c84685e89521cf3043f0f", null],
 ["operator<", "class_go_1_1ft_cell_info.html#ad7af2b236ba39886b8b4d3ec7e5e95a4", null],
 ["dist_", "class_go_1_1ft_cell_info.html#aea93ce0074a9eb7c419a77362ea20390", null],
 ["index_", "class_go_1_1ft_cell_info.html#aafd2de5219d54de3d4f41995ba620282", null]
]
```

Definition at line 1 of file class\_go\_1\_1ft\_cell\_info.js.

## 30.439 doc/html/class\_go\_1\_1ft\_chart\_surface.js File Reference

### Variables

- var [class\\_go\\_1\\_1ft\\_chart\\_surface](#)

#### 30.439.1 Variable Documentation

##### 30.439.1.1 var class\_go\_1\_1ft\_chart\_surface

Definition at line 1 of file class\_go\_1\_1ft\_chart\_surface.js.

## 30.440 doc/html/class\_go\_1\_1ft\_curve.js File Reference

### Variables

- var [class\\_go\\_1\\_1ft\\_curve](#)

#### 30.440.1 Variable Documentation

##### 30.440.1.1 var class\_go\_1\_1ft\_curve

Definition at line 1 of file class\_go\_1\_1ft\_curve.js.

## 30.441 doc/html/class\_go\_1\_1ft\_curve\_segment.js File Reference

### Variables

- var [class\\_go\\_1\\_1ft\\_curve\\_segment](#)

### 30.441.1 Variable Documentation

30.441.1.1 var class\_go\_1\_1ft\_curve\_segment

Definition at line 1 of file class\_go\_1\_1ft\_curve\_segment.js.

## 30.442 doc/html/class\_go\_1\_1ft\_edge.js File Reference

### Variables

- var [class\\_go\\_1\\_1ft\\_edge](#)

### 30.442.1 Variable Documentation

30.442.1.1 var class\_go\_1\_1ft\_edge

Definition at line 1 of file class\_go\_1\_1ft\_edge.js.

## 30.443 doc/html/class\_go\_1\_1ft\_edge\_base.js File Reference

### Variables

- var [class\\_go\\_1\\_1ft\\_edge\\_base](#)

### 30.443.1 Variable Documentation

30.443.1.1 var class\_go\_1\_1ft\_edge\_base

Definition at line 1 of file class\_go\_1\_1ft\_edge\_base.js.

## 30.444 doc/html/class\_go\_1\_1ft\_face\_base.js File Reference

### Variables

- var [class\\_go\\_1\\_1ft\\_face\\_base](#)



### 30.444.1 Variable Documentation

#### 30.444.1.1 var class\_go\_1\_1ft\_face\_base

##### Initial value:

```
=
[
 ["ftFaceBase", "class_go_1_1ft_face_base.html#aeb3a1f454567c2413295b8399f4b49e8", null],
 ["ftFaceBase", "class_go_1_1ft_face_base.html#a013d8c2cb9df9e4522323822df28c61d", null],
 ["~ftFaceBase", "class_go_1_1ft_face_base.html#a069103abb148ef2b0d05a4d2f86a828b", null],
 ["asFtSurface", "class_go_1_1ft_face_base.html#a0cf6138638e06631722761cbbd7105e2", null],
 ["boundingBox", "class_go_1_1ft_face_base.html#ad73654559003e7e2c5255255c06a4af1", null],
 ["clearInitialEdges", "class_go_1_1ft_face_base.html#a44cc2ead4dc2f6c7a09ee91d84d95d4b", null],
 ["closestPoint", "class_go_1_1ft_face_base.html#acc397f0ca4ceae1b3fb6f8c23cae768", null],
 ["createInitialEdges", "class_go_1_1ft_face_base.html#a7630d69756a3487446041ea364b863e3", null],
 ["createSurf", "class_go_1_1ft_face_base.html#ab4a79799a587e2bcbfad8abd4368d629", null],
 ["getError", "class_go_1_1ft_face_base.html#a233bf7917d3bda72e7ffba165cfacdf7", null],
 ["getId", "class_go_1_1ft_face_base.html#a53510da9b831537e94cc77f99dfb4fde", null],
 ["getPrioType", "class_go_1_1ft_face_base.html#a87d1107a1aa219882d2f5707bce975fa", null],
 ["isolateFace", "class_go_1_1ft_face_base.html#a92563f5059c70b0b6a7f8503108549e2", null],
 ["normal", "class_go_1_1ft_face_base.html#a9f903c0a338ee84d697b6244017c2409", null],
 ["point", "class_go_1_1ft_face_base.html#a8bfc890d8a1145f7e0d3586af4a6e0f1", null],
 ["removeGap", "class_go_1_1ft_face_base.html#aa72131ebf12e16648e6ac3fc534aa265", null],
 ["setId", "class_go_1_1ft_face_base.html#ad52dle2593360de63c77830781a1f0ac", null],
 ["startEdges", "class_go_1_1ft_face_base.html#aad97f5f5bef285cc4187afce79e00e903", null],
 ["surface", "class_go_1_1ft_face_base.html#a7682502dea96834e44693b25e4db533a", null],
 ["updateBoundaryLoops", "class_go_1_1ft_face_base.html#aa41f6fd57d5df5959edd9f2ed2de72c5", null],
 ["id_", "class_go_1_1ft_face_base.html#a26ef3f6e5e428b74b4e6aa0a79cd712a", null]
]
```

Definition at line 1 of file class\_go\_1\_1ft\_face\_base.js.

## 30.445 doc/html/class\_go\_1\_1ft\_graph\_edge.js File Reference

### Variables

- var [class\\_go\\_1\\_1ft\\_graph\\_edge](#)

### 30.445.1 Variable Documentation

#### 30.445.1.1 var class\_go\_1\_1ft\_graph\_edge

##### Initial value:

```
=
[
 ["ftGraphEdge", "class_go_1_1ft_graph_edge.html#a6ea3131608912062d5e7af5403fb5f5d", null],
 ["ftGraphEdge", "class_go_1_1ft_graph_edge.html#a96fa14bb1c156fcad495d4789fd5fc33", null],
 ["~ftGraphEdge", "class_go_1_1ft_graph_edge.html#a50ab56d28b980d23b91e9cd55938b308", null],
 ["endparam", "class_go_1_1ft_graph_edge.html#a260f3773935b8d875bc075e51a1f82df", null],
 ["endPoint", "class_go_1_1ft_graph_edge.html#a45c4087b05f202f4e0f4152c21808c75", null],
 ["getFaces", "class_go_1_1ft_graph_edge.html#a7ca2917fa8fae6dea9929f7435c21e28", null],
 ["lower", "class_go_1_1ft_graph_edge.html#a1d387d43635dc8870a4a1f8d1d607c64", null],
 ["point", "class_go_1_1ft_graph_edge.html#ac8253558686f05c30089f5eb5d6c3d98", null],
 ["point", "class_go_1_1ft_graph_edge.html#afe85117484ec43cbc63ca666c4988ed4", null],
 ["startparam", "class_go_1_1ft_graph_edge.html#a4bab461425a733b333930120c0fab079", null],
 ["startPoint", "class_go_1_1ft_graph_edge.html#ab83312300bcd3c4235a60bf6007266a9", null],
 ["upper", "class_go_1_1ft_graph_edge.html#a8069fd01ad452256e44353be62bb91da", null]
]
```

Definition at line 1 of file class\_go\_1\_1ft\_graph\_edge.js.

## 30.446 doc/html/class\_go\_1\_1ft\_group\_geom.js File Reference

### Variables

- var [class\\_go\\_1\\_1ft\\_group\\_geom](#)

### 30.446.1 Variable Documentation

#### 30.446.1.1 var class\_go\_1\_1ft\_group\_geom

##### Initial value:

```
=
[
 ["ftGroupGeom", "class_go_1_1ft_group_geom.html#a81e48684597d986148f78a926375ee22", null],
 ["~ftGroupGeom", "class_go_1_1ft_group_geom.html#af927889ce7acd322f29b03a8f1057bce", null],
 ["addGeomObj", "class_go_1_1ft_group_geom.html#a6deb85d14d38905ac1e98c520efcdf87", null],
 ["getType", "class_go_1_1ft_group_geom.html#a6ae7c836a918765c1da2678fee62098b", null],
 ["operator[]", "class_go_1_1ft_group_geom.html#a8f7683e608575d829d0834679d561854", null],
 ["setType", "class_go_1_1ft_group_geom.html#ac4459cc4013744086035b74fae2bedd3", null],
 ["size", "class_go_1_1ft_group_geom.html#a438c8a5ccec38260202b6597497c64d9", null],
 ["geomobj_", "class_go_1_1ft_group_geom.html#a777549837a91518c25e875647a5b4280", null],
 ["type_", "class_go_1_1ft_group_geom.html#a369b7c907a4ad9176cbac044ef45239b", null]
]
```

Definition at line 1 of file class\_go\_1\_1ft\_group\_geom.js.

## 30.447 doc/html/class\_go\_1\_1ft\_line.js File Reference

### Variables

- var [class\\_go\\_1\\_1ft\\_line](#)

### 30.447.1 Variable Documentation

#### 30.447.1.1 var class\_go\_1\_1ft\_line

##### Initial value:

```
=
[
 ["ftLine", "class_go_1_1ft_line.html#a50680feb6bc47518f7e4ec29e9b3705e", null],
 ["ftLine", "class_go_1_1ft_line.html#a50559440f5f6cbae2c0a1ad31ea6ca2e", null],
 ["~ftLine", "class_go_1_1ft_line.html#a36ab8167ae4398c8eab81e69abaa033f", null],
 ["closeToScaledPoint", "class_go_1_1ft_line.html#a3d7867a61227b67f16c611e57fa38733", null],
 ["direction", "class_go_1_1ft_line.html#a0ef27b802dd865c8d93f8ace19d3f5d5", null],
 ["getTwoPlanes", "class_go_1_1ft_line.html#af583cd5f86689072ce4c153ec374edba", null],
 ["intersectsBox", "class_go_1_1ft_line.html#ac9ab61527c36d5ea0128891bae6f0b21", null],
 ["intersectsSphereOfBox", "class_go_1_1ft_line.html#a98718b75fec0f28dc66c9e71596c29a6", null],
 ["planesIntersectBox", "class_go_1_1ft_line.html#a67e729b07c42c78ce4d845afad402f4", null],
 ["point", "class_go_1_1ft_line.html#af959e5b16b21130f038dd3523d1f4af3", null],
 ["scaled", "class_go_1_1ft_line.html#a05c3f88742e2579e793de0f7bca89c2e", null],
 ["dir_", "class_go_1_1ft_line.html#a4ee4aab40795d2e7414bc20c53742d90", null],
 ["point_", "class_go_1_1ft_line.html#a4717d36b942e5c0309e88af802a808e3", null]
]
```

Definition at line 1 of file class\_go\_1\_1ft\_line.js.

## 30.448 doc/html/class\_go\_1\_1ft\_message.js File Reference

### Variables

- var [class\\_go\\_1\\_1ft\\_message](#)

#### 30.448.1 Variable Documentation

##### 30.448.1.1 var class\_go\_1\_1ft\_message

###### Initial value:

```
=
[
 ["ftMessage", "class_go_1_1ft_message.html#a1f5e69c82a65f1ce04d2a606fd50559b", null],
 ["ftMessage", "class_go_1_1ft_message.html#a8280d5a84f869825940ddd77de4cb49e", null],
 ["~ftMessage", "class_go_1_1ft_message.html#ad2003bdf49ad919afa044983fb4dd113", null],
 ["addWarning", "class_go_1_1ft_message.html#aa4370d4b71dd0e273d9db94cdb84941e", null],
 ["getMessage", "class_go_1_1ft_message.html#acb1cbl277521c8da8581f2157ac89d8", null],
 ["getWarning", "class_go_1_1ft_message.html#a0c7185d252ac2498128ce55aef59ba52", null],
 ["isOK", "class_go_1_1ft_message.html#abbc9876b12c5b0314c8aece95044531d", null],
 ["noOfWarnings", "class_go_1_1ft_message.html#ad3df0b7fe89f8a74551fc340896e6e19", null],
 ["setError", "class_go_1_1ft_message.html#aead9adc3f2830c8cd4710780ee2f725", null]
]
```

Definition at line 1 of file class\_go\_1\_1ft\_message.js.

## 30.449 doc/html/class\_go\_1\_1ft\_planar\_graph.js File Reference

### Variables

- var [class\\_go\\_1\\_1ft\\_planar\\_graph](#)

#### 30.449.1 Variable Documentation

##### 30.449.1.1 var class\_go\_1\_1ft\_planar\_graph

###### Initial value:

```
=
[
 ["ftPlanarGraph", "class_go_1_1ft_planar_graph.html#a55856f344896114fd0d9ffc2b5344e40", null],
 ["ftPlanarGraph", "class_go_1_1ft_planar_graph.html#a995726dc6cbcfd3c62a01aa0b71efaff", null],
 ["~ftPlanarGraph", "class_go_1_1ft_planar_graph.html#a374ad049daa19cb189d9c0182ef4e029", null],
 ["getLocalParameters", "class_go_1_1ft_planar_graph.html#adb3fda7dc1ec6f160ad04438f4fdfe11", null],
 ["locateInGraph", "class_go_1_1ft_planar_graph.html#a02126a5ddc1b4881e27111f03effedf6", null],
 ["setGraph", "class_go_1_1ft_planar_graph.html#a31693e3927059d80cd0f0b1790fb7cc2", null]
]
```

Definition at line 1 of file class\_go\_1\_1ft\_planar\_graph.js.

## 30.450 doc/html/class\_go\_1\_1ft\_plane.js File Reference

### Variables

- var [class\\_go\\_1\\_1ft\\_plane](#)

### 30.450.1 Variable Documentation

#### 30.450.1.1 var class\_go\_1\_1ft\_plane

##### Initial value:

```
=
[
 ["ftPlane", "class_go_1_1ft_plane.html#ad7ba7ffd1fccdd41130daad7c21a0c2", null],
 ["ftPlane", "class_go_1_1ft_plane.html#ac22d30d3b20998b447e9137482142770", null],
 ["ftPlane", "class_go_1_1ft_plane.html#a52c1e63299a028bf909026a188f65cba", null],
 ["ftPlane", "class_go_1_1ft_plane.html#a6925de11b4ca78706fdd8e430ceec640", null],
 ["~ftPlane", "class_go_1_1ft_plane.html#a72f222fc0ca1ab772bb0bcb9510479f2", null],
 ["intersectsBox", "class_go_1_1ft_plane.html#a0cb13a54332b9bd081d02eafcd73ee47", null],
 ["normal", "class_go_1_1ft_plane.html#a0afd0f1c5b74089f089f5b4c9716ac08", null],
 ["point", "class_go_1_1ft_plane.html#a09ad762952a3c02f1adab49d88df9347", null],
 ["normal_", "class_go_1_1ft_plane.html#a1ee6b0d639a4e8b46397d1cfd97fb1a", null],
 ["point_", "class_go_1_1ft_plane.html#a68667709fb05418daafd32c7ec48c31c", null]
]
```

Definition at line 1 of file class\_go\_1\_1ft\_plane.js.

## 30.451 doc/html/class\_go\_1\_1ft\_point.js File Reference

### Variables

- var [class\\_go\\_1\\_1ft\\_point](#)

### 30.451.1 Variable Documentation

#### 30.451.1.1 var class\_go\_1\_1ft\_point

##### Initial value:

```
=
[
 ["ftPoint", "class_go_1_1ft_point.html#a2bef21b59ea7fd6f6f869939696442c4", null],
 ["ftPoint", "class_go_1_1ft_point.html#a71e8cdalfee6e4d77e8e47cc9de67fe7", null],
 ["face", "class_go_1_1ft_point.html#aa32c53f685602d17fa5e4f9ale1f0c4d", null],
 ["normal", "class_go_1_1ft_point.html#ab6216a92b607e1b71e13ba6e5c991f5b", null],
 ["position", "class_go_1_1ft_point.html#af3b40057fbad3323002e27ef98035dd5", null],
 ["u", "class_go_1_1ft_point.html#a6d2038f4d89e1b35e149cc47f782fc29", null],
 ["v", "class_go_1_1ft_point.html#a5f721b935eeefed8e81d0cf416cf0f2a", null],
 ["pt_", "class_go_1_1ft_point.html#ac9649d1cfcf1216802eaedbcae856d13", null],
 ["surface_", "class_go_1_1ft_point.html#af5409e867eadda68994b4facf1e1d49b", null],
 ["u_", "class_go_1_1ft_point.html#a3a514101ecd669aeb1feb67bf095a176", null],
 ["v_", "class_go_1_1ft_point.html#ae66d28de47384fb0fd2f113281a83848", null]
]
```

Definition at line 1 of file class\_go\_1\_1ft\_point.js.

## 30.452 doc/html/class\_go\_1\_1ft\_point\_set.js File Reference

### Variables

- var [class\\_go\\_1\\_1ft\\_point\\_set](#)

#### 30.452.1 Variable Documentation

##### 30.452.1.1 var class\_go\_1\_1ft\_point\_set

Definition at line 1 of file class\_go\_1\_1ft\_point\_set.js.

## 30.453 doc/html/class\_go\_1\_1ft\_s\_sf\_edge.js File Reference

### Variables

- var [class\\_go\\_1\\_1ft\\_s\\_sf\\_edge](#)

#### 30.453.1 Variable Documentation

##### 30.453.1.1 var class\_go\_1\_1ft\_s\_sf\_edge

#### Initial value:

```
=
[
 ["ftSSfEdge", "class_go_1_1ft_s_sf_edge.html#a9cabfea3f1d54e47004a0a61b47fd868", null],
 ["~ftSSfEdge", "class_go_1_1ft_s_sf_edge.html#a74ab8de1f47ed43d3e3c96bcf37cdca2", null],
 ["boundingBox", "class_go_1_1ft_s_sf_edge.html#a2b186aa55a6150d1d26b49d10e2f90e0", null],
 ["closestPoint", "class_go_1_1ft_s_sf_edge.html#aad89474f862f07a2f26226e01b33500c", null],
 ["entryId", "class_go_1_1ft_s_sf_edge.html#af2517c199e5fdb6417ae12ced7f4bc37", null],
 ["face", "class_go_1_1ft_s_sf_edge.html#a2659c6f1a432256dab88ba2d2b110b84", null],
 ["geomEdge", "class_go_1_1ft_s_sf_edge.html#a7195ecf4fdcf77ca4e8664f6d269bd73", null],
 ["isReversed", "class_go_1_1ft_s_sf_edge.html#ad79d781e4c515601f460b0495b6a2dfa", null],
 ["normal", "class_go_1_1ft_s_sf_edge.html#a0aff4574036a73c4d507cb5674ae6ed4", null],
 ["normal", "class_go_1_1ft_s_sf_edge.html#a6b4f82925064c770f2fbfb295e6484fa", null],
 ["point", "class_go_1_1ft_s_sf_edge.html#acbec7c42e8fc53b27ddf66a620ce1d32", null],
 ["setEntryId", "class_go_1_1ft_s_sf_edge.html#a12dba8ed3961568f73783adb5e42c8c9", null],
 ["setFace", "class_go_1_1ft_s_sf_edge.html#a8a9e6849b17317c94e0bbfb770c8ce8b", null],
 ["setReversed", "class_go_1_1ft_s_sf_edge.html#a30fd26f69b7a09999eb9368b25914ac0", null],
 ["split", "class_go_1_1ft_s_sf_edge.html#aec6284ac13d112ae5f4c35373d686015", null],
 ["tangent", "class_go_1_1ft_s_sf_edge.html#aec5af9042c2a159c2b8dce97f88eb4ad", null],
 ["tMax", "class_go_1_1ft_s_sf_edge.html#ade89d8991d27a0ac9a3d6f42d13ed1e5", null],
 ["tMin", "class_go_1_1ft_s_sf_edge.html#a21b6a3c201008dc2eebbe6d91a95b721", null]
]
```

Definition at line 1 of file class\_go\_1\_1ft\_s\_sf\_edge.js.

## 30.454 doc/html/class\_go\_1\_1ft\_sample\_point.js File Reference

### Variables

- var [class\\_go\\_1\\_1ft\\_sample\\_point](#)

### 30.454.1 Variable Documentation

#### 30.454.1.1 var class\_go\_1\_1ft\_sample\_point

Definition at line 1 of file class\_go\_1\_1ft\_sample\_point.js.

## 30.455 doc/html/class\_go\_1\_1ft\_search\_node.js File Reference

### Variables

- var [class\\_go\\_1\\_1ft\\_search\\_node](#)

### 30.455.1 Variable Documentation

#### 30.455.1.1 var class\_go\_1\_1ft\_search\_node

#### Initial value:

```
=
[
 ["ftSearchNode", "class_go_1_1ft_search_node.html#a9c55cef76fa0622368b25392da282f99", null],
 ["~ftSearchNode", "class_go_1_1ft_search_node.html#aebf452021f76a83b39a418404a6c71d3", null],
 ["getOrderedSegments", "class_go_1_1ft_search_node.html#afdca877312a1141ab057e330a5a536a5", null],
 ["node", "class_go_1_1ft_search_node.html#aba44c9cf35c551cca4d4f5828242028a", null],
 ["setOrderedSegments", "class_go_1_1ft_search_node.html#abc57b221eb7c3025d81ca2cfc2cf1011", null]
]
```

Definition at line 1 of file class\_go\_1\_1ft\_search\_node.js.

## 30.456 doc/html/class\_go\_1\_1ft\_smooth\_surf.js File Reference

### Variables

- var [class\\_go\\_1\\_1ft\\_smooth\\_surf](#)

### 30.456.1 Variable Documentation

#### 30.456.1.1 var class\_go\_1\_1ft\_smooth\_surf

#### Initial value:

```
=
[
 ["ftSmoothSurf", "class_go_1_1ft_smooth_surf.html#a187b305c0c5c021395b9d9540f0b5151", null],
 ["~ftSmoothSurf", "class_go_1_1ft_smooth_surf.html#a01388579e24fd359b452409b55224a0e", null],
 ["getApproxWeight", "class_go_1_1ft_smooth_surf.html#aca11835e16f9ce421fe98e518e636e70", null],
 ["getError", "class_go_1_1ft_smooth_surf.html#a31be1435532cae978198475d9f607418", null],
 ["refineSurf", "class_go_1_1ft_smooth_surf.html#ab418789f68680dd0a809117e62b4eb7a", null],
 ["setApproxWeight", "class_go_1_1ft_smooth_surf.html#a71684faf5912cfc9f38dd964f7f14072", null],
 ["setSmoothU", "class_go_1_1ft_smooth_surf.html#aae9f578161e0a2238cf322da89402d8d", null],
 ["setSmoothV", "class_go_1_1ft_smooth_surf.html#a9698869692e3c10328c3b149661cd597", null],
 ["update", "class_go_1_1ft_smooth_surf.html#a3312c5bdd8f78cb80f91c6ee48885d54", null],
 ["init_approx_weight_", "class_go_1_1ft_smooth_surf.html#a2ef7fc8e3c6c1ace11c672ef0ea32035", null],
 ["approx_tol_", "class_go_1_1ft_smooth_surf.html#aec9cdafdc2633142a81e2fd4a8388ba45", null],
 ["ccw_edge_derivs_", "class_go_1_1ft_smooth_surf.html#ac3b463dca9a0004897472a5beead0dc2", null],
 ["init_approx_weight_", "class_go_1_1ft_smooth_surf.html#a3f8e09ac76a0bee1e43db724bc400863", null],
 ["lock_corner_points_", "class_go_1_1ft_smooth_surf.html#ae4675b41bfff97e8184886a852b5128c", null],
 ["max_error_", "class_go_1_1ft_smooth_surf.html#a5948eda4719c97b821472899d2cc6182", null],
 ["maxiter_", "class_go_1_1ft_smooth_surf.html#a1e0ff1517a1e42f0cdd01d3165b22f4b", null],
 ["mean_error_", "class_go_1_1ft_smooth_surf.html#a90a9c488ef8fffbcaf7c2d34333a7ecc", null],
 ["orig_surf_", "class_go_1_1ft_smooth_surf.html#a99cfc4c9accec4aa03bb3a1d080c0d734", null],
 ["seem_", "class_go_1_1ft_smooth_surf.html#abab3501bbd777ad43b21717703a4e23d", null],
 ["surf_", "class_go_1_1ft_smooth_surf.html#aee983c68219bd47b0ec83b0250d4cc2d", null]
]
```

Definition at line 1 of file class\_go\_1\_1ft\_smooth\_surf.js.

## 30.457 doc/html/class\_go\_1\_1ft\_surface.js File Reference

### Variables

- var [class\\_go\\_1\\_1ft\\_surface](#)

### 30.457.1 Variable Documentation

#### 30.457.1.1 var class\_go\_1\_1ft\_surface

Definition at line 1 of file class\_go\_1\_1ft\_surface.js.

## 30.458 doc/html/class\_go\_1\_1ft\_surface\_set.js File Reference

### Variables

- var [class\\_go\\_1\\_1ft\\_surface\\_set](#)

### 30.458.1 Variable Documentation

#### 30.458.1.1 var class\_go\_1\_1ft\_surface\_set

Definition at line 1 of file class\_go\_1\_1ft\_surface\_set.js.

## 30.459 doc/html/class\_go\_1\_1ft\_surface\_set\_point.js File Reference

### Variables

- var [class\\_go\\_1\\_1ft\\_surface\\_set\\_point](#)

### 30.459.1 Variable Documentation

#### 30.459.1.1 var class\_go\_1\_1ft\_surface\_set\_point

#### Initial value:

```
=
[
 ["ftSurfaceSetPoint", "class_go_1_1ft_surface_set_point.html#a2c11652111741e4486588b26b22dab9c", null],
 ["ftSurfaceSetPoint", "class_go_1_1ft_surface_set_point.html#a7abfd754aaba91e4c10a8a7472ea1381", null],
 ["~ftSurfaceSetPoint", "class_go_1_1ft_surface_set_point.html#af662d8194c31ecf162b0e7375bc7002c", null],
 ["addFace", "class_go_1_1ft_surface_set_point.html#a11a3c2b1185b37518ad982f5c3188718", null],
 ["addInfo", "class_go_1_1ft_surface_set_point.html#a5f827e634b8d8987e283882d4279fbc9", null],
 ["addPair", "class_go_1_1ft_surface_set_point.html#abc20be3c9c4ba403d5efc118117ba142", null],
 ["asSurfaceSetPoint", "class_go_1_1ft_surface_set_point.html#a1e0b31c2c6eca592a7e8002e6bc449c2", null],
 ["containsFace", "class_go_1_1ft_surface_set_point.html#a6f694c0103e18c8dd988e3fef15747bc", null],
 ["face", "class_go_1_1ft_surface_set_point.html#a882e43b4b07c97b22f9979ab5d0099ee", null],
 ["getPar", "class_go_1_1ft_surface_set_point.html#ae36505aa31d4df12cd9068298de1114e", null],
 ["nmbFaces", "class_go_1_1ft_surface_set_point.html#acd99898cd40e2fc60fb59c682d555eca", null],
 ["parValue", "class_go_1_1ft_surface_set_point.html#a6a49adffdf46d2768d353c7931c0fa93", null],
 ["resetPosition", "class_go_1_1ft_surface_set_point.html#a96763241dcfcbb9e309ab41f34410ce2", null],
 ["resetPosition", "class_go_1_1ft_surface_set_point.html#af8bbb1ec6824ade183e3dc7b999fc767b", null],
 ["write2Dval", "class_go_1_1ft_surface_set_point.html#a92b64340d0c99eb8523bf9dca4777ee9", null]
]
```

Definition at line 1 of file class\_go\_1\_1ft\_surface\_set\_point.js.

## 30.460 doc/html/class\_go\_1\_1ft\_volume.js File Reference

### Variables

- var [class\\_go\\_1\\_1ft\\_volume](#)

### 30.460.1 Variable Documentation

#### 30.460.1.1 var class\_go\_1\_1ft\_volume

Definition at line 1 of file class\_go\_1\_1ft\_volume.js.

## 30.461 doc/html/class\_go\_1\_1tp\_edge.js File Reference

### Variables

- var [class\\_go\\_1\\_1tp\\_edge](#)

### 30.461.1 Variable Documentation

#### 30.461.1.1 var class\_go\_1\_1tp\_edge

Definition at line 1 of file class\_go\_1\_1tp\_edge.js.

## 30.462 doc/html/class\_go\_1\_1tp\_face.js File Reference

### Variables

- var [class\\_go\\_1\\_1tp\\_face](#)

### 30.462.1 Variable Documentation

#### 30.462.1.1 var class\_go\_1\_1tp\_face

#### Initial value:

```
=
[
 ["~tpFace", "class_go_1_1tp_face.html#a29bcd236ece78bef73e6364517b8c112", null],
 ["boundingBox", "class_go_1_1tp_face.html#a0c64958bd2e7f5d129034b40ef280e16", null],
 ["createInitialEdges", "class_go_1_1tp_face.html#a7945efcd0158e5b183282e0625ac8da5", null],
 ["getId", "class_go_1_1tp_face.html#a1c77809ad89ac49d234bf1c4c904cec2", null],
 ["isolateFace", "class_go_1_1tp_face.html#a1cbd052eab1bd604c192967c8e517636", null],
 ["normal", "class_go_1_1tp_face.html#a3a926d8b81c1a61332ac7eaad6436e06", null],
 ["point", "class_go_1_1tp_face.html#a921bf8edf4ad997f7a890946f90985be", null],
 ["startEdges", "class_go_1_1tp_face.html#a7767aca4a185fbb8ca05bfab02cbb235", null]
]
```

Definition at line 1 of file class\_go\_1\_1tp\_face.js.



## 30.463 doc/html/class\_go\_1\_1tp\_march\_point.js File Reference

### Variables

- var [class\\_go\\_1\\_1tp\\_march\\_point](#)

#### 30.463.1 Variable Documentation

##### 30.463.1.1 var class\_go\_1\_1tp\_march\_point

###### Initial value:

```
=
[
 ["tpMarchPoint", "class_go_1_1tp_march_point.html#a67adf9a6e441ed8caeb02dd152cca489", null],
 ["operator<", "class_go_1_1tp_march_point.html#aa16a43a59721a561a2fcb8d9bde5b82c", null],
 ["cos_ang", "class_go_1_1tp_march_point.html#a6b22182dc006480534a965d063c06aae", null],
 ["dist", "class_go_1_1tp_march_point.html#a660a6dc7deb2c0e9579a018fed56397e", null],
 ["par", "class_go_1_1tp_march_point.html#a1c1248c51b31a2f75f55529ddb1396e7", null],
 ["par2", "class_go_1_1tp_march_point.html#ae2df4e93f348b6b4c60a5e2c2231c2ef", null],
 ["pt", "class_go_1_1tp_march_point.html#ade241004d5a81a0c4d4a50cfac161bc4", null],
 ["status", "class_go_1_1tp_march_point.html#a18526aa481bb814dddabb5f172a3295b", null]
]
```

Definition at line 1 of file [class\\_go\\_1\\_1tp\\_march\\_point.js](#).

## 30.464 doc/html/class\_go\_1\_1tp\_table\_entry.js File Reference

### Variables

- var [class\\_go\\_1\\_1tp\\_table\\_entry](#)

#### 30.464.1 Variable Documentation

##### 30.464.1.1 var class\_go\_1\_1tp\_table\_entry

###### Initial value:

```
=
[
 ["tpTableEntry", "class_go_1_1tp_table_entry.html#ad0felab4b188650b6ac1892dc08d98d6", null],
 ["Info", "class_go_1_1tp_table_entry.html#a1cd7aa94340381cc43831c095bb9f8aa", null],
 ["e1_", "class_go_1_1tp_table_entry.html#ab2e9e30e315ec2fd05d5d37114296884", null],
 ["e2_", "class_go_1_1tp_table_entry.html#a2e23e6a84effa6180933fe8e6fa6e61c", null],
 ["info_", "class_go_1_1tp_table_entry.html#ad64c99903a46aeb303baac34b8a3d9a9", null]
]
```

Definition at line 1 of file [class\\_go\\_1\\_1tp\\_table\\_entry.js](#).

## 30.465 doc/html/class\_go\_1\_1tp\_topology\_table.js File Reference

### Variables

- var [class\\_go\\_1\\_1tp\\_topology\\_table](#)

### 30.465.1 Variable Documentation

#### 30.465.1.1 var class\_go\_1\_1tp\_topology\_table

##### Initial value:

```
=
[
 ["tpTopologyTable", "class_go_1_1tp_topology_table.html#a280f476f704632649026f9416ea9ed72", null],
 ["~tpTopologyTable", "class_go_1_1tp_topology_table.html#ad0f0c163cd7ca7534e75cd982999e7c9", null],
 ["addEntry", "class_go_1_1tp_topology_table.html#aedde77944ab89619f33224b60f8d34df", null],
 ["addFaceToTable", "class_go_1_1tp_topology_table.html#a9ce5f349cd350f63afef12c05a1e2083", null],
 ["BoundaryLoops", "class_go_1_1tp_topology_table.html#af23c20fcd4e757515f4349346ffa5e", null],
 ["BoundaryLoops", "class_go_1_1tp_topology_table.html#a7f8860c61aac68b28cde19466a039adc", null],
 ["connectedEdges", "class_go_1_1tp_topology_table.html#a1220adebcb4f84d74b54cd9f81b33824", null],
 ["connectTwins", "class_go_1_1tp_topology_table.html#adfc4e616b827771b00196f0f26ce12f2", null],
 ["constructTable", "class_go_1_1tp_topology_table.html#a449c512f37a6b24f76c67f5cc52adb40", null],
 ["cornersAndKinks", "class_go_1_1tp_topology_table.html#aed1c391542da8504be6e8a89492da8cb", null],
 ["disjointObjects", "class_go_1_1tp_topology_table.html#af7486be8ef6128f36b4f310b501ea352", null],
 ["edgesBoundingFace", "class_go_1_1tp_topology_table.html#ae258c9c944e48754cf7016a89d726e8c", null],
 ["getInconsistentFaces", "class_go_1_1tp_topology_table.html#aad161c1c0118ca0e226537303b0abcc0", null],
],
 ["getTolerances", "class_go_1_1tp_topology_table.html#ac616042148c3047428ec3b21ee8babf3", null],
 ["info", "class_go_1_1tp_topology_table.html#ac0b1051546e2a5c3062ffcc0a6f1161c", null],
 ["prepareTable", "class_go_1_1tp_topology_table.html#a09af231da32b51ecc65f96366fcea852", null],
 ["removeFaceFromTable", "class_go_1_1tp_topology_table.html#a10fb7df54b8706366a71d25ef9762d3a", null]
],
 ["setTable", "class_go_1_1tp_topology_table.html#ade56576beb580a2a6dd35516ae76a59f", null],
 ["setTolerances", "class_go_1_1tp_topology_table.html#ablcb76026a20dc775539f77bc01221b7", null],
 ["updateTableEntry", "class_go_1_1tp_topology_table.html#a6f183699fa5757ade0cfb85af22d062f", null],
 ["Validate", "class_go_1_1tp_topology_table.html#a82cf7ad03299185589e0ac28bf6c30a6", null],
 ["edges_", "class_go_1_1tp_topology_table.html#abf3e86281c1bdb12fda93bf752f8923e", null],
 ["orientation_inconsist_", "class_go_1_1tp_topology_table.html#af611ebf3e740eef65964c53e827cb463", null],
],
 ["table_", "class_go_1_1tp_topology_table.html#a144bdf017ad7970a1a1d85386652fbd7", null],
 ["tol_", "class_go_1_1tp_topology_table.html#af6ae926cb5d60909655fcc5fd63c0c2f", null]
]
```

Definition at line 1 of file class\_go\_1\_1tp\_topology\_table.js.

## 30.466 doc/html/class\_go\_1\_1ttl\_point.js File Reference

### Variables

- var [class\\_go\\_1\\_1ttl\\_point](#)

### 30.466.1 Variable Documentation

#### 30.466.1.1 var class\_go\_1\_1ttl\_point

##### Initial value:

```
=
[
 ["ttlPoint", "class_go_1_1ttl_point.html#afb76122074f0ccee549b3falbca716b25", null],
 ["~ttlPoint", "class_go_1_1ttl_point.html#a484d35c0299a097f038f3a85b5ec5199", null],
 ["pnt_iter", "class_go_1_1ttl_point.html#a1aa8e7179388f1852041da7fa4e2da8b", null],
 ["x", "class_go_1_1ttl_point.html#a2edfd5b3a2a64c8c3af3e02c06ae2357", null],
 ["y", "class_go_1_1ttl_point.html#aa68070e89bc646943dcff514e3d194f", null],
 ["z", "class_go_1_1ttl_point.html#ac3852f5d65cea6fc03ded418c52b1224", null]
]
```

Definition at line 1 of file class\_go\_1\_1ttl\_point.js.

## 30.467 doc/html/class\_handle.js File Reference

### Variables

- var [class\\_handle](#)

### 30.467.1 Variable Documentation

#### 30.467.1.1 var class\_handle

##### Initial value:

```
=
[
 ["Handle", "class_handle.html#a6a72028918adf79c0ff8d9996e5e4107", null],
 ["Handle", "class_handle.html#a1b0a3e0e3449918fac3bf223a70b8709", null],
 ["Handle", "class_handle.html#ad2cab6ea66970629a381c70738b0b472", null],
 ["Handle", "class_handle.html#a9cabb5daacd62808eb3d3dd059a6fe2e", null],
 ["~Handle", "class_handle.html#af44781eaf3bf4a8dc43c03bbffb6a99b", null],
 ["getPtr", "class_handle.html#ab4ab29f577789c0577d1b44faf6a8f3f", null],
 ["getPtr", "class_handle.html#a904517eada1393456c257bba79f8ee31", null],
 ["getRef", "class_handle.html#aa2b0e477fbab690dc33e4f8751235a54", null],
 ["getRef", "class_handle.html#aa0d40f86fb91881263135e468c500c61", null],
 ["operator!=", "class_handle.html#ab0daceecdfdb7a3c6a986ec4e44c1bef", null],
 ["operator()", "class_handle.html#ae240e5153753350da4c7e12bdca5606e", null],
 ["operator()", "class_handle.html#a71a32d9971601ddc7370f1487b1b0b9f", null],
 ["operator*", "class_handle.html#a400771b1a292b9cc986458a17fdd56a4", null],
 ["operator*", "class_handle.html#aa363f58481614502dd4c81b850689dfe", null],
 ["operator->", "class_handle.html#a8b1e1184484b93dbb43bd2718f09b36a", null],
 ["operator->", "class_handle.html#a86614303c99dd2616bec91f67d09c648", null],
 ["operator<", "class_handle.html#a2f06bad9eb8d12daa63e9ab0d22589d2", null],
 ["operator=", "class_handle.html#a90a09410b859358597334c0e8faa0e72", null],
 ["operator=", "class_handle.html#aa393552601be7cb81793a0633d2b74ed", null],
 ["operator=", "class_handle.html#adc4a85d1badceeb701e514681212e6c9", null],
 ["operator==", "class_handle.html#a091df1aff90d6308007bbd7729ef61b8", null],
 ["operator>", "class_handle.html#a0b7895dd362d162168e6fe1aa0315fc7", null],
 ["rebind", "class_handle.html#ad185ab655996c9f4f4d3c6d60b82f2ab", null],
 ["rebind", "class_handle.html#a150c54fa8c15a76a6566c844ef73e5962", null],
 ["classptr", "class_handle.html#adaead7076a54db6aff6cd20c3d5effa7", null]
]
```

Definition at line 1 of file class\_handle.js.

## 30.468 doc/html/class\_handle\_id.js File Reference

### Variables

- var [class\\_handle\\_id](#)

### 30.468.1 Variable Documentation

#### 30.468.1.1 var class\_handle\_id

##### Initial value:

```
=
[
 ["HandleId", "class_handle_id.html#a01fddf4a8dc585f3e586995abbaf9c18", null],
 ["~HandleId", "class_handle_id.html#aa51e63467689ea379db024af0afb232", null],
 ["decrement", "class_handle_id.html#a60c877566273852f57db0c287e8e20e9", null],
 ["dynamicObj", "class_handle_id.html#a731b580ced4cf492c117d400845ffa9e", null],
 ["getNoRefs", "class_handle_id.html#a7056b61e814da1d6efdaa829abb41113", null],
 ["increment", "class_handle_id.html#a6203b5c7c6abb05b2f29c8867957ed91", null],
 ["isReferenced", "class_handle_id.html#abb43ecc44cbdee2b224ffac88ac93fff", null],
 ["operator delete", "class_handle_id.html#a93674a6e3b006910a7622fb342963c93", null],
 ["operator new", "class_handle_id.html#a35338ac3a826d0b9932787918adb92f6", null],
 ["operator new", "class_handle_id.html#a63c4e34f17f25e7ba5197e81449d39e4", null],
 ["dynamic_object", "class_handle_id.html#a6f50f94d066f1e7ac940446a4a57040e", null],
 ["refcount", "class_handle_id.html#aaafccc3f78cf1734743a9c0a4c85e731d", null]
]
```

Definition at line 1 of file class\_handle\_id.js.

## 30.469 doc/html/class\_heap\_node.js File Reference

### Variables

- var [class\\_heap\\_node](#)

### 30.469.1 Variable Documentation

#### 30.469.1.1 var class\_heap\_node

##### Initial value:

```
=
[
 ["HeapNode", "class_heap_node.html#a3a3dad2d61930cf4c4c45594419958ed", null],
 ["HeapNode", "class_heap_node.html#a1e2d06f1f13c75ccf3431521a8c4fd42", null],
 ["~HeapNode", "class_heap_node.html#ad9a3c1e2d881e7834a1bb7027e40f38c", null],
 ["operator>", "class_heap_node.html#ae5cd253973c2925786d7e3574a0d756c", null],
 ["operator>=", "class_heap_node.html#a41be61494c745036fc31c52128fd0ab5", null],
 ["idx_", "class_heap_node.html#aba2e8f78c366d1cc212d27d717603b72", null],
 ["key_", "class_heap_node.html#ae07076e227f6a24b9a70d2dff7031110", null]
]
```

Definition at line 1 of file class\_heap\_node.js.

## 30.470 doc/html/class\_heap\_node2.js File Reference

### Variables

- var [class\\_heap\\_node2](#)

### 30.470.1 Variable Documentation

#### 30.470.1.1 var class\_heap\_node2

##### Initial value:

```
=
[
 ["HeapNode2", "class_heap_node2.html#a6b8d602cb46fe0701e3c6dab995b051e", null],
 ["HeapNode2", "class_heap_node2.html#a5b22b052c82980a29916634dbf753c47", null],
 ["~HeapNode2", "class_heap_node2.html#a08d62ecd298e35b71c6e627bf627ab6d", null],
 ["operator>", "class_heap_node2.html#a7b164608b4c4a2d4f6139ffcc26340d1", null],
 ["operator>=", "class_heap_node2.html#a743db5dba90e1dc9633b4a7ef8209d12", null],
 ["idx_", "class_heap_node2.html#a5ebe9e5b913a025e8f4c5438bdba056a", null],
 ["key_", "class_heap_node2.html#a5834899269c09b811d1f2cf807e41e22", null]
]
```

Definition at line 1 of file class\_heap\_node2.js.

## 30.471 doc/html/class\_load\_and\_store\_flag.js File Reference

### Variables

- var [class\\_load\\_and\\_store\\_flag](#)

#### 30.471.1 Variable Documentation

##### 30.471.1.1 var class\_load\_and\_store\_flag

#### Initial value:

```
=
[
 ["LoadAndStoreFlag", "class_load_and_store_flag.html#ae2c278f4da8c79b1201f529862b5a09c", null],
 ["LoadAndStoreFlag", "class_load_and_store_flag.html#ab66e21ad7c510154322ec490200a768d", null],
 ["LoadAndStoreFlag", "class_load_and_store_flag.html#a917efd271950913a6e326b7cd8c4276a", null],
 ["LoadAndStoreFlag", "class_load_and_store_flag.html#a57a4a0a3ca90e13866d0424b54289782", null]
]
```

Definition at line 1 of file class\_load\_and\_store\_flag.js.

## 30.472 doc/html/class\_matrix\_col.js File Reference

### Variables

- var [class\\_matrix\\_col](#)

#### 30.472.1 Variable Documentation

##### 30.472.1.1 var class\_matrix\_col

#### Initial value:

```
=
[
 ["MatrixCol", "class_matrix_col.html#a7c8e045a6e5659da6874a6cc7e645414", null],
 ["MatrixCol", "class_matrix_col.html#a602517d82b0fc10f8dee909aa74df436", null],
 ["~MatrixCol", "class_matrix_col.html#a9961882559950e092da0d22500ed0e5a", null],
 ["Next", "class_matrix_col.html#aca56b4e4e476005d501c210627563969", null]
]
```

Definition at line 1 of file class\_matrix\_col.js.

## 30.473 doc/html/class\_matrix\_col\_x.js File Reference

### Variables

- var [class\\_matrix\\_col\\_x](#)

### 30.473.1 Variable Documentation

#### 30.473.1.1 var class\_matrix\_col\_x

**Initial value:**

```
=
[
 ["MatrixColX", "class_matrix_col_x.html#ab3094151ba6d460736e03c9fcaf55ac1", null],
 ["~MatrixColX", "class_matrix_col_x.html#a237c59fd0549030b0f46b013a9963421", null],
 ["Next", "class_matrix_col_x.html#a1d23ac481679b1c07896ce31f9cc5e34", null],
 ["store", "class_matrix_col_x.html#aa0cd295e552adc80f7dba9be15640a04", null]
]
```

Definition at line 1 of file class\_matrix\_col\_x.js.

## 30.474 doc/html/class\_matrix\_row.js File Reference

### Variables

- var [class\\_matrix\\_row](#)

### 30.474.1 Variable Documentation

#### 30.474.1.1 var class\_matrix\_row

**Initial value:**

```
=
[
 ["MatrixRow", "class_matrix_row.html#a0bee1c69c70ae9b6314509c3ff0f20b8", null],
 ["~MatrixRow", "class_matrix_row.html#a8e9d1e2ef5f0ca9de9d4a77c239c0d5c", null],
 ["Next", "class_matrix_row.html#a733500292e7c2a65299c3096f21e7331", null]
]
```

Definition at line 1 of file class\_matrix\_row.js.

## 30.475 doc/html/class\_matrix\_row\_col.js File Reference

### Variables

- var [class\\_matrix\\_row\\_col](#)

### 30.475.1 Variable Documentation

#### 30.475.1.1 var class\_matrix\_row\_col

Definition at line 1 of file class\_matrix\_row\_col.js.

## 30.476 doc/html/class\_model\_\_3pe.js File Reference

### Variables

- var [class\\_model\\_\\_3pe](#)

### 30.476.1 Variable Documentation

#### 30.476.1.1 var class\_model\_\_3pe

#### Initial value:

```
=
[
 ["Model_3pe", "class_model__3pe.html#ae8bf015434dee0b6b5a81839c269af37", null],
 ["Derivatives", "class_model__3pe.html#a2b665c06c033d60364ebc4d14ed3f05b", null],
 ["IsValid", "class_model__3pe.html#a89ed75f468bf008647f1fd1ff8af607d", null],
 ["operator()", "class_model__3pe.html#aa6364b4b037f1abd64077b25746280f2", null]
]
```

Definition at line 1 of file class\_model\_\_3pe.js.

## 30.477 doc/html/class\_multi\_dijkstra.js File Reference

### Variables

- var [class\\_multi\\_dijkstra](#)

### 30.477.1 Variable Documentation

#### 30.477.1.1 var class\_multi\_dijkstra

#### Initial value:

```
=
[
 ["MultiDijkstra", "class_multi_dijkstra.html#a172dde104dc4c23e159334898f60fac7", null],
 ["<MultiDijkstra", "class_multi_dijkstra.html#ad642ce853461abb1a07876bd0a733fef", null],
 ["closestNeighbour", "class_multi_dijkstra.html#aa6a3d03f9862ec3dcb2495d863db42a2", null],
 ["getDistance", "class_multi_dijkstra.html#a169d72adef61b6040ab0968c3d916964", null],
 ["getDistance", "class_multi_dijkstra.html#a1be7065940136798e36b3f2e17ccfb5b", null],
 ["getLabel", "class_multi_dijkstra.html#a763ccc7047d5d74a0023a6edc136b722", null],
 ["initialize", "class_multi_dijkstra.html#abfa5931197be1bd6d8f7715e9630147e", null],
 ["insertInCandidates", "class_multi_dijkstra.html#ab60bfddd5d54180172d09618dd42b9f", null],
 ["isFinished", "class_multi_dijkstra.html#a54908b8519dc81257fb12f3950620767", null],
 ["run", "class_multi_dijkstra.html#a3a7ea26538adblabc6debb41d0c9ab53", null],
 ["run", "class_multi_dijkstra.html#a574042b10007e564f3b9926ed0dec49d", null],
 ["run", "class_multi_dijkstra.html#a1b75b2ecd04a85ceab11d3aa231a4011", null],
 ["setFinished", "class_multi_dijkstra.html#a6222cb416af690c90259720c01c9b048", null],
 ["setGraph", "class_multi_dijkstra.html#a3e658fe99ddb8e3d51ced0dfbbb268ba", null],
 ["setSource", "class_multi_dijkstra.html#ac6e50d5f99fbb3c9d1dae327d10012a5", null],
 ["back_trace", "class_multi_dijkstra.html#ad4aba15e5b77c17e32dbe6e0b6cb2ad8", null],
 ["distances_", "class_multi_dijkstra.html#aa3d7557171ee639aa6c84e93b25e4c9a", null],
 ["flags_", "class_multi_dijkstra.html#a0d6da2fa60a1854c199275fc1218e63f", null],
 ["graph", "class_multi_dijkstra.html#a01cba4633b91a539cb4b663016db6d70", null],
 ["label_", "class_multi_dijkstra.html#a7e2736e3c6d24b2a5b451c5d6f4b02ff", null],
 ["large_distance_", "class_multi_dijkstra.html#a4cc650703fd03dae2d075afb06563c2f", null],
 ["queue_", "class_multi_dijkstra.html#a0c4bdac62aa04c86ef015a9c2c3a5507", null]
]
```

Definition at line 1 of file class\_multi\_dijkstra.js.

## 30.478 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_added\_matrix.js File Reference

### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1\\_added\\_matrix](#)

### 30.478.1 Variable Documentation

#### 30.478.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_added\_matrix

#### Initial value:

```
=
[
 ["AddedMatrix", "class_n_e_w_m_a_t_1_1_added_matrix.html#ad77e556ba16299946e8e98782c83c150", null],
 ["~AddedMatrix", "class_n_e_w_m_a_t_1_1_added_matrix.html#aef827a1b165d1eeae87b6b5b5224f186", null],
 ["BandWidth", "class_n_e_w_m_a_t_1_1_added_matrix.html#a1a06e50e5fb74e5ee21315551294ad82", null],
 ["Evaluate", "class_n_e_w_m_a_t_1_1_added_matrix.html#a51cbf3c7f1f820b935368b28528362e0", null],
 ["BaseMatrix", "class_n_e_w_m_a_t_1_1_added_matrix.html#afcc50b811bd45d547cae53941018fdde", null],
 ["GeneralMatrix", "class_n_e_w_m_a_t_1_1_added_matrix.html#abaf0969f44c36896ac567a558a330771", null],
 ["GenericMatrix", "class_n_e_w_m_a_t_1_1_added_matrix.html#a4ecabb4ff1be9d3dd3fd895b25fd7237", null]
]
```

Definition at line 1 of file `class_n_e_w_m_a_t_1_1_added_matrix.js`.

## 30.479 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_array\_length\_specifier.js File Reference

### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1\\_array\\_length\\_specifier](#)

### 30.479.1 Variable Documentation

#### 30.479.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_array\_length\_specifier

#### Initial value:

```
=
[
 ["ArrayLengthSpecifier", "
 class_n_e_w_m_a_t_1_1_array_length_specifier.html#a3c09f07ca24b7e83ba5d0c13d5359ff0", null],
 ["Value", "class_n_e_w_m_a_t_1_1_array_length_specifier.html#a4cf0a24f1c538cfc06ab285933919abb", null
]
]
```

Definition at line 1 of file `class_n_e_w_m_a_t_1_1_array_length_specifier.js`.

## 30.480 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_band\_l\_u\_matrix.js File Reference

### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1\\_band\\_l\\_u\\_matrix](#)



## 30.480.1 Variable Documentation

### 30.480.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_band\_l\_u\_matrix

#### Initial value:

```
=
[
 ["BandLUMatrix", "class_n_e_w_m_a_t_1_1_band_l_u_matrix.html#abf1a3f2bf633e9c5e1b3d9ab13899ebf", null],
 ["~BandLUMatrix", "class_n_e_w_m_a_t_1_1_band_l_u_matrix.html#a59e4522041a5d8d1b52dbf670da9c092", null],
 ["CleanUp", "class_n_e_w_m_a_t_1_1_band_l_u_matrix.html#a47551f47c5cd3fcd019c033d669904f", null],
 ["GetCol", "class_n_e_w_m_a_t_1_1_band_l_u_matrix.html#ac10b8111c08a815effebb7e65af05542", null],
 ["GetCol", "class_n_e_w_m_a_t_1_1_band_l_u_matrix.html#a9a6126cdebd40da1d56e65f1a4035e15", null],
 ["GetRow", "class_n_e_w_m_a_t_1_1_band_l_u_matrix.html#a7035e511da75d3028f776c8ec89ba66c", null],
 ["IsEqual", "class_n_e_w_m_a_t_1_1_band_l_u_matrix.html#a23b14d1c78863bf424f70d18c242827d", null],
 ["IsSingular", "class_n_e_w_m_a_t_1_1_band_l_u_matrix.html#aba4bc7c5b518dc1f454cf3f59a26b2ec", null],
 ["LogDeterminant", "class_n_e_w_m_a_t_1_1_band_l_u_matrix.html#a17eed4e0e740ff224d12d50f1e1cbfc8", null],
 ["lubksb", "class_n_e_w_m_a_t_1_1_band_l_u_matrix.html#aa6b6344001665e9241bd7a3e6564c7da", null],
 ["MakeSolver", "class_n_e_w_m_a_t_1_1_band_l_u_matrix.html#a3700705a64f6cd1752dea6155b5075c8", null],
 ["operator=", "class_n_e_w_m_a_t_1_1_band_l_u_matrix.html#a87d2ce9e16b90fee44b387af9b032cc7", null],
 ["operator=", "class_n_e_w_m_a_t_1_1_band_l_u_matrix.html#ad45e86466e3969ec2cdabea010901804", null],
 ["Solver", "class_n_e_w_m_a_t_1_1_band_l_u_matrix.html#a38d4bb7d84c9c536e1606e9e2164ad11", null],
 ["Type", "class_n_e_w_m_a_t_1_1_band_l_u_matrix.html#ad7d8038615cc14e3582ec69f9be90b42", null]
]
```

Definition at line 1 of file class\_n\_e\_w\_m\_a\_t\_1\_1\_band\_l\_u\_matrix.js.

## 30.481 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_band\_matrix.js File Reference

### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1\\_band\\_matrix](#)

### 30.481.1 Variable Documentation

#### 30.481.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_band\_matrix

Definition at line 1 of file class\_n\_e\_w\_m\_a\_t\_1\_1\_band\_matrix.js.

## 30.482 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_base\_matrix.js File Reference

### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1\\_base\\_matrix](#)

### 30.482.1 Variable Documentation

#### 30.482.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_base\_matrix

Definition at line 1 of file class\_n\_e\_w\_m\_a\_t\_1\_1\_base\_matrix.js.

## 30.483 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_cannot\_build\_exception.js File Reference

### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1\\_cannot\\_build\\_exception](#)

### 30.483.1 Variable Documentation

#### 30.483.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_cannot\_build\_exception

##### Initial value:

```
=
[
 ["CannotBuildException", "
 class_n_e_w_m_a_t_1_1_cannot_build_exception.html#abd05302f4155f571d4efe275fae63992", null]
]
```

Definition at line 1 of file class\_n\_e\_w\_m\_a\_t\_1\_1\_cannot\_build\_exception.js.

## 30.484 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_coled\_matrix.js File Reference

### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1\\_coled\\_matrix](#)

### 30.484.1 Variable Documentation

#### 30.484.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_coled\_matrix

##### Initial value:

```
=
[
 ["~ColedMatrix", "class_n_e_w_m_a_t_1_1_coled_matrix.html#af183979b1260e8395a26275083c4ce03", null],
 ["BandWidth", "class_n_e_w_m_a_t_1_1_coled_matrix.html#a48ff151a8dd9ccc0c3cc13c427b53b41", null],
 ["Evaluate", "class_n_e_w_m_a_t_1_1_coled_matrix.html#a7f3964e2bf5f19e7928936baeb5d7ea6", null],
 ["BaseMatrix", "class_n_e_w_m_a_t_1_1_coled_matrix.html#afcc50b811bd45d547cae53941018fdde", null]
]
```

Definition at line 1 of file class\_n\_e\_w\_m\_a\_t\_1\_1\_coled\_matrix.js.

## 30.485 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_column\_vector.js File Reference

### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1\\_column\\_vector](#)

### 30.485.1 Variable Documentation

#### 30.485.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_column\_vector

##### Initial value:

```
=
[
 ["ColumnVector", "class_n_e_w_m_a_t_1_1_column_vector.html#a5b0c165a9ecee3313dbc0255579e0e8e", null],
 ["~ColumnVector", "class_n_e_w_m_a_t_1_1_column_vector.html#ab57241ece59a6dfcbe0cd1e7a7fc42a6", null]
],
["ColumnVector", "class_n_e_w_m_a_t_1_1_column_vector.html#a625731389b21c037ea40516b2009fa30", null],
["ColumnVector", "class_n_e_w_m_a_t_1_1_column_vector.html#a7dcc6e2dce014f0be26ff94e79c33994", null],
["ColumnVector", "class_n_e_w_m_a_t_1_1_column_vector.html#a23e4d4faf5af6441fd7203e90c308288", null],
["Cleanup", "class_n_e_w_m_a_t_1_1_column_vector.html#a6627ff0f7aeeccf701cd9e9b374b4223a", null],
["element", "class_n_e_w_m_a_t_1_1_column_vector.html#a3c8bb596dd3700fceb988f59373f9c86", null],
["element", "class_n_e_w_m_a_t_1_1_column_vector.html#a5aeff3a1f6ee88ea872c940dc682ea28", null],
["nric", "class_n_e_w_m_a_t_1_1_column_vector.html#ad47280b00e105c22a9e698d6488c114c", null],
["operator()", "class_n_e_w_m_a_t_1_1_column_vector.html#ac3b7731645747d6753f7eb5887bb44a3", null],
["operator()", "class_n_e_w_m_a_t_1_1_column_vector.html#a7bae66e57d02ee96c2c295f173e5157c", null],
["operator=", "class_n_e_w_m_a_t_1_1_column_vector.html#a31570d20b88a3c3dc6ebc433d9db4417", null],
["operator=", "class_n_e_w_m_a_t_1_1_column_vector.html#a0d2252fb6cd4a2129fc26e9b4c9b4760", null],
["operator=", "class_n_e_w_m_a_t_1_1_column_vector.html#add4db6d278e6d0984906ad46c33a9841", null],
["ReSize", "class_n_e_w_m_a_t_1_1_column_vector.html#afc4ca58241ef5581837ce5f2c9a6f244", null],
["ReSize", "class_n_e_w_m_a_t_1_1_column_vector.html#a9ce10dcae7e68a4da583c869c84368ed", null],
["ReSize", "class_n_e_w_m_a_t_1_1_column_vector.html#aa041a520e6a2aee426fbc4ee4fbbf670", null],
["Transpose", "class_n_e_w_m_a_t_1_1_column_vector.html#a5aed881c8b3c5eec69a3fd190705be41", null],
["Type", "class_n_e_w_m_a_t_1_1_column_vector.html#a8d495b637032fc7b05e39fe69ae58c05", null]
]
```

Definition at line 1 of file class\_n\_e\_w\_m\_a\_t\_1\_1\_column\_vector.js.

## 30.486 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_concatenated\_matrix.js File Reference

### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1\\_concatenated\\_matrix](#)

### 30.486.1 Variable Documentation

#### 30.486.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_concatenated\_matrix

##### Initial value:

```
=
[
 ["ConcatenatedMatrix", "
class_n_e_w_m_a_t_1_1_concatenated_matrix.html#ab81c83f6bc125d9dad45eadd24f8a209", null],
 ["~ConcatenatedMatrix", "
class_n_e_w_m_a_t_1_1_concatenated_matrix.html#a3e7816e53c7b7b9187189f903cab1034", null],
 ["BandWidth", "class_n_e_w_m_a_t_1_1_concatenated_matrix.html#aaac39d73963b04599e75aa47909ee0cd", null
],
 ["Evaluate", "class_n_e_w_m_a_t_1_1_concatenated_matrix.html#a7002dlad0e96c63d585fef8c1a174c21", null
],
 ["BaseMatrix", "class_n_e_w_m_a_t_1_1_concatenated_matrix.html#afcc50b811bd45d547cae53941018fdde",
null],
 ["GeneralMatrix", "class_n_e_w_m_a_t_1_1_concatenated_matrix.html#abaf0969f44c36896ac567a558a330771",
null],
 ["GenericMatrix", "class_n_e_w_m_a_t_1_1_concatenated_matrix.html#a4ecabb4ff1be9d3dd3fd895b25fd7237",
null]
]
```

Definition at line 1 of file class\_n\_e\_w\_m\_a\_t\_1\_1\_concatenated\_matrix.js.

## 30.487 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_convergence\_exception.js File Reference

### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1\\_convergence\\_exception](#)

### 30.487.1 Variable Documentation

#### 30.487.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_convergence\_exception

##### Initial value:

```
=
[
 ["ConvergenceException", "
 class_n_e_w_m_a_t_1_1_convergence_exception.html#a5e4003ffd88f6878ee2d06bbe30ad0a4", null],
 ["ConvergenceException", "
 class_n_e_w_m_a_t_1_1_convergence_exception.html#a5336d432ba1ba5ad7a6b87c0bf0ce8b1", null]
]
```

Definition at line 1 of file class\_n\_e\_w\_m\_a\_t\_1\_1\_convergence\_exception.js.

## 30.488 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_crout\_matrix.js File Reference

### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1\\_crout\\_matrix](#)

### 30.488.1 Variable Documentation

#### 30.488.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_crout\_matrix

##### Initial value:

```
=
[
 ["CroutMatrix", "class_n_e_w_m_a_t_1_1_crout_matrix.html#a9d2bb8a8375726f2244d0286a3ec4ee9", null],
 ["~CroutMatrix", "class_n_e_w_m_a_t_1_1_crout_matrix.html#a0d8066614076ec97635f90695a05f05a", null],
 ["CleanUp", "class_n_e_w_m_a_t_1_1_crout_matrix.html#a3812c09942872971d5b94f4c18e2c165", null],
 ["GetCol", "class_n_e_w_m_a_t_1_1_crout_matrix.html#a2685489fcbcd281a7a6cf824d0c342ad", null],
 ["GetCol", "class_n_e_w_m_a_t_1_1_crout_matrix.html#add3b9113e948eaa102ffd8adba9fa98e", null],
 ["GetRow", "class_n_e_w_m_a_t_1_1_crout_matrix.html#a82cc652216aa77871e6ca337db55664d", null],
 ["IsEqual", "class_n_e_w_m_a_t_1_1_crout_matrix.html#ac1cbbc4791df15e19b82337497e28e92", null],
 ["IsSingular", "class_n_e_w_m_a_t_1_1_crout_matrix.html#a7316146e8f084c57bea844f6e431ca7d", null],
 ["LogDeterminant", "class_n_e_w_m_a_t_1_1_crout_matrix.html#ae8538150bbdb49778b728a9ef7e71031", null]
],
 ["lubksb", "class_n_e_w_m_a_t_1_1_crout_matrix.html#afcfb1bcd9b105d4be4f13b9dc85a0f98", null],
 ["MakeSolver", "class_n_e_w_m_a_t_1_1_crout_matrix.html#a4280df1fcceea0e6283b1f8ed4302cd1", null],
 ["operator=", "class_n_e_w_m_a_t_1_1_crout_matrix.html#aba89f95906ae140fe016ddb507bdc02f", null],
 ["operator=", "class_n_e_w_m_a_t_1_1_crout_matrix.html#a52a6aeec80d85af469ae2f3512bdfd7f", null],
 ["Solver", "class_n_e_w_m_a_t_1_1_crout_matrix.html#a408949e52d007455d1737e31edfa53c0", null],
 ["Type", "class_n_e_w_m_a_t_1_1_crout_matrix.html#a52a29d6a758743dda05cb8be7d3dfb9b", null]
]
```

Definition at line 1 of file class\_n\_e\_w\_m\_a\_t\_1\_1\_crout\_matrix.js.

## 30.489 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_diaged\_matrix.js File Reference

### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1\\_diaged\\_matrix](#)

#### 30.489.1 Variable Documentation

##### 30.489.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_diaged\_matrix

#### Initial value:

```
=
[
 ["~DiagedMatrix", "class_n_e_w_m_a_t_1_1_diaged_matrix.html#aec0265288aa9e33ed7b7827b0226ad5e", null],
 ["BandWidth", "class_n_e_w_m_a_t_1_1_diaged_matrix.html#ad7a7e9f78927564a819f4374b7d8fb50", null],
 ["Evaluate", "class_n_e_w_m_a_t_1_1_diaged_matrix.html#a7d6342a05708d740d7ea5975b002a863", null],
 ["BaseMatrix", "class_n_e_w_m_a_t_1_1_diaged_matrix.html#afcc50b811bd45d547cae53941018fdde", null]
]
```

Definition at line 1 of file class\_n\_e\_w\_m\_a\_t\_1\_1\_diaged\_matrix.js.

## 30.490 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_diagonal\_matrix.js File Reference

### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1\\_diagonal\\_matrix](#)

#### 30.490.1 Variable Documentation

##### 30.490.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_diagonal\_matrix

Definition at line 1 of file class\_n\_e\_w\_m\_a\_t\_1\_1\_diagonal\_matrix.js.

## 30.491 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_find\_maximum2.js File Reference

### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1\\_find\\_maximum2](#)

### 30.491.1 Variable Documentation

#### 30.491.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_find\_maximum2

**Initial value:**

```
=
[
 ["~FindMaximum2", "class_n_e_w_m_a_t_1_1_find_maximum2.html#a5b8f707cdf9b42a853d248097c361b23", null]
 ["Fit", "class_n_e_w_m_a_t_1_1_find_maximum2.html#ac535f364bf2417435b4abaf042669b84", null]
]
```

Definition at line 1 of file class\_n\_e\_w\_m\_a\_t\_1\_1\_find\_maximum2.js.

## 30.492 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_general\_matrix.js File Reference

### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1\\_general\\_matrix](#)

### 30.492.1 Variable Documentation

#### 30.492.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_general\_matrix

Definition at line 1 of file class\_n\_e\_w\_m\_a\_t\_1\_1\_general\_matrix.js.

## 30.493 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_generic\_matrix.js File Reference

### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1\\_generic\\_matrix](#)

### 30.493.1 Variable Documentation

#### 30.493.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_generic\_matrix

**Initial value:**

```

=
[
 ["GenericMatrix", "class_n_e_w_m_a_t_1_1_generic_matrix.html#a19947b73823ee9150213f5f3bcbd24c8", null],
 ["GenericMatrix", "class_n_e_w_m_a_t_1_1_generic_matrix.html#a45820078a0784bcalacada47f4ad0c85", null],
 ["GenericMatrix", "class_n_e_w_m_a_t_1_1_generic_matrix.html#a14afe7f8860aa9fd0d8a906d240fc5c4", null],
 ["~GenericMatrix", "class_n_e_w_m_a_t_1_1_generic_matrix.html#abe3a629cf585506d54d6331223968b3d", null],
 ["BandWidth", "class_n_e_w_m_a_t_1_1_generic_matrix.html#afe1f02c324cd2201997d3054c2a36ad4", null],
 ["CleanUp", "class_n_e_w_m_a_t_1_1_generic_matrix.html#a4f7bf7d205291b933078a96bd4a40580", null],
 ["Evaluate", "class_n_e_w_m_a_t_1_1_generic_matrix.html#a7108ff51fc81ef4ced62b3dd8e5980f0", null],
 ["operator&=", "class_n_e_w_m_a_t_1_1_generic_matrix.html#a8743f0f1e522261f4497bdf135ab820a", null],
 ["operator*=", "class_n_e_w_m_a_t_1_1_generic_matrix.html#a72ddaee7647faa7e47873f308511c485", null],
 ["operator**=", "class_n_e_w_m_a_t_1_1_generic_matrix.html#a16d0d84d9e614394146f6a54312078a8", null],
 ["operator+=", "class_n_e_w_m_a_t_1_1_generic_matrix.html#abe00f1f180355b1af2ff46769bcb2d36", null],
 ["operator+=", "class_n_e_w_m_a_t_1_1_generic_matrix.html#a0d01ea669d8deb1fe34f822a28f2cd81", null],
 ["operator--=", "class_n_e_w_m_a_t_1_1_generic_matrix.html#ae26ce111d0d75bba35cd555215b651de", null],
 ["operator-=", "class_n_e_w_m_a_t_1_1_generic_matrix.html#aae6694a5ba6428ca75c5c44cd22a6c96", null],
 ["operator/=", "class_n_e_w_m_a_t_1_1_generic_matrix.html#aaac78f3ba609a6d4a393ec3ff251a448", null],
 ["operator=", "class_n_e_w_m_a_t_1_1_generic_matrix.html#a44c5a34e643d475ff1f3409f7017c893", null],
 ["operator=", "class_n_e_w_m_a_t_1_1_generic_matrix.html#a88ee4c0dae3036bf45110f4ac1549fd5", null],
 ["operator!=", "class_n_e_w_m_a_t_1_1_generic_matrix.html#a416fcfe8e834866abc39b82433fec2d2", null],
 ["Release", "class_n_e_w_m_a_t_1_1_generic_matrix.html#a998692fc87fa7101da13f10473ce89f8", null],
 ["BaseMatrix", "class_n_e_w_m_a_t_1_1_generic_matrix.html#afcc50b811bd45d547cae53941018fdde", null]
]

```

Definition at line 1 of file class\_n\_e\_w\_m\_a\_t\_1\_1\_generic\_matrix.js.

## 30.494 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_get\_sub\_matrix.js File Reference

### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1\\_get\\_sub\\_matrix](#)

### 30.494.1 Variable Documentation

#### 30.494.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_get\_sub\_matrix

#### Initial value:

```

=
[
 ["GetSubMatrix", "class_n_e_w_m_a_t_1_1_get_sub_matrix.html#a0766a66aa428c36c724f8a36b9dead7d", null],
 ["~GetSubMatrix", "class_n_e_w_m_a_t_1_1_get_sub_matrix.html#a8a417b260c066d14d5fe4a307dfd000a", null],
 ["BandWidth", "class_n_e_w_m_a_t_1_1_get_sub_matrix.html#acf67fbbe9f8638abf617c340f93ae315", null],
 ["Evaluate", "class_n_e_w_m_a_t_1_1_get_sub_matrix.html#a0e4401bdc043ca76cf8a6d2757ab155b", null],
 ["Inject", "class_n_e_w_m_a_t_1_1_get_sub_matrix.html#aeb4da21c26be6132051fc8b393de405a", null],
 ["operator**=", "class_n_e_w_m_a_t_1_1_get_sub_matrix.html#abdd7893f548ee765fb2672f0064f42d7", null],
 ["operator+=", "class_n_e_w_m_a_t_1_1_get_sub_matrix.html#a41c915cc444e70c3cd63ef61cf755f8a", null],
 ["operator+=", "class_n_e_w_m_a_t_1_1_get_sub_matrix.html#a98d05b3640c8d6e3429a855ca81db699", null],
 ["operator--=", "class_n_e_w_m_a_t_1_1_get_sub_matrix.html#a5001d48272ad2d2b8bc538be1736fa85", null],
 ["operator-=", "class_n_e_w_m_a_t_1_1_get_sub_matrix.html#aecdd7669168ec6743f7378e11b4f5551", null],
 ["operator/=", "class_n_e_w_m_a_t_1_1_get_sub_matrix.html#a8d2114ebf339c20e20f2a8846589d50d", null],
 ["operator=", "class_n_e_w_m_a_t_1_1_get_sub_matrix.html#a6d25816fa55b2f04e3b06554516f84c6", null],
 ["operator<<=", "class_n_e_w_m_a_t_1_1_get_sub_matrix.html#ad4350f3a2a9aac7311900ada841dc6cd", null],
 ["operator<<=", "class_n_e_w_m_a_t_1_1_get_sub_matrix.html#aed28ef23d97aaa205607c5fb29e46f23", null],
 ["operator<<=", "class_n_e_w_m_a_t_1_1_get_sub_matrix.html#a659b0c1b17d6b52afb3ea25f175a8b5a", null],
 ["operator=", "class_n_e_w_m_a_t_1_1_get_sub_matrix.html#a15f489f0c5828e3ffdd7d36e00b43231", null],
 ["operator=", "class_n_e_w_m_a_t_1_1_get_sub_matrix.html#a0b45e7db37f29b77ec2d7d168288ae22", null],
 ["operator=", "class_n_e_w_m_a_t_1_1_get_sub_matrix.html#a7f528208d6f7fe11e59ddb6b6f203ca", null],
 ["BaseMatrix", "class_n_e_w_m_a_t_1_1_get_sub_matrix.html#afcc50b811bd45d547cae53941018fdde", null]
]

```

Definition at line 1 of file class\_n\_e\_w\_m\_a\_t\_1\_1\_get\_sub\_matrix.js.

## 30.495 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_identity\_matrix.js File Reference

### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1\\_identity\\_matrix](#)

### 30.495.1 Variable Documentation

#### 30.495.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_identity\_matrix

#### Initial value:

```
=
[
 ["IdentityMatrix", "class_n_e_w_m_a_t_1_1_identity_matrix.html#acd9124ff1b21a69afd415a9e68d073b8",
 null],
 ["~IdentityMatrix", "class_n_e_w_m_a_t_1_1_identity_matrix.html#afe7cae0a323c04f609859e3ce0ba35a6",
 null],
 ["IdentityMatrix", "class_n_e_w_m_a_t_1_1_identity_matrix.html#ad9a24dff9b08f41cda062438666d942f",
 null],
 ["IdentityMatrix", "class_n_e_w_m_a_t_1_1_identity_matrix.html#a4f3706dc33b285d3a8706b945a4dfc17",
 null],
 ["IdentityMatrix", "class_n_e_w_m_a_t_1_1_identity_matrix.html#aeed6ff0975f66d11b1ce9f2464e10b32",
 null],
 ["BandWidth", "class_n_e_w_m_a_t_1_1_identity_matrix.html#aac5be3a3284b7eb8b1a2509e587c7676", null],
 ["GetCol", "class_n_e_w_m_a_t_1_1_identity_matrix.html#a07576967a3a0862fc7d72f78b0206bfb", null],
 ["GetCol", "class_n_e_w_m_a_t_1_1_identity_matrix.html#aa66481cff64f3252d68ee10750e85622", null],
 ["GetRow", "class_n_e_w_m_a_t_1_1_identity_matrix.html#a08b3ea293ddb4a1e0f7a0001667ebd30", null],
 ["LogDeterminant", "class_n_e_w_m_a_t_1_1_identity_matrix.html#a7974e44f060885229d4b40a290f07d1c",
 null],
 ["MakeSolver", "class_n_e_w_m_a_t_1_1_identity_matrix.html#a7a70a662fc02229693131978889c3310", null],
 ["NextCol", "class_n_e_w_m_a_t_1_1_identity_matrix.html#adb2537c21d81ef7ff6f4b8478cc1a027", null],
 ["NextCol", "class_n_e_w_m_a_t_1_1_identity_matrix.html#a8295eb1e6130ab1207e09f3f610214b4", null],
 ["NextRow", "class_n_e_w_m_a_t_1_1_identity_matrix.html#ae6ca2f1dac32d5e2dd357c9206b7bb95", null],
 ["operator=", "class_n_e_w_m_a_t_1_1_identity_matrix.html#a3c6539e1c255ac0e64abc254e07ac9f9", null],
 ["operator=", "class_n_e_w_m_a_t_1_1_identity_matrix.html#a5f742b07e107232fb77c431d0fa5d0f1", null],
 ["ReSize", "class_n_e_w_m_a_t_1_1_identity_matrix.html#a1a5b0ca4f3123cc1802fa603c506d233", null],
 ["ReSize", "class_n_e_w_m_a_t_1_1_identity_matrix.html#a9ccf3675e789c6ad8f4f89c7879f058b", null],
 ["Solver", "class_n_e_w_m_a_t_1_1_identity_matrix.html#a7b10efb4ab2a0c4989963f2d2aff542d", null],
 ["Sum", "class_n_e_w_m_a_t_1_1_identity_matrix.html#a3fd6ffc96fcec64e7f3821b0e7049287", null],
 ["SumAbsoluteValue", "class_n_e_w_m_a_t_1_1_identity_matrix.html#a886ad1bab9224eb18bf87892677da50e",
 null],
 ["SumSquare", "class_n_e_w_m_a_t_1_1_identity_matrix.html#aa82c67bf82f8ee120ecbc30bed16f4cd", null],
 ["Trace", "class_n_e_w_m_a_t_1_1_identity_matrix.html#a23a058aa00a07177f2cae30583c819e", null],
 ["Transpose", "class_n_e_w_m_a_t_1_1_identity_matrix.html#a903a3c67e3ce707a932a18669ae73e1", null],
 ["Type", "class_n_e_w_m_a_t_1_1_identity_matrix.html#a220e75cc95126af4658e247e2b1851db", null]
]
```

Definition at line 1 of file `class_n_e_w_m_a_t_1_1_identity_matrix.js`.

## 30.496 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_incompatible\_dimensions\_exception.js File Reference

### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1\\_incompatible\\_dimensions\\_exception](#)



### 30.496.1 Variable Documentation

#### 30.496.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_incompatible\_dimensions\_exception

**Initial value:**

```
=
[
 ["IncompatibleDimensionsException", "
 class_n_e_w_m_a_t_1_1_incompatible_dimensions_exception.html#aa9fc3f81c3c2f311bad5f19a859e0225", null],
 ["IncompatibleDimensionsException", "
 class_n_e_w_m_a_t_1_1_incompatible_dimensions_exception.html#a649572ecbc15d9b72f6b6b4ea611c57c", null]
]
```

Definition at line 1 of file class\_n\_e\_w\_m\_a\_t\_1\_1\_incompatible\_dimensions\_exception.js.

## 30.497 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_index\_exception.js File Reference

### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1\\_index\\_exception](#)

### 30.497.1 Variable Documentation

#### 30.497.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_index\_exception

**Initial value:**

```
=
[
 ["IndexException", "class_n_e_w_m_a_t_1_1_index_exception.html#a68214876072adca2b9dfc8cb8aba3d49",
 null],
 ["IndexException", "class_n_e_w_m_a_t_1_1_index_exception.html#a4118806c61a30e461d1d78f007143ee0",
 null],
 ["IndexException", "class_n_e_w_m_a_t_1_1_index_exception.html#a1146753e80d758e8a229a1b886f71eaf",
 null],
 ["IndexException", "class_n_e_w_m_a_t_1_1_index_exception.html#a03c22303fcd82829f86cefb1186b114d",
 null]
]
```

Definition at line 1 of file class\_n\_e\_w\_m\_a\_t\_1\_1\_index\_exception.js.

## 30.498 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_internal\_exception.js File Reference

### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1\\_internal\\_exception](#)

### 30.498.1 Variable Documentation

#### 30.498.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_internal\_exception

##### Initial value:

```
=
[
 ["InternalException", "class_n_e_w_m_a_t_1_1_internal_exception.html#a12b6e89c3b1d9b5a64d10f3398192033", null]
]
```

Definition at line 1 of file class\_n\_e\_w\_m\_a\_t\_1\_1\_internal\_exception.js.

## 30.499 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_inverted\_matrix.js File Reference

### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1\\_inverted\\_matrix](#)

### 30.499.1 Variable Documentation

#### 30.499.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_inverted\_matrix

##### Initial value:

```
=
[
 ["~InvertedMatrix", "class_n_e_w_m_a_t_1_1_inverted_matrix.html#a3af182a6d83d0d1f61ab4d527f90cbaa", null],
 ["BandWidth", "class_n_e_w_m_a_t_1_1_inverted_matrix.html#ab127880731fd9c2e5202fbd15df3122a", null],
 ["Evaluate", "class_n_e_w_m_a_t_1_1_inverted_matrix.html#abce7668a90eeb91b60683baddc932c3a", null],
 ["operator*", "class_n_e_w_m_a_t_1_1_inverted_matrix.html#afd48b81d0392f67fbc803b52ee3d9bc9", null],
 ["operator*", "class_n_e_w_m_a_t_1_1_inverted_matrix.html#ada3589f22bec18dc2302742332b35409", null],
 ["BaseMatrix", "class_n_e_w_m_a_t_1_1_inverted_matrix.html#afcc50b811bd45d547cae53941018fdde", null]
]
```

Definition at line 1 of file class\_n\_e\_w\_m\_a\_t\_1\_1\_inverted\_matrix.js.

## 30.500 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_k\_p\_matrix.js File Reference

### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1\\_k\\_p\\_matrix](#)

### 30.500.1 Variable Documentation

#### 30.500.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_k\_p\_matrix

##### Initial value:

```
=
[
 ["KPMatrix", "class_n_e_w_m_a_t_1_1_k_p_matrix.html#ac3bc1829dd4dff55fbb7bb77c231a393", null],
 ["~KPMatrix", "class_n_e_w_m_a_t_1_1_k_p_matrix.html#ae8f939b0eb657d86754a43f7d394203f", null],
 ["BandWidth", "class_n_e_w_m_a_t_1_1_k_p_matrix.html#a709cda05547fd9eb8bf3335dc4b22026", null],
 ["Evaluate", "class_n_e_w_m_a_t_1_1_k_p_matrix.html#a17e0256578b704146c777df48356b81a", null],
 ["BaseMatrix", "class_n_e_w_m_a_t_1_1_k_p_matrix.html#afcc50b811bd45d547cae53941018fdde", null],
 ["GeneralMatrix", "class_n_e_w_m_a_t_1_1_k_p_matrix.html#abaf0969f44c36896ac567a558a330771", null],
 ["GenericMatrix", "class_n_e_w_m_a_t_1_1_k_p_matrix.html#a4ecabb4ff1be9d3dd3fd895b25fd7237", null],
 ["KP", "class_n_e_w_m_a_t_1_1_k_p_matrix.html#a7b84f8e43bac4608bf225867306c9450", null]
]
```

Definition at line 1 of file class\_n\_e\_w\_m\_a\_t\_1\_1\_k\_p\_matrix.js.

## 30.501 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_l\_l\_d\_f\_i.js File Reference

### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1\\_l\\_l\\_d\\_f\\_i](#)

### 30.501.1 Variable Documentation

#### 30.501.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_l\_l\_d\_f\_i

##### Initial value:

```
=
[
 ["~LL_D_FI", "class_n_e_w_m_a_t_1_1_l_l_d_f_i.html#ab965296fff06ea79e2da67a63020c780", null],
 ["Derivatives", "class_n_e_w_m_a_t_1_1_l_l_d_f_i.html#adf4f3d247ce14423a4a56cc71208e2f8", null],
 ["FI", "class_n_e_w_m_a_t_1_1_l_l_d_f_i.html#ac3705dc4e71588f1f783a96ee298a414", null],
 ["IsValid", "class_n_e_w_m_a_t_1_1_l_l_d_f_i.html#a00363cd032220a81358b7174f2054277", null],
 ["IsValid", "class_n_e_w_m_a_t_1_1_l_l_d_f_i.html#ad966a18cf175f6a21c06ecc6f6183d66", null],
 ["LogLikelihood", "class_n_e_w_m_a_t_1_1_l_l_d_f_i.html#aa6f46c203df838a8aa42aeb60a526e2a", null]
 ,
 ["LogLikelihood", "class_n_e_w_m_a_t_1_1_l_l_d_f_i.html#ae42e54f6ff22e50475877585f3fee8e2", null]
 ,
 ["Set", "class_n_e_w_m_a_t_1_1_l_l_d_f_i.html#a9fe13f8c5df42e74475218d512ff8e57", null],
 ["WG", "class_n_e_w_m_a_t_1_1_l_l_d_f_i.html#ae6376b9b2bed94af9ca3f16abcf7926b", null],
 ["para", "class_n_e_w_m_a_t_1_1_l_l_d_f_i.html#afac709c2cc3f61d8c8f25fc34a2bf0f0", null],
 ["wg", "class_n_e_w_m_a_t_1_1_l_l_d_f_i.html#a49099740d783df4fff02c19cf63d2dd0", null]
]
```

Definition at line 1 of file class\_n\_e\_w\_m\_a\_t\_1\_1\_l\_l\_d\_f\_i.js.

## 30.502 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_linear\_equation\_solver.js File Reference

### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1\\_linear\\_equation\\_solver](#)

### 30.502.1 Variable Documentation

#### 30.502.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_linear\_equation\_solver

##### Initial value:

```
=
[
 ["LinearEquationSolver", "
 class_n_e_w_m_a_t_1_1_linear_equation_solver.html#a82d651f3a49527923663e1e3df357dcf", null],
 ["~LinearEquationSolver", "
 class_n_e_w_m_a_t_1_1_linear_equation_solver.html#ac745945f382f7fb07a88738c40d274cc", null],
 ["CleanUp", "class_n_e_w_m_a_t_1_1_linear_equation_solver.html#aeba950921f42308c3edd4ae0bfa20cc3",
 null],
 ["Evaluate", "class_n_e_w_m_a_t_1_1_linear_equation_solver.html#aa8d4e2567a0316d998d7e946e67ce30a",
 null],
 ["BaseMatrix", "class_n_e_w_m_a_t_1_1_linear_equation_solver.html#afcc50b811bd45d547cae53941018fdde",
 null]
]
```

Definition at line 1 of file class\_n\_e\_w\_m\_a\_t\_1\_1\_linear\_equation\_solver.js.

## 30.503 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_log\_and\_sign.js File Reference

### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1\\_log\\_and\\_sign](#)

### 30.503.1 Variable Documentation

#### 30.503.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_log\_and\_sign

##### Initial value:

```
=
[
 ["LogAndSign", "class_n_e_w_m_a_t_1_1_log_and_sign.html#ac42e0b1df7255dec4978506ac3df4cd8", null],
 ["LogAndSign", "class_n_e_w_m_a_t_1_1_log_and_sign.html#ae15f7eb74654dd7ab77a8ed36d750808", null],
 ["ChangeSign", "class_n_e_w_m_a_t_1_1_log_and_sign.html#a73ebcfb9d621135b52de9d6854b6fa4b", null],
 ["LogValue", "class_n_e_w_m_a_t_1_1_log_and_sign.html#a6a39d90313933b380017ba59a6918a98", null],
 ["operator*=", "class_n_e_w_m_a_t_1_1_log_and_sign.html#a66c387cba687d87053c1e6197f226773", null],
 ["PowEq", "class_n_e_w_m_a_t_1_1_log_and_sign.html#a48c7e57757033d2c4f8b01d652897d68", null],
 ["Sign", "class_n_e_w_m_a_t_1_1_log_and_sign.html#a0b02fdbb4f4b7dca9541bc3b39895937", null],
 ["Value", "class_n_e_w_m_a_t_1_1_log_and_sign.html#a4f61ca76c5ce21da7cf88b7e158b9edd", null]
]
```

Definition at line 1 of file class\_n\_e\_w\_m\_a\_t\_1\_1\_log\_and\_sign.js.

## 30.504 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_lower\_band\_matrix.js File Reference

### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1\\_lower\\_band\\_matrix](#)

### 30.504.1 Variable Documentation

#### 30.504.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_lower\_band\_matrix

##### Initial value:

```
=
[
 ["LowerBandMatrix", "class_n_e_w_m_a_t_1_1_lower_band_matrix.html#a12e243740c28b360091ee25e07113e27",
 null],
 ["~LowerBandMatrix", "class_n_e_w_m_a_t_1_1_lower_band_matrix.html#ae578calf8e46ab77e2db35182d56b8f4",
 null],
 ["LowerBandMatrix", "class_n_e_w_m_a_t_1_1_lower_band_matrix.html#a92141d3195b42b665372521d1aefd5d1",
 null],
 ["LowerBandMatrix", "class_n_e_w_m_a_t_1_1_lower_band_matrix.html#a121f58ef4aa72767ac16c3f98b25487c",
 null],
 ["LowerBandMatrix", "class_n_e_w_m_a_t_1_1_lower_band_matrix.html#a6ad80552fbcd29aff0040506db8f576e",
 null],
 ["element", "class_n_e_w_m_a_t_1_1_lower_band_matrix.html#a3f9adfe5aa9ece9e0d7e1bb4f3021795", null],
 ["element", "class_n_e_w_m_a_t_1_1_lower_band_matrix.html#a7750a8015c8ba15bfc791c939f484e22", null],
 ["LogDeterminant", "class_n_e_w_m_a_t_1_1_lower_band_matrix.html#aaf9bf5640a81963818e43c698a490d75",
 null],
 ["MakeSolver", "class_n_e_w_m_a_t_1_1_lower_band_matrix.html#a2038b006aa59bce1428e6c332e2087b0", null
],
 ["operator()", "class_n_e_w_m_a_t_1_1_lower_band_matrix.html#a10cd2a784a238ae526c18fd996e4201b", null
],
 ["operator()", "class_n_e_w_m_a_t_1_1_lower_band_matrix.html#a8418ae7db16357fac6dbab51362119ce", null
],
 ["operator=", "class_n_e_w_m_a_t_1_1_lower_band_matrix.html#a9c75ae6e1c94879737d81954ee6d2f4e", null]
 ,
 ["operator=", "class_n_e_w_m_a_t_1_1_lower_band_matrix.html#ad2023669751f223f226034dc5d5b64fc", null]
 ,
 ["operator=", "class_n_e_w_m_a_t_1_1_lower_band_matrix.html#a100f05b1071da1777cc0c7b2cb93452d", null]
 ,
 ["ReSize", "class_n_e_w_m_a_t_1_1_lower_band_matrix.html#ad43a907dela77e742bf71096e9ff86de", null],
 ["ReSize", "class_n_e_w_m_a_t_1_1_lower_band_matrix.html#a445e12ff06f55f06ef01a252f9c65187", null],
 ["ReSize", "class_n_e_w_m_a_t_1_1_lower_band_matrix.html#a55038bb353ceda56220dcd9b701dd6a7", null],
 ["Solver", "class_n_e_w_m_a_t_1_1_lower_band_matrix.html#ac05534ef98fa35e6bf036c300f8fe16b", null],
 ["Type", "class_n_e_w_m_a_t_1_1_lower_band_matrix.html#abd0b6523049546535d2d04ceeba4e599", null]
]
```

Definition at line 1 of file class\_n\_e\_w\_m\_a\_t\_1\_1\_lower\_band\_matrix.js.

## 30.505 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_lower\_triangular\_matrix.js File Reference

### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1\\_lower\\_triangular\\_matrix](#)

### 30.505.1 Variable Documentation

#### 30.505.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_lower\_triangular\_matrix

##### Initial value:

```

=
[
 ["LowerTriangularMatrix", "
 class_n_e_w_m_a_t_1_1_lower_triangular_matrix.html#a56952dae68203bc29e46d8b9f3b83a5d", null],
 ["~LowerTriangularMatrix", "
 class_n_e_w_m_a_t_1_1_lower_triangular_matrix.html#af2ecf4e88d34b5171a52f8b39f1237b8", null],
 ["LowerTriangularMatrix", "
 class_n_e_w_m_a_t_1_1_lower_triangular_matrix.html#a79c9dcdd376b210bc45adaa77c977b29", null],
 ["LowerTriangularMatrix", "
 class_n_e_w_m_a_t_1_1_lower_triangular_matrix.html#a355167e691bd8d2d2b704bca14d5cbbd", null],
 ["LowerTriangularMatrix", "
 class_n_e_w_m_a_t_1_1_lower_triangular_matrix.html#a3196cc213e94a1981a0335d24d09fc49", null],
 ["BandWidth", "class_n_e_w_m_a_t_1_1_lower_triangular_matrix.html#ac013b9b6637f10717bb95a67a5f8b6f3",
 null],
 ["element", "class_n_e_w_m_a_t_1_1_lower_triangular_matrix.html#a10efe5a7596f2da5296ba91d89fd0a38",
 null],
 ["element", "class_n_e_w_m_a_t_1_1_lower_triangular_matrix.html#aa93224ff0071bbe08c506370a0562ef7",
 null],
 ["GetCol", "class_n_e_w_m_a_t_1_1_lower_triangular_matrix.html#a9b46ebb160718c0d31e2e553b0480244",
 null],
 ["GetCol", "class_n_e_w_m_a_t_1_1_lower_triangular_matrix.html#a56dd64c41b7cc8510a30e05b1df25190",
 null],
 ["GetRow", "class_n_e_w_m_a_t_1_1_lower_triangular_matrix.html#ae9fe462a3691de3526f5cc96b45bc9ca",
 null],
 ["LogDeterminant", "
 class_n_e_w_m_a_t_1_1_lower_triangular_matrix.html#addeb7782b7216f9e8b1e64eb76e6af74", null],
 ["MakeSolver", "class_n_e_w_m_a_t_1_1_lower_triangular_matrix.html#aaf15c79abca72399d73349365ba1bf63",
 null],
 ["NextRow", "class_n_e_w_m_a_t_1_1_lower_triangular_matrix.html#a024858c07ef86d76795b7d87903355b1",
 null],
 ["operator()", "class_n_e_w_m_a_t_1_1_lower_triangular_matrix.html#ae8cda30ee5c6443a986fcedc70013fc",
 null],
 ["operator()", "class_n_e_w_m_a_t_1_1_lower_triangular_matrix.html#ac8e10e826580c79cdfd1aab81e2de998",
 null],
 ["operator=", "class_n_e_w_m_a_t_1_1_lower_triangular_matrix.html#a175e694fd53e2ff4df2c0f9f60af1a32",
 null],
 ["operator=", "class_n_e_w_m_a_t_1_1_lower_triangular_matrix.html#a74b03d90ac5a95107a80efd11dec64ae",
 null],
 ["operator=", "class_n_e_w_m_a_t_1_1_lower_triangular_matrix.html#a37ff031c03dd819a1db6b4f07baf70c9",
 null],
 ["ReSize", "class_n_e_w_m_a_t_1_1_lower_triangular_matrix.html#ala96cec059fc60a14028a778575bef44",
 null],
 ["ReSize", "class_n_e_w_m_a_t_1_1_lower_triangular_matrix.html#a42fac429a1c6dda428f3ee1123152605",
 null],
 ["RestoreCol", "class_n_e_w_m_a_t_1_1_lower_triangular_matrix.html#a5dc8cb06cf924013d7bfa95dacfe9ce9",
 null],
 ["RestoreCol", "class_n_e_w_m_a_t_1_1_lower_triangular_matrix.html#ab0266c190a902bcb14b462a2ecb1784b",
 null],
 ["Solver", "class_n_e_w_m_a_t_1_1_lower_triangular_matrix.html#acc2fc134ef66b3375d0dd5e90b20152e",
 null],
 ["Trace", "class_n_e_w_m_a_t_1_1_lower_triangular_matrix.html#a6e9414133bffc9eddb0b03f216b05120", null
],
 ["Type", "class_n_e_w_m_a_t_1_1_lower_triangular_matrix.html#a83de7e5c9d6dc4bb3d3655cf742b8d6a", null
]
]
]

```

Definition at line 1 of file `class_n_e_w_m_a_t_1_1_lower_triangular_matrix.js`.

## 30.506 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_m\_l\_e\_\_\_d\_\_\_f\_i.js File Reference

### Variables

- `var class_n_e_w_m_a_t_1_1_m_l_e___d___f_i`

### 30.506.1 Variable Documentation

#### 30.506.1.1 `var class_n_e_w_m_a_t_1_1_m_l_e___d___f_i`

**Initial value:**

```
=
[
 ["MLE_D_FI", "class_n_e_w_m_a_t_1_1_m_l_e___d___f_i.html#aa292fc72b7d11982aa7f3f9a23609066", null],
 ["Fit", "class_n_e_w_m_a_t_1_1_m_l_e___d___f_i.html#adf0b21c31ac54c10da3ab58d1d5b7adc", null],
 ["GetCorrelations", "class_n_e_w_m_a_t_1_1_m_l_e___d___f_i.html#a6c5d64ceee4776d6c2fb6e8836a64346",
 null],
 ["GetStandardErrors", "class_n_e_w_m_a_t_1_1_m_l_e___d___f_i.html#a739a01e754b9afb9db843cfd8deaa65db",
 null]
]
```

Definition at line 1 of file class\_n\_e\_w\_m\_a\_t\_1\_1\_m\_l\_e\_\_\_d\_\_\_f\_i.js.

## 30.507 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_mated\_matrix.js File Reference

### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1\\_mated\\_matrix](#)

#### 30.507.1 Variable Documentation

##### 30.507.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_mated\_matrix

###### Initial value:

```
=
[
 ["~MatedMatrix", "class_n_e_w_m_a_t_1_1_mated_matrix.html#ac3bbe550857e1e51eed5487a0779fbf3", null],
 ["BandWidth", "class_n_e_w_m_a_t_1_1_mated_matrix.html#a0f939899325f755b57d4ea52f711f181", null],
 ["Evaluate", "class_n_e_w_m_a_t_1_1_mated_matrix.html#a382011d95b6de199da3ad59fef2966a8", null],
 ["BaseMatrix", "class_n_e_w_m_a_t_1_1_mated_matrix.html#afcc50b811bd45d547cae53941018fdde", null]
]
```

Definition at line 1 of file class\_n\_e\_w\_m\_a\_t\_1\_1\_mated\_matrix.js.

## 30.508 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_matrix.js File Reference

### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1\\_matrix](#)

#### 30.508.1 Variable Documentation

##### 30.508.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_matrix

Definition at line 1 of file class\_n\_e\_w\_m\_a\_t\_1\_1\_matrix.js.

## 30.509 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_matrix\_band\_width.js File Reference

### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1\\_matrix\\_band\\_width](#)

### 30.509.1 Variable Documentation

#### 30.509.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_matrix\_band\_width

##### Initial value:

```
=
[
 ["MatrixBandWidth", "class_n_e_w_m_a_t_1_1_matrix_band_width.html#a4288a21bfba52a769f48f250cd3a4da1",
 null],
 ["MatrixBandWidth", "class_n_e_w_m_a_t_1_1_matrix_band_width.html#aea29765dfcbdb849cf7d1a2bab5fdbad1",
 null],
 ["Lower", "class_n_e_w_m_a_t_1_1_matrix_band_width.html#a99b3d410de03bcd35ae2d6c02c3d06c4", null],
 ["minimum", "class_n_e_w_m_a_t_1_1_matrix_band_width.html#a12e7af8c334838c29c084052f6f452eb", null],
 ["operator!=", "class_n_e_w_m_a_t_1_1_matrix_band_width.html#a2fc0f5876d7a64926aad2c77a5300c2d", null
],
 ["operator*", "class_n_e_w_m_a_t_1_1_matrix_band_width.html#aa78e571f2ff9367fee14ec20a4ebd80e", null]
 ,
 ["operator+", "class_n_e_w_m_a_t_1_1_matrix_band_width.html#aec95298228050e06bb9f7e3852e11f87", null]
 ,
 ["operator==", "class_n_e_w_m_a_t_1_1_matrix_band_width.html#a64d7e146dcb7b1db62c368109c838886", null
],
 ["t", "class_n_e_w_m_a_t_1_1_matrix_band_width.html#a9282c4ccc63ddeadc55483508c5c2794", null],
 ["Upper", "class_n_e_w_m_a_t_1_1_matrix_band_width.html#a01e7a15fda15b5a069e462824cb9927f", null],
 ["lower", "class_n_e_w_m_a_t_1_1_matrix_band_width.html#a6f001a0586946b9c25d86de8d1f0eb3e", null],
 ["upper", "class_n_e_w_m_a_t_1_1_matrix_band_width.html#a6b13bddfdaa1318d25610484b8504e74", null]
]
```

Definition at line 1 of file `class_n_e_w_m_a_t_1_1_matrix_band_width.js`.

## 30.510 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_matrix\_input.js File Reference

### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1\\_matrix\\_input](#)

### 30.510.1 Variable Documentation

#### 30.510.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_matrix\_input

##### Initial value:

```
=
[
 ["MatrixInput", "class_n_e_w_m_a_t_1_1_matrix_input.html#a187dd140a0a0eecf339adf489ece229b", null],
 ["MatrixInput", "class_n_e_w_m_a_t_1_1_matrix_input.html#ada6c50a6ea79741cb20ecf999623f26", null],
 ["~MatrixInput", "class_n_e_w_m_a_t_1_1_matrix_input.html#a47890afd4a2c2e309adaf80ce9025de8", null],
 ["operator<<", "class_n_e_w_m_a_t_1_1_matrix_input.html#adf42553daed4a107ad31c26b4063bdb0", null],
 ["operator<<", "class_n_e_w_m_a_t_1_1_matrix_input.html#ab921d2ec1f1e0ef4e958cede988769dc", null],
 ["GeneralMatrix", "class_n_e_w_m_a_t_1_1_matrix_input.html#abaf0969f44c36896ac567a558a330771", null]
]
```

Definition at line 1 of file `class_n_e_w_m_a_t_1_1_matrix_input.js`.



## 30.511 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_matrix\_type.js File Reference

### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1\\_matrix\\_type](#)

#### 30.511.1 Variable Documentation

##### 30.511.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_matrix\_type

Definition at line 1 of file class\_n\_e\_w\_m\_a\_t\_1\_1\_matrix\_type.js.

## 30.512 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_multi\_radix\_counter.js File Reference

### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1\\_multi\\_radix\\_counter](#)

#### 30.512.1 Variable Documentation

##### 30.512.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_multi\_radix\_counter

#### Initial value:

```
=
[
 ["MultiRadixCounter", "
 class_n_e_w_m_a_t_1_1_multi_radix_counter.html#a8daaaa635a2650e3368e6fba3dd0ea55", null],
 ["Counter", "class_n_e_w_m_a_t_1_1_multi_radix_counter.html#ab50c824e081ee104b74a9e7c10b25a83", null]
 ,
 ["Finish", "class_n_e_w_m_a_t_1_1_multi_radix_counter.html#a97cea9382aa4b253ab923e96a5caa25b", null],
 ["operator++", "class_n_e_w_m_a_t_1_1_multi_radix_counter.html#add6df1b99428c4aecb043f6ccb06ef89",
 null],
 ["Reverse", "class_n_e_w_m_a_t_1_1_multi_radix_counter.html#ad76063e8eef4b31e893b526c0f86cd1a", null]
 ,
 ["Swap", "class_n_e_w_m_a_t_1_1_multi_radix_counter.html#a446f0e6e72ae1b545dda4bb1296075f7", null]
]
```

Definition at line 1 of file class\_n\_e\_w\_m\_a\_t\_1\_1\_multi\_radix\_counter.js.

## 30.513 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_multiplied\_matrix.js File Reference

### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1\\_multiplied\\_matrix](#)

### 30.513.1 Variable Documentation

#### 30.513.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_multiplied\_matrix

##### Initial value:

```
=
[
 ["MultipliedMatrix", "class_n_e_w_m_a_t_1_1_multiplied_matrix.html#a76ed97f5bdbb946fc6b3d8f962d4807d",
 null],
 ["~MultipliedMatrix", "class_n_e_w_m_a_t_1_1_multiplied_matrix.html#a2cca473c4b32b6d380abee3f5c4fef04",
 null],
 ["BandWidth", "class_n_e_w_m_a_t_1_1_multiplied_matrix.html#a366ba8cef9cacd84ab385f48d0580ff", null],
 ["Evaluate", "class_n_e_w_m_a_t_1_1_multiplied_matrix.html#a399f18d0dfef4da879004ea51df54a50", null],
 ["search", "class_n_e_w_m_a_t_1_1_multiplied_matrix.html#a4e77970d04e62fdbb487b5d1fe74b041", null],
 ["BaseMatrix", "class_n_e_w_m_a_t_1_1_multiplied_matrix.html#afcc50b811bd45d547cae53941018fdde", null],
 ["GeneralMatrix", "class_n_e_w_m_a_t_1_1_multiplied_matrix.html#abaf0969f44c36896ac567a558a330771",
 null],
 ["GenericMatrix", "class_n_e_w_m_a_t_1_1_multiplied_matrix.html#a4ecabb4ff1be9d3dd3fd895b25fd7237",
 null],
 ["bm1", "class_n_e_w_m_a_t_1_1_multiplied_matrix.html#ac0fc18145168f1ed15c0e060fdb55362", null],
 ["bm2", "class_n_e_w_m_a_t_1_1_multiplied_matrix.html#a5f2f88b7d4d7853ef12bc9a602ad610d", null],
 ["gm1", "class_n_e_w_m_a_t_1_1_multiplied_matrix.html#a7803a6d673c5d8761701a9bccbe8c91d", null],
 ["gm2", "class_n_e_w_m_a_t_1_1_multiplied_matrix.html#ab52429edb38c4019ebec3accd2806115", null]
]
```

Definition at line 1 of file class\_n\_e\_w\_m\_a\_t\_1\_1\_multiplied\_matrix.js.

### 30.514 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_n\_p\_d\_exception.js File Reference

#### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1\\_n\\_p\\_d\\_exception](#)

### 30.514.1 Variable Documentation

#### 30.514.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_n\_p\_d\_exception

##### Initial value:

```
=
[
 ["NPDEException", "class_n_e_w_m_a_t_1_1_n_p_d_exception.html#ad8029f357546ebd430716d97f52c8756", null]
]
```

Definition at line 1 of file class\_n\_e\_w\_m\_a\_t\_1\_1\_n\_p\_d\_exception.js.

### 30.515 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_neg\_shifted\_matrix.js File Reference

#### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1\\_neg\\_shifted\\_matrix](#)

### 30.515.1 Variable Documentation

#### 30.515.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_neg\_shifted\_matrix

##### Initial value:

```
=
[
 ["NegShiftedMatrix", "class_n_e_w_m_a_t_1_1_neg_shifted_matrix.html#a759d2cc85daee8a6f22f9f07559fc272", null],
 ["~NegShiftedMatrix", "class_n_e_w_m_a_t_1_1_neg_shifted_matrix.html#a746ad3204ac976a759bc91ef5633ddeb", null],
 ["Evaluate", "class_n_e_w_m_a_t_1_1_neg_shifted_matrix.html#aec54c83660bd17362305d8aad991f880", null],
 ["BaseMatrix", "class_n_e_w_m_a_t_1_1_neg_shifted_matrix.html#afcc50b811bd45d547cae53941018fdde", null],
 ["GeneralMatrix", "class_n_e_w_m_a_t_1_1_neg_shifted_matrix.html#abaf0969f44c36896ac567a558a330771", null],
 ["GenericMatrix", "class_n_e_w_m_a_t_1_1_neg_shifted_matrix.html#a4ecabb4ff1be9d3dd3fd895b25fd7237", null],
 ["operator-", "class_n_e_w_m_a_t_1_1_neg_shifted_matrix.html#a12197e802fd1e5829c78adcaccb57840", null]
]
```

Definition at line 1 of file class\_n\_e\_w\_m\_a\_t\_1\_1\_neg\_shifted\_matrix.js.

## 30.516 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_negated\_matrix.js File Reference

### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1\\_negated\\_matrix](#)

### 30.516.1 Variable Documentation

#### 30.516.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_negated\_matrix

##### Initial value:

```
=
[
 ["NegatedMatrix", "class_n_e_w_m_a_t_1_1_negated_matrix.html#a701a017652a4ec29860401e3f2292a30", null],
 ["~NegatedMatrix", "class_n_e_w_m_a_t_1_1_negated_matrix.html#af06146ab7897db6bee09b3ea21c5b5a6", null],
 ["BandWidth", "class_n_e_w_m_a_t_1_1_negated_matrix.html#ad6ebbf17da3a6446ae4a8583bec19e6", null],
 ["Evaluate", "class_n_e_w_m_a_t_1_1_negated_matrix.html#aa70f58a700232f7e510eba5a8df8a88c", null],
 ["search", "class_n_e_w_m_a_t_1_1_negated_matrix.html#afd2bd1bc94cab6482b2c581f52047820", null],
 ["BaseMatrix", "class_n_e_w_m_a_t_1_1_negated_matrix.html#afcc50b811bd45d547cae53941018fdde", null],
 ["bm", "class_n_e_w_m_a_t_1_1_negated_matrix.html#a3ba0a541aad27aba78a4802c1ecc1500", null],
 ["gm", "class_n_e_w_m_a_t_1_1_negated_matrix.html#a52bb6ea0a318be83c029bfa48346884a", null]
]
```

Definition at line 1 of file class\_n\_e\_w\_m\_a\_t\_1\_1\_negated\_matrix.js.

## 30.517 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_non\_linear\_least\_squares.js File Reference

### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1\\_non\\_linear\\_least\\_squares](#)

### 30.517.1 Variable Documentation

#### 30.517.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_non\_linear\_least\_squares

##### Initial value:

```
=
[
 ["NonLinearLeastSquares", "
 class_n_e_w_m_a_t_1_1_non_linear_least_squares.html#a7cd6a0f3d290cba610457b54fa583693", null],
 ["Fit", "class_n_e_w_m_a_t_1_1_non_linear_least_squares.html#a5925a2d291a1a0c222be2760f1efbbf3", null
],
 ["GetCorrelations", "
 class_n_e_w_m_a_t_1_1_non_linear_least_squares.html#a7b5b04c531f7bc3d7d813231e2593e77", null],
 ["GetHatDiagonal", "
 class_n_e_w_m_a_t_1_1_non_linear_least_squares.html#a1e1dc84a4bb63289fcaf186d3e31b47d", null],
 ["GetResiduals", "
 class_n_e_w_m_a_t_1_1_non_linear_least_squares.html#aa5d35896e1de55d252db3a2b2cc082dc", null],
 ["GetStandardErrors", "
 class_n_e_w_m_a_t_1_1_non_linear_least_squares.html#a5d9b36b7fa0d657b53af77ed879403c5", null],
 ["ResidualVariance", "
 class_n_e_w_m_a_t_1_1_non_linear_least_squares.html#a0004dbc736ed51e72a22d279dc62ee68", null]
]
```

Definition at line 1 of file class\_n\_e\_w\_m\_a\_t\_1\_1\_non\_linear\_least\_squares.js.

## 30.518 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_not\_defined\_exception.js File Reference

### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1\\_not\\_defined\\_exception](#)

### 30.518.1 Variable Documentation

#### 30.518.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_not\_defined\_exception

##### Initial value:

```
=
[
 ["NotDefinedException", "
 class_n_e_w_m_a_t_1_1_not_defined_exception.html#a609d715cfedbf7c4fcde1aca71ec55ba", null]
]
```

Definition at line 1 of file class\_n\_e\_w\_m\_a\_t\_1\_1\_not\_defined\_exception.js.

## 30.519 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_not\_square\_exception.js File Reference

### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1\\_not\\_square\\_exception](#)

### 30.519.1 Variable Documentation

#### 30.519.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_not\_square\_exception

**Initial value:**

```
=
[
 ["NotSquareException", "
 class_n_e_w_m_a_t_1_1_not_square_exception.html#a217c00379b3896dad851ee65d0094962", null]
]
```

Definition at line 1 of file class\_n\_e\_w\_m\_a\_t\_1\_1\_not\_square\_exception.js.

## 30.520 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_overflow\_exception.js File Reference

### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1\\_overflow\\_exception](#)

### 30.520.1 Variable Documentation

#### 30.520.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_overflow\_exception

**Initial value:**

```
=
[
 ["OverflowException", "class_n_e_w_m_a_t_1_1_overflow_exception.html#ac1a3b1d77140e5c84f706fb09398fdd3
 ", null]
]
```

Definition at line 1 of file class\_n\_e\_w\_m\_a\_t\_1\_1\_overflow\_exception.js.

## 30.521 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_program\_exception.js File Reference

### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1\\_program\\_exception](#)

### 30.521.1 Variable Documentation

#### 30.521.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_program\_exception

##### Initial value:

```
=
[
 ["ProgramException", "class_n_e_w_m_a_t_1_1_program_exception.html#a2d4cea24b64fb16ee6048f4f86383e06",
 null],
 ["ProgramException", "class_n_e_w_m_a_t_1_1_program_exception.html#a942f03a341dca4cf811218611234aac1",
 null],
 ["ProgramException", "class_n_e_w_m_a_t_1_1_program_exception.html#ab82ffc9d2105d341fc8e50a13afb3ea0",
 null],
 ["ProgramException", "class_n_e_w_m_a_t_1_1_program_exception.html#a4ed405161910ff2df26975d5283cdc75",
 null],
 ["ProgramException", "class_n_e_w_m_a_t_1_1_program_exception.html#a42c8f9124591bfa8b63a46223a9f5249",
 null]
]
```

Definition at line 1 of file class\_n\_e\_w\_m\_a\_t\_1\_1\_program\_exception.js.

## 30.522 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_r1\_\_\_col\_\_\_i\_\_\_d.js File Reference

### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1\\_r1\\_\\_\\_col\\_\\_\\_i\\_\\_\\_d](#)

### 30.522.1 Variable Documentation

#### 30.522.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_r1\_\_\_col\_\_\_i\_\_\_d

##### Initial value:

```
=
[
 ["~R1_Col_I_D", "class_n_e_w_m_a_t_1_1_r1___col___i___d.html#a4b42afe9c3af905fa5b10ae2df03e270", null
],
 ["Derivatives", "class_n_e_w_m_a_t_1_1_r1___col___i___d.html#ade98cf3596655488ca5c6bb2a144d533", null
],
 ["IsValid", "class_n_e_w_m_a_t_1_1_r1___col___i___d.html#a53e337c89f29c82018fed82d26a14c0c", null],
 ["IsValid", "class_n_e_w_m_a_t_1_1_r1___col___i___d.html#aad25b0e1fa8809ae15162ee8e4c3029c", null],
 ["operator()", "class_n_e_w_m_a_t_1_1_r1___col___i___d.html#a471c68a909aefc99a2c63df3a7e3b285", null]
 ,
 ["operator()", "class_n_e_w_m_a_t_1_1_r1___col___i___d.html#a0b1da2aa2df01fd3ed4c25cfbc151131", null]
 ,
 ["Set", "class_n_e_w_m_a_t_1_1_r1___col___i___d.html#a13fb8540d785edaf4c71d3d754ff12ea", null],
 ["para", "class_n_e_w_m_a_t_1_1_r1___col___i___d.html#aa3b313d2a5b5d201c696f2a0aa54605e", null]
]
```

Definition at line 1 of file class\_n\_e\_w\_m\_a\_t\_1\_1\_r1\_\_\_col\_\_\_i\_\_\_d.js.

## 30.523 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_return\_matrix\_x.js File Reference

### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1\\_return\\_matrix\\_x](#)

### 30.523.1 Variable Documentation

#### 30.523.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_return\_matrix\_x

**Initial value:**

```
=
[
 ["~ReturnMatrixX", "class_n_e_w_m_a_t_1_1_return_matrix_x.html#a01eb5f2c346f54a15c437de09e59b85e",
 null],
 ["ReturnMatrixX", "class_n_e_w_m_a_t_1_1_return_matrix_x.html#ad7e94f23ea609d05b888e3a1564d6b8c", null
],
 ["ReturnMatrixX", "class_n_e_w_m_a_t_1_1_return_matrix_x.html#a62362f1296e466dd62139d362100337f", null
],
 ["BandWidth", "class_n_e_w_m_a_t_1_1_return_matrix_x.html#aa4efcd17c7c446df9030443099f783bd", null],
 ["Evaluate", "class_n_e_w_m_a_t_1_1_return_matrix_x.html#a844f43d6099446532b561d8ed85f48dd", null],
 ["BaseMatrix", "class_n_e_w_m_a_t_1_1_return_matrix_x.html#afcc50b811bd45d547cae53941018fdde", null]
]
```

Definition at line 1 of file class\_n\_e\_w\_m\_a\_t\_1\_1\_return\_matrix\_x.js.

## 30.524 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_reversed\_matrix.js File Reference

### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1\\_reversed\\_matrix](#)

### 30.524.1 Variable Documentation

#### 30.524.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_reversed\_matrix

**Initial value:**

```
=
[
 ["~ReversedMatrix", "class_n_e_w_m_a_t_1_1_reversed_matrix.html#a4a92bad182eea4298c73ae5ac21855bb",
 null],
 ["Evaluate", "class_n_e_w_m_a_t_1_1_reversed_matrix.html#a8435563352014a2a67095c09917458b9", null],
 ["BaseMatrix", "class_n_e_w_m_a_t_1_1_reversed_matrix.html#afcc50b811bd45d547cae53941018fdde", null]
]
```

Definition at line 1 of file class\_n\_e\_w\_m\_a\_t\_1\_1\_reversed\_matrix.js.

## 30.525 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_row\_vector.js File Reference

### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1\\_row\\_vector](#)

### 30.525.1 Variable Documentation

#### 30.525.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_row\_vector

##### Initial value:

```
=
[
 ["RowVector", "class_n_e_w_m_a_t_1_1_row_vector.html#a9701e0f58e281c59380ec555ddc62319", null],
 ["~RowVector", "class_n_e_w_m_a_t_1_1_row_vector.html#afe9b63044065492e64fce0d02a3bf07d", null],
 ["RowVector", "class_n_e_w_m_a_t_1_1_row_vector.html#a43708d03410c9440956ed8d2584b28c5", null],
 ["RowVector", "class_n_e_w_m_a_t_1_1_row_vector.html#a6af0f84382f4526f7218bfa6e345da2d", null],
 ["RowVector", "class_n_e_w_m_a_t_1_1_row_vector.html#a9d47128cc46dbcf3f45d9ee88df623f7", null],
 ["CleanUp", "class_n_e_w_m_a_t_1_1_row_vector.html#ac575c694a61bca69175d159cd811d3de", null],
 ["element", "class_n_e_w_m_a_t_1_1_row_vector.html#a2833de5351de97e2650cff8599ad2bf2", null],
 ["element", "class_n_e_w_m_a_t_1_1_row_vector.html#ac59bffd10d1995cc4d1f41eb5302c0e0", null],
 ["GetCol", "class_n_e_w_m_a_t_1_1_row_vector.html#a8c31cac109d8be98cfa5f57eab654677", null],
 ["GetCol", "class_n_e_w_m_a_t_1_1_row_vector.html#a91a8ae10baa2ddf6fdd71a10b3c562bc", null],
 ["NextCol", "class_n_e_w_m_a_t_1_1_row_vector.html#aee35f8beb1fef8e177e89789d75205b6", null],
 ["NextCol", "class_n_e_w_m_a_t_1_1_row_vector.html#a48740232e7c3e8bf5eb9edb5f0cc7e63", null],
 ["nric", "class_n_e_w_m_a_t_1_1_row_vector.html#ad10b182e3ef0b4c517ae72851cf57ca8", null],
 ["operator()", "class_n_e_w_m_a_t_1_1_row_vector.html#a1750a77091b11dbe69be40eeb9b608e3", null],
 ["operator()", "class_n_e_w_m_a_t_1_1_row_vector.html#a1d7305fe114cafef636399a8c6321b0c", null],
 ["operator=", "class_n_e_w_m_a_t_1_1_row_vector.html#afbea4642b4683e39b3306c379bbf4972", null],
 ["operator=", "class_n_e_w_m_a_t_1_1_row_vector.html#a95c22c0a9760b7e4c3cf932be914cbc7", null],
 ["operator=", "class_n_e_w_m_a_t_1_1_row_vector.html#a9b065f86d16ba0e39fcab43efc6ccb8f", null],
 ["ReSize", "class_n_e_w_m_a_t_1_1_row_vector.html#a49e0faad2f87c0f6c7dadd78ffa0bfcc", null],
 ["ReSize", "class_n_e_w_m_a_t_1_1_row_vector.html#a87c33a6b088b49d033cbfa6b4738aa36", null],
 ["ReSize", "class_n_e_w_m_a_t_1_1_row_vector.html#acec6671623eafc77e86399735e1d501e", null],
 ["RestoreCol", "class_n_e_w_m_a_t_1_1_row_vector.html#a03e1f86d7c0a282f1400aalc24a2cbaa", null],
 ["RestoreCol", "class_n_e_w_m_a_t_1_1_row_vector.html#ad0a29256653ede6b3cf7dd1d8e3dedfe", null],
 ["Transpose", "class_n_e_w_m_a_t_1_1_row_vector.html#aec387a7c45e85ac1531619e77f1ca4c2", null],
 ["Type", "class_n_e_w_m_a_t_1_1_row_vector.html#a10cc6acf559410b5c27b7166fecb85e2", null]
]
```

Definition at line 1 of file class\_n\_e\_w\_m\_a\_t\_1\_1\_row\_vector.js.

### 30.526 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_rowed\_matrix.js File Reference

#### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1\\_rowed\\_matrix](#)

### 30.526.1 Variable Documentation

#### 30.526.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_rowed\_matrix

##### Initial value:

```
=
[
 ["~RowedMatrix", "class_n_e_w_m_a_t_1_1_rowed_matrix.html#a5e8746cf82605c26fca5f16460c4a692", null],
 ["BandWidth", "class_n_e_w_m_a_t_1_1_rowed_matrix.html#a9f2e977a31bfleeda306165de5a87096", null],
 ["Evaluate", "class_n_e_w_m_a_t_1_1_rowed_matrix.html#a60efbef4a9bb00bc969acf2fedff64d2", null],
 ["BaseMatrix", "class_n_e_w_m_a_t_1_1_rowed_matrix.html#afcc50b811bd45d547cae53941018fdde", null]
]
```

Definition at line 1 of file class\_n\_e\_w\_m\_a\_t\_1\_1\_rowed\_matrix.js.



## 30.527 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_s\_p\_matrix.js File Reference

### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1\\_s\\_p\\_matrix](#)

### 30.527.1 Variable Documentation

#### 30.527.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_s\_p\_matrix

##### Initial value:

```
=
[
 ["SPMatrix", "class_n_e_w_m_a_t_1_1_s_p_matrix.html#a6035578cfd08291078376d8a4607e615", null],
 ["~SPMatrix", "class_n_e_w_m_a_t_1_1_s_p_matrix.html#a82a219d831f08d839761d645a287dc1d", null],
 ["BandWidth", "class_n_e_w_m_a_t_1_1_s_p_matrix.html#accl3e5905128c73237ccbbda8ac9a09c", null],
 ["Evaluate", "class_n_e_w_m_a_t_1_1_s_p_matrix.html#ab5c83f89a3d119fcccd89e661a062193", null],
 ["BaseMatrix", "class_n_e_w_m_a_t_1_1_s_p_matrix.html#afcc50b811bd45d547cae53941018fdde", null],
 ["GeneralMatrix", "class_n_e_w_m_a_t_1_1_s_p_matrix.html#abaf0969f44c36896ac567a558a330771", null],
 ["GenericMatrix", "class_n_e_w_m_a_t_1_1_s_p_matrix.html#a4ecabb4ff1be9d3dd3fd895b25fd7237", null],
 ["SP", "class_n_e_w_m_a_t_1_1_s_p_matrix.html#a2490eace871fe6269a3511e64cbbba7e5", null]
]
```

Definition at line 1 of file `class_n_e_w_m_a_t_1_1_s_p_matrix.js`.

## 30.528 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_scaled\_matrix.js File Reference

### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1\\_scaled\\_matrix](#)

### 30.528.1 Variable Documentation

#### 30.528.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_scaled\_matrix

##### Initial value:

```
=
[
 ["~ScaledMatrix", "class_n_e_w_m_a_t_1_1_scaled_matrix.html#a2d68eb46f02dc78c1959843947b2adf7", null]
 ,
 ["BandWidth", "class_n_e_w_m_a_t_1_1_scaled_matrix.html#a885f68b8cf7884a78c59f6f5fb162e03", null],
 ["Evaluate", "class_n_e_w_m_a_t_1_1_scaled_matrix.html#a7399ad4622d03f4ba9dbac9cc850c25b", null],
 ["BaseMatrix", "class_n_e_w_m_a_t_1_1_scaled_matrix.html#afcc50b811bd45d547cae53941018fdde", null],
 ["GeneralMatrix", "class_n_e_w_m_a_t_1_1_scaled_matrix.html#abaf0969f44c36896ac567a558a330771", null]
 ,
 ["GenericMatrix", "class_n_e_w_m_a_t_1_1_scaled_matrix.html#a4ecabb4ff1be9d3dd3fd895b25fd7237", null]
 ,
 ["operator*", "class_n_e_w_m_a_t_1_1_scaled_matrix.html#ab1d04e45670e7e54332437ab81f69f69", null]
]
```

Definition at line 1 of file `class_n_e_w_m_a_t_1_1_scaled_matrix.js`.

## 30.529 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_shifted\_matrix.js File Reference

### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1\\_shifted\\_matrix](#)

### 30.529.1 Variable Documentation

#### 30.529.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_shifted\_matrix

##### Initial value:

```
=
[
 ["ShiftedMatrix", "class_n_e_w_m_a_t_1_1_shifted_matrix.html#a56a3ac340fd61c618dd2cbc91a2bd591", null],
 ["~ShiftedMatrix", "class_n_e_w_m_a_t_1_1_shifted_matrix.html#adc048235fff757ac2fe6c2901f893465", null],
 ["Evaluate", "class_n_e_w_m_a_t_1_1_shifted_matrix.html#aa63702a58c882644182636a948e7b4df", null],
 ["search", "class_n_e_w_m_a_t_1_1_shifted_matrix.html#a0344dbc7f75d90841f6bba253ccee476", null],
 ["BaseMatrix", "class_n_e_w_m_a_t_1_1_shifted_matrix.html#afcc50b811bd45d547cae53941018fdde", null],
 ["GeneralMatrix", "class_n_e_w_m_a_t_1_1_shifted_matrix.html#abaf0969f44c36896ac567a558a330771", null],
 ["GenericMatrix", "class_n_e_w_m_a_t_1_1_shifted_matrix.html#a4ecabb4ff1be9d3dd3fd895b25fd7237", null],
 ["operator+", "class_n_e_w_m_a_t_1_1_shifted_matrix.html#afc960ce11532a76ca986ec75542bca02", null],
 ["bm", "class_n_e_w_m_a_t_1_1_shifted_matrix.html#a9e74c66bae8849d269a7ff45b649384e", null],
 ["f", "class_n_e_w_m_a_t_1_1_shifted_matrix.html#ab5a72558c082fafa0a11a708df102156", null],
 ["gm", "class_n_e_w_m_a_t_1_1_shifted_matrix.html#a19701ca94daecd6cdf5f137df145de8f", null]
]
```

Definition at line 1 of file `class_n_e_w_m_a_t_1_1_shifted_matrix.js`.

## 30.530 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_simple\_int\_array.js File Reference

### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1\\_simple\\_int\\_array](#)

### 30.530.1 Variable Documentation

#### 30.530.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_simple\_int\_array

##### Initial value:

```
=
[
 ["SimpleIntArray", "class_n_e_w_m_a_t_1_1_simple_int_array.html#a52278e08217d7e595bffb970f5208be1", null],
 ["~SimpleIntArray", "class_n_e_w_m_a_t_1_1_simple_int_array.html#a64381fcf4a22eaccf9e9d22e40785d23", null],
 ["SimpleIntArray", "class_n_e_w_m_a_t_1_1_simple_int_array.html#af70199c5bc49bf2c0fd817cc48a1d18e", null],
 ["Cleanup", "class_n_e_w_m_a_t_1_1_simple_int_array.html#aabe59095debd58dfefaae6c9ae0b9cba", null],
 ["Data", "class_n_e_w_m_a_t_1_1_simple_int_array.html#a224921d28dcde7d9131ae08d30adba79", null],
 ["Data", "class_n_e_w_m_a_t_1_1_simple_int_array.html#aa2caf98cf436c591d84b0c4ffa1f95d2", null],
 ["operator=", "class_n_e_w_m_a_t_1_1_simple_int_array.html#a238024b853b528a282dc8f93f252981e", null],
 ["operator=", "class_n_e_w_m_a_t_1_1_simple_int_array.html#a82db2ff57ad64ebc4615071c0cf9c060", null],
 ["operator[]", "class_n_e_w_m_a_t_1_1_simple_int_array.html#a870fcd80a615398c3e264b34aa3693b0", null],
 ["operator[]", "class_n_e_w_m_a_t_1_1_simple_int_array.html#afeba4cb444bef4f5a1c4d2a1a4398896", null],
 ["ReSize", "class_n_e_w_m_a_t_1_1_simple_int_array.html#ae47103ee042bb1515c32108427c0c96b", null],
 ["Size", "class_n_e_w_m_a_t_1_1_simple_int_array.html#ab3de14f18ff20c72b2bb8b43a2c7c745", null],
 ["a", "class_n_e_w_m_a_t_1_1_simple_int_array.html#af8aa7920093ee4bde2aa7dfb272cb403", null],
 ["n", "class_n_e_w_m_a_t_1_1_simple_int_array.html#a1bf5d83dbc38b5c9449fce83c9e860cd", null]
]
```

Definition at line 1 of file `class_n_e_w_m_a_t_1_1_simple_int_array.js`.

## 30.531 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_singular\_exception.js File Reference

### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1\\_singular\\_exception](#)

#### 30.531.1 Variable Documentation

##### 30.531.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_singular\_exception

###### Initial value:

```
=
[
 ["SingularException", "class_n_e_w_m_a_t_1_1_singular_exception.html#a2cc112843dfeb208e105780b5052321a", null]
]
```

Definition at line 1 of file class\_n\_e\_w\_m\_a\_t\_1\_1\_singular\_exception.js.

## 30.532 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_solved\_matrix.js File Reference

### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1\\_solved\\_matrix](#)

#### 30.532.1 Variable Documentation

##### 30.532.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_solved\_matrix

###### Initial value:

```
=
[
 ["~SolvedMatrix", "class_n_e_w_m_a_t_1_1_solved_matrix.html#ace985d37b92e55935ded5ab09553e462", null],
 ["BandWidth", "class_n_e_w_m_a_t_1_1_solved_matrix.html#a1163aa0c6804e8d46c35342124c303d1", null],
 ["Evaluate", "class_n_e_w_m_a_t_1_1_solved_matrix.html#a64da575b20101f597f6547be91517392", null],
 ["BaseMatrix", "class_n_e_w_m_a_t_1_1_solved_matrix.html#afcc50b811bd45d547cae53941018fdde", null],
 ["InvertedMatrix", "class_n_e_w_m_a_t_1_1_solved_matrix.html#ae361bc8ebdc01f511bb97ab6516e2e34", null]
]
```

Definition at line 1 of file class\_n\_e\_w\_m\_a\_t\_1\_1\_solved\_matrix.js.

## 30.533 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_stacked\_matrix.js File Reference

### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1\\_stacked\\_matrix](#)

### 30.533.1 Variable Documentation

#### 30.533.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_stacked\_matrix

##### Initial value:

```
=
[
 ["StackedMatrix", "class_n_e_w_m_a_t_1_1_stacked_matrix.html#a97b4a5bfff808fff38f423a9cb21b7e86", null
],
 ["~StackedMatrix", "class_n_e_w_m_a_t_1_1_stacked_matrix.html#a54ed4945bebe26ca97a52875230736d5", null
],
 ["Evaluate", "class_n_e_w_m_a_t_1_1_stacked_matrix.html#a96635ea72a30821f5bbc2888da015694", null],
 ["BaseMatrix", "class_n_e_w_m_a_t_1_1_stacked_matrix.html#afcc50b811bd45d547cae53941018fdde", null],
 ["GeneralMatrix", "class_n_e_w_m_a_t_1_1_stacked_matrix.html#abaf0969f44c36896ac567a558a330771", null
],
 ["GenericMatrix", "class_n_e_w_m_a_t_1_1_stacked_matrix.html#a4ecabb4ff1be9d3dd3fd895b25fd7237", null
]
]
```

Definition at line 1 of file class\_n\_e\_w\_m\_a\_t\_1\_1\_stacked\_matrix.js.

## 30.534 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_sub\_matrix\_dimension\_exception.js File Reference

### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1\\_sub\\_matrix\\_dimension\\_exception](#)

### 30.534.1 Variable Documentation

#### 30.534.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_sub\_matrix\_dimension\_exception

##### Initial value:

```
=
[
 ["SubMatrixDimensionException", "
 class_n_e_w_m_a_t_1_1_sub_matrix_dimension_exception.html#abd25517936c0502b87e217ea1d7d048e", null]
]
```

Definition at line 1 of file class\_n\_e\_w\_m\_a\_t\_1\_1\_sub\_matrix\_dimension\_exception.js.

## 30.535 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_subtracted\_matrix.js File Reference

### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1\\_subtracted\\_matrix](#)

### 30.535.1 Variable Documentation

#### 30.535.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_subtracted\_matrix

**Initial value:**

```
=
[
 ["~SubtractedMatrix", "class_n_e_w_m_a_t_1_1_subtracted_matrix.html#a237be251e398ed213d5957a1acf20fb0"
 , null],
 ["Evaluate", "class_n_e_w_m_a_t_1_1_subtracted_matrix.html#a8aa77be23c224971038968af5a3755b4", null],
 ["BaseMatrix", "class_n_e_w_m_a_t_1_1_subtracted_matrix.html#afcc50b811bd45d547cae53941018fdde", null
],
 ["GeneralMatrix", "class_n_e_w_m_a_t_1_1_subtracted_matrix.html#abaf0969f44c36896ac567a558a330771",
 null],
 ["GenericMatrix", "class_n_e_w_m_a_t_1_1_subtracted_matrix.html#a4ecabb4ff1be9d3dd3fd895b25fd7237",
 null]
]
```

Definition at line 1 of file class\_n\_e\_w\_m\_a\_t\_1\_1\_subtracted\_matrix.js.

## 30.536 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_symmetric\_band\_matrix.js File Reference

### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1\\_symmetric\\_band\\_matrix](#)

### 30.536.1 Variable Documentation

#### 30.536.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_symmetric\_band\_matrix

Definition at line 1 of file class\_n\_e\_w\_m\_a\_t\_1\_1\_symmetric\_band\_matrix.js.

## 30.537 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_symmetric\_eigen\_analysis.js File Reference

### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1\\_symmetric\\_eigen\\_analysis](#)

### 30.537.1 Variable Documentation

#### 30.537.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_symmetric\_eigen\_analysis

**Initial value:**

```
=
[
 ["SymmetricEigenAnalysis", "
 class_n_e_w_m_a_t_1_1_symmetric_eigen_analysis.html#abc9dfa5380987fa5d21f53b2f0cb63d2", null]
]
```

Definition at line 1 of file class\_n\_e\_w\_m\_a\_t\_1\_1\_symmetric\_eigen\_analysis.js.

## 30.538 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_symmetric\_matrix.js File Reference

### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1\\_symmetric\\_matrix](#)

### 30.538.1 Variable Documentation

#### 30.538.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_symmetric\_matrix

#### Initial value:

```
=
[
 ["SymmetricMatrix", "class_n_e_w_m_a_t_1_1_symmetric_matrix.html#ae2c2a54db1d2c0e28bd2cfa380081832",
 null],
 ["~SymmetricMatrix", "class_n_e_w_m_a_t_1_1_symmetric_matrix.html#ae4b06b44e6f0b13f9872b8f1accbaac9",
 null],
 ["SymmetricMatrix", "class_n_e_w_m_a_t_1_1_symmetric_matrix.html#aa2491ee1127b26c9448f6b53456152ac",
 null],
 ["SymmetricMatrix", "class_n_e_w_m_a_t_1_1_symmetric_matrix.html#a00594c6eced7cf7bd0e671dbf60a8361",
 null],
 ["SymmetricMatrix", "class_n_e_w_m_a_t_1_1_symmetric_matrix.html#ac3d2b731017c2cdcccd0ff51a3533ea17",
 null],
 ["element", "class_n_e_w_m_a_t_1_1_symmetric_matrix.html#a5dcb1771f7950ff61f08407c2c4775d9", null],
 ["element", "class_n_e_w_m_a_t_1_1_symmetric_matrix.html#a897fc26dd1c25d7a16eb4aa543fa8c00", null],
 ["GetCol", "class_n_e_w_m_a_t_1_1_symmetric_matrix.html#af5a5246219e82ce63f793d8291465865", null],
 ["GetCol", "class_n_e_w_m_a_t_1_1_symmetric_matrix.html#a888a89c2df9dca382660258c11fbf5f6", null],
 ["GetRow", "class_n_e_w_m_a_t_1_1_symmetric_matrix.html#a6fb84e5efe10694f606b4b4df394ceb0", null],
 ["operator()", "class_n_e_w_m_a_t_1_1_symmetric_matrix.html#a72c62160e3218d363dee73b88c1cbf38", null],
 ["operator()", "class_n_e_w_m_a_t_1_1_symmetric_matrix.html#ae9faab8a7a86aa812e9a474f67564a29", null],
 ["operator=", "class_n_e_w_m_a_t_1_1_symmetric_matrix.html#a6d7784715f5cf7f8747c2680ba6a0fd6", null],
 ["operator=", "class_n_e_w_m_a_t_1_1_symmetric_matrix.html#ac52ed19d3e737524a903241859cbe59", null],
 ["operator=", "class_n_e_w_m_a_t_1_1_symmetric_matrix.html#ab4287c53c6b288d536d4e96d79a44298", null],
 ["ReSize", "class_n_e_w_m_a_t_1_1_symmetric_matrix.html#a5113a1fe376c5594a5e3d639c9f7dd9a", null],
 ["ReSize", "class_n_e_w_m_a_t_1_1_symmetric_matrix.html#af0b59e1223600b7fe9fcb8373b90bc6", null],
 ["RestoreCol", "class_n_e_w_m_a_t_1_1_symmetric_matrix.html#ae1f90e1991bbdbee7133a03ff978ef74", null],
 ["RestoreCol", "class_n_e_w_m_a_t_1_1_symmetric_matrix.html#aabceff4b03eacd072182ae6a0384fcaa", null],
 ["Sum", "class_n_e_w_m_a_t_1_1_symmetric_matrix.html#acf6c7a8178db215880df3371eea89e4f", null],
 ["SumAbsoluteValue", "class_n_e_w_m_a_t_1_1_symmetric_matrix.html#a6f28a4fd67d0eac4bcd003d1a67a0692",
 null],
 ["SumSquare", "class_n_e_w_m_a_t_1_1_symmetric_matrix.html#a807fda06d1be9a7ce68c524b59654e23", null],
 ["Trace", "class_n_e_w_m_a_t_1_1_symmetric_matrix.html#a7f8a432b844e187b231d834e359d33231", null],
 ["Transpose", "class_n_e_w_m_a_t_1_1_symmetric_matrix.html#ace2dc8ef50a08215d179d6692ff6f10c", null],
 ["Type", "class_n_e_w_m_a_t_1_1_symmetric_matrix.html#a7d7568c0effefd6e20bf4c2b04028d5a", null]
]
```

Definition at line 1 of file `class_n_e_w_m_a_t_1_1_symmetric_matrix.js`.

## 30.539 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_transposed\_matrix.js File Reference

### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1\\_transposed\\_matrix](#)

### 30.539.1 Variable Documentation

#### 30.539.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_transposed\_matrix

##### Initial value:

```
=
[
 ["~TransposedMatrix", "class_n_e_w_m_a_t_1_1_transposed_matrix.html#a017982953187584c1c174a62676060ac", null],
 ["BandWidth", "class_n_e_w_m_a_t_1_1_transposed_matrix.html#a9778520c1dc1a3f9eb3961de8e2453b5", null],
 ["Evaluate", "class_n_e_w_m_a_t_1_1_transposed_matrix.html#a55d668cd4b198c0890bbb64f946491de", null],
 ["BaseMatrix", "class_n_e_w_m_a_t_1_1_transposed_matrix.html#afcc50b811bd45d547cae53941018fdde", null]
]
```

Definition at line 1 of file class\_n\_e\_w\_m\_a\_t\_1\_1\_transposed\_matrix.js.

## 30.540 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_upper\_band\_matrix.js File Reference

### Variables

- var class\_n\_e\_w\_m\_a\_t\_1\_1\_upper\_band\_matrix

### 30.540.1 Variable Documentation

#### 30.540.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_upper\_band\_matrix

##### Initial value:

```
=
[
 ["UpperBandMatrix", "class_n_e_w_m_a_t_1_1_upper_band_matrix.html#a05c4f4e8622790e93ed13b4c9d72cdd3", null],
 ["~UpperBandMatrix", "class_n_e_w_m_a_t_1_1_upper_band_matrix.html#a9bd6b0632457947f7b1d9baacdda6908", null],
 ["UpperBandMatrix", "class_n_e_w_m_a_t_1_1_upper_band_matrix.html#aa74ef95dfa3cd81cebd55c9e282aa14f", null],
 ["UpperBandMatrix", "class_n_e_w_m_a_t_1_1_upper_band_matrix.html#ae8b1d68b382f3d5590709f5bad0cfd10", null],
 ["UpperBandMatrix", "class_n_e_w_m_a_t_1_1_upper_band_matrix.html#a02bb7210ca57eda946ebe08e9acf75a1", null],
 ["element", "class_n_e_w_m_a_t_1_1_upper_band_matrix.html#a384ea0a49773eca1ca86ff63089520a2", null],
 ["element", "class_n_e_w_m_a_t_1_1_upper_band_matrix.html#abceb4ef40e37bf9aa17c440542e2ae4b", null],
 ["LogDeterminant", "class_n_e_w_m_a_t_1_1_upper_band_matrix.html#af1983e7ef715eb025fe24e3883ea673f", null],
 ["MakeSolver", "class_n_e_w_m_a_t_1_1_upper_band_matrix.html#a757aa1debf0a6095091a0ffe143ab779", null],
 ["operator()", "class_n_e_w_m_a_t_1_1_upper_band_matrix.html#ab9da207b110f5d0347756283b3f602d6", null],
 ["operator()", "class_n_e_w_m_a_t_1_1_upper_band_matrix.html#a1eaf7c71b4cf361f9f4438ad88072766", null],
 ["operator=", "class_n_e_w_m_a_t_1_1_upper_band_matrix.html#a52473f05645ea98ec740d6092b934d02", null],
 ["operator=", "class_n_e_w_m_a_t_1_1_upper_band_matrix.html#a393f173335757127e29b3f588fa3a551", null],
 ["operator=", "class_n_e_w_m_a_t_1_1_upper_band_matrix.html#a1abad0276818af6fe503cb55416ca806", null],
 ["ReSize", "class_n_e_w_m_a_t_1_1_upper_band_matrix.html#a726258b2cd3f70e7a2753423592bd0ba", null],
 ["ReSize", "class_n_e_w_m_a_t_1_1_upper_band_matrix.html#a66e6cd23aafb315266c8859edff3ac97", null],
 ["ReSize", "class_n_e_w_m_a_t_1_1_upper_band_matrix.html#a45d44da0f4ef4c5f6288866d3a445857", null],
 ["Solver", "class_n_e_w_m_a_t_1_1_upper_band_matrix.html#a1bf82143f11e8602ee26744belca498f", null],
 ["Type", "class_n_e_w_m_a_t_1_1_upper_band_matrix.html#a29318a563416dlac12dd348329d2fc17", null]
]
```

Definition at line 1 of file class\_n\_e\_w\_m\_a\_t\_1\_1\_upper\_band\_matrix.js.

## 30.541 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_upper\_triangular\_matrix.js File Reference

### Variables

- var `class_n_e_w_m_a_t_1_1_upper_triangular_matrix`

### 30.541.1 Variable Documentation

#### 30.541.1.1 var `class_n_e_w_m_a_t_1_1_upper_triangular_matrix`

##### Initial value:

```
=
[
 ["UpperTriangularMatrix", "
 class_n_e_w_m_a_t_1_1_upper_triangular_matrix.html#a60cccd528a0bedf79e99a0bd9bbe983d", null],
 ["~UpperTriangularMatrix", "
 class_n_e_w_m_a_t_1_1_upper_triangular_matrix.html#ae006061da058f7ce411ee781652a96ef", null],
 ["UpperTriangularMatrix", "
 class_n_e_w_m_a_t_1_1_upper_triangular_matrix.html#a27433781a058f05692ab4d1f54b1c3b9", null],
 ["UpperTriangularMatrix", "
 class_n_e_w_m_a_t_1_1_upper_triangular_matrix.html#a27c7b98a6f5db31c1d6f7385f9eaccab", null],
 ["UpperTriangularMatrix", "
 class_n_e_w_m_a_t_1_1_upper_triangular_matrix.html#a348db76c945ac7ca58f35afb435bb6d4", null],
 ["BandWidth", "class_n_e_w_m_a_t_1_1_upper_triangular_matrix.html#ac44568df02e5f05406e7fc9c9327745b",
 null],
 ["element", "class_n_e_w_m_a_t_1_1_upper_triangular_matrix.html#a046b84030cb32feccc089fe91f60074a",
 null],
 ["element", "class_n_e_w_m_a_t_1_1_upper_triangular_matrix.html#a84e297d8a4393ee3ae0f25e38686928e",
 null],
 ["GetCol", "class_n_e_w_m_a_t_1_1_upper_triangular_matrix.html#af944a6f5e1f17ceee80900ee1b555bfb",
 null],
 ["GetCol", "class_n_e_w_m_a_t_1_1_upper_triangular_matrix.html#ad764b09007be88971afef3b21a7d9b25",
 null],
 ["GetRow", "class_n_e_w_m_a_t_1_1_upper_triangular_matrix.html#a28f28ebf1f2aab9958d81d08c749cdce",
 null],
 ["LogDeterminant", "
 class_n_e_w_m_a_t_1_1_upper_triangular_matrix.html#ab6c08c946b89a2ef3cb4836c2a596f34", null],
 ["MakeSolver", "class_n_e_w_m_a_t_1_1_upper_triangular_matrix.html#aec1be799506cbec6b4e32c3d53e499c3",
 null],
 ["NextRow", "class_n_e_w_m_a_t_1_1_upper_triangular_matrix.html#ad88e19cb3b5395f92bb8cfa3ca5ea6db",
 null],
 ["operator()", "class_n_e_w_m_a_t_1_1_upper_triangular_matrix.html#aae00d4aeabea23cbca33031e5413945c",
 null],
 ["operator()", "class_n_e_w_m_a_t_1_1_upper_triangular_matrix.html#a3b9736cf6890e220d38ea3be3a74a67f",
 null],
 ["operator=", "class_n_e_w_m_a_t_1_1_upper_triangular_matrix.html#ae92d73c903f0f897e23b87c29834e3ab",
 null],
 ["operator=", "class_n_e_w_m_a_t_1_1_upper_triangular_matrix.html#a0473d8c547b7cfee5d514d0abb0f7eb3",
 null],
 ["operator=", "class_n_e_w_m_a_t_1_1_upper_triangular_matrix.html#ac62b2c390d605a050f3c69e24291c512",
 null],
 ["ReSize", "class_n_e_w_m_a_t_1_1_upper_triangular_matrix.html#a5b433191db6a5d868b3fd6ba15e3e891",
 null],
 ["ReSize", "class_n_e_w_m_a_t_1_1_upper_triangular_matrix.html#a2d6a62e39a7b3ee89c57365a9df7179c",
 null],
 ["RestoreCol", "class_n_e_w_m_a_t_1_1_upper_triangular_matrix.html#a6cb4b5bab6258887dfbe3d689f18cbd5",
 null],
 ["RestoreCol", "class_n_e_w_m_a_t_1_1_upper_triangular_matrix.html#a4ec93fdcd6582d743f2d8d284ab7a666",
 null],
 ["Solver", "class_n_e_w_m_a_t_1_1_upper_triangular_matrix.html#a2d2d515ff4c2b6dc8813b1545dc11950",
 null],
 ["Trace", "class_n_e_w_m_a_t_1_1_upper_triangular_matrix.html#a533ac255a31c33813f3342e1385f2ba9", null
],
 ["Type", "class_n_e_w_m_a_t_1_1_upper_triangular_matrix.html#ac2b805ae46946254632b51ed6de80622", null
]
]
```

Definition at line 1 of file `class_n_e_w_m_a_t_1_1_upper_triangular_matrix.js`.



## 30.542 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1\_vector\_exception.js File Reference

### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1\\_vector\\_exception](#)

### 30.542.1 Variable Documentation

#### 30.542.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1\_vector\_exception

##### Initial value:

```
=
[
 ["VectorException", "class_n_e_w_m_a_t_1_1_vector_exception.html#aeb9ecef50b98576f64273e32cf7f6408",
 null],
 ["VectorException", "class_n_e_w_m_a_t_1_1_vector_exception.html#ald8cbd2f76b53a840940c8073a923b32",
 null]
]
```

Definition at line 1 of file `class_n_e_w_m_a_t_1_1_vector_exception.js`.

## 30.543 doc/html/class\_n\_e\_w\_m\_a\_t\_1\_1nric\_matrix.js File Reference

### Variables

- var [class\\_n\\_e\\_w\\_m\\_a\\_t\\_1\\_1nric\\_matrix](#)

### 30.543.1 Variable Documentation

#### 30.543.1.1 var class\_n\_e\_w\_m\_a\_t\_1\_1nric\_matrix

##### Initial value:

```
=
[
 ["nricMatrix", "class_n_e_w_m_a_t_1_1nric_matrix.html#a71472915dd795bd4578bc8fcaf4df329", null],
 ["nricMatrix", "class_n_e_w_m_a_t_1_1nric_matrix.html#ab666d6af8ad5d123053868162bc15e8a", null],
 ["nricMatrix", "class_n_e_w_m_a_t_1_1nric_matrix.html#aa6d83044fd2538bb66f7587d219dbcee", null],
 ["nricMatrix", "class_n_e_w_m_a_t_1_1nric_matrix.html#a6fb0d3b24e679e712180eae750c7b0d1", null],
 ["~nricMatrix", "class_n_e_w_m_a_t_1_1nric_matrix.html#af2d9d74b656163c70d3218686f319b99", null],
 ["CleanUp", "class_n_e_w_m_a_t_1_1nric_matrix.html#a25d4c4c4f737472f212e7de2a89e81dd", null],
 ["nric", "class_n_e_w_m_a_t_1_1nric_matrix.html#a5cd5a86793112f0f1892d1dc9a6ea63f", null],
 ["operator<<", "class_n_e_w_m_a_t_1_1nric_matrix.html#ac2da22880599fe369cc0f2f4f8ba3302", null],
 ["operator=", "class_n_e_w_m_a_t_1_1nric_matrix.html#a1514a8b4d4008c579fd3fb5211f84afe", null],
 ["operator=", "class_n_e_w_m_a_t_1_1nric_matrix.html#a6eb27163366d6408ee85996505c5e73b", null],
 ["operator=", "class_n_e_w_m_a_t_1_1nric_matrix.html#ad76180daeeeb6bc55d9cb331b59afa2d", null],
 ["ReSize", "class_n_e_w_m_a_t_1_1nric_matrix.html#a5e4658320bc41472e4b8a3d2a42570b2", null],
 ["ReSize", "class_n_e_w_m_a_t_1_1nric_matrix.html#ae40ad5647e371c2a6c253f1030d7c164", null]
]
```

Definition at line 1 of file `class_n_e_w_m_a_t_1_1nric_matrix.js`.

## 30.544 doc/html/class\_parametric\_surface\_property\_sheet.js File Reference

### Variables

- var [class\\_parametric\\_surface\\_property\\_sheet](#)

### 30.544.1 Variable Documentation

#### 30.544.1.1 var class\_parametric\_surface\_property\_sheet

##### Initial value:

```
=
[
 ["ParametricSurfacePropertySheet", "
 class_parametric_surface_property_sheet.html#a1988d5ca14218b7fe011986e94eeb8bc", null],
 ["ParametricSurfacePropertySheet", "
 class_parametric_surface_property_sheet.html#a8a926e205e3df7a1df5e7dfcf34c40e7", null],
 ["~ParametricSurfacePropertySheet", "
 class_parametric_surface_property_sheet.html#a58d075af3625aba3953bc1aec2c2d2c2", null],
 ["apply", "class_parametric_surface_property_sheet.html#a58153ab0ac1f11e082d73755e522146b", null],
 ["createSheet", "class_parametric_surface_property_sheet.html#a4993f8166745738be7c49633e039f339", null
]
]
```

Definition at line 1 of file class\_parametric\_surface\_property\_sheet.js.

## 30.545 doc/html/class\_path\_type.js File Reference

### Variables

- var [class\\_path\\_type](#)

### 30.545.1 Variable Documentation

#### 30.545.1.1 var class\_path\_type

##### Initial value:

```
=
[
 ["funnelOfPath", "class_path_type.html#a4c49a86198da4ae6ae95b53c577c0f2b", null],
 ["initPathType", "class_path_type.html#a901e079d89cbb67ab32ac2efe75015b0", null],
 ["modification_l_path_", "class_path_type.html#a3c9dc01c75b91db2f7acff58727de110", null],
 ["modification_r_path_", "class_path_type.html#a36be5d505adcbf7c71066f10f6d6c9b7", null],
 ["print", "class_path_type.html#ac9ff569ae379418868d0fb54d79bf3c7", null],
 ["funnel_", "class_path_type.html#aa494c02211096f0fb748cc6e41002b01", null],
 ["l_path_", "class_path_type.html#a3849d634c4440974a3e8664f0adf2c88", null],
 ["r_path_", "class_path_type.html#a6698f5a904d409bc6779015e5ef36aca", null]
]
```

Definition at line 1 of file class\_path\_type.js.

## 30.546 doc/html/class\_point\_cloud\_property\_sheet.js File Reference

### Variables

- var [class\\_point\\_cloud\\_property\\_sheet](#)

### 30.546.1 Variable Documentation

#### 30.546.1.1 var class\_point\_cloud\_property\_sheet

##### Initial value:

```
=
[
 ["PointCloudPropertySheet", "class_point_cloud_property_sheet.html#ac7cee89813a163332d644d6156bce2e1",
 null],
 ["apply", "class_point_cloud_property_sheet.html#aa538a7cc2fb4d934bd70ee45a690ab4c", null],
 ["createSheet", "class_point_cloud_property_sheet.html#ac15fa68bf84eed7ee2ba9d6d39f329f7", null]
]
```

Definition at line 1 of file class\_point\_cloud\_property\_sheet.js.

## 30.547 doc/html/class\_pr\_bi\_c\_g\_stab.js File Reference

### Variables

- var [class\\_pr\\_bi\\_c\\_g\\_stab](#)

### 30.547.1 Variable Documentation

#### 30.547.1.1 var class\_pr\_bi\_c\_g\_stab

##### Initial value:

```
=
[
 ["PrBiCGStab", "class_pr_bi_c_g_stab.html#af29c17ca09bc2aaa00f77683ec9bf6b5", null],
 ["~PrBiCGStab", "class_pr_bi_c_g_stab.html#ae205ad572007535f67b698df4c0e256c", null],
 ["converged", "class_pr_bi_c_g_stab.html#a2ddd5368c09bf5ab2d376cec5286b903", null],
 ["getCPUtime", "class_pr_bi_c_g_stab.html#a6e0dcd08861df584e48d929c986fb72c", null],
 ["getItCount", "class_pr_bi_c_g_stab.html#af8a616464297663ba6f11a9287916156", null],
 ["setMaxIterations", "class_pr_bi_c_g_stab.html#a5c6ba709c4dd7308049c372827600bcc", null],
 ["setTolerance", "class_pr_bi_c_g_stab.html#a85ebae11f3711327bb5e1da225cdab7b", null],
 ["solve", "class_pr_bi_c_g_stab.html#ab823e064ec1a640a4db5660c07921a2b", null],
 ["converged_", "class_pr_bi_c_g_stab.html#abbb29cf5a3f4865b11a18c12980f143c", null],
 ["cpu_time_", "class_pr_bi_c_g_stab.html#a38f46357a634b412e03430e11d86f387", null],
 ["it_count_", "class_pr_bi_c_g_stab.html#a176255ece25cf5d314eeb07c25f496bc", null],
 ["max_iterations_", "class_pr_bi_c_g_stab.html#ae577a4afad04242659e1a607a445f532", null],
 ["tolerance_", "class_pr_bi_c_g_stab.html#a1fbeb8689dac50d78a1c4311c608942", null]
]
```

Definition at line 1 of file class\_pr\_bi\_c\_g\_stab.js.

## 30.548 doc/html/class\_pr\_c\_g.js File Reference

### Variables

- var [class\\_pr\\_c\\_g](#)

### 30.548.1 Variable Documentation

#### 30.548.1.1 var class\_pr\_c\_g

##### Initial value:

```
=
[
 ["PrCG", "class_pr_c_g.html#a2a133e0edcf4255b13cc8ac87e5a2228", null],
 ["~PrCG", "class_pr_c_g.html#acc017672d6843d30f429ba8c96840b3d", null],
 ["converged", "class_pr_c_g.html#acdb4f70d9d04d84ddabcb89ffe70add6", null],
 ["getCPUtime", "class_pr_c_g.html#ada3cf282632645f08ed32c6dbbedf9dc", null],
 ["getItCount", "class_pr_c_g.html#ab086829e4149cc54ba8b681bf4437e73", null],
 ["setMaxIterations", "class_pr_c_g.html#a21f656df247b78293ef2471352e1f58d", null],
 ["setTolerance", "class_pr_c_g.html#a3bad356e25f97e13024d96be2fde5", null],
 ["solve", "class_pr_c_g.html#a502d86eebffa8902ffd6dee4c71985f3", null],
 ["converged_", "class_pr_c_g.html#a27546b7c46973f7fe38c6cd3e49599e6", null],
 ["cpu_time_", "class_pr_c_g.html#ab52a43118de67a07e8bb07db25519e80", null],
 ["it_count_", "class_pr_c_g.html#ac08eafdfd61250a817879da2de1155c2", null],
 ["max_iterations_", "class_pr_c_g.html#a033d3c4a9496d728fe4fed338db96232", null],
 ["tolerance_", "class_pr_c_g.html#aa2637b95f8234776302f534b945768e4", null]
]
```

Definition at line 1 of file class\_pr\_c\_g.js.

## 30.549 doc/html/class\_pr\_cell\_structure.js File Reference

### Variables

- var [class\\_pr\\_cell\\_structure](#)

### 30.549.1 Variable Documentation

#### 30.549.1.1 var class\_pr\_cell\_structure

##### Initial value:

```
=
[
 ["PrCellStructure", "class_pr_cell_structure.html#a16c767a23002dad5f591618d7d6f3fb8", null],
 ["PrCellStructure", "class_pr_cell_structure.html#ad3f64b851371b0367f25ae3e72d96146", null],
 ["~PrCellStructure", "class_pr_cell_structure.html#a043accfc76dc40b62ce7870ca95c57ef", null],
 ["attach", "class_pr_cell_structure.html#a0fd188587af01f15c24ee6a7b40d4cbd", null],
 ["get3dNode", "class_pr_cell_structure.html#adfebfeed2dcf77015397080a85f4b896", null],
 ["getBall", "class_pr_cell_structure.html#a73cf4b75c2d19ac757ecfea9658b47bd", null],
 ["getI", "class_pr_cell_structure.html#a08e9a339747e562d66629f91c2f0e060", null],
 ["getIJK", "class_pr_cell_structure.html#ae6004d1ecd14afcfc5f2dd88995f4559", null],
 ["getKNearest", "class_pr_cell_structure.html#aef17f5f445707a2f79adfdf154218358", null],
 ["getNumCells", "class_pr_cell_structure.html#a9a2fc00537502476fee71c7831ac90d3", null],
 ["getNumNodes", "class_pr_cell_structure.html#a3796af23df95ea0e6735bc5078ab8a46", null],
 ["print", "class_pr_cell_structure.html#ac988110628e043a50e0220078e8ea6a4", null],
 ["scan", "class_pr_cell_structure.html#ac82976828f96e3c317ee708c3dedae01", null],
 ["set3dNode", "class_pr_cell_structure.html#a457b90475c8fcfbf6d837f739e60490", null],
 ["setNumCells", "class_pr_cell_structure.html#a19fde9a07e3263ddef38acd0ca0c8343", null],
 ["whichCell", "class_pr_cell_structure.html#a029bd0e3caec3c0041f84d4fbc9fde07", null]
]
```

Definition at line 1 of file class\_pr\_cell\_structure.js.

## 30.550 doc/html/class\_pr\_explicit\_connectivity.js File Reference

### Variables

- var [class\\_pr\\_explicit\\_connectivity](#)

### 30.550.1 Variable Documentation

#### 30.550.1.1 var class\_pr\_explicit\_connectivity

##### Initial value:

```
=
[
 ["findFace", "class_pr_explicit_connectivity.html#a6dce993f368ce7a7cad039e3b1d1e13b", null],
 ["findGenus", "class_pr_explicit_connectivity.html#affe3cdfef5bec6753d506d4b0dd212f0", null],
 ["findNextEdgeInFace", "class_pr_explicit_connectivity.html#aeeedf5948bdf30cc86d100813ee928c2", null]
,
 ["findNumFaces", "class_pr_explicit_connectivity.html#ac8b48451df85182a5f53a635136f7b3c", null],
 ["isTriangulation", "class_pr_explicit_connectivity.html#a238f3c956286ed7a75d6e0533a82c9bd", null],
 ["printInfo", "class_pr_explicit_connectivity.html#ac740b4ea4b1640d31612630d9f114d97", null],
 ["printTexture", "class_pr_explicit_connectivity.html#a719878295d70b1580ad7cf780a6c58ba", null],
 ["printUVFaces", "class_pr_explicit_connectivity.html#a06eb38d64f4a84aa0ec252bb6e1d0520", null],
 ["printXYZFaces", "class_pr_explicit_connectivity.html#af7fd7c82f20dbbe1ce6f7b308308fb2b", null],
 ["printXYZFacesML", "class_pr_explicit_connectivity.html#acbfd5a2ed61ec4371bb1bd28ed52823c", null],
 ["printXYZFacesVRML", "class_pr_explicit_connectivity.html#ad5c0e34d9019ab2eeb492af581e8c40e", null]
]
```

Definition at line 1 of file class\_pr\_explicit\_connectivity.js.

## 30.551 doc/html/class\_pr\_faber\_\_\_f.js File Reference

### Variables

- var [class\\_pr\\_faber\\_\\_\\_f](#)

### 30.551.1 Variable Documentation

#### 30.551.1.1 var class\_pr\_faber\_\_\_f

##### Initial value:

```
=
[
 ["PrFaber_F", "class_pr_faber___f.html#ae91ae9ab3f5a7245766f37c173ce2fab", null],
 ["~PrFaber_F", "class_pr_faber___f.html#ac0fb6798bb4739ad262bddb592aab18a", null],
 ["compose", "class_pr_faber___f.html#ad17db6cacf6a4df453ab1c40abea20cc", null],
 ["decompose", "class_pr_faber___f.html#a99c5dd908fc81d1720339296513e9c69", null]
]
```

Definition at line 1 of file class\_pr\_faber\_\_\_f.js.

## 30.552 doc/html/class\_pr\_fast\_unorganized\_\_\_o\_p.js File Reference

### Variables

- var [class\\_pr\\_fast\\_unorganized\\_\\_\\_o\\_p](#)

### 30.552.1 Variable Documentation

#### 30.552.1.1 var class\_pr\_fast\_unorganized\_\_\_o\_p

#### Initial value:

```
=
[
 ["PrFastUnorganized_OP", "class_pr_fast_unorganized___o_p.html#ab8182ea9cc5a8e07c3505dd169c5f822",
 null],
 ["PrFastUnorganized_OP", "class_pr_fast_unorganized___o_p.html#abdd99ba6a2d89db4351e0155ce5d0661",
 null],
 ["~PrFastUnorganized_OP", "class_pr_fast_unorganized___o_p.html#a65161b706f068f6aa851a23420110837",
 null],
 ["get3dNode", "class_pr_fast_unorganized___o_p.html#ac2f22f010e169ab276e2067b9318cc96", null],
 ["getKNearest", "class_pr_fast_unorganized___o_p.html#aa2078e02765076407c79067bf0a5ab91", null],
 ["getNeighbours", "class_pr_fast_unorganized___o_p.html#a531b902b4380ca46ea790a2eed8cd6e2", null],
 ["getNumNodes", "class_pr_fast_unorganized___o_p.html#aa165f906680d6a6eaa2c41837fdeb841", null],
 ["getRadius", "class_pr_fast_unorganized___o_p.html#a5e7d347bcb669f63772af8ca7d119bb6", null],
 ["getU", "class_pr_fast_unorganized___o_p.html#ae48782e590d1e8c86c99e36b7e075b06", null],
 ["getV", "class_pr_fast_unorganized___o_p.html#aa3df19fc29a6beedb367195e9c26c9da", null],
 ["initNeighbours", "class_pr_fast_unorganized___o_p.html#adea6d686f19587064315b1b653c9a143", null],
 ["isBoundary", "class_pr_fast_unorganized___o_p.html#a07be1e2277a09ab684845fb04eb6e5dc", null],
 ["print", "class_pr_fast_unorganized___o_p.html#ae55ef2161e07741cfd73a92823afd5", null],
 ["scanRawData", "class_pr_fast_unorganized___o_p.html#aa5b0440ba88b20d36b011b77161de050", null],
 ["set3dNode", "class_pr_fast_unorganized___o_p.html#a4e28572095a5c461866f98c5643cba0d", null],
 ["setKNearest", "class_pr_fast_unorganized___o_p.html#af846a4715a1bce09c74874caal79b304", null],
 ["setRadius", "class_pr_fast_unorganized___o_p.html#ad807e42f2ee8839ef50e10ef9a035497", null],
 ["setU", "class_pr_fast_unorganized___o_p.html#a6f30f46acbe3fbba0594f889c27653f5", null],
 ["setV", "class_pr_fast_unorganized___o_p.html#a8f8e937505e2e009a9760896465d90bb", null],
 ["useK", "class_pr_fast_unorganized___o_p.html#a3a7171a3f301bfc9c9a98fcd69b6fe65", null],
 ["useRadius", "class_pr_fast_unorganized___o_p.html#a9228f86f8f8c016c94c25dad375e80ab", null]
]
```

Definition at line 1 of file class\_pr\_fast\_unorganized\_\_\_o\_p.js.

## 30.553 doc/html/class\_pr\_filterbank.js File Reference

### Variables

- var [class\\_pr\\_filterbank](#)

### 30.553.1 Variable Documentation

#### 30.553.1.1 var class\_pr\_filterbank

#### Initial value:

```
=
[
 ["~PrFilterbank", "class_pr_filterbank.html#aee382f21ce649fca3a4d935a057ed108", null],
 ["attach", "class_pr_filterbank.html#a28a908fc7e930da4a931cfa621334111", null],
 ["compose", "class_pr_filterbank.html#a1c33322f58deb9f95147b815bebd70fd1", null],
 ["composeAll", "class_pr_filterbank.html#a38e75c91887cd1a90e7bb80ebdb9ad0", null],
 ["composeUpTo", "class_pr_filterbank.html#a287e238fec6a5175d8c50b1ea7f22728", null],
 ["decompose", "class_pr_filterbank.html#ad88fd562db95c8995e3363758a9b5ec9", null],
 ["decomposeAll", "class_pr_filterbank.html#a04a34136e554217d24e4f5be34e12cf0", null],
 ["decomposeFrom", "class_pr_filterbank.html#a6b1af55dc2804c168fc2b7e2562561cf", null],
 ["neighbours_", "class_pr_filterbank.html#a5e17246b697fc4e617a1c859701f7c36", null],
 ["t_", "class_pr_filterbank.html#a8ba3a2c1b525d44832be09fc86afe136", null]
]
```

Definition at line 1 of file class\_pr\_filterbank.js.

## 30.554 doc/html/class\_pr\_heap.js File Reference

### Variables

- var [class\\_pr\\_heap](#)

### 30.554.1 Variable Documentation

#### 30.554.1.1 var class\_pr\_heap

##### Initial value:

```
=
[
 ["PrHeap", "class_pr_heap.html#a885b0cd0ac71c273d64717266cae66ed", null],
 ["PrHeap", "class_pr_heap.html#a5f94d31251ae79f78c9158469b57b7ba", null],
 ["PrHeap", "class_pr_heap.html#a9fa6efec2ed32b287d8b8659043a962e", null],
 ["PrHeap", "class_pr_heap.html#af7653207d696ae3e4f8c2151110c61ca", null],
 ["~PrHeap", "class_pr_heap.html#acd55f02207eabf822ed4a2a51b83c611", null],
 ["emptyHeap", "class_pr_heap.html#ab6f635d7175614b79f924854b5f3e34a", null],
 ["getMaxSize", "class_pr_heap.html#ac1bde43c0b065be732d5f22e2119661", null],
 ["getSize", "class_pr_heap.html#a25b63a53b01007cedd3356a7f5568739", null],
 ["modify", "class_pr_heap.html#a40d87c74197e7c601f4c14e4f0cf61ef", null],
 ["pop", "class_pr_heap.html#adb0e42028fb225f697d1762a81b1b2b7", null],
 ["pop", "class_pr_heap.html#adbcb76503eb4c6483bfddd7454f1649", null],
 ["print", "class_pr_heap.html#a3b4c2639f85e5fca72512f2ccla348a3", null],
 ["push", "class_pr_heap.html#afd592c636c5c88af174fd56df80ccd8f", null],
 ["redim", "class_pr_heap.html#a88f289a9bbe49d069c55471eb62c1d73", null]
]
```

Definition at line 1 of file class\_pr\_heap.js.

## 30.555 doc/html/class\_pr\_level\_triangulation\_\_\_o\_p.js File Reference

### Variables

- var [class\\_pr\\_level\\_triangulation\\_\\_\\_o\\_p](#)

### 30.555.1 Variable Documentation

#### 30.555.1.1 var class\_pr\_level\_triangulation\_\_\_o\_p

##### Initial value:

```
=
[
 ["PrLevelTriangulation_OP", "class_pr_level_triangulation___o_p.html#a0c899f975fc8abcea05f0c09f899501b", null],
 ["PrLevelTriangulation_OP", "class_pr_level_triangulation___o_p.html#a921e9bfb0b17959bcf93f6d41b9e719", null],
 ["~PrLevelTriangulation_OP", "class_pr_level_triangulation___o_p.html#a592917a6652ef2cb21733460c2e8fc5b", null],
 ["findNumFaces", "class_pr_level_triangulation___o_p.html#a5c127c36ea718199d7120e3d29dd35ed", null],
 ["get3dNode", "class_pr_level_triangulation___o_p.html#ae9a99c26709ad8718eb85a7717248bc7", null],
 ["getNeighbours", "class_pr_level_triangulation___o_p.html#af231b1530e5db60df4b0dc4efc140c5e", null],
 ["getNumNodes", "class_pr_level_triangulation___o_p.html#a359cd2d453f84ff4e749e687680250c7", null],
 ["getPrNestedNode", "class_pr_level_triangulation___o_p.html#a6f02a1310d843d0703e655b567ed4ff9", null],
]
```

```
[
 ["getPrTriangle", "class_pr_level_triangulation___o_p.html#ae51d8250ed18de53567d3cd1125d82b0", null],
 ["getU", "class_pr_level_triangulation___o_p.html#a5c93378b4b3334e9b480e0461fec6956", null],
 ["getV", "class_pr_level_triangulation___o_p.html#affb10f48fc1ad0635a5787c9d961eab4", null],
 ["isBoundary", "class_pr_level_triangulation___o_p.html#a240f978a14d57d4f2c27cdc78b2b8a67", null],
 ["print", "class_pr_level_triangulation___o_p.html#a358fe4c750cfc5dee4052dea4747d21a", null],
 ["printRawData", "class_pr_level_triangulation___o_p.html#a8032303355737bb267e40013aa690dda", null],
 ["printUV", "class_pr_level_triangulation___o_p.html#ad4a307ed0c86ed9c2e49c6f9aef60aa9", null],
 ["printUVTriangles", "class_pr_level_triangulation___o_p.html#a896d4dc8a46df866ba7d638479d4f965", null],
],
 ["printXYZTriangles", "class_pr_level_triangulation___o_p.html#ac0eea3a31094abf2764b3e2baafecbfe",
 null],
 ["set3dNode", "class_pr_level_triangulation___o_p.html#a571ffc12f02b671e22480338cc2df804", null],
 ["setU", "class_pr_level_triangulation___o_p.html#a3e2590222d6b52a8a0c59837f67c43c9", null],
 ["setV", "class_pr_level_triangulation___o_p.html#a246e9c75cea76bd570704f6a68191790", null]
]
```

Definition at line 1 of file class\_pr\_level\_triangulation\_\_\_o\_p.js.

## 30.556 doc/html/class\_pr\_mat.js File Reference

### Variables

- var [class\\_pr\\_mat](#)

### 30.556.1 Variable Documentation

#### 30.556.1.1 var class\_pr\_mat

#### Initial value:

```
=
[
 ["~PrMat", "class_pr_mat.html#abac68fd201e66418a5f938317a9466e1", null],
 ["PrMat", "class_pr_mat.html#aba56565aee89245b11b9aec00bdd8ea7", null],
 ["PrMat", "class_pr_mat.html#a2323edeb3783c52d2b02a73bf97af7e7", null],
 ["colms", "class_pr_mat.html#aed80d11429e1c2a0566db0d806d80f3f", null],
 ["operator()", "class_pr_mat.html#aa5f4ad3c596380c2eb0048986d55223b", null],
 ["operator()", "class_pr_mat.html#aff57354fbd39fc635924f0754f9d5cdb", null],
 ["prod", "class_pr_mat.html#a6f71a98524b14304d5b1a422da0c4a1a", null],
 ["read", "class_pr_mat.html#a137fb40cee88e91211c2df1a90a0e62c", null],
 ["redim", "class_pr_mat.html#a35b007c13f0732f4b379f1375fa97475", null],
 ["rows", "class_pr_mat.html#a51fc93a045aa5c0d2b6c357269441fbf", null],
 ["a_", "class_pr_mat.html#adc5bf006d2b3c60eb45bc6f374992a3d", null],
 ["m_", "class_pr_mat.html#a4808d661baa0b26acc1c87a5647e18ab", null],
 ["n_", "class_pr_mat.html#a02159b9afb6c616424f920abaae8565", null]
]
```

Definition at line 1 of file class\_pr\_mat.js.

## 30.557 doc/html/class\_pr\_mat\_sparse.js File Reference

### Variables

- var [class\\_pr\\_mat\\_sparse](#)



### 30.557.1 Variable Documentation

#### 30.557.1.1 var class\_pr\_mat\_sparse

##### Initial value:

```
=
[
 ["~PrMatSparse", "class_pr_mat_sparse.html#aed116dlaf35084249dd084d98d709b13", null],
 ["PrMatSparse", "class_pr_mat_sparse.html#a8eaf5655d57c3e281d3d505d3dec103a", null],
 ["PrMatSparse", "class_pr_mat_sparse.html#a514ba7377148959d9c62dca77467a3d4", null],
 ["PrMatSparse", "class_pr_mat_sparse.html#aad0459e91bc81ac8feb5456b42337698", null],
 ["colsms", "class_pr_mat_sparse.html#a22e218eabe43458adaf7acac161da74a", null],
 ["irow", "class_pr_mat_sparse.html#ad43793dd79a6c54bc5673700891c6b1b", null],
 ["irow", "class_pr_mat_sparse.html#a87cd1ddbb1985beefc4ea7d91f95632c", null],
 ["jcol", "class_pr_mat_sparse.html#ac228cc2de5b4e250dc528a3aleb134e8", null],
 ["jcol", "class_pr_mat_sparse.html#ab2e769db2b0ae75bd1e098bf47c5f122", null],
 ["matProd", "class_pr_mat_sparse.html#a277e2097aedc2f6097bd00f56c9bbaaf", null],
 ["operator()", "class_pr_mat_sparse.html#aa90791838142418e3f4b32e292f236d0", null],
 ["operator()", "class_pr_mat_sparse.html#ab1a60337c6f4817d73079357518e0d3d", null],
 ["operator()", "class_pr_mat_sparse.html#ad9a41f7ecf950f83fe759bfa5662f4d6", null],
 ["print", "class_pr_mat_sparse.html#a70aabe01c24310aa91623fba9ebe86e9", null],
 ["printFull", "class_pr_mat_sparse.html#a3f24806e5b6ed1d7c0b23ceaa829cb89", null],
 ["prod", "class_pr_mat_sparse.html#afe016057a1857fbf926d736549488dc2", null],
 ["read", "class_pr_mat_sparse.html#a6ae00848a5bb4eee4b6cc0460ec09721", null],
 ["redim", "class_pr_mat_sparse.html#aff579539f8274becbb9ef2dd10d56804", null],
 ["rows", "class_pr_mat_sparse.html#ad9dae54e8b806e2d30e2e3522fe47665", null],
 ["scalProd", "class_pr_mat_sparse.html#a241293b891b9ff2edec937b6306cad3d", null],
 ["setToMatrix", "class_pr_mat_sparse.html#a6d1ddcc10fb7fe3c6db127cc2e3b5e7c", null]
]
```

Definition at line 1 of file class\_pr\_mat\_sparse.js.

## 30.558 doc/html/class\_pr\_matrix.js File Reference

### Variables

- var [class\\_pr\\_matrix](#)

### 30.558.1 Variable Documentation

#### 30.558.1.1 var class\_pr\_matrix

##### Initial value:

```
=
[
 ["~PrMatrix", "class_pr_matrix.html#a874138ad2523d4e463466cdc22b87bfb", null],
 ["colsms", "class_pr_matrix.html#a8281ab6b6c20077323e3365c566e3df3", null],
 ["operator()", "class_pr_matrix.html#afe0e2980dcb97924cdbfc725f4ca983b", null],
 ["print", "class_pr_matrix.html#aa4232d5dcb7a16e7a7alb9190b2f8c6f", null],
 ["prod", "class_pr_matrix.html#a04f041446176355c38d95e3eea40029b", null],
 ["read", "class_pr_matrix.html#afb99ca11e79ef21802d68cfdfb555bb8", null],
 ["rows", "class_pr_matrix.html#a098ac67f62a0a492f285b6608c9225e4", null]
]
```

Definition at line 1 of file class\_pr\_matrix.js.

## 30.559 doc/html/class\_pr\_nested\_node.js File Reference

### Variables

- var [class\\_pr\\_nested\\_node](#)

### 30.559.1 Variable Documentation

#### 30.559.1.1 var class\_pr\_nested\_node

##### Initial value:

```
=
[
 ["PrNestedNode", "class_pr_nested_node.html#a36fa389153b0aa64bd34c24b759d5c32", null],
 ["PrNestedNode", "class_pr_nested_node.html#a13c3548e0d0e223c732d8f65677bcf30", null],
 ["~PrNestedNode", "class_pr_nested_node.html#a33dd62c2b069247089acb492d38ac2c5", null],
 ["addTrianglePtr", "class_pr_nested_node.html#a91c4e46795b99d0efa7cbcceb5494ae9", null],
 ["init", "class_pr_nested_node.html#a860b7495910de6de3d14ec978cf248e2", null],
 ["level", "class_pr_nested_node.html#ab065c2aa60ce5c6fe4c9ab88afb9ca99", null],
 ["level", "class_pr_nested_node.html#ae3a88ecb71bed3cfae1a0af64a286c51", null],
 ["pnt", "class_pr_nested_node.html#a45d4a054f9161bbe67ad83291ba26375", null],
 ["pnt", "class_pr_nested_node.html#a52aa5fa14965b065f40597b9bcf0af8c", null],
 ["print", "class_pr_nested_node.html#ab38b4c82767fd7b7256c2a98318cf487", null],
 ["printUV", "class_pr_nested_node.html#acc7aa0e5795b71425fda2cde1fd4ba32", null],
 ["printXYZ", "class_pr_nested_node.html#a6648e0579589a09a43a7812d5bb90815", null],
 ["tr", "class_pr_nested_node.html#a3861d22889e30151bb28e3b38f9750a3", null],
 ["tr", "class_pr_nested_node.html#a4a01495e473dbe624e4db0881bedac14", null],
 ["u", "class_pr_nested_node.html#a7f9c2beee25a3bf8806907fe6b675b92", null],
 ["u", "class_pr_nested_node.html#a2fcd5ec45fa3515d66c959305db4a1ce", null],
 ["v", "class_pr_nested_node.html#a4fb696c20bedfb1aa45377050dfc8509", null],
 ["v", "class_pr_nested_node.html#aa0688bdd1fcd5d49dd45e3cbb56e0cec", null],
 ["x", "class_pr_nested_node.html#a69747eb489112d0560cebcf169df57f6", null],
 ["x", "class_pr_nested_node.html#a613b6c4885ec0097d356410119e8a4b9", null],
 ["y", "class_pr_nested_node.html#ac2aa9365d31b0e0bc8526ee52ca48f40", null],
 ["y", "class_pr_nested_node.html#a437a7f299f4aeb739d3c86fd434a5673", null],
 ["z", "class_pr_nested_node.html#ae5e6fab6058c33f1c08c5e7b335502ba", null],
 ["z", "class_pr_nested_node.html#a6b14152feaf1cdbe28caae15b42e1514", null]
]
```

Definition at line 1 of file `class_pr_nested_node.js`.

## 30.560 doc/html/class\_pr\_nested\_triangulation.js File Reference

### Variables

- var [class\\_pr\\_nested\\_triangulation](#)

### 30.560.1 Variable Documentation

#### 30.560.1.1 var class\_pr\_nested\_triangulation

##### Initial value:

```
=
[
 ["~PrNestedTriangulation", "class_pr_nested_triangulation.html#a59cad8d7ce5a400c389b0d5e69359ac9", null],
 ["getFinestLevel", "class_pr_nested_triangulation.html#adb5ff821ff46e6b1c74a2dfd83b8b861", null],
 ["getNeighbours", "class_pr_nested_triangulation.html#a670fe9a46eb1b32d538076b12f07928b", null],
 ["getNumNodes", "class_pr_nested_triangulation.html#ad24aad8c7d57742c6b1f5b6f2ab34b9e", null],
 ["getParents", "class_pr_nested_triangulation.html#a532957adaa30f7830cf126f0e6df362e", null],
 ["getX", "class_pr_nested_triangulation.html#abf3afb15cfe97f7be5bc45f1d43b0134a", null],
 ["getY", "class_pr_nested_triangulation.html#a971a214705cbb3860596dd4de59c2cf7", null],
 ["getZ", "class_pr_nested_triangulation.html#a3760b237a4bc2ccbfc5f64d35ad2c81", null],
 ["isBoundary", "class_pr_nested_triangulation.html#aa27033447f3f9c878c305d7d9243b7af", null],
 ["setX", "class_pr_nested_triangulation.html#aec4b07cb47b9444d811a917d55598bec", null],
 ["setY", "class_pr_nested_triangulation.html#abb034ddc03fb21360173a46f71d7cf39", null],
 ["setZ", "class_pr_nested_triangulation.html#ae6e08bb3cf714164ec08b811adc923a2", null]
]
```

Definition at line 1 of file `class_pr_nested_triangulation.js`.

## 30.561 doc/html/class\_pr\_node.js File Reference

### Variables

- var [class\\_pr\\_node](#)

### 30.561.1 Variable Documentation

#### 30.561.1.1 var class\_pr\_node

#### Initial value:

```
=
[
 ["PrNode", "class_pr_node.html#a46c035195d075cb92942e269ffa2f7f8", null],
 ["PrNode", "class_pr_node.html#aca41d00932e46e5a124a85b72987d6e0", null],
 ["init", "class_pr_node.html#abf10cdcbcc995052ca52112148e7e33b", null],
 ["point", "class_pr_node.html#a33a088c48aa4e8096ec27a855c110163", null],
 ["print", "class_pr_node.html#a1f1a2cc9dfc628f58efe93002b14e4a8", null],
 ["printUV", "class_pr_node.html#a069cc534c30ec1585b62eb2274c6fe", null],
 ["printXYZ", "class_pr_node.html#aae3da68a46e072df2ddcd4c4b7134a10", null],
 ["scan", "class_pr_node.html#a7fbc39d120b154a7ec36555061dab3ec", null],
 ["tr", "class_pr_node.html#ae9306b9497e98c93b895729879af2bb0", null],
 ["tr", "class_pr_node.html#ada7d8355857835362e85a9b487660b86", null],
 ["u", "class_pr_node.html#af32cd62efd6c560666a3ad4992e32c9d", null],
 ["u", "class_pr_node.html#affa228a107f5648aac56a8c79102b120", null],
 ["v", "class_pr_node.html#a7001ff1832d48c579c1b69deddb00db9", null],
 ["v", "class_pr_node.html#a8b59204989f4b21c4d88ea06f32e5203", null],
 ["x", "class_pr_node.html#a5a5865db2f78a4ccfd762634b4899e39", null],
 ["x", "class_pr_node.html#af1089d11e7ee670efb2007351752c1b6", null],
 ["y", "class_pr_node.html#ae837126218bc3668de1896a232be37bf", null],
 ["y", "class_pr_node.html#a5aa29384b673ee37acb1cf7f95059e5", null],
 ["z", "class_pr_node.html#ad6eec21be11c8ba6ee3a0e4fe9d4693a", null],
 ["z", "class_pr_node.html#a323e95d93b0be8200fbfb5cf0b98afc1", null]
]
```

Definition at line 1 of file class\_pr\_node.js.

## 30.562 doc/html/class\_pr\_organized\_points.js File Reference

### Variables

- var [class\\_pr\\_organized\\_points](#)

### 30.562.1 Variable Documentation

#### 30.562.1.1 var class\_pr\_organized\_points

Definition at line 1 of file class\_pr\_organized\_points.js.

## 30.563 doc/html/class\_pr\_p\_g\_node.js File Reference

### Variables

- var [class\\_pr\\_p\\_g\\_node](#)

### 30.563.1 Variable Documentation

#### 30.563.1.1 var class\_pr\_p\_g\_node

##### Initial value:

```
=
[
 ["PrPGNode", "class_pr_p_g_node.html#ae5c39ee78d7a13b8ae2ac418112a3c4", null],
 ["PrPGNode", "class_pr_p_g_node.html#aa609dd0d7d9317a5bb73dc0824086b92", null],
 ["PrPGNode", "class_pr_p_g_node.html#a7d8bfc8633cebc859a06391ee78d535e", null],
 ["ePrPGNode", "class_pr_p_g_node.html#aa2f5d24905661967c18e594767e65f4f", null],
 ["end", "class_pr_p_g_node.html#aed58c74a3dfea62e78c27438e263fd33", null],
 ["end", "class_pr_p_g_node.html#a17e4d0c73dcafa69d714cdb08e94387a", null],
 ["init", "class_pr_p_g_node.html#ae6a0717fc07928335c7466eaf4f44441", null],
 ["pnt", "class_pr_p_g_node.html#a6d7f992aaf478dc1ec5a1b0ff800e5f1", null],
 ["pnt", "class_pr_p_g_node.html#aa9ce8b832f049466aa8cffece2544de33", null],
 ["print", "class_pr_p_g_node.html#ad4c4ab93b1a9da4e5422a5a16dad37ae", null],
 ["printUV", "class_pr_p_g_node.html#a67dc61af63470497690c19fdb3ccd761", null],
 ["printXYZ", "class_pr_p_g_node.html#a5d79e6445006faff03b604684ecf7d4d", null],
 ["scan", "class_pr_p_g_node.html#a63a8e6a16953fbe0a2ab428d1cc948f3", null],
 ["u", "class_pr_p_g_node.html#adc28baec0a789a1d7e523cdb295b7754", null],
 ["u", "class_pr_p_g_node.html#aa05ad30e1b28840fa147e7a1cc7ae164", null],
 ["v", "class_pr_p_g_node.html#adaaf1049de852bdad29fdd715d39419", null],
 ["v", "class_pr_p_g_node.html#a8476c3c6b52627b627a31ec7f0b0d046", null],
 ["x", "class_pr_p_g_node.html#af56272e39fbc9e4b0273b83d9d9aa970", null],
 ["x", "class_pr_p_g_node.html#a373e004d4ae77aa11f3aee38ecb7985d", null],
 ["y", "class_pr_p_g_node.html#a73c1340c393a2d29be3ab2aca97dfb88", null],
 ["y", "class_pr_p_g_node.html#acf61cce8b39c5f5f30f26961f447e00f", null],
 ["z", "class_pr_p_g_node.html#a302fb7445d4022c990244d61768aaa22", null],
 ["z", "class_pr_p_g_node.html#aab8d9580b6112f5342804c503aa5aa78", null]
]
```

Definition at line 1 of file class\_pr\_p\_g\_node.js.

## 30.564 doc/html/class\_pr\_param\_triangulation.js File Reference

### Variables

- var [class\\_pr\\_param\\_triangulation](#)

### 30.564.1 Variable Documentation

#### 30.564.1.1 var class\_pr\_param\_triangulation

##### Initial value:

```
=
[
 ["PrParamTriangulation", "class_pr_param_triangulation.html#a1b09c2d6db5bf76c8bd8ffb4d32cfdb6", null]
 ,
 ["~PrParamTriangulation", "class_pr_param_triangulation.html#ad1f859a31b8952b06ef8b335aa5894fd", null
],
 ["attach", "class_pr_param_triangulation.html#ace2c563aaa857b9b1157fad40dbf095f", null],
 ["findTriangleContainingPoint", "class_pr_param_triangulation.html#a597e6f8e8bbda983ce0abb0b0c8d2685",
 null],
 ["getCoarseMesh", "class_pr_param_triangulation.html#a636d5a0f9e0eb3b0d27d6807c88b86d6", null],
 ["getFineMesh", "class_pr_param_triangulation.html#a0d9bcbal959fa0ae2bc9f1dafc1def0f", null],
 ["getParamPoint", "class_pr_param_triangulation.html#ad43dcfa0a43c0775f7433d74a5445c0a", null],
 ["getSurfPoint", "class_pr_param_triangulation.html#a09a872a911f0f7c7dbaecec2dd042bd1", null],
 ["getUV", "class_pr_param_triangulation.html#abd7bd619debd92e3ed7f553885e8e31", null],
 ["makeCorrespondences", "class_pr_param_triangulation.html#a6840f62f964b2405c1af121ece03d6c4", null],
 ["open", "class_pr_param_triangulation.html#a6d33b52c43d63bdf26064db11bac5aba", null],
 ["printBaseTriangles", "class_pr_param_triangulation.html#a39e63cae865241283494337bcf532b8c", null],
 ["save", "class_pr_param_triangulation.html#a4357af63c58ab2d7ca2d1dcd1a9ac5e6", null],
 ["triangleContainsPoint", "class_pr_param_triangulation.html#a322daf0457b7f54f6c359549cfebda05", null
]
]
```

Definition at line 1 of file class\_pr\_param\_triangulation.js.

## 30.565 doc/html/class\_pr\_parametrize\_bdy.js File Reference

### Variables

- var [class\\_pr\\_parametrize\\_bdy](#)

#### 30.565.1 Variable Documentation

##### 30.565.1.1 var class\_pr\_parametrize\_bdy

#### Initial value:

```
=
[
 ["PrParametrizeBdy", "class_pr_parametrize_bdy.html#af83d3a2afa9a786b69c49c6690f3289b", null],
 ["~PrParametrizeBdy", "class_pr_parametrize_bdy.html#a627f8fe265dc7ac62c79ac95101c660f", null],
 ["attach", "class_pr_parametrize_bdy.html#ab443522f631906821bac709deb346b36", null],
 ["boundaryLength", "class_pr_parametrize_bdy.html#a9caf971d4acabc686c9a375ac9780f5b", null],
 ["chord", "class_pr_parametrize_bdy.html#a09d84cf45677b66329f49cfc9961fb91", null],
 ["findBdyNode", "class_pr_parametrize_bdy.html#a76706ea5112df0c83b3ee94e3bbald9c", null],
 ["findCornersFromUV", "class_pr_parametrize_bdy.html#ab681f78edcb2cd451699a35f65baff64", null],
 ["findCornersFromXYZ", "class_pr_parametrize_bdy.html#ad9174f05f6ed621e4bbc93e956e714f7", null],
 ["getNextBdyNode", "class_pr_parametrize_bdy.html#ad01b4c92b3008c7f114724c7a7d60d52", null],
 ["parametrize", "class_pr_parametrize_bdy.html#ad07b22a033ba09ee3c5cf17eebled502", null],
 ["parametrize", "class_pr_parametrize_bdy.html#aa6c678f93bd412749362062e14c1789b", null],
 ["parametrizeSide", "class_pr_parametrize_bdy.html#a2f8f49ea0573e924c8cfc47ee91643f2", null],
 ["setParamKind", "class_pr_parametrize_bdy.html#ac3f08e8815786f179062e5dfcd6d2141", null],
 ["bdyparamtype_", "class_pr_parametrize_bdy.html#a0f10aae3995d2a4527fa4ac317e135c4", null],
 ["g_", "class_pr_parametrize_bdy.html#aflee487b52611457d3a81147f130021a", null],
 ["neighbours_", "class_pr_parametrize_bdy.html#a103dac1fc0b6bb14b310072af50798aa", null]
]
```

Definition at line 1 of file class\_pr\_parametrize\_bdy.js.

## 30.566 doc/html/class\_pr\_parametrize\_int.js File Reference

### Variables

- var [class\\_pr\\_parametrize\\_int](#)

#### 30.566.1 Variable Documentation

##### 30.566.1.1 var class\_pr\_parametrize\_int

#### Initial value:

```

=
[
 ["PrParametrizeInt", "class_pr_parametrize_int.html#a24cf818aea74952ccf98d515f2e3065e", null],
 ["~PrParametrizeInt", "class_pr_parametrize_int.html#ab4e3043d2490cb8ad8c107d0f65a5687", null],
 ["attach", "class_pr_parametrize_int.html#a25381707496c9d61bf40b999902e3402", null],
 ["computeWeights", "class_pr_parametrize_int.html#a2d863f767eafcd7cfac417e763384f86", null],
 ["findBarycentre", "class_pr_parametrize_int.html#a0ba460868d49d4ffa0afe10444fff1dd", null],
 ["findFixedPntsFromXYZ", "class_pr_parametrize_int.html#ad89fb4b2331e39317a085ef62a8744f5", null],
 ["getAllNeighbours", "class_pr_parametrize_int.html#ac384e242d3d9581e9b3ec762be728c53", null],
 ["getAllWeights", "class_pr_parametrize_int.html#a97ebfeb2702480841edb7c1cceffe406", null],
 ["getNumInt2Nghrs", "class_pr_parametrize_int.html#a85909ba2abad18917f0fbe20222baed3", null],
 ["getNumIntNghrs", "class_pr_parametrize_int.html#ad70ea861d7d8e723bd0dbfaadcaaaa13", null],
 ["isFixed", "class_pr_parametrize_int.html#ab6e93adb91358c18bfbe2a9ab12c4485", null],
 ["makeWeights", "class_pr_parametrize_int.html#ae4aa3fd1290361879b3f8d1906ad8a81", null],
 ["new_parametrize3d", "class_pr_parametrize_int.html#ac6e89a09aba3bf0918ce57ed3a4d453a", null],
 ["parametrize", "class_pr_parametrize_int.html#ab3308ded36a9fb72dal110b5a4a897e1", null],
 ["parametrize3d", "class_pr_parametrize_int.html#ab234f339fb378fb20facf911f0bacfb0", null],
 ["setBiCGTolerance", "class_pr_parametrize_int.html#ab9e09db9a0bfa4e0d3d62051abce3b5e", null],
 ["setStartVectorKind", "class_pr_parametrize_int.html#a35d88cfb426318723e7256c7a3804b10", null],
 ["smooth", "class_pr_parametrize_int.html#a6a17ada738a6a503ea37201ad58ba149", null],
 ["allNeighbours_", "class_pr_parametrize_int.html#a376adea9f4931c23acf517e5d06bf7d7", null],
 ["allWeights_", "class_pr_parametrize_int.html#a3b24d5ea6f4c97b04aa17e197f425328", null],
 ["g_", "class_pr_parametrize_int.html#a36919da9715f5f1625e062c6e6b27176", null],
 ["neighbours_", "class_pr_parametrize_int.html#a5b263f48a8e23507920c2c374dee353a", null],
 ["startvectortype_", "class_pr_parametrize_int.html#a3995ebc67005b13192fc020b3d847ac7", null],
 ["tolerance_", "class_pr_parametrize_int.html#a987817e47bcf9c3d9edd20492b95b493", null],
 ["weights_", "class_pr_parametrize_int.html#ae212bbd37bc09f289a0dbe076dab9b77", null]
]

```

Definition at line 1 of file class\_pr\_parametrize\_int.js.

## 30.567 doc/html/class\_pr\_parametrize\_mesh.js File Reference

### Variables

- var [class\\_pr\\_parametrize\\_mesh](#)

### 30.567.1 Variable Documentation

#### 30.567.1.1 var class\_pr\_parametrize\_mesh

#### Initial value:

```

=
[
 ["PrParametrizeMesh", "class_pr_parametrize_mesh.html#a3f70470d1231c4c93622b4a47ff719f5", null],
 ["~PrParametrizeMesh", "class_pr_parametrize_mesh.html#a0cab32c2ad55181a84da8541be25051d", null],
 ["attach", "class_pr_parametrize_mesh.html#a08c0b84e36568b0658e781276a935c9d", null],
 ["castRay", "class_pr_parametrize_mesh.html#a2777f2de0749ea44553f9364cb0d4fbf", null],
 ["edgeInPolygon", "class_pr_parametrize_mesh.html#a9bfb2ca8b0695348bc0efc623ee4blec", null],
 ["getConnectedNodes", "class_pr_parametrize_mesh.html#aa72b55874cac79c4f5f7881477fd884f", null],
 ["getInteriorNeighbours", "class_pr_parametrize_mesh.html#afff4c9666391c41b43622ce01d7a8909", null],
 ["getLeftNode", "class_pr_parametrize_mesh.html#a0b6949852805a6ba115bb24f3626d32b", null],
 ["getNextV", "class_pr_parametrize_mesh.html#a8766950e57fc6b06adc40db4277b9361", null],
 ["makePath", "class_pr_parametrize_mesh.html#a59b8c11febe89f74455da50664e349", null],
 ["makePolygon", "class_pr_parametrize_mesh.html#aa347cedd501066bfbb3a368cc6bd8134", null],
 ["makeSubTriangulation", "class_pr_parametrize_mesh.html#a05e09c6448074f7296fe94e470f8ba16", null],
 ["makeSubTriangulationFromTriangle", "class_pr_parametrize_mesh.html#aadebefb84d2a41173383b1c244e7e9a7", null],
 ["makeSubTriangulationInsidePolygon", "class_pr_parametrize_mesh.html#a05daf4e4f908667fe2c1ff8369e89a7b", null],
 ["parametrize", "class_pr_parametrize_mesh.html#a1892514375d679df356d416fc368637f", null],
 ["parametrizeSubTriangulation", "class_pr_parametrize_mesh.html#a0006fc40d52b533107e681efae7b3a3", null]
]

```

Definition at line 1 of file class\_pr\_parametrize\_mesh.js.

## 30.568 doc/html/class\_pr\_planar\_graph\_\_\_o\_p.js File Reference

### Variables

- var [class\\_pr\\_planar\\_graph\\_\\_\\_o\\_p](#)

### 30.568.1 Variable Documentation

#### 30.568.1.1 var class\_pr\_planar\_graph\_\_\_o\_p

##### Initial value:

```
=
[
 ["PrPlanarGraph_OP", "class_pr_planar_graph___o_p.html#acdd1ecf42cde4b5f6d050c376da9b4e4", null],
 ["PrPlanarGraph_OP", "class_pr_planar_graph___o_p.html#a4ef2049e2f56526f0eb003a0e52f9448", null],
 ["PrPlanarGraph_OP", "class_pr_planar_graph___o_p.html#ae832ec3a817be617fcef10421e7ae2f7", null],
 ["PrPlanarGraph_OP", "class_pr_planar_graph___o_p.html#aebdecc6e7bd1a4b9edc88bb6c1de711d", null],
 ["~PrPlanarGraph_OP", "class_pr_planar_graph___o_p.html#a399ecc526fb7260e9ee8893b212bc2dc", null],
 ["get3dNode", "class_pr_planar_graph___o_p.html#acece125574a3544bd0953b0dd9f30c8f", null],
 ["getNeighbours", "class_pr_planar_graph___o_p.html#ac3ba946d0f6544649f661bf70dfba187", null],
 ["getNumNodes", "class_pr_planar_graph___o_p.html#a3ecb2f22f125055704b0195db86229ba", null],
 ["getU", "class_pr_planar_graph___o_p.html#af0c9e3a090195b7c43c7efadd484f788", null],
 ["getV", "class_pr_planar_graph___o_p.html#a74d1e42a46193ab8f3750191a52f8732", null],
 ["isBoundary", "class_pr_planar_graph___o_p.html#ab6ab662d212b28b6c5c73708e5b5c92a", null],
 ["print", "class_pr_planar_graph___o_p.html#acc6cabed592875afc2b429207a8d9451", null],
 ["scan", "class_pr_planar_graph___o_p.html#a566ac1a6b7e23039f6484a94ec04dc16", null],
 ["scan2", "class_pr_planar_graph___o_p.html#a17f31d0d75eb90a6b1d8a7c3b791a719", null],
 ["set3dNode", "class_pr_planar_graph___o_p.html#af8c207c61a33753fe17b81487f736037", null],
 ["setU", "class_pr_planar_graph___o_p.html#a1beb13db1d9a2a4c26a590a17c666aad", null],
 ["setV", "class_pr_planar_graph___o_p.html#ac23c3ea9c628b7bc2e1b1c9da4c370b8", null]
]
```

Definition at line 1 of file class\_pr\_planar\_graph\_\_\_o\_p.js.

## 30.569 doc/html/class\_pr\_prewavelet\_\_\_f.js File Reference

### Variables

- var [class\\_pr\\_prewavelet\\_\\_\\_f](#)

### 30.569.1 Variable Documentation

#### 30.569.1.1 var class\_pr\_prewavelet\_\_\_f

##### Initial value:

```
=
[
 ["PrPrewavelet_F", "class_pr_prewavelet___f.html#aa85a80c864f6fffc2a33702626b467f2c", null],
 ["~PrPrewavelet_F", "class_pr_prewavelet___f.html#a1b324392f12544892e36bc62de2374a1", null],
 ["compose", "class_pr_prewavelet___f.html#aaa1ed3fe48ff6033d93aded872e9aa3c", null],
 ["decompose", "class_pr_prewavelet___f.html#a245bb4614348a94405e9b9f4839fb18d", null],
 ["setCGTolerance", "class_pr_prewavelet___f.html#aaf21ff9f24fa97fa8fb1b670792d13ee", null]
]
```

Definition at line 1 of file class\_pr\_prewavelet\_\_\_f.js.

## 30.570 doc/html/class\_pr\_prm\_e\_d\_d\_h\_l\_s.js File Reference

### Variables

- var [class\\_pr\\_prm\\_e\\_d\\_d\\_h\\_l\\_s](#)

### 30.570.1 Variable Documentation

#### 30.570.1.1 var class\_pr\_prm\_e\_d\_d\_h\_l\_s

##### Initial value:

```
=
[
 ["PrPrmEDDHLS", "class_pr_prm_e_d_d_h_l_s.html#a4aca895bc74e1869c633c543b21d5f56", null],
 ["~PrPrmEDDHLS", "class_pr_prm_e_d_d_h_l_s.html#ad8103707329ece41917b00bf0240e815", null],
 ["makeWeights", "class_pr_prm_e_d_d_h_l_s.html#a3d2b2dfa84fe08ed37ba3427f117519c", null]
]
```

Definition at line 1 of file class\_pr\_prm\_e\_d\_d\_h\_l\_s.js.

## 30.571 doc/html/class\_pr\_prm\_experimental.js File Reference

### Variables

- var [class\\_pr\\_prm\\_experimental](#)

### 30.571.1 Variable Documentation

#### 30.571.1.1 var class\_pr\_prm\_experimental

##### Initial value:

```
=
[
 ["PrPrmExperimental", "class_pr_prm_experimental.html#a17314ff9cdba3b8dae30b2ba6b3531d4", null],
 ["~PrPrmExperimental", "class_pr_prm_experimental.html#add1e25c2e5f4ae5ca319a2b58318d103", null],
 ["makeWeights", "class_pr_prm_experimental.html#a44da5f1f3af2f70a5e9c74e03cd75e3d", null]
]
```

Definition at line 1 of file class\_pr\_prm\_experimental.js.

## 30.572 doc/html/class\_pr\_prm\_least\_square.js File Reference

### Variables

- var [class\\_pr\\_prm\\_least\\_square](#)



### 30.572.1 Variable Documentation

#### 30.572.1.1 var class\_pr\_prm\_least\_square

**Initial value:**

```
=
[
 ["PrPrmLeastSquare", "class_pr_prm_least_square.html#ad71809cbda773a57063cd4673551c07e", null],
 ["~PrPrmLeastSquare", "class_pr_prm_least_square.html#a8afbcac333705371b86a5e1ff72afb3a", null],
 ["makeWeights", "class_pr_prm_least_square.html#adae63a2fe8e0378cc03fce5d197f9493", null]
]
```

Definition at line 1 of file class\_pr\_prm\_least\_square.js.

## 30.573 doc/html/class\_pr\_prm\_mean\_value.js File Reference

### Variables

- var [class\\_pr\\_prm\\_mean\\_value](#)

### 30.573.1 Variable Documentation

#### 30.573.1.1 var class\_pr\_prm\_mean\_value

**Initial value:**

```
=
[
 ["PrPrmMeanValue", "class_pr_prm_mean_value.html#a84fb043277622730787d8cfed9b47fff", null],
 ["~PrPrmMeanValue", "class_pr_prm_mean_value.html#a95cbcc3e056f333d69d8e233bb6b235b", null],
 ["makeWeights", "class_pr_prm_mean_value.html#a7b036031623cdc0502b4b9872a826e6a", null]
]
```

Definition at line 1 of file class\_pr\_prm\_mean\_value.js.

## 30.574 doc/html/class\_pr\_prm\_shp\_pres.js File Reference

### Variables

- var [class\\_pr\\_prm\\_shp\\_pres](#)

### 30.574.1 Variable Documentation

#### 30.574.1.1 var class\_pr\_prm\_shp\_pres

##### Initial value:

```
=
[
 ["PrPrmShpPres", "class_pr_prm_shp_pres.html#ab858d1904f8ee36165c95e9492d9d307", null],
 ["~PrPrmShpPres", "class_pr_prm_shp_pres.html#a71920b9b800b2c97f6ebd4354506c1", null],
 ["localParam", "class_pr_prm_shp_pres.html#ac623174dd58929231e2b9e24df4a51f9", null],
 ["makeWeights", "class_pr_prm_shp_pres.html#a663b1e47c88b2c7a979fd8cfcb659e44", null],
 ["alpha_", "class_pr_prm_shp_pres.html#a0d2c2caee95dc86a0d4e4a20daa6f407", null],
 ["len_", "class_pr_prm_shp_pres.html#a20fefbf6207938021d3a0fe033ac278a", null],
 ["u_", "class_pr_prm_shp_pres.html#ae67db719b7c94fed334b2d6711c224d7", null],
 ["v_", "class_pr_prm_shp_pres.html#a1b91934fd02891b562a62a7def09b645", null]
]
```

Definition at line 1 of file class\_pr\_prm\_shp\_pres.js.

## 30.575 doc/html/class\_pr\_prm\_surface.js File Reference

### Variables

- var [class\\_pr\\_prm\\_surface](#)

### 30.575.1 Variable Documentation

#### 30.575.1.1 var class\_pr\_prm\_surface

##### Initial value:

```
=
[
 ["PrPrmSurface", "class_pr_prm_surface.html#aff8bf946b045f35cb5397ffba54c418f", null],
 ["~PrPrmSurface", "class_pr_prm_surface.html#a2587ba04d6401f30727b05ca7eac67ae", null]
]
```

Definition at line 1 of file class\_pr\_prm\_surface.js.

## 30.576 doc/html/class\_pr\_prm\_sym\_mean\_value.js File Reference

### Variables

- var [class\\_pr\\_prm\\_sym\\_mean\\_value](#)

### 30.576.1 Variable Documentation

#### 30.576.1.1 var class\_pr\_prm\_sym\_mean\_value

**Initial value:**

```
=
[
 ["PrPrmSymMeanValue", "class_pr_prm_sym_mean_value.html#a0941f6c5e0d3335d9b8ddeb8e6d1baad", null],
 ["~PrPrmSymMeanValue", "class_pr_prm_sym_mean_value.html#a246a6c28c070ff36c28196e95ec2ef31", null],
 ["makeWeights", "class_pr_prm_sym_mean_value.html#a5b7950ef23ff8c00baffd9e15a05fe33", null],
 ["tanThetaOverTwo", "class_pr_prm_sym_mean_value.html#a9f07830f6d01f9430204d6b090da8823", null]
]
```

Definition at line 1 of file class\_pr\_prm\_sym\_mean\_value.js.

## 30.577 doc/html/class\_pr\_prm\_uniform.js File Reference

### Variables

- var [class\\_pr\\_prm\\_uniform](#)

### 30.577.1 Variable Documentation

#### 30.577.1.1 var class\_pr\_prm\_uniform

**Initial value:**

```
=
[
 ["PrPrmUniform", "class_pr_prm_uniform.html#a2e6c4bca0a0ec47428b4afad5d24ce33", null],
 ["~PrPrmUniform", "class_pr_prm_uniform.html#a783b707a6b0f3cc9b152f1b6ebba4a91", null]
]
```

Definition at line 1 of file class\_pr\_prm\_uniform.js.

## 30.578 doc/html/class\_pr\_prm\_wachspress.js File Reference

### Variables

- var [class\\_pr\\_prm\\_wachspress](#)

### 30.578.1 Variable Documentation

#### 30.578.1.1 var class\_pr\_prm\_wachspress

##### Initial value:

```
=
[
 ["PrPrmWachspress", "class_pr_prm_wachspress.html#a3c710e3936980bbb8de34036dbeae394", null],
 ["~PrPrmWachspress", "class_pr_prm_wachspress.html#a130125112b132d41232d3f1b97148731", null],
 ["makeWeights", "class_pr_prm_wachspress.html#a4264c4ddf0430c00cdadb4f873b48d5", null],
 ["tanThetaOverTwo", "class_pr_prm_wachspress.html#a3b09c67987bea5e0a6f2ac95755ca258", null]
]
```

Definition at line 1 of file class\_pr\_prm\_wachspress.js.

## 30.579 doc/html/class\_pr\_rectangular\_grid\_\_\_o\_p.js File Reference

### Variables

- var [class\\_pr\\_rectangular\\_grid\\_\\_\\_o\\_p](#)

### 30.579.1 Variable Documentation

#### 30.579.1.1 var class\_pr\_rectangular\_grid\_\_\_o\_p

Definition at line 1 of file class\_pr\_rectangular\_grid\_\_\_o\_p.js.

## 30.580 doc/html/class\_pr\_sub\_triangulation.js File Reference

### Variables

- var [class\\_pr\\_sub\\_triangulation](#)

### 30.580.1 Variable Documentation

#### 30.580.1.1 var class\_pr\_sub\_triangulation

##### Initial value:

```
=
[
 ["PrSubTriangulation", "class_pr_sub_triangulation.html#acae41fa48ebba273c83304a7fe2dff5", null],
 ["PrSubTriangulation", "class_pr_sub_triangulation.html#ae1d252f0333ffdbfb942393efc3b40a0", null],
 ["~PrSubTriangulation", "class_pr_sub_triangulation.html#a2b2923c2714c7f28c15de4e04b8974ac", null],
 ["attach", "class_pr_sub_triangulation.html#a6a0f9f273be6eb137e79adea9a393298", null],
 ["findNumBdyNodes", "class_pr_sub_triangulation.html#ae65468c1c45af012950f0dc812983627", null],
 ["get3dNode", "class_pr_sub_triangulation.html#a19f7d1ec142380461fea4e6003e06829", null],
 ["getGlobalIndex", "class_pr_sub_triangulation.html#a7cd54dad5740c697d496b7912913368d", null],
 ["getLocalIndex", "class_pr_sub_triangulation.html#adb9f0ec2feb4a500dd8df9a0730e8ec4", null],
 ["getNeighbours", "class_pr_sub_triangulation.html#ad6530d164037d105410a2226c42aa46a", null],
 ["getNeighbourTriangles", "class_pr_sub_triangulation.html#a352f903c564f38e460efe48d17e5eca6", null],
 ["getNumNodes", "class_pr_sub_triangulation.html#a3b7e78ebb6858bc5483bf597c9115433", null],
 ["getTriangleIndices", "class_pr_sub_triangulation.html#a38d983c887a578d0526c0d447cb4ab99", null],
 ["getU", "class_pr_sub_triangulation.html#a4ae8edeca415b480b73ddec89d7cbae1", null],
 ["getV", "class_pr_sub_triangulation.html#a98aa8d4b33d168e73c6e544eea54154f", null],
 ["initialize", "class_pr_sub_triangulation.html#acb9317f32e5707d5609dd557f83e18c1", null],
 ["isBoundary", "class_pr_sub_triangulation.html#ac9264ba7e7c69f5d3eace29974a63d2f", null],
 ["set3dNode", "class_pr_sub_triangulation.html#ab50ffc1774c9ff1a118c0dc237bcc0e8", null],
 ["setU", "class_pr_sub_triangulation.html#ae98031d8c7f02e66591a3c43243697dd", null],
 ["setV", "class_pr_sub_triangulation.html#a62f00c16103a9c2a2933be4a400528ef", null],
 ["triangleInThis", "class_pr_sub_triangulation.html#a6cb4ec410f7622613c88c9d23f1e161d", null]
]
```

Definition at line 1 of file class\_pr\_sub\_triangulation.js.

## 30.581 doc/html/class\_pr\_thin.js File Reference

### Variables

- var [class\\_pr\\_thin](#)

### 30.581.1 Variable Documentation

#### 30.581.1.1 var class\_pr\_thin

#### Initial value:

```
=
[
 ["PrThin", "class_pr_thin.html#a7d094956ec19027e5a0bd83bb8961bae", null],
 ["~PrThin", "class_pr_thin.html#a31c43bf3df499c37d4fe5cbf3cc844df", null],
 ["attach", "class_pr_thin.html#ab6d927ae13c78b834c8682ac378f8d45", null],
 ["findError", "class_pr_thin.html#a87454694b580dce6614277a29cd2ed8a", null],
 ["findNearestNeighbour", "class_pr_thin.html#a0255645d4dde7222c3994d0c721282b2", null],
 ["halfEdgeCollapse", "class_pr_thin.html#a8562782f8ebd8645b6ef5dee869f913e", null],
 ["makeHeap", "class_pr_thin.html#a34b5f6d53ab702f83680ef2a3063ac3e", null],
 ["printHeap", "class_pr_thin.html#ab747d680498fab1237c14a9e0bd5e154", null],
 ["removePoint", "class_pr_thin.html#a13252df15cbe0d681da674e4f273bb46", null],
 ["setError", "class_pr_thin.html#a29132e6d0c24ff6562772e10c8e52114", null],
 ["setStepNumber", "class_pr_thin.html#a22dd0842274c5d9c935d99cb069b2c24", null],
 ["thin", "class_pr_thin.html#aea4838b641ddc6727f51e43a03be1d5b", null],
 ["error_", "class_pr_thin.html#a1cc62841af30289e446e2d7b3ac0170a", null],
 ["heap_", "class_pr_thin.html#a6c903d183c116bcf9c8efde3a7e1d1d1", null],
 ["nlist_ptr", "class_pr_thin.html#ac9cc59553e50f6ebc73edcbd6cf9a55a", null],
 ["steps_", "class_pr_thin.html#a9ca72625ae44bb1cf6fe5ef07b502b1a", null],
 ["tlist_ptr", "class_pr_thin.html#a3a369157ba7218c70863f042afa1643c", null],
 ["trn_", "class_pr_thin.html#a61c134e572850c889c6e1ec67d879bf7", null]
]
```

Definition at line 1 of file class\_pr\_thin.js.

## 30.582 doc/html/class\_pr\_threshold.js File Reference

### Variables

- var [class\\_pr\\_threshold](#)

### 30.582.1 Variable Documentation

#### 30.582.1.1 var class\_pr\_threshold

#### Initial value:

```
=
[
 ["PrThreshold", "class_pr_threshold.html#ac492cec733aa0796b73d4f19e09b900e", null],
 ["~PrThreshold", "class_pr_threshold.html#adee2c54b802b8c7256d9300c0a6fbbf0", null],
 ["attach", "class_pr_threshold.html#a9316864e4bae15474367f329bfff1c5", null],
 ["averageAbsValue", "class_pr_threshold.html#a831eec1d2955d601036ce58a8af2112f", null],
 ["findThreshold", "class_pr_threshold.html#ad6c76d88dd579e7cfdb791cd444df958", null],
 ["leaveLevel", "class_pr_threshold.html#a816cbbad25faf259db527bf3c302bc5e", null],
 ["maxNorm", "class_pr_threshold.html#a693025401fd91e94a8dd94037b1dcb02", null],
 ["setThreshold", "class_pr_threshold.html#a1271a04eb09a374d158c0ec5e56aee94", null],
 ["threshold", "class_pr_threshold.html#a0670f0ae29c865b724e978e3d864c0fd", null],
 ["thresholdByCompRate", "class_pr_threshold.html#a5d7af83376ba54cbfbed5d3c501df406", null],
 ["triangleNorm", "class_pr_threshold.html#abc5b829433f418cf95e63026cc340293", null],
 ["truncateLevel", "class_pr_threshold.html#a012ab15d37653fcc76869cb5ac370de0", null],
 ["weightedL2Norm", "class_pr_threshold.html#a49faf4a4837b4986080824fe64df929c", null],
 ["_", "class_pr_threshold.html#a569fc7ba663b6a439069fe47f3babb5c", null],
 ["threshold_", "class_pr_threshold.html#a684efc618a0e74f8d0ed3ac08a837db7", null]
]
```

Definition at line 1 of file class\_pr\_threshold.js.

## 30.583 doc/html/class\_pr\_triangle.js File Reference

### Variables

- var [class\\_pr\\_triangle](#)

### 30.583.1 Variable Documentation

#### 30.583.1.1 var class\_pr\_triangle

Definition at line 1 of file class\_pr\_triangle.js.

## 30.584 doc/html/class\_pr\_triangulation\_\_\_n\_t.js File Reference

### Variables

- var [class\\_pr\\_triangulation\\_\\_\\_n\\_t](#)

### 30.584.1 Variable Documentation

#### 30.584.1.1 var class\_pr\_triangulation\_\_\_n\_t

#### Initial value:

```
=
[
 ["PrTriangulation_NT", "class_pr_triangulation___n_t.html#af157aed886430f59184492833aafe4da", null],
 ["PrTriangulation_NT", "class_pr_triangulation___n_t.html#af9373ad1c5343f05d695003e4f2a48dd", null],
 ["~PrTriangulation_NT", "class_pr_triangulation___n_t.html#a525165ba2e98871226cfc9a0d3f6a5c5", null],
 ["getFinestLevel", "class_pr_triangulation___n_t.html#a49069a4eb0308c4590064b5f141019bc", null],
 ["getLevel", "class_pr_triangulation___n_t.html#a3d781299cb59ab6c4d081b784dd68f14", null],
 ["getNeighbours", "class_pr_triangulation___n_t.html#a0d82289eb370854b5ba1244092eb7ec4", null],
 ["getNumNodes", "class_pr_triangulation___n_t.html#a617b4f8515733a8db93133649bf68f6e", null],
 ["getTopLevel", "class_pr_triangulation___n_t.html#a98d7a7833e15fe6c3127e27f58ce332d", null],
 ["getX", "class_pr_triangulation___n_t.html#aef076938bde917aea39d89f192266eb6", null],
 ["getY", "class_pr_triangulation___n_t.html#ac63bcb7a301116a4898645f8cef0b78b", null],
 ["getZ", "class_pr_triangulation___n_t.html#ae13f8f97dc07538940d3240f643d14c5", null],
 ["isBoundary", "class_pr_triangulation___n_t.html#aeb1c6afe089af3c7c1708193ale863dc", null],
 ["lift", "class_pr_triangulation___n_t.html#a3347c0d19b44b3b2e5ff1f420f17b466", null],
 ["refine", "class_pr_triangulation___n_t.html#a49dd5eb2bd50cd976c3bbe98638fc904", null],
 ["setX", "class_pr_triangulation___n_t.html#a5a7096a8b586e941217bbde751d09572", null],
 ["setY", "class_pr_triangulation___n_t.html#af4ceec67cc9bd0f8e512cec2054b589b", null],
 ["setZ", "class_pr_triangulation___n_t.html#ac436058e9e9849b298baecb66d38b43d", null]
]
```

Definition at line 1 of file class\_pr\_triangulation\_\_\_n\_t.js.

## 30.585 doc/html/class\_pr\_triangulation\_\_\_o\_p.js File Reference

### Variables

- var [class\\_pr\\_triangulation\\_\\_\\_o\\_p](#)

### 30.585.1 Variable Documentation

#### 30.585.1.1 var class\_pr\_triangulation\_\_\_o\_p

Definition at line 1 of file class\_pr\_triangulation\_\_\_o\_p.js.

## 30.586 doc/html/class\_pr\_unorganized\_\_\_o\_p.js File Reference

### Variables

- var [class\\_pr\\_unorganized\\_\\_\\_o\\_p](#)

### 30.586.1 Variable Documentation

#### 30.586.1.1 var class\_pr\_unorganized\_\_\_o\_p

#### Initial value:

```
=
[
 ["PrUnorganized_OP", "class_pr_unorganized___o_p.html#a6af54585405b6c3d0bc9a3887aa38476", null],
 ["PrUnorganized_OP", "class_pr_unorganized___o_p.html#a8d8886f223a6ddd397254ce560e4335b", null],
 ["~PrUnorganized_OP", "class_pr_unorganized___o_p.html#a4664ec62817650d35f49d0d6f435a2bb", null],
 ["get3dNode", "class_pr_unorganized___o_p.html#ae5dc1474062e533a55bc30a3b4698849", null],
 ["getNeighbours", "class_pr_unorganized___o_p.html#ac6cead07bb2a3df8d0220a442929f09c", null],
 ["getNumNodes", "class_pr_unorganized___o_p.html#a6f45913608f66000f774f73cd06295d3", null],
 ["getRadius", "class_pr_unorganized___o_p.html#af063ca3220ff5e5fa5623e3420589513", null],
 ["getU", "class_pr_unorganized___o_p.html#af2adc4bc1012c3571af302963be2141e", null],
 ["getV", "class_pr_unorganized___o_p.html#a905a2425d54e32398ff2b271a984fec6", null],
 ["isBoundary", "class_pr_unorganized___o_p.html#a5a3ecf644836e23f79c44b97b5818ae2", null],
 ["print", "class_pr_unorganized___o_p.html#a43d2fa9eb4188818bc9270cba6d7c159", null],
 ["printRawData", "class_pr_unorganized___o_p.html#ab334661004f656184cb4a37fc7ba83ed", null],
 ["scan", "class_pr_unorganized___o_p.html#ad76a888ca9b4f217835dba42053dae3d", null],
 ["scanRawData", "class_pr_unorganized___o_p.html#a0f09baa27e8cdcd4ed570f2df9203577", null],
 ["set3dNode", "class_pr_unorganized___o_p.html#a4de7406aa62ded74b619dcf0ffdb2d7a", null],
 ["setKNearest", "class_pr_unorganized___o_p.html#a19d9492977b44d34175d394c33c0f6d7", null],
 ["setRadius", "class_pr_unorganized___o_p.html#ad4e9cd1ae3a3079fa2bf2e0dafc24529", null],
 ["setU", "class_pr_unorganized___o_p.html#ac2ba839dad21aec851533314ed5fe802", null],
 ["setV", "class_pr_unorganized___o_p.html#afc98367785c160eaffdc6820e912c034", null],
 ["useK", "class_pr_unorganized___o_p.html#a762cb0156e940b783262ba5acedc3af4", null],
 ["useRadius", "class_pr_unorganized___o_p.html#a5458b764bede44fa0b1a1a032aef275f", null]
]
```

Definition at line 1 of file class\_pr\_unorganized\_\_\_o\_p.js.

## 30.587 doc/html/class\_pr\_vec.js File Reference

### Variables

- var [class\\_pr\\_vec](#)

### 30.587.1 Variable Documentation

#### 30.587.1.1 var class\_pr\_vec

##### Initial value:

```
=
[
 ["PrVec", "class_pr_vec.html#ac6139765e0a7a175aeda55405c3e0c38", null],
 ["PrVec", "class_pr_vec.html#abef46803bbe9c53b8328d74b52334757", null],
 ["PrVec", "class_pr_vec.html#a09e3eb5ce8a0f762a32014586af9aaf4", null],
 ["inner", "class_pr_vec.html#a4d14a8bb523f541ea4de6dcfe8565f8e", null],
 ["operator()", "class_pr_vec.html#ad55487ab0c699f7dd9a3657492bcc368", null],
 ["operator()", "class_pr_vec.html#ad65a3d6d217b30c65f1e57e5d959b9dd", null],
 ["operator[]", "class_pr_vec.html#ae5841830f8c7bdd95c684b61df8d614c", null],
 ["operator[]", "class_pr_vec.html#a528e2d56cda18a3f94d88597db6e84d4", null],
 ["print", "class_pr_vec.html#aa7204d6b00b252bc54e486ff3d209664", null],
 ["read", "class_pr_vec.html#a5bfbbe99e8a711b5d93908f4ce3255d8", null],
 ["redim", "class_pr_vec.html#a4ef99ea22a657f5fdee5e953605ca8de", null],
 ["size", "class_pr_vec.html#a6293878c1fe04ef3257fb83350495bd7", null],
 ["a_", "class_pr_vec.html#afda074f05b8bb9e81086ba29f31f2762", null]
]
```

Definition at line 1 of file class\_pr\_vec.js.

## 30.588 doc/html/class\_print\_counter.js File Reference

### Variables

- var [class\\_print\\_counter](#)

### 30.588.1 Variable Documentation

#### 30.588.1.1 var class\_print\_counter

##### Initial value:

```
=
[
 ["~PrintCounter", "class_print_counter.html#a44daffd64a249f3ca428087f17f4953d", null],
 ["PrintCounter", "class_print_counter.html#a0d3886579d29194cd6d82f677a927cf1", null],
 ["operator++", "class_print_counter.html#a5cde49b2daaa853c6e4bbf14001def95", null]
]
```

Definition at line 1 of file class\_print\_counter.js.

## 30.589 doc/html/class\_r\_b\_d\_\_\_c\_o\_m\_m\_o\_n\_1\_1\_bad\_alloc.js File Reference

### Variables

- var [class\\_r\\_b\\_d\\_\\_\\_c\\_o\\_m\\_m\\_o\\_n\\_1\\_1\\_bad\\_alloc](#)



### 30.589.1 Variable Documentation

#### 30.589.1.1 var class\_r\_b\_d\_\_\_c\_o\_m\_m\_o\_n\_1\_1\_bad\_alloc

**Initial value:**

```
=
[
 ["Bad_alloc", "class_r_b_d___c_o_m_m_o_n_1_1_bad_alloc.html#a696395534c332086b90ace212721ddd2", null
]
]
```

Definition at line 1 of file class\_r\_b\_d\_\_\_c\_o\_m\_m\_o\_n\_1\_1\_bad\_alloc.js.

## 30.590 doc/html/class\_r\_b\_d\_\_\_c\_o\_m\_m\_o\_n\_1\_1\_base\_exception.js File Reference

### Variables

- var [class\\_r\\_b\\_d\\_\\_\\_c\\_o\\_m\\_m\\_o\\_n\\_1\\_1\\_base\\_exception](#)

### 30.590.1 Variable Documentation

#### 30.590.1.1 var class\_r\_b\_d\_\_\_c\_o\_m\_m\_o\_n\_1\_1\_base\_exception

**Initial value:**

```
=
[
 ["BaseException", "class_r_b_d___c_o_m_m_o_n_1_1_base_exception.html#a275b7fc020a5786cdcafcb325b550426", null]
]
```

Definition at line 1 of file class\_r\_b\_d\_\_\_c\_o\_m\_m\_o\_n\_1\_1\_base\_exception.js.

## 30.591 doc/html/class\_r\_b\_d\_\_\_c\_o\_m\_m\_o\_n\_1\_1\_domain\_\_error.js File Reference

### Variables

- var [class\\_r\\_b\\_d\\_\\_\\_c\\_o\\_m\\_m\\_o\\_n\\_1\\_1\\_domain\\_\\_error](#)

### 30.591.1 Variable Documentation

#### 30.591.1.1 var class\_r\_b\_d\_\_\_c\_o\_m\_m\_o\_n\_1\_1\_domain\_\_error

**Initial value:**

```
=
[
 ["Domain_error", "class_r_b_d___c_o_m_m_o_n_1_1_domain__error.html#a95fa4017ebb608fee2527948419e087b", null]
]
```

Definition at line 1 of file class\_r\_b\_d\_\_\_c\_o\_m\_m\_o\_n\_1\_1\_domain\_\_error.js.

## 30.592 doc/html/class\_r\_b\_d\_\_\_c\_o\_m\_m\_o\_n\_1\_1\_invalid\_\_argument.js File Reference

### Variables

- var [class\\_r\\_b\\_d\\_\\_\\_c\\_o\\_m\\_m\\_o\\_n\\_1\\_1\\_invalid\\_\\_argument](#)

### 30.592.1 Variable Documentation

#### 30.592.1.1 var class\_r\_b\_d\_\_\_c\_o\_m\_m\_o\_n\_1\_1\_invalid\_\_argument

#### Initial value:

```
=
[
 ["Invalid_argument", "
 class_r_b_d___c_o_m_m_o_n_1_1_invalid__argument.html#a3a3918a4ae0bf2f28cb3446f5f2c5103", null]
]
```

Definition at line 1 of file class\_r\_b\_d\_\_\_c\_o\_m\_m\_o\_n\_1\_1\_invalid\_\_argument.js.

## 30.593 doc/html/class\_r\_b\_d\_\_\_c\_o\_m\_m\_o\_n\_1\_1\_janitor.js File Reference

### Variables

- var [class\\_r\\_b\\_d\\_\\_\\_c\\_o\\_m\\_m\\_o\\_n\\_1\\_1\\_janitor](#)

### 30.593.1 Variable Documentation

#### 30.593.1.1 var class\_r\_b\_d\_\_\_c\_o\_m\_m\_o\_n\_1\_1\_janitor

#### Initial value:

```
=
[
 ["Janitor", "class_r_b_d___c_o_m_m_o_n_1_1_janitor.html#ace39d4394d9bfec39ef5a44eafc70f88", null],
 ["~Janitor", "class_r_b_d___c_o_m_m_o_n_1_1_janitor.html#a16a5acc6e8dbbf97cd97d8a0b2b3a880", null],
 ["CleanUp", "class_r_b_d___c_o_m_m_o_n_1_1_janitor.html#ad98693ba2eaf1ac688565a91d5307cb4", null]
]
```

Definition at line 1 of file class\_r\_b\_d\_\_\_c\_o\_m\_m\_o\_n\_1\_1\_janitor.js.

## 30.594 doc/html/class\_r\_b\_d\_\_\_c\_o\_m\_m\_o\_n\_1\_1\_length\_\_error.js File Reference

### Variables

- var [class\\_r\\_b\\_d\\_\\_\\_c\\_o\\_m\\_m\\_o\\_n\\_1\\_1\\_length\\_\\_error](#)

### 30.594.1 Variable Documentation

#### 30.594.1.1 var class\_r\_b\_d\_\_\_c\_o\_m\_m\_o\_n\_1\_1\_length\_\_error

**Initial value:**

```
=
[
 ["Length_error", "class_r_b_d___c_o_m_m_o_n_1_1_length__error.html#aaa6de69c775aee8e7fbf15521ffa5733",
 null]
]
```

Definition at line 1 of file class\_r\_b\_d\_\_\_c\_o\_m\_m\_o\_n\_1\_1\_length\_\_error.js.

## 30.595 doc/html/class\_r\_b\_d\_\_\_c\_o\_m\_m\_o\_n\_1\_1\_logic\_\_error.js File Reference

### Variables

- var [class\\_r\\_b\\_d\\_\\_\\_c\\_o\\_m\\_m\\_o\\_n\\_1\\_1\\_logic\\_\\_error](#)

### 30.595.1 Variable Documentation

#### 30.595.1.1 var class\_r\_b\_d\_\_\_c\_o\_m\_m\_o\_n\_1\_1\_logic\_\_error

**Initial value:**

```
=
[
 ["Logic_error", "class_r_b_d___c_o_m_m_o_n_1_1_logic__error.html#a8ebe36bc99c7e80f23b5f1ef4d1510be",
 null]
]
```

Definition at line 1 of file class\_r\_b\_d\_\_\_c\_o\_m\_m\_o\_n\_1\_1\_logic\_\_error.js.

## 30.596 doc/html/class\_r\_b\_d\_\_\_c\_o\_m\_m\_o\_n\_1\_1\_one\_dim\_solve.js File Reference

### Variables

- var [class\\_r\\_b\\_d\\_\\_\\_c\\_o\\_m\\_m\\_o\\_n\\_1\\_1\\_one\\_dim\\_solve](#)

### 30.596.1 Variable Documentation

#### 30.596.1.1 var class\_r\_b\_d\_\_\_c\_o\_m\_m\_o\_n\_1\_1\_one\_dim\_solve

**Initial value:**

```
=
[
 ["OneDimSolve", "class_r_b_d___c_o_m_m_o_n_1_1_one_dim_solve.html#abcce09038399f61e0867e96cc1278980",
 null],
 ["Solve", "class_r_b_d___c_o_m_m_o_n_1_1_one_dim_solve.html#a1bfcad6abecc3935ce6f3dd6f03bf4fd", null]
]
```

Definition at line 1 of file class\_r\_b\_d\_\_\_c\_o\_m\_m\_o\_n\_1\_1\_one\_dim\_solve.js.

## 30.597 doc/html/class\_r\_b\_d\_\_\_c\_o\_m\_m\_o\_n\_1\_1\_out\_of\_range.js File Reference

### Variables

- var [class\\_r\\_b\\_d\\_\\_\\_c\\_o\\_m\\_m\\_o\\_n\\_1\\_1\\_out\\_of\\_range](#)

### 30.597.1 Variable Documentation

#### 30.597.1.1 var class\_r\_b\_d\_\_\_c\_o\_m\_m\_o\_n\_1\_1\_out\_of\_range

##### Initial value:

```
=
[
 ["Out_of_range", "class_r_b_d___c_o_m_m_o_n_1_1_out_of_range.html#a1016261ef3ac9d5eff1f38a95e02da79"
 , null]
]
```

Definition at line 1 of file class\_r\_b\_d\_\_\_c\_o\_m\_m\_o\_n\_1\_1\_out\_of\_range.js.

## 30.598 doc/html/class\_r\_b\_d\_\_\_c\_o\_m\_m\_o\_n\_1\_1\_overflow\_error.js File Reference

### Variables

- var [class\\_r\\_b\\_d\\_\\_\\_c\\_o\\_m\\_m\\_o\\_n\\_1\\_1\\_overflow\\_error](#)

### 30.598.1 Variable Documentation

#### 30.598.1.1 var class\_r\_b\_d\_\_\_c\_o\_m\_m\_o\_n\_1\_1\_overflow\_error

##### Initial value:

```
=
[
 ["Overflow_error", "
 class_r_b_d___c_o_m_m_o_n_1_1_overflow_error.html#ae071d4588c5143776418c00d2ce93b8d", null]
]
```

Definition at line 1 of file class\_r\_b\_d\_\_\_c\_o\_m\_m\_o\_n\_1\_1\_overflow\_error.js.

## 30.599 doc/html/class\_r\_b\_d\_\_\_c\_o\_m\_m\_o\_n\_1\_1\_r1\_\_\_r1.js File Reference

### Variables

- var [class\\_r\\_b\\_d\\_\\_\\_c\\_o\\_m\\_m\\_o\\_n\\_1\\_1\\_r1\\_\\_\\_r1](#)

### 30.599.1 Variable Documentation

#### 30.599.1.1 var class\_r\_b\_d\_\_\_c\_o\_m\_m\_o\_n\_1\_1\_r1\_\_\_r1

##### Initial value:

```
=
[
 ["R1_R1", "class_r_b_d___c_o_m_m_o_n_1_1_r1___r1.html#a278e42bea46cadb3e7579f8357975e34", null],
 ["~R1_R1", "class_r_b_d___c_o_m_m_o_n_1_1_r1___r1.html#abea87c3752ac41bc5454a550e7b3bbde", null],
 ["IsValid", "class_r_b_d___c_o_m_m_o_n_1_1_r1___r1.html#a27068635b33bce4ea9c3c3289b9e5348", null],
 ["operator Real", "class_r_b_d___c_o_m_m_o_n_1_1_r1___r1.html#acff1830109f596a46700b4912dd37f66", null],
],
 ["operator()", "class_r_b_d___c_o_m_m_o_n_1_1_r1___r1.html#ae745606ca696d139fde52cc6ae30196c", null],
 ["operator()", "class_r_b_d___c_o_m_m_o_n_1_1_r1___r1.html#a5d09b0f9772a83f9ad1931c26078355b", null],
 ["Set", "class_r_b_d___c_o_m_m_o_n_1_1_r1___r1.html#aa464970d29a49718c732350474671d77", null],
 ["maxX", "class_r_b_d___c_o_m_m_o_n_1_1_r1___r1.html#a80d4dbe4d3a8b542eef0815e305c126d", null],
 ["maxXinf", "class_r_b_d___c_o_m_m_o_n_1_1_r1___r1.html#af3ddf907eb8fe8b8d245ac42dadd1564", null],
 ["minX", "class_r_b_d___c_o_m_m_o_n_1_1_r1___r1.html#a0e21d76c982beca3b12e4887ded817c5", null],
 ["minXinf", "class_r_b_d___c_o_m_m_o_n_1_1_r1___r1.html#addd2ebaaafd948a822574b4d154501cfe", null],
 ["x", "class_r_b_d___c_o_m_m_o_n_1_1_r1___r1.html#a4812b378711ea5f78a5441b96134f569", null],
 ["xSet", "class_r_b_d___c_o_m_m_o_n_1_1_r1___r1.html#a9c4468e6a3619646aba36b92dd36b681", null]
]
```

Definition at line 1 of file class\_r\_b\_d\_\_\_c\_o\_m\_m\_o\_n\_1\_1\_r1\_\_\_r1.js.

## 30.600 doc/html/class\_r\_b\_d\_\_\_c\_o\_m\_m\_o\_n\_1\_1\_range\_\_error.js File Reference

### Variables

- var [class\\_r\\_b\\_d\\_\\_\\_c\\_o\\_m\\_m\\_o\\_n\\_1\\_1\\_range\\_\\_error](#)

### 30.600.1 Variable Documentation

#### 30.600.1.1 var class\_r\_b\_d\_\_\_c\_o\_m\_m\_o\_n\_1\_1\_range\_\_error

##### Initial value:

```
=
[
 ["Range_error", "class_r_b_d___c_o_m_m_o_n_1_1_range__error.html#a8bc12b7074abb363df50ddb82c1c9db", null]
]
```

Definition at line 1 of file class\_r\_b\_d\_\_\_c\_o\_m\_m\_o\_n\_1\_1\_range\_\_error.js.

## 30.601 doc/html/class\_r\_b\_d\_\_\_c\_o\_m\_m\_o\_n\_1\_1\_runtime\_\_error.js File Reference

### Variables

- var [class\\_r\\_b\\_d\\_\\_\\_c\\_o\\_m\\_m\\_o\\_n\\_1\\_1\\_runtime\\_\\_error](#)

### 30.601.1 Variable Documentation

30.601.1.1 `var class_r_b_d__c_o_m_m_o_n_1_1_runtime__error`

#### Initial value:

```
=
[
 ["Runtime_error", "class_r_b_d__c_o_m_m_o_n_1_1_runtime__error.html#a8b242deab8e1b4a25d24bbfde345d8c2", null]
]
```

Definition at line 1 of file `class_r_b_d__c_o_m_m_o_n_1_1_runtime__error.js`.

## 30.602 `doc/html/class_r_b_d__c_o_m_m_o_n_1_1_solution_exception.js` File Reference

### Variables

- `var class_r_b_d__c_o_m_m_o_n_1_1_solution_exception`

### 30.602.1 Variable Documentation

30.602.1.1 `var class_r_b_d__c_o_m_m_o_n_1_1_solution_exception`

#### Initial value:

```
=
[
 ["SolutionException", "class_r_b_d__c_o_m_m_o_n_1_1_solution_exception.html#a8f8e2990ffdaed96bf24083c6a16d2fe", null]
]
```

Definition at line 1 of file `class_r_b_d__c_o_m_m_o_n_1_1_solution_exception.js`.

## 30.603 `doc/html/class_r_b_d__c_o_m_m_o_n_1_1_tracer.js` File Reference

### Variables

- `var class_r_b_d__c_o_m_m_o_n_1_1_tracer`

### 30.603.1 Variable Documentation

#### 30.603.1.1 var class\_r\_b\_d\_\_\_c\_o\_m\_m\_o\_n\_1\_1\_tracer

##### Initial value:

```
=
[
 ["Tracer", "class_r_b_d___c_o_m_m_o_n_1_1_tracer.html#a19dca2902d689ed75b6dc96f5a5dc362", null],
 ["~Tracer", "class_r_b_d___c_o_m_m_o_n_1_1_tracer.html#ac2d6d4fb4949e941a49fb6a6afd15d60", null],
 ["ReName", "class_r_b_d___c_o_m_m_o_n_1_1_tracer.html#afb685530404bf08109ee2be8dc388f90", null],
 ["BaseException", "class_r_b_d___c_o_m_m_o_n_1_1_tracer.html#acc29c9a717ece42fb764f4a48a04b6cb", null
]
]
```

Definition at line 1 of file class\_r\_b\_d\_\_\_c\_o\_m\_m\_o\_n\_1\_1\_tracer.js.

## 30.604 doc/html/class\_rect\_matrix\_col.js File Reference

### Variables

- var [class\\_rect\\_matrix\\_col](#)

### 30.604.1 Variable Documentation

#### 30.604.1.1 var class\_rect\_matrix\_col

##### Initial value:

```
=
[
 ["RectMatrixCol", "class_rect_matrix_col.html#af8b4292bc6eedae5bc641e860c5ccfc8", null],
 ["RectMatrixCol", "class_rect_matrix_col.html#aec39989b21bcd9a7e02332a17f42d49f", null],
 ["Down", "class_rect_matrix_col.html#ae47c1ddf71053f44b6a98fb9ecb5a192", null],
 ["Left", "class_rect_matrix_col.html#a749656e672061aca5b6be61cb2ce1f2d", null],
 ["Reset", "class_rect_matrix_col.html#a64a1b6471b2e8c1a9e8eb2495934df7f", null],
 ["Reset", "class_rect_matrix_col.html#ad12aac0a25f4a6d0130df853a4db67e6", null],
 ["Right", "class_rect_matrix_col.html#aed238ccd4b925b8ac753f8eca4b564c0", null],
 ["Up", "class_rect_matrix_col.html#a57b2bb743fb4fa46bfc887a335f5b338", null],
 ["ComplexScale", "class_rect_matrix_col.html#ae131888d49cb5f48944aa641f5998af5", null],
 ["Rotate", "class_rect_matrix_col.html#a23acf0c66955f76462c923517f1f64be", null]
]
```

Definition at line 1 of file class\_rect\_matrix\_col.js.

## 30.605 doc/html/class\_rect\_matrix\_diag.js File Reference

### Variables

- var [class\\_rect\\_matrix\\_diag](#)

### 30.605.1 Variable Documentation

#### 30.605.1.1 var class\_rect\_matrix\_diag

##### Initial value:

```
=
[
 ["RectMatrixDiag", "class_rect_matrix_diag.html#a18b7d187dac91afcc43350b74a5b0810", null],
 ["DownDiag", "class_rect_matrix_diag.html#a418e6c0eb6f7f2f36de5772273275817", null],
 ["operator[]", "class_rect_matrix_diag.html#a81acf762fed3b7f4cecbf6c403058274", null],
 ["UpDiag", "class_rect_matrix_diag.html#aceb400b730818c6e77e83b2dd23750b3", null]
]
```

Definition at line 1 of file class\_rect\_matrix\_diag.js.

## 30.606 doc/html/class\_rect\_matrix\_row.js File Reference

### Variables

- var [class\\_rect\\_matrix\\_row](#)

### 30.606.1 Variable Documentation

#### 30.606.1.1 var class\_rect\_matrix\_row

##### Initial value:

```
=
[
 ["RectMatrixRow", "class_rect_matrix_row.html#af401cf2eb90b93a094a2435af46340ce", null],
 ["RectMatrixRow", "class_rect_matrix_row.html#af149e9f46a2f7bf9ce845668d76462a6", null],
 ["Down", "class_rect_matrix_row.html#a4a7bf5ec50be2e769e1394aea3d39243", null],
 ["Left", "class_rect_matrix_row.html#a459e0abe296fe1d3dc3000eda2e5a8f4", null],
 ["operator[]", "class_rect_matrix_row.html#af5881bc072938d306183c43f5166deff", null],
 ["Reset", "class_rect_matrix_row.html#aaa395cb08fed4a807ca5525f5aa45e4d", null],
 ["Reset", "class_rect_matrix_row.html#a4a2d16927e55415837ee34af837bdf7e", null],
 ["Right", "class_rect_matrix_row.html#aea28d4fd921cbea791b7672ddea94c07", null],
 ["Up", "class_rect_matrix_row.html#a5ec8b50ff768dc4140566c76b20605c2", null]
]
```

Definition at line 1 of file class\_rect\_matrix\_row.js.

## 30.607 doc/html/class\_rect\_matrix\_row\_col.js File Reference

### Variables

- var [class\\_rect\\_matrix\\_row\\_col](#)



### 30.607.1 Variable Documentation

#### 30.607.1.1 var class\_rect\_matrix\_row\_col

##### Initial value:

```
=
[
 ["RectMatrixRowCol", "class_rect_matrix_row_col.html#a0a1557801206f93c6d65fa35e6ef5c1b", null],
 ["AddScaled", "class_rect_matrix_row_col.html#a3d5a30a8f4a16f227745c8853e432292", null],
 ["Divide", "class_rect_matrix_row_col.html#af8a83ec516bf0e3e6c16df2eb0b79b7a", null],
 ["Divide", "class_rect_matrix_row_col.html#a7208776742fc69ed34bfb1f309b258db", null],
 ["DownDiag", "class_rect_matrix_row_col.html#a32dae8a5c17539b6ca2bc9adb48c6b0c", null],
 ["First", "class_rect_matrix_row_col.html#a86dff62cdf5b164af51438f2cd41b474", null],
 ["Negate", "class_rect_matrix_row_col.html#a5c6d68fb80759c68d09a0b5d12b84822", null],
 ["operator*", "class_rect_matrix_row_col.html#a387f09d14f7eb953a86c64be5edd8ac2", null],
 ["operator[]", "class_rect_matrix_row_col.html#a90021608c8ec2825325dd933b5d58e15", null],
 ["Reset", "class_rect_matrix_row_col.html#a7d8b0a570e37aeef12c548b47ac9e5bb", null],
 ["SumSquare", "class_rect_matrix_row_col.html#aaed10cc0174d1d8bdfc58829e97e2c57", null],
 ["UpDiag", "class_rect_matrix_row_col.html#a8e2a7a6d4cd227c1ea884b7d65585942", null],
 ["Zero", "class_rect_matrix_row_col.html#a935c0ec5f6ed575e62f2a52a3ce5ef12", null],
 ["ComplexScale", "class_rect_matrix_row_col.html#ae131888d49cb5f48944aa641f5998af5", null],
 ["Rotate", "class_rect_matrix_row_col.html#a23acf0c66955f76462c923517f1f64be", null],
 ["n", "class_rect_matrix_row_col.html#abaf49227eb2926f998c2630b93794ff6", null],
 ["shift", "class_rect_matrix_row_col.html#a4942d4ac243bd96cf3109fc0b966f28d", null],
 ["spacing", "class_rect_matrix_row_col.html#a7791524b848b624c08508bd2acd93d09", null],
 ["store", "class_rect_matrix_row_col.html#a11f41f07466d14f4a51e8556b677d11f", null]
]
```

Definition at line 1 of file class\_rect\_matrix\_row\_col.js.

## 30.608 doc/html/class\_rectangular\_surface\_property\_sheet.js File Reference

### Variables

- var [class\\_rectangular\\_surface\\_property\\_sheet](#)

### 30.608.1 Variable Documentation

#### 30.608.1.1 var class\_rectangular\_surface\_property\_sheet

##### Initial value:

```
=
[
 ["RectangularSurfacePropertySheet", "
class_rectangular_surface_property_sheet.html#a0605650510f46099247cf545431ff2e9", null],
 ["~RectangularSurfacePropertySheet", "
class_rectangular_surface_property_sheet.html#ad54201ab500dc9cb9d79f6e290414b58", null],
 ["apply", "class_rectangular_surface_property_sheet.html#a461dc5278bceebc3de5a1834746fa379", null],
 ["createSheet", "class_rectangular_surface_property_sheet.html#a40728d169b57c7b88c08124467a827de",
null]
]
```

Definition at line 1 of file class\_rectangular\_surface\_property\_sheet.js.

## 30.609 doc/html/class\_rectangular\_volume\_property\_sheet.js File Reference

### Variables

- var [class\\_rectangular\\_volume\\_property\\_sheet](#)

### 30.609.1 Variable Documentation

#### 30.609.1.1 var class\_rectangular\_volume\_property\_sheet

##### Initial value:

```
=
[
 ["RectangularVolumePropertySheet", "
 class_rectangular_volume_property_sheet.html#aec20832a403f214753eee0886315a959", null],
 ["~RectangularVolumePropertySheet", "
 class_rectangular_volume_property_sheet.html#a6febaac17cb2df37b84d86af73c55401", null],
 ["apply", "class_rectangular_volume_property_sheet.html#a245e3f5088ee386f6dffaae3bfa98563", null],
 ["createSheet", "class_rectangular_volume_property_sheet.html#a045630f8482d58abc7b46d0e5f6f01ed", null
]
]
```

Definition at line 1 of file class\_rectangular\_volume\_property\_sheet.js.

## 30.610 doc/html/class\_spline\_curve\_property\_sheet.js File Reference

### Variables

- var [class\\_spline\\_curve\\_property\\_sheet](#)

### 30.610.1 Variable Documentation

#### 30.610.1.1 var class\_spline\_curve\_property\_sheet

##### Initial value:

```
=
[
 ["SplineCurvePropertySheet", "class_spline_curve_property_sheet.html#a963427a253594aca33b1e463203ff18a
 ", null],
 ["apply", "class_spline_curve_property_sheet.html#a2cf173c21e74febd15828bcda200952c", null],
 ["createSheet", "class_spline_curve_property_sheet.html#a418643f8988530029bfc03f8c9a0535", null]
]
```

Definition at line 1 of file class\_spline\_curve\_property\_sheet.js.

## 30.611 doc/html/class\_surface\_resolution\_sheet.js File Reference

### Variables

- var [class\\_surface\\_resolution\\_sheet](#)

### 30.611.1 Variable Documentation

#### 30.611.1.1 var class\_surface\_resolution\_sheet

**Initial value:**

```
=
[
 ["SurfaceResolutionSheet", "class_surface_resolution_sheet.html#a5c8fa0a9cf743a23ebabd704e7a8d4d7",
 null],
 ["createSheet", "class_surface_resolution_sheet.html#a098963d6e66d7de5ae59b1c5f061566f", null],
 ["ok", "class_surface_resolution_sheet.html#aae61ee8fb4c499ef66de2af38e69b532", null],
 ["return_value", "class_surface_resolution_sheet.html#a9ef93d4170e8ca385d7ed80fcbce4e02", null]
]
```

Definition at line 1 of file class\_surface\_resolution\_sheet.js.

## 30.612 doc/html/class\_test\_class.js File Reference

### Variables

- var [class\\_test\\_class](#)

### 30.612.1 Variable Documentation

#### 30.612.1.1 var class\_test\_class

**Initial value:**

```
=
[
 ["TestClass", "class_test_class.html#a09903432ed60c7c38f1beccc4cf18987", null],
 ["Sum", "class_test_class.html#acc8678bdebaf258b6a56a320d58264e3", null]
]
```

Definition at line 1 of file class\_test\_class.js.

## 30.613 doc/html/class\_unf\_node\_type.js File Reference

### Variables

- var [class\\_unf\\_node\\_type](#)

### 30.613.1 Variable Documentation

#### 30.613.1.1 var class\_unf\_node\_type

##### Initial value:

```
=
[
 ["printNode", "class_unf_node_type.html#a4cace6758a4d5820a6b07b3f55ef1a42", null],
 ["n_", "class_unf_node_type.html#a37fde4efbecadca31c42c5ea266467e0", null],
 ["origine_", "class_unf_node_type.html#a6473043efe97dd34ceb497ee310efb63", null]
]
```

Definition at line 1 of file class\_unf\_node\_type.js.

## 30.614 doc/html/classbool.js File Reference

### Variables

- var [classbool](#)

### 30.614.1 Variable Documentation

#### 30.614.1.1 var classbool

##### Initial value:

```
=
[
 ["bool", "classbool.html#ab73d34fbac668d5fbba277dd263df834", null],
 ["bool", "classbool.html#a8d55b7ccfa7efec5e42a83ba0affe101", null],
 ["bool", "classbool.html#aebef315d83f84ed5dfd02573fb6d741e", null],
 ["operator int", "classbool.html#a6949f0cf4f737e6e9344bdfe822ba520", null],
 ["operator!", "classbool.html#abca2cc3627077f5a54408861dd16a6b3", null]
]
```

Definition at line 1 of file classbool.js.

## 30.615 doc/html/classgv\_action\_sheet.js File Reference

### Variables

- var [classgv\\_action\\_sheet](#)

### 30.615.1 Variable Documentation

#### 30.615.1.1 var classgv\_action\_sheet

**Initial value:**

```
=
[
 ["~gvActionSheet", "classgv_action_sheet.html#a2bda238d086bf141576e75543d7f0864", null],
 ["showDialog", "classgv_action_sheet.html#ace90b4317a54b079acabfda0172b1634", null],
 ["form_", "classgv_action_sheet.html#aeaf5f0304b7c950f719f027fd6999792", null]
]
```

Definition at line 1 of file classgv\_action\_sheet.js.

## 30.616 doc/html/classgv\_application.js File Reference

### Variables

- var [classgv\\_application](#)

### 30.616.1 Variable Documentation

#### 30.616.1.1 var classgv\_application

Definition at line 1 of file classgv\_application.js.

## 30.617 doc/html/classgv\_application\_vol\_and\_l\_r.js File Reference

### Variables

- var [classgv\\_application\\_vol\\_and\\_l\\_r](#)

### 30.617.1 Variable Documentation

#### 30.617.1.1 var classgv\_application\_vol\_and\_l\_r

**Initial value:**

```
=
[
 ["gvApplicationVolAndLR", "classgv_application_vol_and_l_r.html#acce6f6d76921b8338e19af9cecfbc6df5",
 null],
 ["~gvApplicationVolAndLR", "classgv_application_vol_and_l_r.html#aaadf77b8d7d5c4bc060aac0a1731b37b",
 null],
 ["buildExtraGUI", "classgv_application_vol_and_l_r.html#af0efcf10aee68a8775b5695daea12945", null],
 ["move_vertices_to_origin", "classgv_application_vol_and_l_r.html#a4b986c08bfff616b1adcdf05fba3d09e7",
 null],
 ["translate_to_origin", "classgv_application_vol_and_l_r.html#a376ed2331b131a9430b3a58e4dda0251", null
],
 ["view_reset", "classgv_application_vol_and_l_r.html#a7a5210fcb605729a0a7027b27d47ffa3", null]
]
```

Definition at line 1 of file classgv\_application\_vol\_and\_l\_r.js.

## 30.618 doc/html/classgv\_camera.js File Reference

### Variables

- var [classgv\\_camera](#)

### 30.618.1 Variable Documentation

#### 30.618.1.1 var classgv\_camera

Definition at line 1 of file classgv\_camera.js.

## 30.619 doc/html/classgv\_curve\_paintable.js File Reference

### Variables

- var [classgv\\_curve\\_paintable](#)

### 30.619.1 Variable Documentation

#### 30.619.1.1 var classgv\_curve\_paintable

#### Initial value:

```
=
[
 ["gvCurvePaintable", "classgv_curve_paintable.html#a6c92732be34ff2592b7fc478a119ff87", null],
 ["gvCurvePaintable", "classgv_curve_paintable.html#a2b3d88dea2ba6b0b5d32cb9df0857289", null],
 ["~gvCurvePaintable", "classgv_curve_paintable.html#af5f8192a2455315140d1e5bdbd1e6765", null],
 ["drawPoints", "classgv_curve_paintable.html#a8e8f813fec18aeb861f1b5ca3767fe9c", null],
 ["paint", "classgv_curve_paintable.html#a48991e757dd23c3cc8198c6b5c06dd8a", null],
 ["setDrawPoints", "classgv_curve_paintable.html#aefebd027820715c892715cafd7de9b26", null],
 ["draw_pts_", "classgv_curve_paintable.html#aca14ac01e98cc0cb6b7403ddcaab6c19", null],
 ["line_", "classgv_curve_paintable.html#ab1899d9822ddf163d025ec8f22b4ff99", null]
]
```

Definition at line 1 of file classgv\_curve\_paintable.js.

## 30.620 doc/html/classgv\_data.js File Reference

### Variables

- var [classgv\\_data](#)

### 30.620.1 Variable Documentation

#### 30.620.1.1 var classgv\_data

Definition at line 1 of file classgv\_data.js.

## 30.621 doc/html/classgv\_generic\_tri\_paintable.js File Reference

### Variables

- var [classgv\\_generic\\_tri\\_paintable](#)

### 30.621.1 Variable Documentation

#### 30.621.1.1 var classgv\_generic\_tri\_paintable

##### Initial value:

```
=
[
 ["gvGenericTriPaintable", "classgv_generic_tri_paintable.html#a8a92cf64e5e78aea9e154237048d3e3e", null],
 ["gvGenericTriPaintable", "classgv_generic_tri_paintable.html#aeafa3555dc46d731727ff7002881306aa", null],
 ["~gvGenericTriPaintable", "classgv_generic_tri_paintable.html#a425348f0209058815c4e77d3840c9907", null],
 ["paint", "classgv_generic_tri_paintable.html#a16da12e7feb462445f72e4a6c77f9afa", null],
 ["tri_", "classgv_generic_tri_paintable.html#a81e5efc1bc3f50e26e31e33050de2658", null]
]
```

Definition at line 1 of file classgv\_generic\_tri\_paintable.js.

## 30.622 doc/html/classgv\_generic\_tri\_quad\_mesh.js File Reference

### Variables

- var [classgv\\_generic\\_tri\\_quad\\_mesh](#)

### 30.622.1 Variable Documentation

#### 30.622.1.1 var classgv\_generic\_tri\_quad\_mesh

##### Initial value:

```
=
[
 ["gvGenericTriQuadMesh", "classgv_generic_tri_quad_mesh.html#a6f92d36a82eebdb3ed80e96d5dda4cf", null],
 ["normalArray", "classgv_generic_tri_quad_mesh.html#a35d51765dc89258deaaaebbf0418767", null],
 ["numQuads", "classgv_generic_tri_quad_mesh.html#a867e4c4f671a4829b2b51d753f3093cd", null],
 ["numTriangles", "classgv_generic_tri_quad_mesh.html#a4f42cd2dc2430832d5afeeee55ae8369", null],
 ["numVertices", "classgv_generic_tri_quad_mesh.html#ac047150f16fbcf016beef94b759c6745", null],
 ["quadIndexArray", "classgv_generic_tri_quad_mesh.html#a9b98121e98e22434a7977610ebf2e240", null],
 ["resize", "classgv_generic_tri_quad_mesh.html#a00a26ca803a923e94a64d18249340fa9", null],
 ["texcoordArray", "classgv_generic_tri_quad_mesh.html#afb502bc32f98646ddc1fb39fced5d0da", null],
 ["triangleIndexArray", "classgv_generic_tri_quad_mesh.html#a7f5a917fef1e914fbbdbb3011cf0a465", null],
 ["useNormals", "classgv_generic_tri_quad_mesh.html#a7aaf4c32dd58c096b8b1ca8f6f439252", null],
 ["useTexCoords", "classgv_generic_tri_quad_mesh.html#a8d182ec97fd8949b2a31b5e3bb07e110", null],
 ["vertexArray", "classgv_generic_tri_quad_mesh.html#aeab35227128abea55454bd79d4cb4ce6", null]
]
```

Definition at line 1 of file classgv\_generic\_tri\_quad\_mesh.js.

## 30.623 doc/html/classgv\_generic\_tri\_quad\_paintable.js File Reference

### Variables

- var [classgv\\_generic\\_tri\\_quad\\_paintable](#)

### 30.623.1 Variable Documentation

#### 30.623.1.1 var classgv\_generic\_tri\_quad\_paintable

##### Initial value:

```
=
[
 ["gvGenericTriQuadPaintable", "
 classgv_generic_tri_quad_paintable.html#a48b8f370ee4fdd61b1745b2342cc6414", null],
 ["gvGenericTriQuadPaintable", "
 classgv_generic_tri_quad_paintable.html#a9eb49aa90212ff7f8c30befc3a69ca04", null],
 ["~gvGenericTriQuadPaintable", "
 classgv_generic_tri_quad_paintable.html#aaa0bac5a08d07f054b9b1c7970fb8f99", null],
 ["paint", "classgv_generic_tri_quad_paintable.html#aa04334da96ae1551b788bc0fd08dcadb", null],
 ["mesh_", "classgv_generic_tri_quad_paintable.html#a49c8c5382467b36a654de0ed02b36b3a", null]
]
```

Definition at line 1 of file classgv\_generic\_tri\_quad\_paintable.js.

## 30.624 doc/html/classgv\_group.js File Reference

### Variables

- var [classgv\\_group](#)

### 30.624.1 Variable Documentation

#### 30.624.1.1 var classgv\_group

##### Initial value:

```
=
[
 ["gvGroup", "classgv_group.html#ae5a83bbc4371dd60b7dae860f31996d7", null],
 ["~gvGroup", "classgv_group.html#a6e21e0a155b4d655170d3206896346a9", null],
 ["name", "classgv_group.html#a6032ce853c50bb71260ecf96d90720cc", null],
 ["operator[]", "classgv_group.html#a8738f55bdbc90fe88e997a38a920f8e9", null],
 ["size", "classgv_group.html#a757a4bed030f67881f508cce6302d4eb", null]
]
```

Definition at line 1 of file classgv\_group.js.



## 30.625 doc/html/classgv\_group\_property\_sheet.js File Reference

### Variables

- var [classgv\\_group\\_property\\_sheet](#)

### 30.625.1 Variable Documentation

#### 30.625.1.1 var classgv\_group\_property\_sheet

##### Initial value:

```
=
[
 ["gvGroupPropertySheet", "classgv_group_property_sheet.html#acf186d05304921b9d222c6c290822bd4", null],
 ["accept", "classgv_group_property_sheet.html#aa9f5d9d4ada94afa84cfb1d45f3c0568", null],
 ["createSheet", "classgv_group_property_sheet.html#a4284ec0e95ab2a67b2e66e27012a5731", null],
 ["getMembers", "classgv_group_property_sheet.html#a09abcf2cc6f11f0493299460875b3830", null],
 ["value_changed", "classgv_group_property_sheet.html#a11b1dc2efecd44ea4144b5b7e6093558", null]
]
```

Definition at line 1 of file classgv\_group\_property\_sheet.js.

## 30.626 doc/html/classgv\_line\_cloud\_paintable.js File Reference

### Variables

- var [classgv\\_line\\_cloud\\_paintable](#)

### 30.626.1 Variable Documentation

#### 30.626.1.1 var classgv\_line\_cloud\_paintable

##### Initial value:

```
=
[
 ["gvLineCloudPaintable", "classgv_line_cloud_paintable.html#ae4d09aa070032eaa979f4e7d979b2d1c", null],
 ["gvLineCloudPaintable", "classgv_line_cloud_paintable.html#a57fe51aa347bc554f4c0a05b78d56ec6", null],
 ["~gvLineCloudPaintable", "classgv_line_cloud_paintable.html#a8c0888908c9c558a4f4f04203962ada3", null],
 ["fractionRendered", "classgv_line_cloud_paintable.html#ac639ff659a37d19ab19bd493c9e75d29", null],
 ["paint", "classgv_line_cloud_paintable.html#a48d2d8e9855a31c4d125fe629e1c089a", null],
 ["setFractionRendered", "classgv_line_cloud_paintable.html#abc46fa8094d7d434fef5e2185d0bde75", null],
 ["fractionrendered", "classgv_line_cloud_paintable.html#a090b222902dd687ec00d31dda642327c", null],
 ["lc", "classgv_line_cloud_paintable.html#a1b4d49f084bd47ec978735885ea95e32", null]
]
```

Definition at line 1 of file classgv\_line\_cloud\_paintable.js.

## 30.627 doc/html/classgv\_noop\_paintable.js File Reference

### Variables

- var [classgv\\_noop\\_paintable](#)

### 30.627.1 Variable Documentation

#### 30.627.1.1 var classgv\_noop\_paintable

##### Initial value:

```
=
[
 ["gvNoopPaintable", "classgv_noop_paintable.html#a7718749f030a806b3ed6d2e38fc38dce", null],
 ["gvNoopPaintable", "classgv_noop_paintable.html#a0f29a8db1226c25088e427a9d30ff367", null],
 ["~gvNoopPaintable", "classgv_noop_paintable.html#a8336bbc9cdce68061c38ad57b832dbe1", null],
 ["paint", "classgv_noop_paintable.html#a253b35d9049d004240ca45759d31c050", null]
]
```

Definition at line 1 of file classgv\_noop\_paintable.js.

## 30.628 doc/html/classgv\_object\_list.js File Reference

### Variables

- var [classgv\\_object\\_list](#)

### 30.628.1 Variable Documentation

#### 30.628.1.1 var classgv\_object\_list

##### Initial value:

```
=
[
 ["gvObjectList", "classgv_object_list.html#a9e7a69c68c247398c1b481e3e6b553fc", null],
 ["~gvObjectList", "classgv_object_list.html#a730e36cc31b61babcb49ccbc70f2be06", null],
 ["buildGUI", "classgv_object_list.html#a396aa917a2bcf70c0c9aee7ea9ebd930", null],
 ["clicked", "classgv_object_list.html#aa925bed4363e8f463e2cad3b984c8171", null],
 ["observedChanged", "classgv_object_list.html#a0d97f6f2f1d506f2e143ee91f5b679ac", null]
]
```

Definition at line 1 of file classgv\_object\_list.js.

## 30.629 doc/html/classgv\_observer.js File Reference

### Variables

- var [classgv\\_observer](#)

## 30.629.1 Variable Documentation

### 30.629.1.1 var classgv\_observer

#### Initial value:

```
=
[
 ["~gvObserver", "classgv_observer.html#aea825b41a2d01e9bfc03aaf1847b7f1b", null],
 ["observedChanged", "classgv_observer.html#ad965a9002fb8e250f4a364c2c994836c", null]
]
```

Definition at line 1 of file classgv\_observer.js.

## 30.630 doc/html/classgv\_paintable.js File Reference

### Variables

- var [classgv\\_paintable](#)

## 30.630.1 Variable Documentation

### 30.630.1.1 var classgv\_paintable

#### Initial value:

```
=
[
 ["gvPaintable", "classgv_paintable.html#a8df8126a8cb701340ee064aa42a3ec8c", null],
 ["gvPaintable", "classgv_paintable.html#a25e129d0a129024d9249083ce0d4b2d5", null],
 ["~gvPaintable", "classgv_paintable.html#a5aa5df86a5c1e322d812e3525c6644ed", null],
 ["getNormalColor", "classgv_paintable.html#af090ff10f14d137bea23c339730cb231", null],
 ["id", "classgv_paintable.html#a7a31121be20df12925c3aac0e838d598", null],
 ["paint", "classgv_paintable.html#aefcb86d3492c1c34ad2bf399dd6926d7", null],
 ["paintGL", "classgv_paintable.html#a79d06b44efa40fc9699ed915101295e3", null],
 ["selected", "classgv_paintable.html#a1aa50110d878509eccf158270f7240a7", null],
 ["setColor", "classgv_paintable.html#aa39bd4354c652188d8a1c107cd4a7c33", null],
 ["setColor", "classgv_paintable.html#acb76b57f883ec097bf86aa8e00fba27a", null],
 ["setId", "classgv_paintable.html#a14ec063f074e74c13e3f0d31469f4991", null],
 ["setSelected", "classgv_paintable.html#a4e186e429b430bcda2fac5fc4173504a", null],
 ["setVisible", "classgv_paintable.html#abdf058f399c59c72b5fe0b553475245c", null],
 ["visible", "classgv_paintable.html#abadf8d50aeba7e567fe9aa63f2a2dbd", null],
 ["id_", "classgv_paintable.html#abccdedd9ecd5d4ff8e6b0b20fbf573f7", null],
 ["normal_color_", "classgv_paintable.html#aae952db4fea5d902b81696c380dd3d8b", null],
 ["selected_", "classgv_paintable.html#a7690410c11d5b3e149221987dc59dbbc", null],
 ["selected_color_", "classgv_paintable.html#a693658ba4594dc9c79204ee0a9192853", null],
 ["visible_", "classgv_paintable.html#a629c8d39d5467e02154457ee1c7dde41", null]
]
```

Definition at line 1 of file classgv\_paintable.js.

## 30.631 doc/html/classgvPainter.js File Reference

### Variables

- var [classgvPainter](#)

### 30.631.1 Variable Documentation

#### 30.631.1.1 var classgvPainter

##### Initial value:

```
=
[
 ["gvPainter", "classgvPainter.html#a1c306705688cded43e274e5f7185ef63", null],
 ["~gvPainter", "classgvPainter.html#acdaef69360467db79f5c5218254da7d7", null],
 ["addPaintable", "classgvPainter.html#a3f6bf2d0cd46b2404d65248640bd0c8e", null],
 ["drawScene", "classgvPainter.html#a9cc69acb664783ce268dcca1a404e3f", null],
 ["getPaintable", "classgvPainter.html#af91a029f96cf87589e56083d20f647a4", null],
 ["getTexture", "classgvPainter.html#ade39026bc5fdb5c5104dd3d78f3f98a0", null],
 ["removeAllPaintables", "classgvPainter.html#a63138449340e6ee2c609d573690a8a71", null],
 ["removeLastPaintable", "classgvPainter.html#a94daa42d484dae6f3598a221da835789", null],
 ["removePaintable", "classgvPainter.html#a5475cb8ad291cb3eed0f328a4c5bc6e", null],
 ["setTexture", "classgvPainter.html#af7feec222a64232f596f871f2ef47928", null],
 ["setTexture", "classgvPainter.html#a8b24f7720584050b3099f60593d87ae3", null],
 ["paintables_", "classgvPainter.html#a35c501b0cfdc0d5ec1730c25fbcf16da", null],
 ["textures_", "classgvPainter.html#a6819b4074c19021ce222adfe83f4d1aa", null]
]
```

Definition at line 1 of file classgvPainter.js.

### 30.632 doc/html/classgv\_parametric\_surface\_paintable.js File Reference

#### Variables

- var [classgv\\_parametric\\_surface\\_paintable](#)

### 30.632.1 Variable Documentation

#### 30.632.1.1 var classgv\_parametric\_surface\_paintable

##### Initial value:

```
=
[
 ["gvParametricSurfacePaintable", "
classgv_parametric_surface_paintable.html#a3cecaac8a83ec0d401434dede6e4ccd7", null],
 ["gvParametricSurfacePaintable", "
classgv_parametric_surface_paintable.html#ac7e402ef62d65fabdb3b1ede7f72dd83", null],
 ["~gvParametricSurfacePaintable", "
classgv_parametric_surface_paintable.html#a4b45806196622dd270231001832ee89f", null],
 ["createSurface", "classgv_parametric_surface_paintable.html#a756d38eda355bceff1c2a07blae01b3f", null
],
 ["drawSurface", "classgv_parametric_surface_paintable.html#ab72a299bbe6a880f835180e05f556cf0", null],
 ["paint", "classgv_parametric_surface_paintable.html#a5fd844d4e71681fe07fec1971b7e495c", null],
 ["tri_", "classgv_parametric_surface_paintable.html#a6b708b7b2091f0a67a4f849d4f15922a", null]
]
```

Definition at line 1 of file classgv\_parametric\_surface\_paintable.js.

### 30.633 doc/html/classgv\_point\_cloud\_paintable.js File Reference

#### Variables

- var [classgv\\_point\\_cloud\\_paintable](#)

### 30.633.1 Variable Documentation

#### 30.633.1.1 var classgv\_point\_cloud\_paintable

##### Initial value:

```
=
[
 ["gvPointCloudPaintable", "classgv_point_cloud_paintable.html#ab21a8642a6b1a31fbdd75d9313526e33", null],
 ["gvPointCloudPaintable", "classgv_point_cloud_paintable.html#a71f1ald5b7840803d9bb602a20783a98", null],
 ["~gvPointCloudPaintable", "classgv_point_cloud_paintable.html#a88930f8e37abbacd60a78b94ed7993e", null],
 ["fractionRendered", "classgv_point_cloud_paintable.html#aa24e3954a55c104a875dd14294c5f75e", null],
 ["getPaintId", "classgv_point_cloud_paintable.html#a3071d123cea31b81c32d5462ac5f5723", null],
 ["paint", "classgv_point_cloud_paintable.html#a23fa7ad4a47c8335cceaafd37d9a57c6", null],
 ["paintGL", "classgv_point_cloud_paintable.html#a0cad9ec6abc581b43eddecf50b772747", null],
 ["pointSize", "classgv_point_cloud_paintable.html#a34583ecf6a2dd48058ee86264854da73", null],
 ["setFractionRendered", "classgv_point_cloud_paintable.html#a5f7278ff6de4a61eff8aafa159b6dd77", null],
 ["setPaintId", "classgv_point_cloud_paintable.html#ab237b8b151dcc251271e9f80fd2f1bb6", null],
 ["setPointSize", "classgv_point_cloud_paintable.html#a722340833dff8bb9d3cc72850c12de17", null],
 ["ttranslate", "classgv_point_cloud_paintable.html#aa53d3f733ff45208aaae0dead6e99b92", null],
 ["fractionrendered_", "classgv_point_cloud_paintable.html#a1a860092e7b53459ea85b06b3b605f49", null],
 ["paintId_", "classgv_point_cloud_paintable.html#a3b5df088f32941c3594f2bea134e0c39", null],
 ["pc_", "classgv_point_cloud_paintable.html#ac7f225401ac7b1cb5992de117751fa32", null],
 ["pointsize_", "classgv_point_cloud_paintable.html#ad0b4e484772f222b4e8b52ec4514bdde", null],
 ["vert_translation_", "classgv_point_cloud_paintable.html#a377d65530f2f4b7fb754d9b6232171a4", null]
]
```

Definition at line 1 of file classgv\_point\_cloud\_paintable.js.

## 30.634 doc/html/classgv\_property\_sheet.js File Reference

### Variables

- var [classgv\\_property\\_sheet](#)

### 30.634.1 Variable Documentation

#### 30.634.1.1 var classgv\_property\_sheet

##### Initial value:

```
=
[
 ["~gvPropertySheet", "classgv_property_sheet.html#a16bd27bc9854d27f6541988ebccce52a", null],
 ["createSheet", "classgv_property_sheet.html#a1644a548b3e7b7716bbf2a99fd7fa4b8", null]
]
```

Definition at line 1 of file classgv\_property\_sheet.js.

## 30.635 doc/html/classgv\_quads\_paintable.js File Reference

### Variables

- var [classgv\\_quads\\_paintable](#)

### 30.635.1 Variable Documentation

#### 30.635.1.1 var classgv\_quads\_paintable

##### Initial value:

```
=
[
 ["gvQuadsPaintable", "classgv_quads_paintable.html#abb6db6f7ac6b7c8f73bb1f1a5081c2f5", null],
 ["gvQuadsPaintable", "classgv_quads_paintable.html#ae406eb43609fe834476e1a82266fa32", null],
 ["~gvQuadsPaintable", "classgv_quads_paintable.html#af58ff384f9aea851a8e714feed4f7773", null],
 ["paint", "classgv_quads_paintable.html#a240f5343199e122e61a6a207bf14e5eb", null],
 ["quads_", "classgv_quads_paintable.html#a80b1bdf3b938aee5229b96ba7a5959b0", null]
]
```

Definition at line 1 of file classgv\_quads\_paintable.js.

## 30.636 doc/html/classgv\_rectangular\_surface\_paintable.js File Reference

### Variables

- var [classgv\\_rectangular\\_surface\\_paintable](#)

### 30.636.1 Variable Documentation

#### 30.636.1.1 var classgv\_rectangular\_surface\_paintable

##### Initial value:

```
=
[
 ["gvRectangularSurfacePaintable", "
classgv_rectangular_surface_paintable.html#a2acbaa371b3b75bd3507c847f3bb3251", null],
 ["gvRectangularSurfacePaintable", "
classgv_rectangular_surface_paintable.html#a06783fa9d80589c69f81eaa171016f75", null],
 ["~gvRectangularSurfacePaintable", "
classgv_rectangular_surface_paintable.html#ad5aa8d77080d3a66decb282b1f54cddc", null],
 ["paint", "classgv_rectangular_surface_paintable.html#a1bd977bee27a43c358afae53bbfc0f05", null],
 ["tri_", "classgv_rectangular_surface_paintable.html#aa25fd874ae47ae3942aa08b23a223602", null]
]
```

Definition at line 1 of file classgv\_rectangular\_surface\_paintable.js.

## 30.637 doc/html/classgv\_rectangular\_volume\_paintable.js File Reference

### Variables

- var [classgv\\_rectangular\\_volume\\_paintable](#)

### 30.637.1 Variable Documentation

#### 30.637.1.1 var classgv\_rectangular\_volume\_paintable

##### Initial value:

```
=
[
 ["gvRectangularVolumePaintable", "
 classgv_rectangular_volume_paintable.html#a927e3e9a95c78f037815cd31a7d6827d", null],
 ["gvRectangularVolumePaintable", "
 classgv_rectangular_volume_paintable.html#a88da42073d2bedb05ac14d7daf054622", null],
 ["~gvRectangularVolumePaintable", "
 classgv_rectangular_volume_paintable.html#a2f2771c05cf99a36954d85413ce48176", null],
 ["paint", "classgv_rectangular_volume_paintable.html#adac9a39d5da128fc7aa6e119d751ace7", null],
 ["tri_", "classgv_rectangular_volume_paintable.html#a25e42855a7f495c0d5f0acd03f7eaf65", null]
]
```

Definition at line 1 of file classgv\_rectangular\_volume\_paintable.js.

## 30.638 doc/html/classgv\_resolution\_dialog.js File Reference

### Variables

- var [classgv\\_resolution\\_dialog](#)

### 30.638.1 Variable Documentation

#### 30.638.1.1 var classgv\_resolution\_dialog

##### Initial value:

```
=
[
 ["gvResolutionDialog", "classgv_resolution_dialog.html#a65ef3c997d5e774f3dbbb86140a75d08", null],
 ["~gvResolutionDialog", "classgv_resolution_dialog.html#a4011f5d8768f74900f66affccdc09e44", null],
 ["accept", "classgv_resolution_dialog.html#af0ad74ab49cc2771b1492ac6fe8fc257", null],
 ["apply", "classgv_resolution_dialog.html#a4a6f85c2cf416be678cdf0211a1e9ca", null],
 ["valuesChanged", "classgv_resolution_dialog.html#a274a1a67cb61adbae73297941c000800", null],
 ["ures_", "classgv_resolution_dialog.html#a80928c1c8369d3a57c58dc7c3f45b4e4", null],
 ["uslide_", "classgv_resolution_dialog.html#ad1d66a6de7b9177c1a871720cdeda823", null],
 ["vres_", "classgv_resolution_dialog.html#a09508da0c8d1c00b9da9493802a5be51", null],
 ["vslide_", "classgv_resolution_dialog.html#a967e9ab226b8ee2a6ed004306ef8d6e9", null]
]
```

Definition at line 1 of file classgv\_resolution\_dialog.js.

## 30.639 doc/html/classgv\_standard\_mouse\_handler.js File Reference

### Variables

- var [classgv\\_standard\\_mouse\\_handler](#)

### 30.639.1 Variable Documentation

#### 30.639.1.1 var classgv\_standard\_mouse\_handler

##### Initial value:

```
=
[
 ["gvStandardMouseHandler", "classgv_standard_mouse_handler.html#a3aad1205e51624a9e801b66b654ab7b7",
 null],
 ["~gvStandardMouseHandler", "classgv_standard_mouse_handler.html#afb27b6a1df7dc568a4869d9822dbbe44",
 null]
]
```

Definition at line 1 of file classgv\_standard\_mouse\_handler.js.

## 30.640 doc/html/classgv\_texture.js File Reference

### Variables

- var [classgv\\_texture](#)

### 30.640.1 Variable Documentation

#### 30.640.1.1 var classgv\_texture

##### Initial value:

```
=
[
 ["gvTexture", "classgv_texture.html#a9bdbc3ec11abed50e4640779333da23e", null],
 ["gvTexture", "classgv_texture.html#a0d609acaed1499af8fcf8d99e54c80", null],
 ["gvTexture", "classgv_texture.html#ae06d6508e876fe0d9e45381baecf2420", null],
 ["gvTexture", "classgv_texture.html#ac21d410a98136315714c503693300a73", null],
 ["~gvTexture", "classgv_texture.html#a1fbcca34e28da27a5616832316e4009d", null],
 ["bind", "classgv_texture.html#a51a170497dba9efb9c591b3d35c60604", null],
 ["genTexture", "classgv_texture.html#aba546d26953db4e862c4a9b786664a64", null],
 ["getEnvMode", "classgv_texture.html#a6b34ab48a07d548c6bef11e3c6a46b4a", null],
 ["getMagFilter", "classgv_texture.html#ae2b2ef88114a3e120ebfc4128d99fdc4", null],
 ["getMinFilter", "classgv_texture.html#a4436004b003c008026cb1ae694f9ee53", null],
 ["getQImage", "classgv_texture.html#a5e10844c6710261a16999aa4caa30c8d", null],
 ["height", "classgv_texture.html#a050bf69adedded794a353bb94e8badd4", null],
 ["operator=", "classgv_texture.html#a5fa3c3107691fb6cba6fe812196d5805", null],
 ["readFile", "classgv_texture.html#a7d8986529a3e53d2be2cb46b371253a3", null],
 ["setAlphaValue", "classgv_texture.html#a029d82579d1ea0216dd509493f081043", null],
 ["setCenterEdgeTexels", "classgv_texture.html#acb22b5e3b07ef4620bf115ef95d3886b", null],
 ["setEnvMode", "classgv_texture.html#a2e293c7eb659dd703a596975c2ed594a", null],
 ["setIdentMatrix", "classgv_texture.html#ad59b1840cc7484d0865659667a7866c6", null],
 ["setMagFilter", "classgv_texture.html#a8940ddc697502f7d1cd558b228345d85", null],
 ["setMinFilter", "classgv_texture.html#a1d6c6eb31d5bc74bf29c3186667f6561", null],
 ["setTextureMatrix", "classgv_texture.html#a28c94abfca75ba8623dc52f7a4d35b74", null],
 ["setWrapMode", "classgv_texture.html#a0161ef23ff0a881d08656f61f812388d", null],
 ["width", "classgv_texture.html#aa9078e4da5c899ca528cf48791788f71", null]
]
```

Definition at line 1 of file classgv\_texture.js.



## 30.641 doc/html/classgv\_view.js File Reference

### Variables

- var [classgv\\_view](#)

#### 30.641.1 Variable Documentation

##### 30.641.1.1 var classgv\_view

Definition at line 1 of file classgv\_view.js.

## 30.642 doc/html/classhed\_1\_1\_dart.js File Reference

### Variables

- var [classhed\\_1\\_1\\_dart](#)

#### 30.642.1 Variable Documentation

##### 30.642.1.1 var classhed\_1\_1\_dart

### Initial value:

```
=
[
 ["Dart", "classhed_1_1_dart.html#a141548d2908ef5847fd81370d57c07a0", null],
 ["Dart", "classhed_1_1_dart.html#a40d2832ae9a9d61a1cc16c9b171a2d14", null],
 ["Dart", "classhed_1_1_dart.html#a75fe6236394e8e039a1b3ad44582bf4f", null],
 ["~Dart", "classhed_1_1_dart.html#a10f0f2b42173bda49c781fe56f1c0a8a", null],
 ["alpha0", "classhed_1_1_dart.html#aabce4f3568d0729332d583c4e130d815", null],
 ["alpha1", "classhed_1_1_dart.html#a18a7132ba7a19cd64a9614785db0f8bc", null],
 ["alpha2", "classhed_1_1_dart.html#a07596e714ce02057c6934ffab585458c", null],
 ["getEdge", "classhed_1_1_dart.html#a54be6f9667fd38f286c7532bc3b375cf", null],
 ["getNode", "classhed_1_1_dart.html#ad3e2a078d1cf035228e9d7707ebcc0d5", null],
 ["getOppositeNode", "classhed_1_1_dart.html#a26f9b2a8d611a8fbc9f150ed3f8a3ef1", null],
 ["init", "classhed_1_1_dart.html#a5d99a7eeb15b6099d66d69e6d38cdf08", null],
 ["isCounterClockwise", "classhed_1_1_dart.html#ac7db17ef56b3fe80b2fb6ba5acf1e3ed", null],
 ["operator!=" , "classhed_1_1_dart.html#a1ceee38a5d1e74e32e423ec67fd5ede0", null],
 ["operator=" , "classhed_1_1_dart.html#a35ablebaa3eae59d566d8f6a6106fd6", null],
 ["operator==" , "classhed_1_1_dart.html#aa593dd5d95f05237b0b175ec917d1e76", null],
 ["x", "classhed_1_1_dart.html#a39d20327f3eeae53f06f79f53d840", null],
 ["y", "classhed_1_1_dart.html#a20849660f038888f05ba65daf0a99ae5", null]
]
```

Definition at line 1 of file classhed\_1\_1\_dart.js.

## 30.643 doc/html/classhed\_1\_1\_edge.js File Reference

### Variables

- var [classhed\\_1\\_1\\_edge](#)

### 30.643.1 Variable Documentation

#### 30.643.1.1 var classed\_1\_1\_edge

##### Initial value:

```
=
[
 ["Edge", "classed_1_1_edge.html#afcf345e2e15a83e18dbf14dd2eaecad7", null],
 ["~Edge", "classed_1_1_edge.html#a5fc895fb958290c3ad807139824de291", null],
 ["getNextEdgeInFace", "classed_1_1_edge.html#a64a08a620e11cd3336ad0ff06aa13013", null],
 ["getSourceNode", "classed_1_1_edge.html#a156eabe70e853a905bfb2e9a511a48d1", null],
 ["getTargetNode", "classed_1_1_edge.html#a9d7d33ee335595bae685ae4a1428a484", null],
 ["getTwinEdge", "classed_1_1_edge.html#ae1358d12578911b00084d40269569f56", null],
 ["isConstrained", "classed_1_1_edge.html#a4f204d397b8502db9b15f571677d305c", null],
 ["isLeadingEdge", "classed_1_1_edge.html#a56ee267c453135610c91a05a97ec3478", null],
 ["setAsLeadingEdge", "classed_1_1_edge.html#ac2a34c4cc6d002696a73d6e814a11cb2", null],
 ["setConstrained", "classed_1_1_edge.html#a3ffa453b77c2d6f8e01c3c673c2aca36", null],
 ["setNextEdgeInFace", "classed_1_1_edge.html#a63e9f4ae2705fa6d52f2734e7635c053", null],
 ["setSourceNode", "classed_1_1_edge.html#a4583f7d0e199c30e574c14b66f4ba539", null],
 ["setTwinEdge", "classed_1_1_edge.html#a7505edb3562f36e6d08082b00f69d7b5", null],
 ["isConstrained_", "classed_1_1_edge.html#ae285190503f792364ab699f77f4194d7", null],
 ["isLeadingEdge_", "classed_1_1_edge.html#a19b8a593101978a592db9525f2100137", null]
]
```

Definition at line 1 of file classed\_1\_1\_edge.js.

## 30.644 doc/html/classed\_1\_1\_node.js File Reference

### Variables

- var [classed\\_1\\_1\\_node](#)

### 30.644.1 Variable Documentation

#### 30.644.1.1 var classed\_1\_1\_node

##### Initial value:

```
=
[
 ["Node", "classed_1_1_node.html#a12967fb66dbfb30869ce45216ee12e8", null],
 ["Node", "classed_1_1_node.html#ad829f17633a0b6cf4dda8f73d4f13358", null],
 ["~Node", "classed_1_1_node.html#a44bd6321beeb4983395f79bc36832d3", null],
 ["getFlag", "classed_1_1_node.html#aa5026220174ela840860e32982ec0e82", null],
 ["id", "classed_1_1_node.html#ae0ae30c021c0e665b409db77bf604afa", null],
 ["init", "classed_1_1_node.html#aecf4e4b054c363b753812c28f5c00a21", null],
 ["setFlag", "classed_1_1_node.html#a43d9a3a9079143034b370be7551d1492", null],
 ["x", "classed_1_1_node.html#ae5240e90452911b4635f8668f2d4129c", null],
 ["y", "classed_1_1_node.html#ab71715188f02759f2b2a632adf9dd4cb", null],
 ["z", "classed_1_1_node.html#a812130c3e029cdec9bc9a47a1e53a7", null]
]
```

Definition at line 1 of file classed\_1\_1\_node.js.

## 30.645 doc/html/classhed\_1\_1\_triangulation.js File Reference

### Variables

- var [classhed\\_1\\_1\\_triangulation](#)

### 30.645.1 Variable Documentation

#### 30.645.1.1 var classhed\_1\_1\_triangulation

##### Initial value:

```
=
[
 ["Triangulation", "classhed_1_1_triangulation.html#a66c7768aa1fb45e26e2ecdaf3c61c73e", null],
 ["Triangulation", "classhed_1_1_triangulation.html#a3ca85712664a867caa4f5ce945890f78", null],
 ["~Triangulation", "classhed_1_1_triangulation.html#a1a66db896d007b1a57cc2ffd0b3c3b6a", null],
 ["addLeadingEdge", "classhed_1_1_triangulation.html#a431ff0642c094a073ac78b45dd83c416", null],
 ["checkDelaunay", "classhed_1_1_triangulation.html#ae55e6f576e7d892bc1aaa00f831c514a", null],
 ["cleanAll", "classhed_1_1_triangulation.html#a6880897ae6a171379671880f76be1fa", null],
 ["createDart", "classhed_1_1_triangulation.html#ae6f30af09e8be13c3d21eef7b42c8e99", null],
 ["createDelaunay", "classhed_1_1_triangulation.html#acc1c7ee89bf8d5e5a39d0cb3219ca8d7", null],
 ["flagNodes", "classhed_1_1_triangulation.html#a47f1f095a8d48536d91a0315f8fde1bc", null],
 ["getBoundaryEdge", "classhed_1_1_triangulation.html#a5ad0ba9b943eedc8b57dfc8ff0e84a1", null],
 ["getEdges", "classhed_1_1_triangulation.html#aca48dd9df4dda24cac14e231397f6727", null],
 ["getInteriorNode", "classhed_1_1_triangulation.html#a671e04032ce039a909d8343ab98aa863", null],
 ["getLeadingEdges", "classhed_1_1_triangulation.html#a8b4faaa7ca0a7cdccc49f2dbe86e65bd", null],
 ["getNodes", "classhed_1_1_triangulation.html#a57f48cb368376503174bef523d5c54ea", null],
 ["initTwoEnclosingTriangles", "classhed_1_1_triangulation.html#a0cc9befb4f0f65144188d174fd4355d1",
 null],
 ["noTriangles", "classhed_1_1_triangulation.html#a685e36e668bfe3781486134b4fc1e97a", null],
 ["optimizeDelaunay", "classhed_1_1_triangulation.html#ac416bb53882ddb732c27ba2a9fecdb63", null],
 ["printEdges", "classhed_1_1_triangulation.html#ac67f9b08a01bc956088fee34615d4bc1", null],
 ["removeLeadingEdgeFromList", "classhed_1_1_triangulation.html#a0dca1788bdf9f99b9a0707827cb2edd6",
 null],
 ["removeTriangle", "classhed_1_1_triangulation.html#aeff0c075bc2a4e5b33fa03e3406ce2ed", null],
 ["reverse_splitTriangle", "classhed_1_1_triangulation.html#a064a6ae468a82186a66cb076eedd575e", null],
 ["splitTriangle", "classhed_1_1_triangulation.html#a720e4b0e8ec20650bef7537b5b5606c6", null],
 ["swapEdge", "classhed_1_1_triangulation.html#a74cbc329d1f7dfac664d6a821d491f31", null],
 ["leadingEdges_", "classhed_1_1_triangulation.html#a36ac06b34d61da338e01d448ea588e40", null]
]
```

Definition at line 1 of file classhed\_1\_1\_triangulation.js.

## 30.646 doc/html/classhetriang\_1\_1\_dart.js File Reference

### Variables

- var [classhetriang\\_1\\_1\\_dart](#)

### 30.646.1 Variable Documentation

#### 30.646.1.1 var classshetriang\_1\_1\_dart

##### Initial value:

```
=
[
 ["Dart", "classshetriang_1_1_dart.html#ab39198f2792ee004668b8fc7f6a2d54f", null],
 ["Dart", "classshetriang_1_1_dart.html#ad19d89b2aed2b3a8d928be1625c41522", null],
 ["Dart", "classshetriang_1_1_dart.html#a38272a3789a247be3b18bd270c7f7e0e", null],
 ["~Dart", "classshetriang_1_1_dart.html#ae08502655542ccd2b3cbf6e603ef73dc", null],
 ["alpha0", "classshetriang_1_1_dart.html#a6d878ddfd8a31ba8e4b23698e100872b", null],
 ["alpha1", "classshetriang_1_1_dart.html#a95db6d9a41551204844e02a8575d44bc", null],
 ["alpha2", "classshetriang_1_1_dart.html#a192635c8ff9df52bfa6c390e2f4d5536", null],
 ["controllConstDart", "classshetriang_1_1_dart.html#a8d6797ceec27b9e48d57205950aa44fc", null],
 ["getEdge", "classshetriang_1_1_dart.html#ab94fb992ea3a3c45e343d0dd84dd48ab", null],
 ["getNode", "classshetriang_1_1_dart.html#aae77ac412e6b22956aaa7c6b1d3a1c78", null],
 ["getOppositeNode", "classshetriang_1_1_dart.html#a1862f77e025467e122c1218367ee4676", null],
 ["init", "classshetriang_1_1_dart.html#a8f631b0ed26c011717753efc84994138", null],
 ["isCounterClockWise", "classshetriang_1_1_dart.html#a50d9018f82a02e6cca7704d9bd5c88b5", null],
 ["makeCopy", "classshetriang_1_1_dart.html#a2c8a3fee654f5b5e2ca9fc308541beb1", null],
 ["operator!=" , "classshetriang_1_1_dart.html#ae338fecbf1cb8f7085a7bde60440fe2b", null],
 ["operator=", "classshetriang_1_1_dart.html#aa0e9f7916775b8b8315b6b551b2458ce", null],
 ["operator==", "classshetriang_1_1_dart.html#a077ef450040dc0299b860502c092b6db", null],
 ["printDart", "classshetriang_1_1_dart.html#a0ca6545f0936282a42d2877e74c8d409", null],
 ["x", "classshetriang_1_1_dart.html#a54e0bbd87448ac2b3e9bddd8db856c59", null],
 ["y", "classshetriang_1_1_dart.html#af210f3f33a71b195c745e54cd7165070", null]
]
```

Definition at line 1 of file classshetriang\_1\_1\_dart.js.

## 30.647 doc/html/classshetriang\_1\_1\_edge.js File Reference

### Variables

- var [classshetriang\\_1\\_1\\_edge](#)

### 30.647.1 Variable Documentation

#### 30.647.1.1 var classshetriang\_1\_1\_edge

##### Initial value:

```
=
[
 ["Edge", "classshetriang_1_1_edge.html#ac66aa8601c31b0f664fb020e02d5e0bc", null],
 ["~Edge", "classshetriang_1_1_edge.html#a895709f6162fcd9d1fe4fb7b9d216f00", null],
 ["getNextEdgeInFace", "classshetriang_1_1_edge.html#a7d67ecaf416f2bdf5fa860c1c945008b", null],
 ["getSourceNode", "classshetriang_1_1_edge.html#a58f01f66e0c0d09dfcbe5f139ba5de13", null],
 ["getTargetNode", "classshetriang_1_1_edge.html#a575cc2013925af2a64c1d8c0889fa008", null],
 ["getTwinEdge", "classshetriang_1_1_edge.html#af05da617f77eae7847bc7c62a9f953c5", null],
 ["isLeadingEdge", "classshetriang_1_1_edge.html#ae9768c5a551c1c1d0f4bcae4aeb61210", null],
 ["setAsLeadingEdge", "classshetriang_1_1_edge.html#aed73071cf0278c599fb4907852b98870", null],
 ["setNextEdgeInFace", "classshetriang_1_1_edge.html#a16b6f10e7931ea9fa76dbc5fc8697e19", null],
 ["setSourceNode", "classshetriang_1_1_edge.html#a013164a18ec0fecaea5bc3569749c010", null],
 ["setTwinEdge", "classshetriang_1_1_edge.html#aae96f3d2686c6a0f2f0722af3071d695", null]
]
```

Definition at line 1 of file classshetriang\_1\_1\_edge.js.

## 30.648 doc/html/classhetriang\_1\_1\_node.js File Reference

### Variables

- var [classhetriang\\_1\\_1\\_node](#)

### 30.648.1 Variable Documentation

#### 30.648.1.1 var classhetriang\_1\_1\_node

##### Initial value:

```
=
[
 ["Node", "classhetriang_1_1_node.html#af180c24bc79338e59137679f0bd2bb41", null],
 ["Node", "classhetriang_1_1_node.html#ad2662e20a591a4e5508c52665a0c7d4f", null],
 ["Node", "classhetriang_1_1_node.html#ad2820a0242e6196055b28af4c7b05532", null],
 ["~Node", "classhetriang_1_1_node.html#a23aefd4c8c46c97814295553cbc8e84c", null],
 ["init", "classhetriang_1_1_node.html#aeb2bfc6f53bb23084f84b28ab60b01fa", null],
 ["pointIter", "classhetriang_1_1_node.html#a37efc562994a0f62b05ed6d2cc3f4deb", null],
 ["x", "classhetriang_1_1_node.html#a6d6619cfedeafeaf9e468da7670da5f", null],
 ["y", "classhetriang_1_1_node.html#a9e14e02fa7d2ec93a21f26108a6d1b36", null],
 ["z", "classhetriang_1_1_node.html#a3a3e75bffc96f2230699b7ec641afa70", null]
]
```

Definition at line 1 of file classhetriang\_1\_1\_node.js.

## 30.649 doc/html/classhetriang\_1\_1\_triangulation.js File Reference

### Variables

- var [classhetriang\\_1\\_1\\_triangulation](#)

### 30.649.1 Variable Documentation

#### 30.649.1.1 var classhetriang\_1\_1\_triangulation

##### Initial value:

```
=
[
 ["Triangulation", "classhetriang_1_1_triangulation.html#a5c275e8328957704b33555ad911ab587", null],
 ["~Triangulation", "classhetriang_1_1_triangulation.html#a99de211fd9c7e1e58dc4bd4ea3297b7f", null],
 ["addLeadingEdge", "classhetriang_1_1_triangulation.html#ae2794933e5e468b4056014e6cfc0f9a9", null],
 ["checkDelaunay", "classhetriang_1_1_triangulation.html#a45f97dc5781fe6ee0bbb95925e5460f3", null],
 ["cleanAll", "classhetriang_1_1_triangulation.html#a9455e4c7acfac72abe55d9aa81fcc512", null],
 ["createDart", "classhetriang_1_1_triangulation.html#a19be33d3b6a558aa7ef24e9408f15642", null],
 ["createDelaunay", "classhetriang_1_1_triangulation.html#aelaf06802fa3c3589b47c3e4273d2967", null],
 ["getBoundaryEdge", "classhetriang_1_1_triangulation.html#a768422ebc4abaf0d2464357d4a526a9c", null],
 ["getConstrainedEdges", "classhetriang_1_1_triangulation.html#ad54febf9b746624dcdc05ecc1ccdd752", null],
 ["getEdges", "classhetriang_1_1_triangulation.html#af704386eda232efcd59cdd65ee4117df", null],
 ["getInteriorNode", "classhetriang_1_1_triangulation.html#a2681b074818dd30161ba59432c0299b7", null],
 ["getLeadingEdges", "classhetriang_1_1_triangulation.html#ae582df5cb14aaac6a5634cd6308475c1", null],
 ["initTwoEnclosingTriangles", "classhetriang_1_1_triangulation.html#ac41010f48023ee255e3ea8fd4cccd154", null],
 ["noTriangles", "classhetriang_1_1_triangulation.html#aad57852ccbacc4206356f81c5e831eeaf", null],
 ["optimizeDelaunay", "classhetriang_1_1_triangulation.html#a06ee88d2a74dcd6bcfcc9036764fa483", null],
 ["printEdges", "classhetriang_1_1_triangulation.html#a801c6425f24252708e8ae83afe3f8810", null],
 ["removeLeadingEdgeFromList", "classhetriang_1_1_triangulation.html#a8ea06774eae80bc636dbd17a84551427", null],
 ["removeTriangle", "classhetriang_1_1_triangulation.html#a3358efc08b5d7aec9ee3025d94221a84", null],
 ["reverse_splitTriangle", "classhetriang_1_1_triangulation.html#a3e504982c51a506486900654b85d5c74", null],
 ["splitTriangle", "classhetriang_1_1_triangulation.html#a100090e62f19ca59a9db7913b33ba477", null],
 ["swapEdge", "classhetriang_1_1_triangulation.html#ab1f108065d213e088f65a3516bab8830", null],
 ["leadingEdges", "classhetriang_1_1_triangulation.html#a94625ecc7008e84a98fe7e5e55d572e4", null]
]
```

Definition at line 1 of file classhetriang\_1\_1\_triangulation.js.

## 30.650 doc/html/classmaterial\_\_appearance.js File Reference

### Variables

- var [classmaterial\\_\\_appearance](#)

### 30.650.1 Variable Documentation

#### 30.650.1.1 var classmaterial\_\_appearance

##### Initial value:

```
=
[
 ["material_appearance", "classmaterial__appearance.html#a8466b0b1fc034232d439d7836247eb2a", null],
 ["col_scheme_select", "classmaterial__appearance.html#a9b42c9958f67223ed59b7957e5460ccd", null],
 ["col_scheme_selected", "classmaterial__appearance.html#a6e3e961804250b37fb94fd4f8d9b91d4", null],
 ["Key", "classmaterial__appearance.html#a74882356eb34bffa5a8683cccbee97bc", null],
 ["redefine_material_f", "classmaterial__appearance.html#a7dc96d53ef17ed996f1a440f6401affe", null],
 ["set_material", "classmaterial__appearance.html#a4dcb592c91edfd83bf8071a93ae825e1", null],
 ["ztrans_delta", "classmaterial__appearance.html#ad623f36c5f08defe334ddebdc713213", null]
]
```

Definition at line 1 of file classmaterial\_\_appearance.js.

## 30.651 doc/html/classtime\_\_lapse.js File Reference

### Variables

- var [classtime\\_\\_lapse](#)

### 30.651.1 Variable Documentation

#### 30.651.1.1 var classtime\_\_lapse

##### Initial value:

```
=
[
 ["time_lapse", "classtime__lapse.html#a5d9d61b9c4788939518d2ba53134c7db", null],
 ["~time_lapse", "classtime__lapse.html#a4e009ca3428fea87ec471fd8928ecec5", null]
]
```

Definition at line 1 of file classtime\_\_lapse.js.

## 30.652 doc/html/classvector3t.js File Reference

### Variables

- var [classvector3t](#)

### 30.652.1 Variable Documentation

#### 30.652.1.1 var classvector3t

Definition at line 1 of file classvector3t.js.

## 30.653 doc/html/cm\_utils\_8h.js File Reference

### Variables

- var [cm\\_utils\\_8h](#)

### 30.653.1 Variable Documentation

#### 30.653.1.1 var cm\_utils\_8h

#### Initial value:

```
=
[
 ["RotationInfo", "struct_go_l_l_rotation_info.html", "struct_go_l_l_rotation_info"],
 ["abovePlane", "cm_utils_8h.html#a26e0f32f6d3a0730cab21ab3f6f5b2d7", null],
 ["ccwAngle", "cm_utils_8h.html#a01ec874c95173e8dd46783ba680f6cef", null],
 ["cwOrientation", "cm_utils_8h.html#a0e8d24cd5737086fc3e72524d818d2d6", null],
 ["cwOrientation2", "cm_utils_8h.html#a08a484efc37055f021aa7673e9e751c6", null],
 ["estimatedCurveLength", "cm_utils_8h.html#a8e9fb0320366e2f460d75b94516c9fb9", null],
 ["extendWithDegBd", "cm_utils_8h.html#a545c0187155c2c9eaf902c750e147bed", null],
 ["geometricParamDomain", "cm_utils_8h.html#ac2652cdc78ac020412157fd7a00dfb35", null],
 ["getG1FaceCurves", "cm_utils_8h.html#ac2b99ce2e93099cdd1b2c57f52f0badc", null],
 ["insideBdTrain", "cm_utils_8h.html#af9162aba2837232e341703617772f70f", null],
 ["removeInnerCorners", "cm_utils_8h.html#a8cf4e6373645963f9ca720691179845d", null],
 ["reparametrizeBdCvs", "cm_utils_8h.html#ad8dcc777e87bfb4635a08d64ffb6a8c5", null],
 ["reparametrizeBdCvs2", "cm_utils_8h.html#abdf559c1cd0036c494c2fa46fd513e00", null],
 ["splitEdgesInCorners", "cm_utils_8h.html#acf2000d3cda91fa89a0aa605160bb52e", null],
 ["updateWithNewCorners", "cm_utils_8h.html#a49a6b9a1873b425e7b4d58aa45676936", null]
]
```

Definition at line 1 of file cm\_utils\_8h.js.

## 30.654 doc/html/config\_8h.js File Reference

### Variables

- var [config\\_8h](#)

### 30.654.1 Variable Documentation

#### 30.654.1.1 var config\_8h

#### Initial value:

```
=
[
 ["GO_API", "config_8h.html#aa9fe0b1a0029021ae339602cdfc4d8dd", null]
]
```

Definition at line 1 of file config\_8h.js.

## 30.655 doc/html/construct\_8c.js File Reference

### Variables

- var [construct\\_8c](#)

### 30.655.1 Variable Documentation

#### 30.655.1.1 var construct\_8c

#### Initial value:

```
=
[
 ["CONSTRUCT", "construct_8c.html#a158cc7b98afd3caa1f2cf9c39be593b5", null],
 ["copyCurve", "construct_8c.html#aa3001587ca8d75c55a0cfdab5b00cdb0", null],
 ["copyIntpt", "construct_8c.html#aa7064efc6725011c38239accdafe31c8", null],
 ["copySurface", "construct_8c.html#aad0f7067335e41a2661cfdad0af2a3aa", null],
 ["hp_copyIntpt", "construct_8c.html#ace471e941e47cb4e67a1092f0f24332f", null],
 ["hp_newIntpt", "construct_8c.html#a4103bb7dd12c92e34a011792650d03c1", null],
 ["newbox", "construct_8c.html#aa43a378c821800eb747e91656ee4025f", null],
 ["newCurve", "construct_8c.html#a7bdc93268907b69b88098c5aeb2c4cd9", null],
 ["newdir", "construct_8c.html#a12f510a1d7f0ceda1ddc34dd746e9c7", null],
 ["newEdge", "construct_8c.html#af3c3222a7a78c75358f1eb0e4d31a11f", null],
 ["newIntcurve", "construct_8c.html#a809278570971cee58e55e46c740d39eb", null],
 ["newIntdat", "construct_8c.html#a9fa9789d8486e935781121bf46477d07", null],
 ["newIntlist", "construct_8c.html#ab32c3a857ac6220e6313e86f16136423", null],
 ["newIntpt", "construct_8c.html#af6f2e968a7128a48bdbb0f08110bb591", null],
 ["newIntsurf", "construct_8c.html#ac9b1e743272484ca8ee89b5c39b668e8", null],
 ["newObject", "construct_8c.html#a5b46207bedb046db4cb153dd3296b9f3", null],
 ["newPoint", "construct_8c.html#aeea6e5cb7fb2d4e10f2f49ba51efe64f", null],
 ["newPtedge", "construct_8c.html#a5b193661893ad8eba9c64ccc4a1cb8ea", null],
 ["newSurf", "construct_8c.html#a8602c3c5b4facb0920af3f4ba76998e4", null],
 ["newTrack", "construct_8c.html#a6bf58012098fdda7c58599ecf3313a", null],
 ["newTrimpar", "construct_8c.html#a6cb323fc23f259cc85ccbcaec5da613a", null]
]
```

Definition at line 1 of file `construct_8c.js`.

## 30.656 doc/html/controlw\_8h.js File Reference

### Variables

- var [controlw\\_8h](#)

### 30.656.1 Variable Documentation

#### 30.656.1.1 var controlw\_8h

#### Initial value:

```
=
[
 ["ControlWord", "class_control_word.html", "class_control_word"],
 ["CONTROL_WORD_LIB", "controlw_8h.html#ac3f4cc42dfc44a8de07e00956f1f192e", null]
]
```

Definition at line 1 of file `controlw_8h.js`.



## 30.657 doc/html/crvarctang\_8c.js File Reference

### Variables

- var [crvarctang\\_8c](#)

#### 30.657.1 Variable Documentation

##### 30.657.1.1 var crvarctang\_8c

###### Initial value:

```
=
[
 ["CRV_ARC_TANG", "crvarctang_8c.html#a17200ba2d085d7b9c14b679c3da87cbe", null],
 ["crv_arc_tang", "crvarctang_8c.html#a7adfd2ca3966585e5529e6ca32d4a81e", null]
]
```

Definition at line 1 of file crvarctang\_8c.js.

## 30.658 doc/html/crvcrvtang\_8c.js File Reference

### Variables

- var [crvcrvtang\\_8c](#)

#### 30.658.1 Variable Documentation

##### 30.658.1.1 var crvcrvtang\_8c

###### Initial value:

```
=
[
 ["CRV_CRV_TANG", "crvcrvtang_8c.html#a9487b21dbbf1d4857d4ed51fcc0e8db1", null],
 ["crv_crv_tang", "crvcrvtang_8c.html#a72743885abdacd2d0cf39ddd07514f71", null]
]
```

Definition at line 1 of file crvcrvtang\_8c.js.

## 30.659 doc/html/crvlintang\_8c.js File Reference

### Variables

- var [crvlintang\\_8c](#)

### 30.659.1 Variable Documentation

#### 30.659.1.1 var crvlintang\_8c

##### Initial value:

```
=
[
 ["CRV_LIN_TANG", "crvlintang_8c.html#a1e18059c1583f19c55dd5e69c0314906", null],
 ["crv_lin_tang", "crvlintang_8c.html#a83d7f10b2e18d02eba8ab8ec634b48e6", null]
]
```

Definition at line 1 of file crvlintang\_8c.js.

### 30.660 doc/html/destruct\_8c.js File Reference

#### Variables

- var [destruct\\_8c](#)

### 30.660.1 Variable Documentation

#### 30.660.1.1 var destruct\_8c

##### Initial value:

```
=
[
 ["DESTRUCT", "destruct_8c.html#a9512e89b4c1c559e674c9903198452eb", null],
 ["freeCurve", "destruct_8c.html#a81fad8c55f6f41ef2e21849beeb8791a", null],
 ["freeEdge", "destruct_8c.html#a104289e6cd925030b80dd28e7ccb8aaa", null],
 ["freeIntcrvlist", "destruct_8c.html#a67efal3cf2156e880d13ffd31c1d32c4", null],
 ["freeIntcurve", "destruct_8c.html#aae3c8b6b942b0944dfb8a92bc3ce2e78", null],
 ["freeIntdat", "destruct_8c.html#ac6fe5785f75f0009a29fcdaaf2084322", null],
 ["freeIntlist", "destruct_8c.html#ab8527ad1c7alb02c15c6d6682aee47a8", null],
 ["freeIntpt", "destruct_8c.html#a2ecde8800945071536e77a0878c0f44c", null],
 ["freeIntsurf", "destruct_8c.html#a0af6ad02906a7101ec91a6cf658e0903", null],
 ["freeObject", "destruct_8c.html#a39971c3e7bc50f706a83c7202e7b1822", null],
 ["freePoint", "destruct_8c.html#aabbfe0b8406369039c17f5d3214765d9", null],
 ["freePtedge", "destruct_8c.html#a7665f3a185f509fb5a86465716cce656", null],
 ["freeSurf", "destruct_8c.html#acbf603611b685b194721e1aaedb217f2", null],
 ["freeTrack", "destruct_8c.html#af0068b3a230501f40ba8f8a4cb956fc9", null],
 ["freeTrimpar", "destruct_8c.html#ae86eb459eeb4221d7514132bf3f6b33d", null]
]
```

Definition at line 1 of file destruct\_8c.js.

### 30.661 doc/html/dir\_0015c891e2eac5fd9e489e846b834700.js File Reference

#### Variables

- var [dir\\_0015c891e2eac5fd9e489e846b834700](#)

### 30.661.1 Variable Documentation

30.661.1.1 `var dir_0015c891e2eac5fd9e489e846b834700`

#### Initial value:

```
=
[
 ["topology", "dir_3a8e8a1c81bc0d4ea45d413753d6b9d6.html", "dir_3a8e8a1c81bc0d4ea45d413753d6b9d6"]
]
```

Definition at line 1 of file dir\_0015c891e2eac5fd9e489e846b834700.js.

## 30.662 doc/html/dir\_038b4c324d5349c9475465ffee024b1c.js File Reference

### Variables

- var [dir\\_038b4c324d5349c9475465ffee024b1c](#)

### 30.662.1 Variable Documentation

30.662.1.1 `var dir_038b4c324d5349c9475465ffee024b1c`

Definition at line 1 of file dir\_038b4c324d5349c9475465ffee024b1c.js.

## 30.663 doc/html/dir\_04f3dbfb06d4e757244c31fc16142d05.js File Reference

### Variables

- var [dir\\_04f3dbfb06d4e757244c31fc16142d05](#)

### 30.663.1 Variable Documentation

30.663.1.1 `var dir_04f3dbfb06d4e757244c31fc16142d05`

#### Initial value:

```
=
[
 ["halfedge", "dir_787e29cc9b8f49ef343dc9ff9fbda368.html", "dir_787e29cc9b8f49ef343dc9ff9fbda368"],
 ["utils", "dir_4dc6ea3c21164f9c4718441280129d09.html", "dir_4dc6ea3c21164f9c4718441280129d09"],
 ["api.h", "api_8h.html", null],
 ["mainpage_ttl.h", "mainpage__ttl_8h.html", null],
 ["ttl.h", "ttl_8h.html", "ttl_8h"],
 ["ttl_constr.h", "ttl__constr_8h.html", "ttl__constr_8h"],
 ["ttl_util.h", "ttl__util_8h.html", "ttl__util_8h"]
]
```

Definition at line 1 of file dir\_04f3dbfb06d4e757244c31fc16142d05.js.

## 30.664 doc/html/dir\_0a492bd6eb1e5e04630449d6b6b92955.js File Reference

### Variables

- var [dir\\_0a492bd6eb1e5e04630449d6b6b92955](#)

### 30.664.1 Variable Documentation

#### 30.664.1.1 var dir\_0a492bd6eb1e5e04630449d6b6b92955

##### Initial value:

```
=
[
 ["GoTools", "dir_27047e9e82ebbd34bcfea6195a1d237f.html", "dir_27047e9e82ebbd34bcfea6195a1d237f"]
]
```

Definition at line 1 of file dir\_0a492bd6eb1e5e04630449d6b6b92955.js.

## 30.665 doc/html/dir\_0df052b1ef5b53594ca9be5a6074277d.js File Reference

### Variables

- var [dir\\_0df052b1ef5b53594ca9be5a6074277d](#)

### 30.665.1 Variable Documentation

#### 30.665.1.1 var dir\_0df052b1ef5b53594ca9be5a6074277d

##### Initial value:

```
=
[
 ["include", "dir_0a492bd6eb1e5e04630449d6b6b92955.html", "dir_0a492bd6eb1e5e04630449d6b6b92955"]
]
```

Definition at line 1 of file dir\_0df052b1ef5b53594ca9be5a6074277d.js.

## 30.666 doc/html/dir\_15757c9e0b20f079697e78f9f294c1ee.js File Reference

### Variables

- var [dir\\_15757c9e0b20f079697e78f9f294c1ee](#)

### 30.666.1 Variable Documentation

30.666.1.1 `var dir_15757c9e0b20f079697e78f9f294c1ee`

Definition at line 1 of file `dir_15757c9e0b20f079697e78f9f294c1ee.js`.

## 30.667 doc/html/dir\_1c9e4414cb2b8eb9526a3aa2069941f8.js File Reference

### Variables

- `var dir\_1c9e4414cb2b8eb9526a3aa2069941f8`

### 30.667.1 Variable Documentation

30.667.1.1 `var dir_1c9e4414cb2b8eb9526a3aa2069941f8`

#### Initial value:

```
=
[
 ["GoReadWrite.h", "_go_read_write_8h.html", "_go_read_write_8h"]
]
```

Definition at line 1 of file `dir_1c9e4414cb2b8eb9526a3aa2069941f8.js`.

## 30.668 doc/html/dir\_1ce88f9178ec82ce3d3b5366984a4a76.js File Reference

### Variables

- `var dir\_1ce88f9178ec82ce3d3b5366984a4a76`

### 30.668.1 Variable Documentation

30.668.1.1 `var dir_1ce88f9178ec82ce3d3b5366984a4a76`

#### Initial value:

```
=
[
 ["doc", "dir_8946de0dbd7af8713b300be94ace6b21.html", "dir_8946de0dbd7af8713b300be94ace6b21"],
 ["include", "dir_61f4d2722e3a1702decd4dcefd169ded.html", "dir_61f4d2722e3a1702decd4dcefd169ded"]
]
```

Definition at line 1 of file `dir_1ce88f9178ec82ce3d3b5366984a4a76.js`.

## 30.669 doc/html/dir\_26434c1759c56a49292c328c033e76d7.js File Reference

### Variables

- var [dir\\_26434c1759c56a49292c328c033e76d7](#)

### 30.669.1 Variable Documentation

#### 30.669.1.1 var dir\_26434c1759c56a49292c328c033e76d7

Definition at line 1 of file dir\_26434c1759c56a49292c328c033e76d7.js.

## 30.670 doc/html/dir\_27047e9e82ebbd34bcfea6195a1d237f.js File Reference

### Variables

- var [dir\\_27047e9e82ebbd34bcfea6195a1d237f](#)

### 30.670.1 Variable Documentation

#### 30.670.1.1 var dir\_27047e9e82ebbd34bcfea6195a1d237f

#### Initial value:

```
=
[
 ["viewlib", "dir_038b4c324d5349c9475465ffee024b1c.html", "dir_038b4c324d5349c9475465ffee024b1c"]
]
```

Definition at line 1 of file dir\_27047e9e82ebbd34bcfea6195a1d237f.js.

## 30.671 doc/html/dir\_287d50cea2dbd2d342307d58f17ae566.js File Reference

### Variables

- var [dir\\_287d50cea2dbd2d342307d58f17ae566](#)

### 30.671.1 Variable Documentation

#### 30.671.1.1 var dir\_287d50cea2dbd2d342307d58f17ae566

#### Initial value:

```
=
[
 ["ftGroupGeom.h", "ft_group_geom_8h.html", [
 ["ftGroupGeom", "class_go_1_lft_group_geom.html", "class_go_1_lft_group_geom"]
]],
 ["ftTangPriority.h", "ft_tang_priority_8h.html", "ft_tang_priority_8h"],
 ["IGESconverter.h", "_i_g_e_converter_8h.html", "_i_g_e_converter_8h"],
 ["igeslib_doxymain.h", "igeslib__doxymain_8h.html", "igeslib__doxymain_8h"]
]
```

Definition at line 1 of file dir\_287d50cea2dbd2d342307d58f17ae566.js.

## 30.672 doc/html/dir\_28d725a1074140e3875acd99df863a9a.js File Reference

### Variables

- var [dir\\_28d725a1074140e3875acd99df863a9a](#)

#### 30.672.1 Variable Documentation

##### 30.672.1.1 var dir\_28d725a1074140e3875acd99df863a9a

Definition at line 1 of file dir\_28d725a1074140e3875acd99df863a9a.js.

## 30.673 doc/html/dir\_2e9f58fed2f8cb33783712134eca6f54.js File Reference

### Variables

- var [dir\\_2e9f58fed2f8cb33783712134eca6f54](#)

#### 30.673.1 Variable Documentation

##### 30.673.1.1 var dir\_2e9f58fed2f8cb33783712134eca6f54

Definition at line 1 of file dir\_2e9f58fed2f8cb33783712134eca6f54.js.

## 30.674 doc/html/dir\_2ea40cee675de4bd87b06c9233dde18b.js File Reference

### Variables

- var [dir\\_2ea40cee675de4bd87b06c9233dde18b](#)

#### 30.674.1 Variable Documentation

##### 30.674.1.1 var dir\_2ea40cee675de4bd87b06c9233dde18b

#### Initial value:

```
=
[
 ["intersections", "dir_90f926ef93e71fdbf5723e062c9da4.html", "dir_90f926ef93e71fdbf5723e062c9da4"
]
]
```

Definition at line 1 of file dir\_2ea40cee675de4bd87b06c9233dde18b.js.

## 30.675 doc/html/dir\_34a54fc55cc4c76e2db47ca198785905.js File Reference

### Variables

- var [dir\\_34a54fc55cc4c76e2db47ca198785905](#)

### 30.675.1 Variable Documentation

#### 30.675.1.1 var dir\_34a54fc55cc4c76e2db47ca198785905

##### Initial value:

```
=
[
 ["GoTools", "dir_ab08e349df88865f421593b417138732.html", "dir_ab08e349df88865f421593b417138732"]
]
```

Definition at line 1 of file dir\_34a54fc55cc4c76e2db47ca198785905.js.

## 30.676 doc/html/dir\_34a7c6eb6b0d9060dd547ea008608408.js File Reference

### Variables

- var [dir\\_34a7c6eb6b0d9060dd547ea008608408](#)

### 30.676.1 Variable Documentation

#### 30.676.1.1 var dir\_34a7c6eb6b0d9060dd547ea008608408

##### Initial value:

```
=
[
 ["GoTools", "dir_b28ca551f00416d098dfb2d5d86cb85e.html", "dir_b28ca551f00416d098dfb2d5d86cb85e"]
]
```

Definition at line 1 of file dir\_34a7c6eb6b0d9060dd547ea008608408.js.

## 30.677 doc/html/dir\_3706bca45f06e8690f2585ec746205ce.js File Reference

### Variables

- var [dir\\_3706bca45f06e8690f2585ec746205ce](#)



### 30.677.1 Variable Documentation

30.677.1.1 `var dir_3706bca45f06e8690f2585ec746205ce`

#### Initial value:

```
=
[
 ["igeslib", "dir_287d50cea2dbd2d342307d58f17ae566.html", "dir_287d50cea2dbd2d342307d58f17ae566"]
]
```

Definition at line 1 of file dir\_3706bca45f06e8690f2585ec746205ce.js.

## 30.678 doc/html/dir\_37bdf26acb6f0e99095c72f4b261057d.js File Reference

### Variables

- var [dir\\_37bdf26acb6f0e99095c72f4b261057d](#)

### 30.678.1 Variable Documentation

30.678.1.1 `var dir_37bdf26acb6f0e99095c72f4b261057d`

#### Initial value:

```
=
[
 ["include", "dir_eb531be94dc3f2da4c4acbfa84e0e7dc.html", "dir_eb531be94dc3f2da4c4acbfa84e0e7dc"]
]
```

Definition at line 1 of file dir\_37bdf26acb6f0e99095c72f4b261057d.js.

## 30.679 doc/html/dir\_3a8e8a1c81bc0d4ea45d413753d6b9d6.js File Reference

### Variables

- var [dir\\_3a8e8a1c81bc0d4ea45d413753d6b9d6](#)

### 30.679.1 Variable Documentation

30.679.1.1 `var dir_3a8e8a1c81bc0d4ea45d413753d6b9d6`

Definition at line 1 of file dir\_3a8e8a1c81bc0d4ea45d413753d6b9d6.js.

## 30.680 doc/html/dir\_3b29025e4f1a16d99c0ac7281d29a749.js File Reference

### Variables

- var [dir\\_3b29025e4f1a16d99c0ac7281d29a749](#)

### 30.680.1 Variable Documentation

#### 30.680.1.1 var dir\_3b29025e4f1a16d99c0ac7281d29a749

##### Initial value:

```
=
[
 ["include", "dir_34a7c6eb6b0d9060dd547ea008608408.html", "dir_34a7c6eb6b0d9060dd547ea008608408"]
]
```

Definition at line 1 of file dir\_3b29025e4f1a16d99c0ac7281d29a749.js.

## 30.681 doc/html/dir\_3cd14e887b3be1bb0699a0f4b9d7471b.js File Reference

### Variables

- var [dir\\_3cd14e887b3be1bb0699a0f4b9d7471b](#)

### 30.681.1 Variable Documentation

#### 30.681.1.1 var dir\_3cd14e887b3be1bb0699a0f4b9d7471b

Definition at line 1 of file dir\_3cd14e887b3be1bb0699a0f4b9d7471b.js.

## 30.682 doc/html/dir\_442358e1b75f159df2b4b10f68cb6669.js File Reference

### Variables

- var [dir\\_442358e1b75f159df2b4b10f68cb6669](#)

### 30.682.1 Variable Documentation

#### 30.682.1.1 var dir\_442358e1b75f159df2b4b10f68cb6669

##### Initial value:

```
=
[
 ["aux2.h", "aux2_8h.html", "aux2_8h"],
 ["gl_aux.h", "gl__aux_8h.html", "gl__aux_8h"],
 ["glutils.h", "glutils_8h.html", "glutils_8h"],
 ["jonvec.h", "jonvec_8h.html", [
 ["vector3t", "classvector3t.html", "classvector3t"]
]],
 ["mouse.h", "mouse_8h.html", "mouse_8h"],
 ["sisl_aux.h", "sisl__aux_8h.html", "sisl__aux_8h"],
 ["transfutils.h", "transfutils_8h.html", "transfutils_8h"]
]
```

Definition at line 1 of file dir\_442358e1b75f159df2b4b10f68cb6669.js.

## 30.683 doc/html/dir\_44eb60e2f3e103c46810fc3a6f6f0b38.js File Reference

### Variables

- var [dir\\_44eb60e2f3e103c46810fc3a6f6f0b38](#)

### 30.683.1 Variable Documentation

#### 30.683.1.1 var dir\_44eb60e2f3e103c46810fc3a6f6f0b38

##### Initial value:

```
=
[
 ["include", "dir_46fb2897033f53c047d4860af5672505.html", "dir_46fb2897033f53c047d4860af5672505"]
]
```

Definition at line 1 of file dir\_44eb60e2f3e103c46810fc3a6f6f0b38.js.

## 30.684 doc/html/dir\_46fb2897033f53c047d4860af5672505.js File Reference

### Variables

- var [dir\\_46fb2897033f53c047d4860af5672505](#)

### 30.684.1 Variable Documentation

#### 30.684.1.1 var dir\_46fb2897033f53c047d4860af5672505

##### Initial value:

```
=
[
 ["GoTools", "dir_a702eb1d615fc8e494327e00bfd64e2d.html", "dir_a702eb1d615fc8e494327e00bfd64e2d"]
]
```

Definition at line 1 of file dir\_46fb2897033f53c047d4860af5672505.js.

### 30.685 doc/html/dir\_4dc6ea3c21164f9c4718441280129d09.js File Reference

#### Variables

- var [dir\\_4dc6ea3c21164f9c4718441280129d09](#)

### 30.685.1 Variable Documentation

#### 30.685.1.1 var dir\_4dc6ea3c21164f9c4718441280129d09

##### Initial value:

```
=
[
 ["Handle.h", "_handle_8h.html", [
 ["Handle", "class_handle.html", "class_handle"]
]],
 ["HandleId.h", "_handle_id_8h.html", [
 ["HandleId", "class_handle_id.html", "class_handle_id"]
]]
]
```

Definition at line 1 of file dir\_4dc6ea3c21164f9c4718441280129d09.js.

### 30.686 doc/html/dir\_4ed957b398f8a342f35e202c6dbeff82.js File Reference

#### Variables

- var [dir\\_4ed957b398f8a342f35e202c6dbeff82](#)

### 30.686.1 Variable Documentation

#### 30.686.1.1 var dir\_4ed957b398f8a342f35e202c6dbeff82

##### Initial value:

```
=
[
 ["trivariatemodel", "dir_4f584829c2589cd0dd901c7bd8d4b8ca.html", "dir_4f584829c2589cd0dd901c7bd8d4b8ca"
]
]
```

Definition at line 1 of file dir\_4ed957b398f8a342f35e202c6dbeff82.js.

## 30.687 doc/html/dir\_4f584829c2589cd0dd901c7bd8d4b8ca.js File Reference

### Variables

- var [dir\\_4f584829c2589cd0dd901c7bd8d4b8ca](#)

### 30.687.1 Variable Documentation

#### 30.687.1.1 var dir\_4f584829c2589cd0dd901c7bd8d4b8ca

##### Initial value:

```
=
[
 ["examples_trivariatemodel_doxyman.h", "examples__trivariatemodel__doxyman_8h.html", null],
 ["ftVolume.h", "ft_volume_8h.html", [
 ["VolumeAdjacencyInfo", "struct_go_1_1_volume_adjacency_info.html", "
 struct_go_1_1_volume_adjacency_info"],
 ["ftVolume", "class_go_1_1ft_volume.html", "class_go_1_1ft_volume"]
]],
 ["ftVolumeTools.h", "ft_volume_tools_8h.html", "ft_volume_tools_8h"],
 ["trivariatemodel_doxyman.h", "trivariatemodel__doxyman_8h.html", null],
 ["VolumeAdjacency.h", "_volume_adjacency_8h.html", [
 ["VolumeAdjacency", "class_go_1_1_volume_adjacency.html", "class_go_1_1_volume_adjacency"]
]],
 ["VolumeModel.h", "_volume_model_8h.html", [
 ["VolumeModel", "class_go_1_1_volume_model.html", "class_go_1_1_volume_model"]
]],
 ["VolumeModelCreator.h", "_volume_model_creator_8h.html", "_volume_model_creator_8h"],
 ["VolumeModelFileHandler.h", "_volume_model_file_handler_8h.html", "_volume_model_file_handler_8h"]
]
```

Definition at line 1 of file dir\_4f584829c2589cd0dd901c7bd8d4b8ca.js.

## 30.688 doc/html/dir\_54c0807dc9dc58401cdf98175b301107.js File Reference

### Variables

- var [dir\\_54c0807dc9dc58401cdf98175b301107](#)

### 30.688.1 Variable Documentation

30.688.1.1 `var dir_54c0807dc9dc58401cdf98175b301107`

Definition at line 1 of file `dir_54c0807dc9dc58401cdf98175b301107.js`.

## 30.689 doc/html/dir\_5eaf107a44af8927c3e88d7693acb3ec.js File Reference

### Variables

- `var dir_5eaf107a44af8927c3e88d7693acb3ec`

### 30.689.1 Variable Documentation

30.689.1.1 `var dir_5eaf107a44af8927c3e88d7693acb3ec`

#### Initial value:

```
=
[
 ["jquery.js", "jquery_8js.html", "jquery_8js"]
]
```

Definition at line 1 of file `dir_5eaf107a44af8927c3e88d7693acb3ec.js`.

## 30.690 doc/html/dir\_61c47cbd77726dc04327162bb82f2acf.js File Reference

### Variables

- `var dir_61c47cbd77726dc04327162bb82f2acf`

### 30.690.1 Variable Documentation

30.690.1.1 `var dir_61c47cbd77726dc04327162bb82f2acf`

#### Initial value:

```
=
[
 ["include", "dir_7efa0b394c677aa9d932dff09cb14a2d.html", "dir_7efa0b394c677aa9d932dff09cb14a2d"]
]
```

Definition at line 1 of file `dir_61c47cbd77726dc04327162bb82f2acf.js`.

## 30.691 doc/html/dir\_61f4d2722e3a1702decd4dcefd169ded.js File Reference

### Variables

- var [dir\\_61f4d2722e3a1702decd4dcefd169ded](#)

#### 30.691.1 Variable Documentation

30.691.1.1 var [dir\\_61f4d2722e3a1702decd4dcefd169ded](#)

##### Initial value:

```
=
[
 ["GoTools", "dir_ffa09c8575bbfbed340eb641d2e0178f.html", "dir_ffa09c8575bbfbed340eb641d2e0178f"]
]
```

Definition at line 1 of file [dir\\_61f4d2722e3a1702decd4dcefd169ded.js](#).

## 30.692 doc/html/dir\_662c58e926d5c8a2e92c3a0efd184e27.js File Reference

### Variables

- var [dir\\_662c58e926d5c8a2e92c3a0efd184e27](#)

#### 30.692.1 Variable Documentation

30.692.1.1 var [dir\\_662c58e926d5c8a2e92c3a0efd184e27](#)

##### Initial value:

```
=
[
 ["include", "dir_9e7f8af3df212ec4d479463341bb72a6.html", "dir_9e7f8af3df212ec4d479463341bb72a6"]
]
```

Definition at line 1 of file [dir\\_662c58e926d5c8a2e92c3a0efd184e27.js](#).

## 30.693 doc/html/dir\_69009d91722e8158a6e1821a3d198fd0.js File Reference

### Variables

- var [dir\\_69009d91722e8158a6e1821a3d198fd0](#)

### 30.693.1 Variable Documentation

30.693.1.1 `var dir_69009d91722e8158a6e1821a3d198fd0`

Definition at line 1 of file `dir_69009d91722e8158a6e1821a3d198fd0.js`.

## 30.694 doc/html/dir\_6917c6a4538152e6852497b387e02df2.js File Reference

### Variables

- `var dir_6917c6a4538152e6852497b387e02df2`

### 30.694.1 Variable Documentation

30.694.1.1 `var dir_6917c6a4538152e6852497b387e02df2`

#### Initial value:

```
=
[
 ["main.cpp", "main_8cpp.html", "main_8cpp"]
]
```

Definition at line 1 of file `dir_6917c6a4538152e6852497b387e02df2.js`.

## 30.695 doc/html/dir\_6958061072da6881959e38b0a737a4c6.js File Reference

### Variables

- `var dir_6958061072da6881959e38b0a737a4c6`

### 30.695.1 Variable Documentation

30.695.1.1 `var dir_6958061072da6881959e38b0a737a4c6`

#### Initial value:

```
=
[
 ["halfedge", "dir_8dcf46dfbb2e2ca3d570ad71ab7f5984.html", "dir_8dcf46dfbb2e2ca3d570ad71ab7f5984"],
 ["utils", "dir_e846bdbb82b84d132cb98df96cb07bf0.html", "dir_e846bdbb82b84d132cb98df96cb07bf0"]
]
```

Definition at line 1 of file `dir_6958061072da6881959e38b0a737a4c6.js`.



## 30.696 doc/html/dir\_6a9f218ac24eb7b754dc250ffd508dae.js File Reference

### Variables

- var [dir\\_6a9f218ac24eb7b754dc250ffd508dae](#)

#### 30.696.1 Variable Documentation

##### 30.696.1.1 var dir\_6a9f218ac24eb7b754dc250ffd508dae

###### Initial value:

```
=
[
 ["GoTools", "dir_ce2be4821eafb03cec507f192e45eceb.html", "dir_ce2be4821eafb03cec507f192e45eceb"]
]
```

Definition at line 1 of file dir\_6a9f218ac24eb7b754dc250ffd508dae.js.

## 30.697 doc/html/dir\_764e20a7fa93cafa3199ac45b6e6b986.js File Reference

### Variables

- var [dir\\_764e20a7fa93cafa3199ac45b6e6b986](#)

#### 30.697.1 Variable Documentation

##### 30.697.1.1 var dir\_764e20a7fa93cafa3199ac45b6e6b986

###### Initial value:

```
=
[
 ["ttl", "dir_04f3dbfb06d4e757244c31fc16142d05.html", "dir_04f3dbfb06d4e757244c31fc16142d05"]
]
```

Definition at line 1 of file dir\_764e20a7fa93cafa3199ac45b6e6b986.js.

## 30.698 doc/html/dir\_787e29cc9b8f49ef343dc9ff9fbda368.js File Reference

### Variables

- var [dir\\_787e29cc9b8f49ef343dc9ff9fbda368](#)

### 30.698.1 Variable Documentation

#### 30.698.1.1 var dir\_787e29cc9b8f49ef343dc9ff9fbd368

##### Initial value:

```
=
[
 ["HeDart.h", "_he_dart_8h.html", [
 ["Dart", "classhed_l_l_dart.html", "classhed_l_l_dart"]
]],
 ["HeTraits.h", "_he_traits_8h.html", [
 ["TTLtraits", "structhed_l_l_t_t_ltraits.html", "structhed_l_l_t_t_ltraits"]
]],
 ["HeTriang.h", "_he_triang_8h.html", "_he_triang_8h"],
 ["main_he_ref.h", "main_he_ref_8h.html", null],
 ["mainpage_hed.h", "mainpage__hed_8h.html", null]
]
```

Definition at line 1 of file dir\_787e29cc9b8f49ef343dc9ff9fbd368.js.

### 30.699 doc/html/dir\_79b3003bf1bdb319185970350806bf23.js File Reference

#### Variables

- var [dir\\_79b3003bf1bdb319185970350806bf23](#)

### 30.699.1 Variable Documentation

#### 30.699.1.1 var dir\_79b3003bf1bdb319185970350806bf23

##### Initial value:

```
=
[
 ["app", "dir_15757c9e0b20f079697e78f9f294c1ee.html", "dir_15757c9e0b20f079697e78f9f294c1ee"],
 ["include", "dir_7f10ec973083a3eaca908361800e49c7.html", "dir_7f10ec973083a3eaca908361800e49c7"],
 ["src", "dir_892cdac2bfffabe6fa625049467eefe0a.html", "dir_892cdac2bfffabe6fa625049467eefe0a"]
]
```

Definition at line 1 of file dir\_79b3003bf1bdb319185970350806bf23.js.

### 30.700 doc/html/dir\_7b91ef4997e28bdb148c447548446353.js File Reference

#### Variables

- var [dir\\_7b91ef4997e28bdb148c447548446353](#)

### 30.700.1 Variable Documentation

#### 30.700.1.1 var dir\_7b91ef4997e28bdb148c447548446353

##### Initial value:

```
=
[
 ["include", "dir_34a54fc55cc4c76e2db47ca198785905.html", "dir_34a54fc55cc4c76e2db47ca198785905"]
]
```

Definition at line 1 of file dir\_7b91ef4997e28bdb148c447548446353.js.

## 30.701 doc/html/dir\_7c72f061721a5863212d64c180702103.js File Reference

### Variables

- var [dir\\_7c72f061721a5863212d64c180702103](#)

### 30.701.1 Variable Documentation

#### 30.701.1.1 var dir\_7c72f061721a5863212d64c180702103

##### Initial value:

```
=
[
 ["FaceSetQuality.h", "_face_set_quality_8h.html", [
 ["FaceSetQuality", "class_go_1_1_face_set_quality.html", "class_go_1_1_face_set_quality"]
]],
 ["FaceSetRepair.h", "_face_set_repair_8h.html", [
 ["FaceSetRepair", "class_go_1_1_face_set_repair.html", "class_go_1_1_face_set_repair"]
]],
 ["ModelQuality.h", "_model_quality_8h.html", [
 ["ModelQuality", "class_go_1_1_model_quality.html", "class_go_1_1_model_quality"]
]],
 ["ModelRepair.h", "_model_repair_8h.html", [
 ["ModelRepair", "class_go_1_1_model_repair.html", "class_go_1_1_model_repair"]
]],
 ["qualitymodule-doxymain.h", "qualitymodule-doxymain_8h.html", null],
 ["QualityResults.h", "_quality_results_8h.html", [
 ["QualityResults", "class_go_1_1_quality_results.html", "class_go_1_1_quality_results"]
]],
 ["QualityUtils.h", "_quality_utils_8h.html", "_quality_utils_8h"],
 ["testSuite.h", "test_suite_8h.html", "test_suite_8h"]
]
```

Definition at line 1 of file dir\_7c72f061721a5863212d64c180702103.js.

## 30.702 doc/html/dir\_7efa0b394c677aa9d932dff09cb14a2d.js File Reference

### Variables

- var [dir\\_7efa0b394c677aa9d932dff09cb14a2d](#)

### 30.702.1 Variable Documentation

#### 30.702.1.1 var dir\_7efa0b394c677aa9d932dff09cb14a2d

##### Initial value:

```
=
[
 ["GoTools", "dir_3706bca45f06e8690f2585ec746205ce.html", "dir_3706bca45f06e8690f2585ec746205ce"]
]
```

Definition at line 1 of file dir\_7efa0b394c677aa9d932dff09cb14a2d.js.

## 30.703 doc/html/dir\_7f10ec973083a3eaca908361800e49c7.js File Reference

### Variables

- var [dir\\_7f10ec973083a3eaca908361800e49c7](#)

### 30.703.1 Variable Documentation

#### 30.703.1.1 var dir\_7f10ec973083a3eaca908361800e49c7

##### Initial value:

```
=
[
 ["boolean.h", "boolean_8h.html", "boolean_8h"],
 ["controlw.h", "controlw_8h.html", "controlw_8h"],
 ["include.h", "include_8h.html", "include_8h"],
 ["myexcept.h", "myexcept_8h.html", "myexcept_8h"],
 ["newmat.h", "newmat_8h.html", "newmat_8h"],
 ["newmatap.h", "newmatap_8h.html", "newmatap_8h"],
 ["newmatio.h", "newmatio_8h.html", "newmatio_8h"],
 ["newmatnl.h", "newmatnl_8h.html", "newmatnl_8h"],
 ["newmatrc.h", "newmatrc_8h.html", "newmatrc_8h"],
 ["newmatrm.h", "newmatrm_8h.html", "newmatrm_8h"],
 ["precisio.h", "precisio_8h.html", "precisio_8h"],
 ["solution.h", "solution_8h.html", [
 ["R1_R1", "class_r_b_d___c_o_m_m_o_n_l_l_r1___r1.html", "class_r_b_d___c_o_m_m_o_n_l_l_r1___r1"],
 ["SolutionException", "class_r_b_d___c_o_m_m_o_n_l_l_solution_exception.html", "
 class_r_b_d___c_o_m_m_o_n_l_l_solution_exception"],
 ["OneDimSolve", "class_r_b_d___c_o_m_m_o_n_l_l_one_dim_solve.html", "
 class_r_b_d___c_o_m_m_o_n_l_l_one_dim_solve"]
]],
 ["tmt.h", "tmt_8h.html", "tmt_8h"]
]
```

Definition at line 1 of file dir\_7f10ec973083a3eaca908361800e49c7.js.

## 30.704 doc/html/dir\_84497794bb37d283dbd68163bed8ef05.js File Reference

### Variables

- var [dir\\_84497794bb37d283dbd68163bed8ef05](#)

### 30.704.1 Variable Documentation

30.704.1.1 var dir\_84497794bb37d283dbd68163bed8ef05

**Initial value:**

```
=
[
 ["include", "dir_a537b1e5a645cb4471f6768545d20c2b.html", "dir_a537b1e5a645cb4471f6768545d20c2b"]
]
```

Definition at line 1 of file dir\_84497794bb37d283dbd68163bed8ef05.js.

## 30.705 doc/html/dir\_85c45df6158cba585b9d394ec2ca8ba6.js File Reference

### Variables

- var [dir\\_85c45df6158cba585b9d394ec2ca8ba6](#)

### 30.705.1 Variable Documentation

30.705.1.1 var dir\_85c45df6158cba585b9d394ec2ca8ba6

**Initial value:**

```
=
[
 ["qualitymodule", "dir_7c72f061721a5863212d64c180702103.html", "dir_7c72f061721a5863212d64c180702103"
]
]
```

Definition at line 1 of file dir\_85c45df6158cba585b9d394ec2ca8ba6.js.

## 30.706 doc/html/dir\_892cdac2bffabe6fa625049467eefe0a.js File Reference

### Variables

- var [dir\\_892cdac2bffabe6fa625049467eefe0a](#)

### 30.706.1 Variable Documentation

#### 30.706.1.1 var dir\_892cdac2bffabe6fa625049467eefe0a

##### Initial value:

```
=
[
 ["bandmat.cpp", "bandmat_8cpp.html", "bandmat_8cpp"],
 ["cholesky.cpp", "cholesky_8cpp.html", "cholesky_8cpp"],
 ["evaluate.cpp", "evaluate_8cpp.html", "evaluate_8cpp"],
 ["fft.cpp", "fft_8cpp.html", "fft_8cpp"],
 ["hholder.cpp", "hholder_8cpp.html", "hholder_8cpp"],
 ["jacobi.cpp", "jacobi_8cpp.html", "jacobi_8cpp"],
 ["myexcept.cpp", "myexcept_8cpp.html", "myexcept_8cpp"],
 ["newfft.cpp", "newfft_8cpp.html", "newfft_8cpp"],
 ["newmat1.cpp", "newmat1_8cpp.html", "newmat1_8cpp"],
 ["newmat2.cpp", "newmat2_8cpp.html", "newmat2_8cpp"],
 ["newmat3.cpp", "newmat3_8cpp.html", "newmat3_8cpp"],
 ["newmat4.cpp", "newmat4_8cpp.html", "newmat4_8cpp"],
 ["newmat5.cpp", "newmat5_8cpp.html", "newmat5_8cpp"],
 ["newmat6.cpp", "newmat6_8cpp.html", "newmat6_8cpp"],
 ["newmat7.cpp", "newmat7_8cpp.html", "newmat7_8cpp"],
 ["newmat8.cpp", "newmat8_8cpp.html", "newmat8_8cpp"],
 ["newmat9.cpp", "newmat9_8cpp.html", "newmat9_8cpp"],
 ["newmatex.cpp", "newmatex_8cpp.html", "newmatex_8cpp"],
 ["newmatn1.cpp", "newmatn1_8cpp.html", "newmatn1_8cpp"],
 ["newmatrm.cpp", "newmatrm_8cpp.html", "newmatrm_8cpp"],
 ["solution.cpp", "solution_8cpp.html", "solution_8cpp"],
 ["sort.cpp", "sort_8cpp.html", "sort_8cpp"],
 ["submat.cpp", "submat_8cpp.html", "submat_8cpp"],
 ["svd.cpp", "svd_8cpp.html", "svd_8cpp"]
]
```

Definition at line 1 of file dir\_892cdac2bffabe6fa625049467eefe0a.js.

## 30.707 doc/html/dir\_8946de0dbd7af8713b300be94ace6b21.js File Reference

### Variables

- var [dir\\_8946de0dbd7af8713b300be94ace6b21](#)

### 30.707.1 Variable Documentation

#### 30.707.1.1 var dir\_8946de0dbd7af8713b300be94ace6b21

##### Initial value:

```
=
[
 ["html", "dir_5eaf107a44af8927c3e88d7693acb3ec.html", "dir_5eaf107a44af8927c3e88d7693acb3ec"]
]
```

Definition at line 1 of file dir\_8946de0dbd7af8713b300be94ace6b21.js.

## 30.708 doc/html/dir\_8c65b503da15ddfa217ea827bbddedc7.js File Reference

### Variables

- var [dir\\_8c65b503da15ddfa217ea827bbddedc7](#)

### 30.708.1 Variable Documentation

#### 30.708.1.1 var dir\_8c65b503da15ddfa217ea827bbddedc7

##### Initial value:

```
=
[
 ["include", "dir_b3f8b92e7ea1946157ca7a805e4d2267.html", "dir_b3f8b92e7ea1946157ca7a805e4d2267"]
]
```

Definition at line 1 of file dir\_8c65b503da15ddfa217ea827bbddedc7.js.

## 30.709 doc/html/dir\_8dcf46dfbb2e2ca3d570ad71ab7f5984.js File Reference

### Variables

- var [dir\\_8dcf46dfbb2e2ca3d570ad71ab7f5984](#)

### 30.709.1 Variable Documentation

#### 30.709.1.1 var dir\_8dcf46dfbb2e2ca3d570ad71ab7f5984

##### Initial value:

```
=
[
 ["HeTriang.cpp", "_he_triang_8cpp.html", null]
]
```

Definition at line 1 of file dir\_8dcf46dfbb2e2ca3d570ad71ab7f5984.js.

## 30.710 doc/html/dir\_90fbe926ef93e71fdbf5723e062c9da4.js File Reference

### Variables

- var [dir\\_90fbe926ef93e71fdbf5723e062c9da4](#)

### 30.710.1 Variable Documentation

30.710.1.1 `var dir_90fbe926ef93e71fdbf5723e062c9da4`

Definition at line 1 of file `dir_90fbe926ef93e71fdbf5723e062c9da4.js`.

## 30.711 `doc/html/dir_981871de7611ee7a54f2d0f3790a0f82.js` File Reference

### Variables

- `var dir\_981871de7611ee7a54f2d0f3790a0f82`

### 30.711.1 Variable Documentation

30.711.1.1 `var dir_981871de7611ee7a54f2d0f3790a0f82`

#### Initial value:

```
=
[
 ["GoTools", "dir_85c45df6158cba585b9d394ec2ca8ba6.html", "dir_85c45df6158cba585b9d394ec2ca8ba6"]
]
```

Definition at line 1 of file `dir_981871de7611ee7a54f2d0f3790a0f82.js`.

## 30.712 `doc/html/dir_99c721c896ff1c980c5b18b3dcbaff94.js` File Reference

### Variables

- `var dir\_99c721c896ff1c980c5b18b3dcbaff94`

### 30.712.1 Variable Documentation

30.712.1.1 `var dir_99c721c896ff1c980c5b18b3dcbaff94`

#### Initial value:

```
=
[
 ["GoReadWrite.cpp", "_go_read_write_8cpp.html", "_go_read_write_8cpp"]
]
```

Definition at line 1 of file `dir_99c721c896ff1c980c5b18b3dcbaff94.js`.



## 30.713 doc/html/dir\_9df88d95c92a16255dc1e6f9b3433e60.js File Reference

### Variables

- var [dir\\_9df88d95c92a16255dc1e6f9b3433e60](#)

### 30.713.1 Variable Documentation

#### 30.713.1.1 var dir\_9df88d95c92a16255dc1e6f9b3433e60

Definition at line 1 of file dir\_9df88d95c92a16255dc1e6f9b3433e60.js.

## 30.714 doc/html/dir\_9e6bb4020941d0be38bb0d27bbe7cccb.js File Reference

### Variables

- var [dir\\_9e6bb4020941d0be38bb0d27bbe7cccb](#)

### 30.714.1 Variable Documentation

#### 30.714.1.1 var dir\_9e6bb4020941d0be38bb0d27bbe7cccb

#### Initial value:

```
=
[
 ["aux2.cpp", "aux2_8cpp.html", "aux2_8cpp"],
 ["gl_aux.cpp", "gl__aux_8cpp.html", "gl__aux_8cpp"],
 ["glutils.cpp", "glutils_8cpp.html", "glutils_8cpp"],
 ["mouse.cpp", "mouse_8cpp.html", "mouse_8cpp"],
 ["sisl_aux.cpp", "sisl__aux_8cpp.html", "sisl__aux_8cpp"],
 ["transfutils.cpp", "transfutils_8cpp.html", "transfutils_8cpp"]
]
```

Definition at line 1 of file dir\_9e6bb4020941d0be38bb0d27bbe7cccb.js.

## 30.715 doc/html/dir\_9e7f8af3df212ec4d479463341bb72a6.js File Reference

### Variables

- var [dir\\_9e7f8af3df212ec4d479463341bb72a6](#)

### 30.715.1 Variable Documentation

#### 30.715.1.1 var dir\_9e7f8af3df212ec4d479463341bb72a6

##### Initial value:

```
=
[
 ["GoTools", "dir_b07f39590aec0282ad5866778d6f956e.html", "dir_b07f39590aec0282ad5866778d6f956e"]
]
```

Definition at line 1 of file dir\_9e7f8af3df212ec4d479463341bb72a6.js.

## 30.716 doc/html/dir\_a0fa69fab3be50a1b4c8c4ebaca5098c.js File Reference

### Variables

- var [dir\\_a0fa69fab3be50a1b4c8c4ebaca5098c](#)

### 30.716.1 Variable Documentation

#### 30.716.1.1 var dir\_a0fa69fab3be50a1b4c8c4ebaca5098c

##### Initial value:

```
=
[
 ["include", "dir_442358e1b75f159df2b4b10f68cb6669.html", "dir_442358e1b75f159df2b4b10f68cb6669"],
 ["src", "dir_9e6bb4020941d0be38bb0d27bbe7ccb.html", "dir_9e6bb4020941d0be38bb0d27bbe7ccb"],
 ["sisl_view_demo.cpp", "sisl__view__demo_8cpp.html", "sisl__view__demo_8cpp"]
]
```

Definition at line 1 of file dir\_a0fa69fab3be50a1b4c8c4ebaca5098c.js.

## 30.717 doc/html/dir\_a39cd4dcd495fe456e1f661046bd5530.js File Reference

### Variables

- var [dir\\_a39cd4dcd495fe456e1f661046bd5530](#)

### 30.717.1 Variable Documentation

#### 30.717.1.1 var dir\_a39cd4dcd495fe456e1f661046bd5530

Definition at line 1 of file dir\_a39cd4dcd495fe456e1f661046bd5530.js.

## 30.718 doc/html/dir\_a537b1e5a645cb4471f6768545d20c2b.js File Reference

### Variables

- var [dir\\_a537b1e5a645cb4471f6768545d20c2b](#)

#### 30.718.1 Variable Documentation

##### 30.718.1.1 var dir\_a537b1e5a645cb4471f6768545d20c2b

###### Initial value:

```
=
[
 ["GoTools", "dir_0015c891e2eac5fd9e489e846b834700.html", "dir_0015c891e2eac5fd9e489e846b834700"]
]
```

Definition at line 1 of file dir\_a537b1e5a645cb4471f6768545d20c2b.js.

## 30.719 doc/html/dir\_a6ecec081e9e6ebaf06ff4a52c078f1.js File Reference

### Variables

- var [dir\\_a6ecec081e9e6ebaf06ff4a52c078f1](#)

#### 30.719.1 Variable Documentation

##### 30.719.1.1 var dir\_a6ecec081e9e6ebaf06ff4a52c078f1

###### Initial value:

```
=
[
 ["DataHandlerVolAndLR.h", "_data_handler_vol_and_l_r_8h.html", [
 ["DataHandlerVolAndLR", "class_data_handler_vol_and_l_r.html", "class_data_handler_vol_and_l_r"]
]],
 ["gvApplicationVolAndLR.h", "gv_application_vol_and_l_r_8h.html", [
 ["gvApplicationVolAndLR", "classgv_application_vol_and_l_r.html", "classgv_application_vol_and_l_r"]
]],
 ["gvRectangularVolumePaintable.h", "gv_rectangular_volume_paintable_8h.html", [
 ["gvRectangularVolumePaintable", "classgv_rectangular_volume_paintable.html", "classgv_rectangular_volume_paintable"]
]],
 ["RectangularVolumePropertySheet.h", "_rectangular_volume_property_sheet_8h.html", [
 ["RectangularVolumePropertySheet", "class_rectangular_volume_property_sheet.html", "class_rectangular_volume_property_sheet"]
]]
]
```

Definition at line 1 of file dir\_a6ecec081e9e6ebaf06ff4a52c078f1.js.

## 30.720 doc/html/dir\_a702eb1d615fc8e494327e00bfd64e2d.js File Reference

### Variables

- var [dir\\_a702eb1d615fc8e494327e00bfd64e2d](#)

### 30.720.1 Variable Documentation

#### 30.720.1.1 var dir\_a702eb1d615fc8e494327e00bfd64e2d

##### Initial value:

```
=
[
 ["trivariate", "dir_26434c1759c56a49292c328c033e76d7.html", "dir_26434c1759c56a49292c328c033e76d7"]
]
```

Definition at line 1 of file dir\_a702eb1d615fc8e494327e00bfd64e2d.js.

## 30.721 doc/html/dir\_ab08e349df88865f421593b417138732.js File Reference

### Variables

- var [dir\\_ab08e349df88865f421593b417138732](#)

### 30.721.1 Variable Documentation

#### 30.721.1.1 var dir\_ab08e349df88865f421593b417138732

##### Initial value:

```
=
[
 ["parametrization", "dir_3cd14e887b3be1bb0699a0f4b9d7471b.html", "dir_3cd14e887b3be1bb0699a0f4b9d7471b"
]
]
```

Definition at line 1 of file dir\_ab08e349df88865f421593b417138732.js.

## 30.722 doc/html/dir\_b07f39590aec0282ad5866778d6f956e.js File Reference

### Variables

- var [dir\\_b07f39590aec0282ad5866778d6f956e](#)

### 30.722.1 Variable Documentation

#### 30.722.1.1 var dir\_b07f39590aec0282ad5866778d6f956e

**Initial value:**

```
=
[
 ["isogeometric_model", "dir_e98c7b4edc7740dfc859667cd423ef3f.html", "
 dir_e98c7b4edc7740dfc859667cd423ef3f"]
]
```

Definition at line 1 of file dir\_b07f39590aec0282ad5866778d6f956e.js.

## 30.723 doc/html/dir\_b168ed9c8c05331b63ad454a11f635f2.js File Reference

### Variables

- var [dir\\_b168ed9c8c05331b63ad454a11f635f2](#)

### 30.723.1 Variable Documentation

#### 30.723.1.1 var dir\_b168ed9c8c05331b63ad454a11f635f2

**Initial value:**

```
=
[
 ["examples", "dir_cfbe617369797e2c088f7718a7b296e4.html", "dir_cfbe617369797e2c088f7718a7b296e4"],
 ["include", "dir_e3e0722e0dd9c71678af19110aeec8f8.html", "dir_e3e0722e0dd9c71678af19110aeec8f8"],
 ["src", "dir_54c0807dc9dc58401cdf98175b301107.html", "dir_54c0807dc9dc58401cdf98175b301107"]
]
```

Definition at line 1 of file dir\_b168ed9c8c05331b63ad454a11f635f2.js.

## 30.724 doc/html/dir\_b28ca551f00416d098dfb2d5d86cb85e.js File Reference

### Variables

- var [dir\\_b28ca551f00416d098dfb2d5d86cb85e](#)

### 30.724.1 Variable Documentation

#### 30.724.1.1 var dir\_b28ca551f00416d098dfb2d5d86cb85e

**Initial value:**

```
=
[
 ["lrsplines2D", "dir_9df88d95c92a16255dc1e6f9b3433e60.html", "dir_9df88d95c92a16255dc1e6f9b3433e60"]
]
```

Definition at line 1 of file dir\_b28ca551f00416d098dfb2d5d86cb85e.js.

## 30.725 doc/html/dir\_b3f8b92e7ea1946157ca7a805e4d2267.js File Reference

### Variables

- var [dir\\_b3f8b92e7ea1946157ca7a805e4d2267](#)

### 30.725.1 Variable Documentation

#### 30.725.1.1 var dir\_b3f8b92e7ea1946157ca7a805e4d2267

##### Initial value:

```
=
[
 ["GoTools", "dir_2ea40cee675de4bd87b06c9233dde18b.html", "dir_2ea40cee675de4bd87b06c9233dde18b"]
]
```

Definition at line 1 of file dir\_b3f8b92e7ea1946157ca7a805e4d2267.js.

## 30.726 doc/html/dir\_bf6a253d933d2fc523d4a21708f60d86.js File Reference

### Variables

- var [dir\\_bf6a253d933d2fc523d4a21708f60d86](#)

### 30.726.1 Variable Documentation

#### 30.726.1.1 var dir\_bf6a253d933d2fc523d4a21708f60d86

##### Initial value:

```
=
[
 ["compositemodel", "dir_28d725a1074140e3875acd99df863a9a.html", "dir_28d725a1074140e3875acd99df863a9a"
]
]
```

Definition at line 1 of file dir\_bf6a253d933d2fc523d4a21708f60d86.js.

## 30.727 doc/html/dir\_ce2be4821eafb03cec507f192e45eceb.js File Reference

### Variables

- var [dir\\_ce2be4821eafb03cec507f192e45eceb](#)

### 30.727.1 Variable Documentation

30.727.1.1 var dir\_ce2be4821eafb03cec507f192e45eceb

#### Initial value:

```
=
[
 ["implicitization", "dir_f2bf6ebad7a5cab5a643ce43ec016ae9.html", "dir_f2bf6ebad7a5cab5a643ce43ec016ae9"
]]
]
```

Definition at line 1 of file dir\_ce2be4821eafb03cec507f192e45eceb.js.

## 30.728 doc/html/dir\_ceb2d94ff415c9faa44cf694ce53cd4a.js File Reference

### Variables

- var [dir\\_ceb2d94ff415c9faa44cf694ce53cd4a](#)

### 30.728.1 Variable Documentation

30.728.1.1 var dir\_ceb2d94ff415c9faa44cf694ce53cd4a

#### Initial value:

```
=
[
 ["GoTools", "dir_bf6a253d933d2fc523d4a21708f60d86.html", "dir_bf6a253d933d2fc523d4a21708f60d86"]
]
```

Definition at line 1 of file dir\_ceb2d94ff415c9faa44cf694ce53cd4a.js.

## 30.729 doc/html/dir\_cfbe617369797e2c088f7718a7b296e4.js File Reference

### Variables

- var [dir\\_cfbe617369797e2c088f7718a7b296e4](#)

### 30.729.1 Variable Documentation

#### 30.729.1.1 var dir\_cfbe617369797e2c088f7718a7b296e4

##### Initial value:

```
=
[
 ["streaming", "dir_dcc2523c85224d60248ee4ee7462cfa8.html", "dir_dcc2523c85224d60248ee4ee7462cfa8"],
 ["viewer", "dir_a0fa69fab3be50a1b4c8c4ebaca5098c.html", "dir_a0fa69fab3be50a1b4c8c4ebaca5098c"],
 ["example01.cpp", "example01_8cpp.html", "example01_8cpp"],
 ["example02.cpp", "example02_8cpp.html", "example02_8cpp"],
 ["example03.cpp", "example03_8cpp.html", "example03_8cpp"],
 ["example04.cpp", "example04_8cpp.html", "example04_8cpp"],
 ["example05.cpp", "example05_8cpp.html", "example05_8cpp"],
 ["example06.cpp", "example06_8cpp.html", "example06_8cpp"],
 ["example07.cpp", "example07_8cpp.html", "example07_8cpp"],
 ["example08.cpp", "example08_8cpp.html", "example08_8cpp"],
 ["example09.cpp", "example09_8cpp.html", "example09_8cpp"],
 ["example10.cpp", "example10_8cpp.html", "example10_8cpp"],
 ["example11.cpp", "example11_8cpp.html", "example11_8cpp"],
 ["example12.cpp", "example12_8cpp.html", "example12_8cpp"],
 ["example13.cpp", "example13_8cpp.html", "example13_8cpp"],
 ["example14.cpp", "example14_8cpp.html", "example14_8cpp"],
 ["example15.cpp", "example15_8cpp.html", "example15_8cpp"]
]
```

Definition at line 1 of file dir\_cfbe617369797e2c088f7718a7b296e4.js.

### 30.730 doc/html/dir\_d3b744872420378775da279872820ed2.js File Reference

#### Variables

- var [dir\\_d3b744872420378775da279872820ed2](#)

#### 30.730.1 Variable Documentation

##### 30.730.1.1 var dir\_d3b744872420378775da279872820ed2

##### Initial value:

```
=
[
 ["include", "dir_981871de7611ee7a54f2d0f3790a0f82.html", "dir_981871de7611ee7a54f2d0f3790a0f82"]
]
```

Definition at line 1 of file dir\_d3b744872420378775da279872820ed2.js.

### 30.731 doc/html/dir\_d3d5f8ee42e1e2dc7017f3f7157ca5bc.js File Reference

#### Variables

- var [dir\\_d3d5f8ee42e1e2dc7017f3f7157ca5bc](#)



### 30.731.1 Variable Documentation

30.731.1.1 `var dir_d3d5f8ee42e1e2dc7017f3f7157ca5bc`

Definition at line 1 of file `dir_d3d5f8ee42e1e2dc7017f3f7157ca5bc.js`.

## 30.732 doc/html/dir\_dcc2523c85224d60248ee4ee7462cfa8.js File Reference

### Variables

- `var dir_dcc2523c85224d60248ee4ee7462cfa8`

### 30.732.1 Variable Documentation

30.732.1.1 `var dir_dcc2523c85224d60248ee4ee7462cfa8`

#### Initial value:

```
=
[
 ["include", "dir_1c9e4414cb2b8eb9526a3aa2069941f8.html", "dir_1c9e4414cb2b8eb9526a3aa2069941f8"],
 ["src", "dir_99c721c896ff1c980c5b18b3dcbaff94.html", "dir_99c721c896ff1c980c5b18b3dcbaff94"]
]
```

Definition at line 1 of file `dir_dcc2523c85224d60248ee4ee7462cfa8.js`.

## 30.733 doc/html/dir\_e2ef5fb48cb01c9435f4dad81bf7eab.js File Reference

### Variables

- `var dir_e2ef5fb48cb01c9435f4dad81bf7eab`

### 30.733.1 Variable Documentation

30.733.1.1 `var dir_e2ef5fb48cb01c9435f4dad81bf7eab`

#### Initial value:

```
=
[
 ["include", "dir_6a9f218ac24eb7b754dc250ffd508dae.html", "dir_6a9f218ac24eb7b754dc250ffd508dae"]
]
```

Definition at line 1 of file `dir_e2ef5fb48cb01c9435f4dad81bf7eab.js`.

## 30.734 doc/html/dir\_e3e0722e0dd9c71678af19110aeec8f8.js File Reference

### Variables

- var [dir\\_e3e0722e0dd9c71678af19110aeec8f8](#)

#### 30.734.1 Variable Documentation

##### 30.734.1.1 var dir\_e3e0722e0dd9c71678af19110aeec8f8

#### Initial value:

```
=
[
 ["sisl-copyright.h", "sisl-copyright_8h.html", null],
 ["sisl.h", "sisl_8h.html", "sisl_8h"],
 ["sislP.h", "sisl_p_8h.html", "sisl_p_8h"]
]
```

Definition at line 1 of file dir\_e3e0722e0dd9c71678af19110aeec8f8.js.

## 30.735 doc/html/dir\_e66b996f8b4be0681f8e9eb9869a3079.js File Reference

### Variables

- var [dir\\_e66b996f8b4be0681f8e9eb9869a3079](#)

#### 30.735.1 Variable Documentation

##### 30.735.1.1 var dir\_e66b996f8b4be0681f8e9eb9869a3079

#### Initial value:

```
=
[
 ["hesimplest", "dir_6917c6a4538152e6852497b387e02df2.html", "dir_6917c6a4538152e6852497b387e02df2"]
]
```

Definition at line 1 of file dir\_e66b996f8b4be0681f8e9eb9869a3079.js.

## 30.736 doc/html/dir\_e846bdbb82b84d132cb98df96cb07bf0.js File Reference

### Variables

- var [dir\\_e846bdbb82b84d132cb98df96cb07bf0](#)

### 30.736.1 Variable Documentation

30.736.1.1 `var dir_e846bdbb82b84d132cb98df96cb07bf0`

#### Initial value:

```
=
[
 ["HandleId.cpp", "_handle_id_8cpp.html", null]
]
```

Definition at line 1 of file `dir_e846bdbb82b84d132cb98df96cb07bf0.js`.

## 30.737 doc/html/dir\_e98c7b4edc7740dfc859667cd423ef3f.js File Reference

### Variables

- `var dir_e98c7b4edc7740dfc859667cd423ef3f`

### 30.737.1 Variable Documentation

30.737.1.1 `var dir_e98c7b4edc7740dfc859667cd423ef3f`

Definition at line 1 of file `dir_e98c7b4edc7740dfc859667cd423ef3f.js`.

## 30.738 doc/html/dir\_eb531be94dc3f2da4c4acbfa84e0e7dc.js File Reference

### Variables

- `var dir_eb531be94dc3f2da4c4acbfa84e0e7dc`

### 30.738.1 Variable Documentation

30.738.1.1 `var dir_eb531be94dc3f2da4c4acbfa84e0e7dc`

#### Initial value:

```
=
[
 ["GoTools", "dir_4ed957b398f8a342f35e202c6dbeff82.html", "dir_4ed957b398f8a342f35e202c6dbeff82"]
]
```

Definition at line 1 of file `dir_eb531be94dc3f2da4c4acbfa84e0e7dc.js`.

## 30.739 doc/html/dir\_f2117bb2bb20a60155ae701cb86e7351.js File Reference

### Variables

- var [dir\\_f2117bb2bb20a60155ae701cb86e7351](#)

### 30.739.1 Variable Documentation

#### 30.739.1.1 var dir\_f2117bb2bb20a60155ae701cb86e7351

##### Initial value:

```
=
[
 ["examples", "dir_e66b996f8b4be0681f8e9eb9869a3079.html", "dir_e66b996f8b4be0681f8e9eb9869a3079"],
 ["include", "dir_764e20a7fa93cafa3199ac45b6e6b986.html", "dir_764e20a7fa93cafa3199ac45b6e6b986"],
 ["src", "dir_6958061072da6881959e38b0a737a4c6.html", "dir_6958061072da6881959e38b0a737a4c6"]
]
```

Definition at line 1 of file dir\_f2117bb2bb20a60155ae701cb86e7351.js.

## 30.740 doc/html/dir\_f2bf6ebad7a5cab5a643ce43ec016ae9.js File Reference

### Variables

- var [dir\\_f2bf6ebad7a5cab5a643ce43ec016ae9](#)

### 30.740.1 Variable Documentation

#### 30.740.1.1 var dir\_f2bf6ebad7a5cab5a643ce43ec016ae9

##### Initial value:

```
=
[
 ["BernsteinMulti.h", "_bernstein_multi_8h.html", "_bernstein_multi_8h"],
 ["BernsteinPoly.h", "_bernstein_poly_8h.html", "_bernstein_poly_8h"],
 ["BernsteinTetrahedralPoly.h", "_bernstein_tetrahedral_poly_8h.html", "_bernstein_tetrahedral_poly_8h"],
 ["BernsteinTriangularPoly.h", "_bernstein_triangular_poly_8h.html", "_bernstein_triangular_poly_8h"],
 ["BernsteinUtils.h", "_bernstein_utils_8h.html", "_bernstein_utils_8h"],
 ["BezierTriangle.h", "_bezier_triangle_8h.html", "_bezier_triangle_8h"],
 ["Binomial.h", "_binomial_8h.html", [
 ["Binomial", "class_go_1_1_binomial.html", "class_go_1_1_binomial"]
]],
 ["GpuMatrix.hpp", "_gpu_matrix_8hpp.html", [
 ["GpuMatrix", "class_go_1_1_gpu_matrix.html", "class_go_1_1_gpu_matrix"]
]],
 ["implicitization-doxymain.h", "implicitization-doxymain_8h.html", null],
 ["ImplicitizeCurveAlgo.h", "_implicitize_curve_algo_8h.html", [
 ["ImplicitizeCurveAlgo", "class_go_1_1_implicitize_curve_algo.html", "class_go_1_1_implicitize_curve_algo"]
]],
 ["ImplicitizeCurveAndVectorAlgo.h", "_implicitize_curve_and_vector_algo_8h.html", [
 ["ImplicitizeCurveAndVectorAlgo", "class_go_1_1_implicitize_curve_and_vector_algo.html", "class_go_1_1_implicitize_curve_and_vector_algo"]
]],
 ["ImplicitizePointCloudAlgo.h", "_implicitize_point_cloud_algo_8h.html", [
 ["ImplicitizePointCloudAlgo", "class_go_1_1_implicitize_point_cloud_algo.html", "class_go_1_1_implicitize_point_cloud_algo"]
]],
 ["ImplicitizeSurfaceAlgo.h", "_implicitize_surface_algo_8h.html", [
 ["ImplicitizeSurfaceAlgo", "class_go_1_1_implicitize_surface_algo.html", "class_go_1_1_implicitize_surface_algo"]
]],
 ["ImplicitUtils.h", "_implicit_utils_8h.html", "_implicit_utils_8h"]
]
```

Definition at line 1 of file dir\_f2bf6ebad7a5cab5a643ce43ec016ae9.js.

## 30.741 doc/html/dir\_f30d67bf21cfa80c94c2f56f2c05062d.js File Reference

### Variables

- var [dir\\_f30d67bf21cfa80c94c2f56f2c05062d](#)

#### 30.741.1 Variable Documentation

##### 30.741.1.1 var dir\_f30d67bf21cfa80c94c2f56f2c05062d

#### Initial value:

```
=
[
 ["include", "dir_ceb2d94ff415c9faa44cf694ce53cd4a.html", "dir_ceb2d94ff415c9faa44cf694ce53cd4a"]
]
```

Definition at line 1 of file dir\_f30d67bf21cfa80c94c2f56f2c05062d.js.

## 30.742 doc/html/dir\_ffa09c8575bbfbed340eb641d2e0178f.js File Reference

### Variables

- var [dir\\_ffa09c8575bbfbed340eb641d2e0178f](#)

#### 30.742.1 Variable Documentation

##### 30.742.1.1 var dir\_ffa09c8575bbfbed340eb641d2e0178f

#### Initial value:

```
=
[
 ["creators", "dir_d3d5f8ee42e1e2dc7017f3f7157ca5bc.html", "dir_d3d5f8ee42e1e2dc7017f3f7157ca5bc"],
 ["geometry", "dir_2e9f58fed2f8cb33783712134eca6f54.html", "dir_2e9f58fed2f8cb33783712134eca6f54"],
 ["tesselator", "dir_a39cd4dcd495fe456e1f661046bd5530.html", "dir_a39cd4dcd495fe456e1f661046bd5530"],
 ["utils", "dir_69009d91722e8158a6e1821a3d198fd0.html", "dir_69009d91722e8158a6e1821a3d198fd0"]
]
```

Definition at line 1 of file dir\_ffa09c8575bbfbed340eb641d2e0178f.js.

## 30.743 doc/html/dynsections.js File Reference

### Functions

- function [toggleVisibility](#) (linkObj)
- function [updateStripes](#) ()
- function [toggleLevel](#) (level)
- function [toggleFolder](#) (id)
- function [toggleInherit](#) (id)

### 30.743.1 Function Documentation

#### 30.743.1.1 function toggleFolder ( *id* )

Definition at line 49 of file dynsections.js.

#### 30.743.1.2 function toggleInherit ( *id* )

Definition at line 84 of file dynsections.js.

#### 30.743.1.3 function toggleLevel ( *level* )

Definition at line 28 of file dynsections.js.

#### 30.743.1.4 function toggleVisibility ( *linkObj* )

Definition at line 1 of file dynsections.js.

#### 30.743.1.5 function updateStripes ( )

Definition at line 22 of file dynsections.js.

## 30.744 doc/html/errormacros\_8h.js File Reference

### Variables

- var [errormacros\\_8h](#)

### 30.744.1 Variable Documentation

#### 30.744.1.1 var errormacros\_8h

#### Initial value:

```
=
[
 ["ALWAYS_ERROR_IF", "errormacros_8h.html#a86611c37e530eb06c9b077fed130b2dc", null],
 ["ASSERT", "errormacros_8h.html#a0966b817b229d48e5ffc7feab19a0be6", null],
 ["ASSERT2", "errormacros_8h.html#ad81384ed0233964bf967b981016d2081", null],
 ["DEBUG_ERROR_IF", "errormacros_8h.html#ad369e2a5e24adc64ecb0e600750372f5", null],
 ["GO_NO_CHECKS", "errormacros_8h.html#a827eef1c7aa36edd20c48299b46e9fce", null],
 ["MESSAGE", "errormacros_8h.html#a73d1a3db252b0e93921262ab0cb92210", null],
 ["MESSAGE_IF", "errormacros_8h.html#ad17bdc9b890075ed493b6707fd3d4916", null],
 ["REPORT", "errormacros_8h.html#a787099914a94ed31fa544b985b55752f", null],
 ["THROW", "errormacros_8h.html#a27264efd631c4f584ddcb2f5888ae6ed", null]
]
```

Definition at line 1 of file errormacros\_8h.js.

## 30.745 doc/html/ev\_\_cv\_\_off\_8c.js File Reference

### Variables

- var [ev\\_\\_cv\\_\\_off\\_8c](#)

#### 30.745.1 Variable Documentation

##### 30.745.1.1 var ev\_\_cv\_\_off\_8c

###### Initial value:

```
=
[
 ["EV_CV_OFF", "ev__cv__off_8c.html#af570ffd133e823ba65e4594bbba6e69a", null],
 ["ev__cv__off", "ev__cv__off_8c.html#a917a316dd26a3899e617cf7974927fd6", null]
]
```

Definition at line 1 of file ev\_\_cv\_\_off\_8c.js.

## 30.746 doc/html/eval\_\_2\_\_crv\_8c.js File Reference

### Variables

- var [eval\\_\\_2\\_\\_crv\\_8c](#)

#### 30.746.1 Variable Documentation

##### 30.746.1.1 var eval\_\_2\_\_crv\_8c

###### Initial value:

```
=
[
 ["EVAL_2_CRV", "eval__2__crv_8c.html#ae21399dc6a70f8168d0e3926598430cc", null],
 ["eval__2__crv", "eval__2__crv_8c.html#ace3f80122e830126d213152daaf589e2", null]
]
```

Definition at line 1 of file eval\_\_2\_\_crv\_8c.js.

## 30.747 doc/html/evalcrvarc\_8c.js File Reference

### Variables

- var [evalcrvarc\\_8c](#)

### 30.747.1 Variable Documentation

#### 30.747.1.1 var evalcrvarc\_8c

**Initial value:**

```
=
[
 ["EVAL_CRV_ARC", "evalcrvarc_8c.html#a8c6fa6c95e08223f26bb7f9a6ba8eccc", null],
 ["eval_crv_arc", "evalcrvarc_8c.html#a8aeb4956698363bcde3e6f4d94164bd8", null]
]
```

Definition at line 1 of file evalcrvarc\_8c.js.

## 30.748 doc/html/evalvalue\_8cpp.js File Reference

### Variables

- var [evalvalue\\_8cpp](#)

### 30.748.1 Variable Documentation

#### 30.748.1.1 var evalvalue\_8cpp

**Initial value:**

```
=
[
 ["REPORT", "evalvalue_8cpp.html#a787099914a94ed31fa544b985b55752f", null],
 ["WANT_MATH", "evalvalue_8cpp.html#a8335b327d416f961a5dbaa0e058cc515", null],
 ["EigenValues", "evalvalue_8cpp.html#a95ab9a4434fa780d21f7710c569fe0a2", null],
 ["EigenValues", "evalvalue_8cpp.html#a29e2963671b3510e0b5018d79f06e049", null],
 ["EigenValues", "evalvalue_8cpp.html#abe74f036daba16f89e0e75bf18542f6a", null]
]
```

Definition at line 1 of file evalvalue\_8cpp.js.

## 30.749 doc/html/example01\_8cpp.js File Reference

### Variables

- var [example01\\_8cpp](#)



### 30.749.1 Variable Documentation

#### 30.749.1.1 var example01\_8cpp

**Initial value:**

```
=
[
 ["main", "example01_8cpp.html#a8a6028caf5642454245a766ab66990f7", null]
]
```

Definition at line 1 of file example01\_8cpp.js.

## 30.750 doc/html/example02\_8cpp.js File Reference

### Variables

- var [example02\\_8cpp](#)

### 30.750.1 Variable Documentation

#### 30.750.1.1 var example02\_8cpp

**Initial value:**

```
=
[
 ["main", "example02_8cpp.html#a8a6028caf5642454245a766ab66990f7", null]
]
```

Definition at line 1 of file example02\_8cpp.js.

## 30.751 doc/html/example03\_8cpp.js File Reference

### Variables

- var [example03\\_8cpp](#)

### 30.751.1 Variable Documentation

#### 30.751.1.1 var example03\_8cpp

**Initial value:**

```
=
[
 ["main", "example03_8cpp.html#a8a6028caf5642454245a766ab66990f7", null]
]
```

Definition at line 1 of file example03\_8cpp.js.

## 30.752 doc/html/example04\_8cpp.js File Reference

### Variables

- var [example04\\_8cpp](#)

### 30.752.1 Variable Documentation

#### 30.752.1.1 var example04\_8cpp

##### Initial value:

```
=
[
 ["main", "example04_8cpp.html#a8a6028caf5642454245a766ab66990f7", null]
]
```

Definition at line 1 of file example04\_8cpp.js.

## 30.753 doc/html/example05\_8cpp.js File Reference

### Variables

- var [example05\\_8cpp](#)

### 30.753.1 Variable Documentation

#### 30.753.1.1 var example05\_8cpp

##### Initial value:

```
=
[
 ["main", "example05_8cpp.html#a8a6028caf5642454245a766ab66990f7", null]
]
```

Definition at line 1 of file example05\_8cpp.js.

## 30.754 doc/html/example06\_8cpp.js File Reference

### Variables

- var [example06\\_8cpp](#)

### 30.754.1 Variable Documentation

#### 30.754.1.1 var example06\_8cpp

**Initial value:**

```
=
[
 ["main", "example06_8cpp.html#a8a6028caf5642454245a766ab66990f7", null]
]
```

Definition at line 1 of file example06\_8cpp.js.

## 30.755 doc/html/example07\_8cpp.js File Reference

### Variables

- var [example07\\_8cpp](#)

### 30.755.1 Variable Documentation

#### 30.755.1.1 var example07\_8cpp

**Initial value:**

```
=
[
 ["main", "example07_8cpp.html#a8a6028caf5642454245a766ab66990f7", null]
]
```

Definition at line 1 of file example07\_8cpp.js.

## 30.756 doc/html/example08\_8cpp.js File Reference

### Variables

- var [example08\\_8cpp](#)

### 30.756.1 Variable Documentation

#### 30.756.1.1 var example08\_8cpp

**Initial value:**

```
=
[
 ["main", "example08_8cpp.html#a8a6028caf5642454245a766ab66990f7", null]
]
```

Definition at line 1 of file example08\_8cpp.js.

## 30.757 doc/html/example09\_8cpp.js File Reference

### Variables

- var [example09\\_8cpp](#)

### 30.757.1 Variable Documentation

#### 30.757.1.1 var example09\_8cpp

##### Initial value:

```
=
[
 ["main", "example09_8cpp.html#a8a6028caf5642454245a766ab66990f7", null]
]
```

Definition at line 1 of file example09\_8cpp.js.

## 30.758 doc/html/example10\_8cpp.js File Reference

### Variables

- var [example10\\_8cpp](#)

### 30.758.1 Variable Documentation

#### 30.758.1.1 var example10\_8cpp

##### Initial value:

```
=
[
 ["main", "example10_8cpp.html#a8a6028caf5642454245a766ab66990f7", null]
]
```

Definition at line 1 of file example10\_8cpp.js.

## 30.759 doc/html/example11\_8cpp.js File Reference

### Variables

- var [example11\\_8cpp](#)

### 30.759.1 Variable Documentation

#### 30.759.1.1 var example11\_8cpp

**Initial value:**

```
=
[
 ["main", "example11_8cpp.html#a8a6028caf5642454245a766ab66990f7", null]
]
```

Definition at line 1 of file example11\_8cpp.js.

## 30.760 doc/html/example12\_8cpp.js File Reference

### Variables

- var [example12\\_8cpp](#)

### 30.760.1 Variable Documentation

#### 30.760.1.1 var example12\_8cpp

**Initial value:**

```
=
[
 ["main", "example12_8cpp.html#a8a6028caf5642454245a766ab66990f7", null]
]
```

Definition at line 1 of file example12\_8cpp.js.

## 30.761 doc/html/example13\_8cpp.js File Reference

### Variables

- var [example13\\_8cpp](#)

### 30.761.1 Variable Documentation

#### 30.761.1.1 var example13\_8cpp

**Initial value:**

```
=
[
 ["main", "example13_8cpp.html#a8a6028caf5642454245a766ab66990f7", null]
]
```

Definition at line 1 of file example13\_8cpp.js.

## 30.762 doc/html/example14\_8cpp.js File Reference

### Variables

- var [example14\\_8cpp](#)

### 30.762.1 Variable Documentation

#### 30.762.1.1 var example14\_8cpp

##### Initial value:

```
=
[
 ["main", "example14_8cpp.html#a8a6028caf5642454245a766ab66990f7", null]
]
```

Definition at line 1 of file example14\_8cpp.js.

## 30.763 doc/html/example15\_8cpp.js File Reference

### Variables

- var [example15\\_8cpp](#)

### 30.763.1 Variable Documentation

#### 30.763.1.1 var example15\_8cpp

##### Initial value:

```
=
[
 ["main", "example15_8cpp.html#adb598a5dc855499911128c370ce738bf", null]
]
```

Definition at line 1 of file example15\_8cpp.js.

## 30.764 doc/html/example\_8cpp.js File Reference

### Variables

- var [example\\_8cpp](#)

### 30.764.1 Variable Documentation

#### 30.764.1.1 var example\_8cpp

**Initial value:**

```
=
[
 ["WANT_MATH", "example_8cpp.html#a8335b327d416f961a5dbaa0e058cc515", null],
 ["WANT_STREAM", "example_8cpp.html#a6ed6c2d6e68d8f0e7900326b4e30850c", null],
 ["main", "example_8cpp.html#ae66f6b31b5ad750f1fe042a706a4e3d4", null],
 ["test1", "example_8cpp.html#ac0697ad85fb379446222fcc2df02c778", null],
 ["test2", "example_8cpp.html#ac7865afb3b9f6d710b1df18e501c94d6", null],
 ["test3", "example_8cpp.html#a727a32bdd7e2491824ca4acd62be9b1a", null],
 ["test4", "example_8cpp.html#acd897830273f9673b3894e831ceb3423", null],
 ["test5", "example_8cpp.html#aad4ba83ae697b892cf4711aa7170c1f3", null]
]
```

Definition at line 1 of file example\_8cpp.js.

## 30.765 doc/html/examples.js File Reference

### Variables

- var [examples](#)

### 30.765.1 Variable Documentation

#### 30.765.1.1 var examples

Definition at line 1 of file examples.js.

## 30.766 doc/html/extremal\_pt\_surf\_surf\_8h.js File Reference

### Variables

- var [extremal\\_pt\\_surf\\_surf\\_8h](#)

### 30.766.1 Variable Documentation

#### 30.766.1.1 var extremal\_pt\_surf\_surf\_8h

**Initial value:**

```
=
[
 ["extremalPtSurfSurf", "extremal_pt_surf_surf_8h.html#af3dc8bb230bbfc10dbb46175d10f1923", null]
]
```

Definition at line 1 of file extremal\_pt\_surf\_surf\_8h.js.

## 30.767 doc/html/fft\_8cpp.js File Reference

### Variables

- var [fft\\_8cpp](#)

### 30.767.1 Variable Documentation

#### 30.767.1.1 var fft\_8cpp

##### Initial value:

```
=
[
 ["REPORT", "fft_8cpp.html#a787099914a94ed31fa544b985b55752f", null],
 ["WANT_MATH", "fft_8cpp.html#a8335b327d416f961a5dbaa0e058cc515", null],
 ["DCT", "fft_8cpp.html#a924dff49d847f7fc827760a9adad7f75", null],
 ["DCT_II", "fft_8cpp.html#a51ac1b5cbac7a28127244a633cf77a76", null],
 ["DCT_II_inverse", "fft_8cpp.html#a9ebdbefe2d3e834b8961236e19393ecf", null],
 ["DCT_inverse", "fft_8cpp.html#aea0007d2d616102dc268a5e232e561a2", null],
 ["DST", "fft_8cpp.html#aef32720ef2218fd3bb3490a495a6bcf1", null],
 ["DST_II", "fft_8cpp.html#aa95d1a1c29ea05fa5e92337428741e33", null],
 ["DST_II_inverse", "fft_8cpp.html#a981f7302c951d712b9f43alec329ec88", null],
 ["DST_inverse", "fft_8cpp.html#a41f17823e96fff24b8d786ab9bfc010", null],
 ["FFT", "fft_8cpp.html#a52ecc4818418779fbd2d2e134625ec7d", null],
 ["FFTI", "fft_8cpp.html#a5ea0edd00363abc20801d57f602fd2ec", null],
 ["RealFFT", "fft_8cpp.html#a361917bfded54fc546c81866013feaf", null],
 ["RealFFTI", "fft_8cpp.html#a8dbcb6ee1b476fd11097415fa7cdab8b", null]
]
```

Definition at line 1 of file `fft_8cpp.js`.

## 30.768 doc/html/files.js File Reference

### Variables

- var [files](#)

### 30.768.1 Variable Documentation

#### 30.768.1.1 var files

##### Initial value:

```
=
[
 ["compositemodel", "dir_f30d67bf21cfa80c94c2f56f2c05062d.html", "dir_f30d67bf21cfa80c94c2f56f2c05062d"],
 ["gotools-core", "dir_1ce88f9178ec82ce3d3b5366984a4a76.html", "dir_1ce88f9178ec82ce3d3b5366984a4a76"],
 ["igeslib", "dir_61c47cbd77726dc04327162bb82f2acf.html", "dir_61c47cbd77726dc04327162bb82f2acf"],
 ["implicitization", "dir_e2ef5fb48cb01c9435f4dadc81bf7eab.html", "dir_e2ef5fb48cb01c9435f4dadc81bf7eab"],
 ["intersections", "dir_8c65b503da15ddfa217ea827bbddedc7.html", "dir_8c65b503da15ddfa217ea827bbddedc7"],
 ["isogeometric_model", "dir_662c58e926d5c8a2e92c3a0efd184e27.html", "dir_662c58e926d5c8a2e92c3a0efd184e27"],
 ["lrsplines2D", "dir_3b29025e4f1a16d99c0ac7281d29a749.html", "dir_3b29025e4f1a16d99c0ac7281d29a749"],
```



```
["newmat", "dir_79b3003bf1bdb319185970350806bf23.html", "dir_79b3003bf1bdb319185970350806bf23"],
["parametrization", "dir_7b91ef4997e28bdb148c447548446353.html", "dir_7b91ef4997e28bdb148c447548446353"
],
["qualitymodule", "dir_d3b744872420378775da279872820ed2.html", "dir_d3b744872420378775da279872820ed2"
],
["sisl", "dir_b168ed9c8c05331b63ad454a11f635f2.html", "dir_b168ed9c8c05331b63ad454a11f635f2"],
["topology", "dir_84497794bb37d283dbd68163bed8ef05.html", "dir_84497794bb37d283dbd68163bed8ef05"],
["trivariate", "dir_44eb60e2f3e103c46810fc3a6f6f0b38.html", "dir_44eb60e2f3e103c46810fc3a6f6f0b38"],
["trivariatemodel", "dir_37bdf26acb6f0e99095c72f4b261057d.html", "dir_37bdf26acb6f0e99095c72f4b261057d"
],
["ttl", "dir_f2117bb2bb20a60155ae701cb86e7351.html", "dir_f2117bb2bb20a60155ae701cb86e7351"],
["viewlib", "dir_0df052b1ef5b53594ca9be5a6074277d.html", "dir_0df052b1ef5b53594ca9be5a6074277d"]
]
```

Definition at line 1 of file files.js.

## 30.769 doc/html/ft\_curve\_8h.js File Reference

### Variables

- [var ft\\_curve\\_8h](#)

### 30.769.1 Variable Documentation

#### 30.769.1.1 var ft\_curve\_8h

#### Initial value:

```
=
[
["ftCurveSegment", "class_go_1_lft_curve_segment.html", "class_go_1_lft_curve_segment"],
["ftCurve", "class_go_1_lft_curve.html", "class_go_1_lft_curve"],
["ftCurveType", "ft_curve_8h.html#ad80c2d41480925fa1892bb443097b4ae", [
["CURVE_NOTYPE", "
ft_curve_8h.html#ad80c2d41480925fa1892bb443097b4aea4850f0595a3d416eb47b55ba12e986e4", null],
["CURVE_INTERSECTION", "
ft_curve_8h.html#ad80c2d41480925fa1892bb443097b4aea94ea0bee5fcf96f724ff261c3cdc4ccb", null],
["CURVE_EDGE", "ft_curve_8h.html#ad80c2d41480925fa1892bb443097b4aea28b19a1936ae927688022ac26774ddf"
, null],
["CURVE_KINK", "ft_curve_8h.html#ad80c2d41480925fa1892bb443097b4aea4773147fa5a7f77b4e4a250df95c06ae"
, null],
["CURVE_CORNER", "
ft_curve_8h.html#ad80c2d41480925fa1892bb443097b4aea011c2d5cf25aad8b7744dcff45ac6b91", null],
["CURVE_GAP", "ft_curve_8h.html#ad80c2d41480925fa1892bb443097b4aeaa5907ce2e3b120df73671542a0fe02e4",
null],
["CURVE_SINGULAR", "
ft_curve_8h.html#ad80c2d41480925fa1892bb443097b4aea4af8e9b72a6a995ed4727f54d8527482", null]
]]
]
```

Definition at line 1 of file ft\_curve\_8h.js.

## 30.770 doc/html/ft\_message\_8h.js File Reference

### Variables

- [var ft\\_message\\_8h](#)

### 30.770.1 Variable Documentation

#### 30.770.1.1 var ft\_message\_8h

Definition at line 1 of file ft\_message\_8h.js.

## 30.771 doc/html/ft\_planar\_graph\_8h.js File Reference

### Variables

- var [ft\\_planar\\_graph\\_8h](#)

### 30.771.1 Variable Documentation

#### 30.771.1.1 var ft\_planar\_graph\_8h

#### Initial value:

```
=
[
 ["ftGraphEdge", "class_go_1_lft_graph_edge.html", "class_go_1_lft_graph_edge"],
 ["ftSearchNode", "class_go_1_lft_search_node.html", "class_go_1_lft_search_node"],
 ["ftPlanarGraph", "class_go_1_lft_planar_graph.html", "class_go_1_lft_planar_graph"],
 ["GraphIter", "ft_planar_graph_8h.html#a428f052620c22896093395bb8d6ddcae", null],
 ["areaTriangle", "ft_planar_graph_8h.html#aff2c4a14aaf8dd923abf7d70102acba", null]
]
```

Definition at line 1 of file ft\_planar\_graph\_8h.js.

## 30.772 doc/html/ft\_point\_set\_8h.js File Reference

### Variables

- var [ft\\_point\\_set\\_8h](#)

### 30.772.1 Variable Documentation

#### 30.772.1.1 var ft\_point\_set\_8h

#### Initial value:

```
=
[
 ["ftSamplePoint", "class_go_1_lft_sample_point.html", "class_go_1_lft_sample_point"],
 ["ftPointSet", "class_go_1_lft_point_set.html", "class_go_1_lft_point_set"],
 ["PointIter", "ft_point_set_8h.html#a7f84dc87fe8867d883bf99185910011b", null],
 ["PointList", "ft_point_set_8h.html#aa8698132dc8666f88d17eb50e8458d17", null]
]
```

Definition at line 1 of file ft\_point\_set\_8h.js.

## 30.773 doc/html/ft\_surface\_set\_8h.js File Reference

### Variables

- var [ft\\_surface\\_set\\_8h](#)

### 30.773.1 Variable Documentation

#### 30.773.1.1 var ft\_surface\_set\_8h

##### Initial value:

```
=
[
 ["ftSurfaceSet", "class_go_1_1ft_surface_set.html", "class_go_1_1ft_surface_set"],
 ["BoundaryPiece", "ft_surface_set_8h.html#a5ffb8f679caa5a03117315a175a828b5", null]
]
```

Definition at line 1 of file ft\_surface\_set\_8h.js.

## 30.774 doc/html/ft\_tang\_priority\_8h.js File Reference

### Variables

- var [ft\\_tang\\_priority\\_8h](#)

### 30.774.1 Variable Documentation

#### 30.774.1.1 var ft\_tang\_priority\_8h

##### Initial value:

```
=
[
 ["ftTangPriority", "ft_tang_priority_8h.html#ab182c80bd5555f4f7adee434150ede71", [
 ["ftNoType", "
ft_tang_priority_8h.html#ab182c80bd5555f4f7adee434150ede71a2b7e041553fb5fe69125d28c24f4a9b0", null],
 ["ftMaster", "
ft_tang_priority_8h.html#ab182c80bd5555f4f7adee434150ede71ad68002a44b2c93dc9d1381ff22e13b46", null],
 ["ftSlave", "
ft_tang_priority_8h.html#ab182c80bd5555f4f7adee434150ede71a0bd29cf0d5cfaff4ab70c8afb8bc85f08", null]
]]
]
```

Definition at line 1 of file ft\_tang\_priority\_8h.js.

## 30.775 doc/html/ft\_volume\_tools\_8h.js File Reference

### Variables

- var [ft\\_volume\\_tools\\_8h](#)

### 30.775.1 Variable Documentation

#### 30.775.1.1 var ft\_volume\_tools\_8h

##### Initial value:

```
=
[
 ["boundaryStatus", "ft_volume_tools_8h.html#ac4c77cfd11a0139070fa0c274ac2479", null],
 ["splitVolumes", "ft_volume_tools_8h.html#a1150e611f3e4cbbf1db852a6b8269cb9", null],
 ["splitVolumes", "ft_volume_tools_8h.html#a54a673f3473edb627a41ce607b668580", null],
 ["updateWithSplitFaces", "ft_volume_tools_8h.html#a193f33a893de6a1b26896a18d890579a", null]
]
```

Definition at line 1 of file ft\_volume\_tools\_8h.js.

## 30.776 doc/html/functions\_dup.js File Reference

### Variables

- var [functions\\_dup](#)

### 30.776.1 Variable Documentation

#### 30.776.1.1 var functions\_dup

##### Initial value:

```
=
[
 ["a", "functions.html", null],
 ["b", "functions_b.html", null],
 ["c", "functions_c.html", null],
 ["d", "functions_d.html", null],
 ["e", "functions_e.html", null],
 ["f", "functions_f.html", null],
 ["g", "functions_g.html", null],
 ["h", "functions_h.html", null],
 ["i", "functions_i.html", null],
 ["j", "functions_j.html", null],
 ["k", "functions_k.html", null],
 ["l", "functions_l.html", null],
 ["m", "functions_m.html", null],
 ["n", "functions_n.html", null],
 ["o", "functions_o.html", null],
 ["p", "functions_p.html", null],
 ["q", "functions_q.html", null],
 ["r", "functions_r.html", null],
 ["s", "functions_s.html", null],
 ["t", "functions_t.html", null],
 ["u", "functions_u.html", null],
 ["v", "functions_v.html", null],
 ["w", "functions_w.html", null],
 ["x", "functions_x.html", null],
 ["y", "functions_y.html", null],
 ["z", "functions_z.html", null],
 ["~", "functions_0x7e.html", null]
]
```

Definition at line 1 of file functions\_dup.js.

## 30.777 doc/html/functions\_func.js File Reference

### Variables

- var [functions\\_func](#)

### 30.777.1 Variable Documentation

#### 30.777.1.1 var functions\_func

##### Initial value:

```
=
[
 ["a", "functions_func.html", null],
 ["b", "functions_func_b.html", null],
 ["c", "functions_func_c.html", null],
 ["d", "functions_func_d.html", null],
 ["e", "functions_func_e.html", null],
 ["f", "functions_func_f.html", null],
 ["g", "functions_func_g.html", null],
 ["h", "functions_func_h.html", null],
 ["i", "functions_func_i.html", null],
 ["j", "functions_func_j.html", null],
 ["k", "functions_func_k.html", null],
 ["l", "functions_func_l.html", null],
 ["m", "functions_func_m.html", null],
 ["n", "functions_func_n.html", null],
 ["o", "functions_func_o.html", null],
 ["p", "functions_func_p.html", null],
 ["q", "functions_func_q.html", null],
 ["r", "functions_func_r.html", null],
 ["s", "functions_func_s.html", null],
 ["t", "functions_func_t.html", null],
 ["u", "functions_func_u.html", null],
 ["v", "functions_func_v.html", null],
 ["w", "functions_func_w.html", null],
 ["x", "functions_func_x.html", null],
 ["y", "functions_func_y.html", null],
 ["z", "functions_func_z.html", null],
 ["~", "functions_func_0x7e.html", null]
]
```

Definition at line 1 of file `functions_func.js`.

## 30.778 doc/html/functions\_vars.js File Reference

### Variables

- var [functions\\_vars](#)

### 30.778.1 Variable Documentation

#### 30.778.1.1 var functions\_vars

##### Initial value:

```
=
[
 ["a", "functions_vars.html", null],
 ["b", "functions_vars_b.html", null],
 ["c", "functions_vars_c.html", null],
 ["d", "functions_vars_d.html", null],
 ["e", "functions_vars_e.html", null],
 ["f", "functions_vars_f.html", null],
 ["g", "functions_vars_g.html", null],
 ["h", "functions_vars_h.html", null],
 ["i", "functions_vars_i.html", null],
 ["j", "functions_vars_j.html", null],
 ["k", "functions_vars_k.html", null],
 ["l", "functions_vars_l.html", null],
 ["m", "functions_vars_m.html", null],
 ["n", "functions_vars_n.html", null],
 ["o", "functions_vars_o.html", null],
 ["p", "functions_vars_p.html", null],
 ["q", "functions_vars_q.html", null],
 ["r", "functions_vars_r.html", null],
 ["s", "functions_vars_s.html", null],
 ["t", "functions_vars_t.html", null],
 ["u", "functions_vars_u.html", null],
 ["v", "functions_vars_v.html", null],
 ["w", "functions_vars_w.html", null],
 ["x", "functions_vars_x.html", null],
 ["y", "functions_vars_y.html", null],
 ["z", "functions_vars_z.html", null]
]
```

Definition at line 1 of file functions\_vars.js.

## 30.779 doc/html/garch\_8cpp.js File Reference

### Variables

- var [garch\\_8cpp](#)

### 30.779.1 Variable Documentation

#### 30.779.1.1 var garch\_8cpp

##### Initial value:

```
=
[
 ["GARCH11_LL", "class_g_a_r_c_h11__l_l.html", "class_g_a_r_c_h11__l_l"],
 ["WANT_FSTREAM", "garch_8cpp.html#a37d625862c54954e237886ad4bb5e75f", null],
 ["WANT_MATH", "garch_8cpp.html#a8335b327d416f961a5dbaa0e058cc515", null],
 ["WANT_STREAM", "garch_8cpp.html#a6ed6c2d6e68d8f0e7900326b4e30850c", null],
 ["main", "garch_8cpp.html#ae66f6b31b5ad750f1fe042a706a4e3d4", null],
 ["square", "garch_8cpp.html#ae0c42a7620957500708a95cbb31e4260", null]
]
```

Definition at line 1 of file garch\_8cpp.js.

## 30.780 doc/html/generic\_\_graph\_\_algorithms\_8h.js File Reference

### Variables

- var [generic\\_\\_graph\\_\\_algorithms\\_8h](#)

### 30.780.1 Variable Documentation

#### 30.780.1.1 var generic\_\_graph\_\_algorithms\_8h

##### Initial value:

```
=
[
 ["get_fundamental_cycle_set", "generic__graph__algorithms_8h.html#a00083edbeaa34fef2850f154badedf51",
 null],
 ["get_fundamental_cycle_set", "generic__graph__algorithms_8h.html#ad67d17b9bad9725acf6c310416d333b2",
 null],
 ["get_individual_paths", "generic__graph__algorithms_8h.html#a40855b300f757c224478a917bb94d658", null
],
 ["is_path_connected", "generic__graph__algorithms_8h.html#ac110501fae3c10bf52cc0d51cfffb248", null]
]
```

Definition at line 1 of file `generic__graph__algorithms_8h.js`.

## 30.781 doc/html/generic\_\_graph\_\_algorithms\_\_implementation\_8h.js File Reference

### Variables

- var [generic\\_\\_graph\\_\\_algorithms\\_\\_implementation\\_8h](#)

### 30.781.1 Variable Documentation

#### 30.781.1.1 var generic\_\_graph\_\_algorithms\_\_implementation\_8h

##### Initial value:

```
=
[
 ["NodeStatus", "generic__graph__algorithms__implementation_8h.html#a8c61272fccb797a8d283bb4ac2d86cdd",
 null],
 ["get_fundamental_cycle_set", "
 generic__graph__algorithms__implementation_8h.html#a00083edbeaa34fef2850f154badedf51", null],
 ["get_fundamental_cycle_set", "
 generic__graph__algorithms__implementation_8h.html#ad67d17b9bad9725acf6c310416d333b2", null],
 ["get_individual_paths", "
 generic__graph__algorithms__implementation_8h.html#a40855b300f757c224478a917bb94d658", null],
 ["is_path_connected", "
 generic__graph__algorithms__implementation_8h.html#ac110501fae3c10bf52cc0d51cfffb248", null]
]
```

Definition at line 1 of file `generic__graph__algorithms__implementation_8h.js`.

## 30.782 doc/html/gl\_\_aux\_8cpp.js File Reference

### Variables

- var [gl\\_\\_aux\\_8cpp](#)

### 30.782.1 Variable Documentation

#### 30.782.1.1 var gl\_\_aux\_8cpp

#### Initial value:

```
=
[
 ["PI", "gl__aux_8cpp.html#a598a3330b3c21701223ee0ca14316eca", null],
 ["draw_cylinder", "gl__aux_8cpp.html#abf07a538d2deb9fa6232d6b9d5391754", null],
 ["draw_gl_axes_old", "gl__aux_8cpp.html#a4336edca7a1084abe30270885cfac6e1", null],
 ["draw_grid", "gl__aux_8cpp.html#acacfc5a44871d8cd00cd16d5cb4be51e", null],
 ["draw_grid_planes", "gl__aux_8cpp.html#a526dbelb746c6923dde13931872950f5", null],
 ["gl_init", "gl__aux_8cpp.html#ab8f4b1abce1b460a5185c8e50f7c24e", null],
 ["print_gl_matrix", "gl__aux_8cpp.html#a95a6dbe79fb7644dc0deebd1ede245a5", null],
 ["read_gl_matrices", "gl__aux_8cpp.html#aabb2eae1c53cf1b829442db5524cd900", null],
 ["reshape_window", "gl__aux_8cpp.html#a73b6df8ac67794cf1ad91f18c71c4a87", null],
 ["transpose_matrix", "gl__aux_8cpp.html#a7528be4278105e7567474f17bb263734", null],
 ["write_gl_matrices", "gl__aux_8cpp.html#a9f517cb0717bfb7c4374fb1fad9a450", null],
 ["predef_col_table", "gl__aux_8cpp.html#ad52c28cb5bb2182ef58d64f931b81ca3", null],
 ["predefined_colours", "gl__aux_8cpp.html#a26c538147e66a3fb949847b33ee2d816", null]
]
```

Definition at line 1 of file [gl\\_\\_aux\\_8cpp.js](#).

## 30.783 doc/html/gl\_\_aux\_8h.js File Reference

### Variables

- var [gl\\_\\_aux\\_8h](#)

### 30.783.1 Variable Documentation

#### 30.783.1.1 var gl\_\_aux\_8h

#### Initial value:

```
=
[
 ["material_appearance", "classmaterial__appearance.html", "classmaterial__appearance"],
 ["draw_cylinder", "gl__aux_8h.html#abf07a538d2deb9fa6232d6b9d5391754", null],
 ["draw_gl_axes_old", "gl__aux_8h.html#ad53e20c22f9b7f20514b852b10f7f51e", null],
 ["draw_grid", "gl__aux_8h.html#acacfc5a44871d8cd00cd16d5cb4be51e", null],
 ["draw_grid_planes", "gl__aux_8h.html#a526dbelb746c6923dde13931872950f5", null],
 ["gl_init", "gl__aux_8h.html#af380a369c4aff5af718acdcf16159c72", null],
 ["print_gl_matrix", "gl__aux_8h.html#a95a6dbe79fb7644dc0deebd1ede245a5", null],
 ["read_gl_matrices", "gl__aux_8h.html#aabb2eae1c53cf1b829442db5524cd900", null],
 ["reshape_window", "gl__aux_8h.html#a73b6df8ac67794cf1ad91f18c71c4a87", null],
 ["write_gl_matrices", "gl__aux_8h.html#a9f517cb0717bfb7c4374fb1fad9a450", null],
 ["col_init", "gl__aux_8h.html#a431343a72dcf7229c56c2caaaa5df39b", null],
 ["col_setting_back", "gl__aux_8h.html#a431343a72dcf7229c56c2caaaa5df39b", null],
 ["col_setting_back_old", "gl__aux_8h.html#a5809b494e89c6e481127d73239df96", null],
 ["predef_col_table", "gl__aux_8h.html#a65f053d21104a990cb14d130bb91a033", null],
 ["predefined_colours", "gl__aux_8h.html#a26c538147e66a3fb949847b33ee2d816", null]
]
```

Definition at line 1 of file [gl\\_\\_aux\\_8h.js](#).



## 30.784 doc/html/globals\_defs.js File Reference

### Variables

- var [globals\\_defs](#)

### 30.784.1 Variable Documentation

#### 30.784.1.1 var globals\_defs

#### Initial value:

```
=
[
 ["_", "globals_defs.html", null],
 ["a", "globals_defs_a.html", null],
 ["b", "globals_defs_b.html", null],
 ["c", "globals_defs_c.html", null],
 ["d", "globals_defs_d.html", null],
 ["e", "globals_defs_e.html", null],
 ["f", "globals_defs_f.html", null],
 ["g", "globals_defs_g.html", null],
 ["h", "globals_defs_h.html", null],
 ["i", "globals_defs_i.html", null],
 ["m", "globals_defs_m.html", null],
 ["n", "globals_defs_n.html", null],
 ["p", "globals_defs_p.html", null],
 ["r", "globals_defs_r.html", null],
 ["s", "globals_defs_s.html", null],
 ["t", "globals_defs_t.html", null],
 ["u", "globals_defs_u.html", null],
 ["v", "globals_defs_v.html", null],
 ["w", "globals_defs_w.html", null]
]
```

Definition at line 1 of file globals\_defs.js.

## 30.785 doc/html/globals\_dup.js File Reference

### Variables

- var [globals\\_dup](#)

### 30.785.1 Variable Documentation

#### 30.785.1.1 var globals\_dup

#### Initial value:

```
=
[
 [" ", "globals.html", null],
 ["a", "globals_a.html", null],
 ["b", "globals_b.html", null],
 ["c", "globals_c.html", null],
 ["d", "globals_d.html", null],
 ["e", "globals_e.html", null],
 ["f", "globals_f.html", null],
 ["g", "globals_g.html", null],
 ["h", "globals_h.html", null],
 ["i", "globals_i.html", null],
 ["j", "globals_j.html", null],
 ["k", "globals_k.html", null],
 ["l", "globals_l.html", null],
 ["m", "globals_m.html", null],
 ["n", "globals_n.html", null],
 ["o", "globals_o.html", null],
 ["p", "globals_p.html", null],
 ["q", "globals_q.html", null],
 ["r", "globals_r.html", null],
 ["s", "globals_s.html", null],
 ["t", "globals_t.html", null],
 ["u", "globals_u.html", null],
 ["v", "globals_v.html", null],
 ["w", "globals_w.html", null],
 ["x", "globals_x.html", null],
 ["y", "globals_y.html", null],
 ["z", "globals_z.html", null]
]
```

Definition at line 1 of file `globals_dup.js`.

## 30.786 doc/html/globals\_func.js File Reference

### Variables

- var `globals_func`

### 30.786.1 Variable Documentation

#### 30.786.1.1 var `globals_func`

##### Initial value:

```
=
[
 ["a", "globals_func.html", null],
 ["b", "globals_func_b.html", null],
 ["c", "globals_func_c.html", null],
 ["d", "globals_func_d.html", null],
 ["e", "globals_func_e.html", null],
 ["f", "globals_func_f.html", null],
 ["g", "globals_func_g.html", null],
 ["h", "globals_func_h.html", null],
 ["i", "globals_func_i.html", null],
 ["j", "globals_func_j.html", null],
 ["k", "globals_func_k.html", null],
 ["l", "globals_func_l.html", null],
 ["m", "globals_func_m.html", null],
 ["n", "globals_func_n.html", null],
 ["o", "globals_func_o.html", null],
 ["p", "globals_func_p.html", null],
 ["q", "globals_func_q.html", null],
 ["r", "globals_func_r.html", null],
 ["s", "globals_func_s.html", null],
 ["t", "globals_func_t.html", null],
 ["u", "globals_func_u.html", null],
 ["v", "globals_func_v.html", null],
 ["w", "globals_func_w.html", null]
]
```

Definition at line 1 of file `globals_func.js`.

## 30.787 doc/html/glutils\_8cpp.js File Reference

### Variables

- var [glutils\\_8cpp](#)

### 30.787.1 Variable Documentation

#### 30.787.1.1 var glutils\_8cpp

##### Initial value:

```
=
[
 ["M_PI", "glutils_8cpp.html#ae71449b1cc6e6250b91f539153a7a0d3", null],
 ["assert_gl_dummy_and_empty", "glutils_8cpp.html#a225fee730085969bbcf7799558a3deb6", null],
 ["assert_gl_m", "glutils_8cpp.html#a44ecc72ec49de6cf300fe76038fae5a6", null],
 ["draw_sphere", "glutils_8cpp.html#acf2f32d904c8fe28c46e5eaeaf13fb34", null],
 ["list_FBConfigs", "glutils_8cpp.html#af42ffe9d015401a84178981a9f99e18b", null]
]
```

Definition at line 1 of file `glutils_8cpp.js`.

## 30.788 doc/html/glutils\_8h.js File Reference

### Variables

- var [glutils\\_8h](#)

### 30.788.1 Variable Documentation

#### 30.788.1.1 var glutils\_8h

##### Initial value:

```
=
[
 ["ASSERT_GL", "glutils_8h.html#a9d7b8f8cdfadc51ff6f6207aeac37cc3", null],
 ["assert_gl", "glutils_8h.html#aa2089f5fc7da3d61a5780540796158d1", null],
 ["ASSERT_GL_DONT_EXIT", "glutils_8h.html#ac39bf0e6af8e87a0f9c6d202a81ea522", null],
 ["GLUTILS_H_INCLUDED", "glutils_8h.html#acd507b0c3c98dc8199ba953051d907ff", null],
 ["assert_gl_dummy_and_empty", "glutils_8h.html#a225fee730085969bbcf7799558a3deb6", null],
 ["assert_gl_m", "glutils_8h.html#a253d1e974110dd84bd9b73a5344253a8", null],
 ["draw_sphere", "glutils_8h.html#acf2f32d904c8fe28c46e5eaeaf13fb34", null],
 ["list_FBConfigs", "glutils_8h.html#af42ffe9d015401a84178981a9f99e18b", null]
]
```

Definition at line 1 of file `glutils_8h.js`.

## 30.789 doc/html/gotools-core\_2include\_2\_go\_tools\_2geometry\_2copyright\_8h.js File Reference

### Variables

- var [gotools\\_core\\_2include\\_2\\_go\\_tools\\_2geometry\\_2copyright\\_8h](#)

### 30.789.1 Variable Documentation

#### 30.789.1.1 var gotools\_core\_2include\_2\_go\_tools\_2geometry\_2copyright\_8h

##### Initial value:

```
=
[
 ["_COPYRIGHT_H", "
 gotools-core_2include_2_go_tools_2geometry_2copyright_8h.html#abdfb6b142e6341a6d5e35b87ccbfa07", null]
]
```

Definition at line 1 of file [gotools-core\\_2include\\_2\\_go\\_tools\\_2geometry\\_2copyright\\_8h.js](#).

## 30.790 doc/html/group\_\_rbd\_\_common.js File Reference

### Variables

- var [group\\_\\_rbd\\_\\_common](#)

### 30.790.1 Variable Documentation

#### 30.790.1.1 var group\_\_rbd\_\_common

##### Initial value:

```
=
[
 ["include.h", "include_8h.html", null],
 ["myexcept.h", "myexcept_8h.html", null],
 ["myexcept.cpp", "myexcept_8cpp.html", null],
 ["RBD_COMMON", "namespace_r_b_d__c_o_m_m_o_n.html", null],
 ["RBD_LIBRARIES", "namespace_r_b_d__l_i_b_r_a_r_i_e_s.html", null],
 ["ATandT", "group__rbd__common.html#gad1d2a32306b125d9045bbd215c45950e", null],
 ["bool_LIB", "group__rbd__common.html#gae034ee292f37bf7f1a5488c0afe12cfd", null],
 ["DEFAULT_HEADER", "group__rbd__common.html#ga17256595f49d7ecb94970a6f6113da0e", null],
 ["TypeDefException", "group__rbd__common.html#ga38f7e78dddca5ba5093f9e0b0a4051e5", null],
 ["use_namespace", "group__rbd__common.html#gac108aaae7094c39fa1cb4412c5c5369c", null],
 ["UseExceptions", "group__rbd__common.html#ga120be4c0912bac8cbefc71604f25a418", null],
 ["USING_DOUBLE", "group__rbd__common.html#ga70bcbf72f842cb5298e720110eb7b7e7", null]
]
```

Definition at line 1 of file [group\\_\\_rbd\\_\\_common.js](#).

## 30.791 doc/html/gv\_application\_8h.js File Reference

### Variables

- var [gv\\_application\\_8h](#)

#### 30.791.1 Variable Documentation

##### 30.791.1.1 var gv\_application\_8h

###### Initial value:

```
=
[
 ["gvApplication", "classgv_application.html", "classgv_application"],
 ["ColContainer", "gv_application_8h.html#a4fb71c3010e59d6637a56f27a1d9d57d", null],
 ["ObjContainer", "gv_application_8h.html#a930ed0f609554d54ba43b401b20dif37", null]
]
```

Definition at line 1 of file gv\_application\_8h.js.

## 30.792 doc/html/gv\_parametric\_surface\_paintable\_8h.js File Reference

### Variables

- var [gv\\_parametric\\_surface\\_paintable\\_8h](#)

#### 30.792.1 Variable Documentation

##### 30.792.1.1 var gv\_parametric\_surface\_paintable\_8h

###### Initial value:

```
=
[
 ["gvParametricSurfacePaintable", "classgv_parametric_surface_paintable.html", "
 classgv_parametric_surface_paintable"],
 ["genMesh", "gv_parametric_surface_paintable_8h.html#a97b8df4fe8839d115decc99c9eda4a35", null]
]
```

Definition at line 1 of file gv\_parametric\_surface\_paintable\_8h.js.

## 30.793 doc/html/gv\_standard\_mouse\_handler\_8h.js File Reference

### Variables

- var [gv\\_standard\\_mouse\\_handler\\_8h](#)

### 30.793.1 Variable Documentation

#### 30.793.1.1 var gv\_standard\_mouse\_handler\_8h

##### Initial value:

```
=
[
 ["gvStandardMouseHandler", "classgv_standard_mouse_handler.html", "classgv_standard_mouse_handler"],
 ["GV_STANDARD_MOUSEHANDLER_H_INCLUDED", "
 gv_standard_mouse_handler_8h.html#aa5025f5c2d9844b0d7b5775954a0b9a3", null]
]
```

Definition at line 1 of file gv\_standard\_mouse\_handler\_8h.js.

## 30.794 doc/html/gv\_texture\_8h.js File Reference

### Variables

- var [gv\\_texture\\_8h](#)

### 30.794.1 Variable Documentation

#### 30.794.1.1 var gv\_texture\_8h

##### Initial value:

```
=
[
 ["gvTexture", "classgv_texture.html", "classgv_texture"],
 ["EnvModeSet", "gv_texture_8h.html#abfb34fc206cb7e7d2ede80704c1fcc86", [
 ["envDecal", "gv_texture_8h.html#abfb34fc206cb7e7d2ede80704c1fcc86ae9b27fa588bfa1b339103a8f9cd7d92d"
 , null],
 ["envReplace", "
 gv_texture_8h.html#abfb34fc206cb7e7d2ede80704c1fcc86a74718d64656d27245ca720852575e124", null],
 ["envModulate", "
 gv_texture_8h.html#abfb34fc206cb7e7d2ede80704c1fcc86a76402e7f3a9c65225ad37d3cd49ce1f3", null],
 ["envBlend", "gv_texture_8h.html#abfb34fc206cb7e7d2ede80704c1fcc86a66850f3a790c942c59dc2567c41cfebf"
 , null]
]],
 ["MagFilterSet", "gv_texture_8h.html#abbb19582793c79e4ec8fe451f9e236e6", [
 ["magNearest", "
 gv_texture_8h.html#abbb19582793c79e4ec8fe451f9e236e6a656ed1cddd73643821e62a1fdedc48b8", null],
 ["magLinear", "gv_texture_8h.html#abbb19582793c79e4ec8fe451f9e236e6abe6738841e17db775ec39e0479af16b8"
 , null]
]],
 ["MinFilterSet", "gv_texture_8h.html#a22888cb2b2c0262ed85c7ac8aa7c1c2f", [
 ["minNearest", "
 gv_texture_8h.html#a22888cb2b2c0262ed85c7ac8aa7c1c2fa529132446a8de28d302618413fd64095", null],
 ["minLinear", "gv_texture_8h.html#a22888cb2b2c0262ed85c7ac8aa7c1c2fa34564aeb9a54aec0ae6daa30d0ca5965"
 , null],
 ["minNearestMipmapNearest", "
 gv_texture_8h.html#a22888cb2b2c0262ed85c7ac8aa7c1c2fa51d5f598d2c42800336631f9ca5d6b1f", null],
 ["minNearestMipmapLinear", "
 gv_texture_8h.html#a22888cb2b2c0262ed85c7ac8aa7c1c2fae65a99cf93d997f52b3d3244e7e9e7a7", null],
 ["minLinearMipmapNearest", "
 gv_texture_8h.html#a22888cb2b2c0262ed85c7ac8aa7c1c2fa8579243951c8b2cd902ef07b1c94f84c", null],
 ["minLinearMipmapLinear", "
 gv_texture_8h.html#a22888cb2b2c0262ed85c7ac8aa7c1c2fa5bf26bc6347132e8118dc0eda75dd2c9", null]
]],
 ["WrapModeSet", "gv_texture_8h.html#adee2e25c021585161c4cab1b6751d90d", [
 ["wrapClamp", "gv_texture_8h.html#adee2e25c021585161c4cab1b6751d90da81991e50cc3280139282b93608fa871d"
 , null],
 ["wrapRepeat", "
 gv_texture_8h.html#adee2e25c021585161c4cab1b6751d90da0004245ea176bc058284f0652a8416e9", null]
]]
]
```

Definition at line 1 of file gv\_texture\_8h.js.

## 30.795 doc/html/gv\_utilities\_8h.js File Reference

### Variables

- var [gv\\_utilities\\_8h](#)

### 30.795.1 Variable Documentation

#### 30.795.1.1 var gv\_utilities\_8h

##### Initial value:

```
=
[
 ["FLOAT_TYPE", "gv_utilities_8h.html#aa8546893c54ec8a7ec036925e52edee0", null],
 ["MATRIX3", "gv_utilities_8h.html#a5277c85f00358938e80963e1f28bd448", null],
 ["MATRIX4", "gv_utilities_8h.html#a17ca410317577d99816160c68b070fc7", null],
 ["draw_cylinder", "gv_utilities_8h.html#abf07a538d2deb9fa6232d6b9d5391754", null],
 ["draw_gl_axes", "gv_utilities_8h.html#ad3e05c2e1ba69a3e8314f1ab819c114a", null],
 ["draw_gl_axes", "gv_utilities_8h.html#a5fc0d3129255fc8f9a14161e2d20d59b", null],
 ["m3_det", "gv_utilities_8h.html#a15f8dd61e74e6e8637e610a774df9607", null],
 ["m3_identity", "gv_utilities_8h.html#a6091a5b9ab587c9a4b692565e6e1d6c6", null],
 ["m3_inverse", "gv_utilities_8h.html#a3347b853ddaa80d8f92360e764b433e8", null],
 ["m4_det", "gv_utilities_8h.html#acdaa244d22eb4f9c9c0bda4b5d94e5b3", null],
 ["m4_inverse", "gv_utilities_8h.html#a708f7916c5b2c579c1096435f8f481f4", null],
 ["m4_submat", "gv_utilities_8h.html#a94b4064202801e77aa5bd9b975b37d0c", null]
]
```

Definition at line 1 of file gv\_utilities\_8h.js.

## 30.796 doc/html/hholder\_8cpp.js File Reference

### Variables

- var [hholder\\_8cpp](#)

### 30.796.1 Variable Documentation

#### 30.796.1.1 var hholder\_8cpp

##### Initial value:

```
=
[
 ["REPORT", "hholder_8cpp.html#a787099914a94ed31fa544b985b55752f", null],
 ["WANT_MATH", "hholder_8cpp.html#a8335b327d416f961a5dbaa0e058cc515", null],
 ["QRZ", "hholder_8cpp.html#abf43bc890b60a01fda0c7db9631bcf64", null],
 ["QRZ", "hholder_8cpp.html#a57d85feed3907075c9ad5d179793d7a9", null],
 ["QRZT", "hholder_8cpp.html#a259bc631fd14f593e155517ac22c4bf0", null],
 ["QRZT", "hholder_8cpp.html#a223f34bc8e3e034d4679388e0d57447f", null],
 ["square", "hholder_8cpp.html#ae0c42a7620957500708a95cbb31e4260", null]
]
```

Definition at line 1 of file hholder\_8cpp.js.

## 30.797 doc/html/hierarchy.js File Reference

### Variables

- var [hierarchy](#)

### 30.797.1 Variable Documentation

#### 30.797.1.1 var hierarchy

Definition at line 1 of file hierarchy.js.

## 30.798 doc/html/hp\_\_s1880\_8c.js File Reference

### Variables

- var [hp\\_\\_s1880\\_8c](#)

### 30.798.1 Variable Documentation

#### 30.798.1.1 var hp\_\_s1880\_8c

#### Initial value:

```
=
[
 ["HP_S1880", "hp__s1880_8c.html#ae6033f654b8655d58b42ee58d81bce23", null],
 ["hp_s1880", "hp__s1880_8c.html#afb95e042012b5d2b20a97b88fc8c53cb", null]
]
```

Definition at line 1 of file hp\_\_s1880\_8c.js.

## 30.799 doc/html/igeslib\_\_doxymain\_8h.js File Reference

### Variables

- var [igeslib\\_\\_doxymain\\_8h](#)

### 30.799.1 Variable Documentation

#### 30.799.1.1 var igeslib\_\_doxymain\_8h

#### Initial value:

```
=
[
 ["_IGESLIB", "igeslib__doxymain_8h.html#aa60f7acaa57d814a1403f7bb16981c1d", null]
]
```

Definition at line 1 of file igeslib\_\_doxymain\_8h.js.



## 30.800 doc/html/include\_8h.js File Reference

### Variables

- var [include\\_8h](#)

#### 30.800.1 Variable Documentation

##### 30.800.1.1 var include\_8h

#### Initial value:

```
=
[
 ["ATandT", "group__rbd__common.html#gad1d2a32306b125d9045bbd215c45950e", null],
 ["bool_LIB", "group__rbd__common.html#gae034ee292f37bf7f1a5488c0afe12cfd", null],
 ["DEFAULT_HEADER", "group__rbd__common.html#ga17256595f49d7ecb94970a6f6113da0e", null],
 ["TypeDefException", "group__rbd__common.html#ga38f7e78dddca5ba5093f9e0b0a4051e5", null],
 ["use_namespace", "group__rbd__common.html#gac108aaae7094c39fa1cb4412c5c5369c", null],
 ["UseExceptions", "group__rbd__common.html#ga120be4c0912bac8cbefc71604f25a418", null],
 ["USING_DOUBLE", "group__rbd__common.html#ga70bcbf72f842cb5298e720110eb7b7e7", null],
 ["long_Real", "include_8h.html#ab577ec287616d0a76ae90f9b579f43c1", null],
 ["Real", "include_8h.html#a221583c51910bd7c44613484867d36be", null]
]
```

Definition at line 1 of file include\_8h.js.

## 30.801 doc/html/intjoinper\_8c.js File Reference

### Variables

- var [intjoinper\\_8c](#)

#### 30.801.1 Variable Documentation

##### 30.801.1.1 var intjoinper\_8c

#### Initial value:

```
=
[
 ["INT_JOIN_PER", "intjoinper_8c.html#a029faabe83b1750e5eff0ca8c7005ae1", null],
 ["int_join_per", "intjoinper_8c.html#ab4e5654f7ea3cab7e1fdada009fb0992", null]
]
```

Definition at line 1 of file intjoinper\_8c.js.

## 30.802 doc/html/jacobi\_8cpp.js File Reference

### Variables

- var [jacobi\\_8cpp](#)

## 30.802.1 Variable Documentation

### 30.802.1.1 var jacobi\_8cpp

#### Initial value:

```
=
[
 ["REPORT", "jacobi_8cpp.html#a787099914a94ed31fa544b985b55752f", null],
 ["WANT_MATH", "jacobi_8cpp.html#a8335b327d416f961a5dbaa0e058cc515", null],
 ["Jacobi", "jacobi_8cpp.html#ad370e3456b419cfe2eae25f2300fd5b1", null],
 ["Jacobi", "jacobi_8cpp.html#ac6a0bb5dfe7a0d639ce053deab66b033", null],
 ["Jacobi", "jacobi_8cpp.html#ac0814314484221101aa36eb519164b07", null],
 ["Jacobi", "jacobi_8cpp.html#a72a2a2c9c0238ab873dc056026fec97f", null]
]
```

Definition at line 1 of file jacobi\_8cpp.js.

## 30.803 doc/html/jquery.js File Reference

### Functions

- **b extend** ({cssHooks:{opacity:{get:function(bw, bv){if(bv){var e=Z(bw,"opacity","opacity");return e===""?"1"↵:e}else{return bw.style.opacity}}}}, cssNumber:{fillOpacity:true, fontWeight:true, lineHeight:true, opacity↵:true, orphans:true, widows:true, zIndex:true, zoom:true}, cssProps:{float:"b.support.cssFloat?"css↵Float":"styleFloat"}, style:function(bx, bw, bD, by){if(!bx||bx.nodeType===3||bx.nodeType===8||bx.↵style){return}var bB, bC, bz=b.camelCase(bw), bv=bx.style, bE=b.cssHooks[bz];bw=b.cssProps[bz]||bz;if(b↵D!==L){bC=typeof bD;if(bC==="string"&&(bB=l.exec(bD))){bD=(+(bB[1]+1)\*+bB[2])+parseFloat(b.css(bx, bw));bC="number"}if(bD===null||bC==="number"&&isNaN(bD)){return}if(bC==="number"&&!b.cssNumber[bz]){bD+="px"}if(!bE↵|bE||(bD=bE.set(bx, bD))!==L){try{bv[bw]=bD}catch(bA){}}else{if(bE &&"get" in bE &&(bB=bE.get(bx, false, by))!==L){return bB}return bv[bw]}}, css:function(by, bx, bv){var bw, e;bx=b.camelCase(bx);e=b.css↵Hooks[bx];bx=b.cssProps[bx]||bx;if(bx==="cssFloat"){bx="float"}if(e &&"get" in e &&(bw=e.get(by, true, bv))!==L){return bw}else{if(Z){return Z(by, bx)}}}, swap:function(bx, bw, by){var e={};for(var bv in bw){e[bv]=bx.style[bv];bx.style[bv]=bw[bv]}by.call(bx);for(bv in bw){bx.style[bv]=e[bv]}}
- **b each** (["height","width"], function(bv, e){b.cssHooks[e]={get:function(by, bx, bw){var bz;if(bx){if(by.offset↵Width!==0){return p(by, e, bw)}else{b.swap(by, a7, function(){bz=p(by, e, bw)}})return bz}}, set:function(bw, bx){if(bc.test(bx)){bx=parseFloat(bx);if(bx >=0){return bx+"px"}}else{return bx}}})
- **if** (!b.support.opacity)
- **b** (function(){if(!b.support.reliableMarginRight){b.cssHooks.marginRight={get:function(bw, bv){var e;b.↵swap(bw,{display:"inline-block"}, function(){if(bv){e=Z(bw,"margin-right","marginRight")}else{e=bw.style.↵marginRight}});return e}}})
- **if** (av.defaultView &&av.defaultView.getComputedStyle)
- **if** (av.documentElement.currentStyle)
- **function p** (by, bw, bv)
- **if** (b.expr &&b.expr.filters)

## Variables

- function **bb**
- function **L** {var av=bb.document,bu=bb.navigator,bl=bb.location
- var **b**
- var **au** =/opacity=(^[^)]\*)/;z=/([A-Z]|^ms)/g;bc=/^-?\d+(?:px)?\$/i;bn=/^-?\d/;l=/^([\+])=([\+.\de]+)/;a7={position↵:"absolute",visibility:"hidden",display:"block"},an=["Left","Right"],a1=["Top","Bottom"],Z,aI,aX
- **b fn css** =function(e,bv){if(arguments.length===2&&bv===L){return this}return b.access(this,e,bv,true,function(bx,bw,by){return by!==L?b.style(bx,bw,by):b.css(bx,bw)})}
- **b curCSS** =b.css
- **Z** =aI||aX
- var **k** =/%20/g
- var **ap** =/[\]\$/
- var **bs** =/r?\n/g
- var **bq** =/#.\*\$/
- var **aD** =/^(.\*?):[\ \t]\*([^\r\n]\*)r?\$/mg
- var **aZ** =/^(?:color|date|datetime|datetime-local|email|hidden|month|number|password|range|search|tel|text|time|url|week)\$/i
- var **aM** =/^(?:about|app|app-storage|.+\-extension|file|res|widget):\$/
- var **aQ** =/^(?:GET|HEAD)\$/
- var **c**

## 30.803.1 Function Documentation

30.803.1.1 **b ( function(){if(!b.support.reliableMarginRight){b.cssHooks.marginRight={get:function(bw, bv){var e;b.swap(bw,{display:"inline-block"}, function(){if(bv){e=Z(bw,"margin-right","marginRight")}else{e=bw.style.↵marginRight}};return e}} } )**

30.803.1.2 **b each ( function(bv, e){b.cssHooks[e]={get:function(by, bx, bw){var bz;if(bx){if(by.offsetWidth!==0){return p(by, e, bw)}else{b.swap(by, a7, function(){bz=p(by, e, bw)}}}return bz}}, set:function(bw, bx){if(bc.test(bx)){bx=parseFloat(bx);if(bx >=0){return bx+"px"}}else{return bx}} } )**

30.803.1.3 **b extend ( {cssHooks:{opacity:{get:function(bw, bv){if(bv){var e=Z(bw,"opacity","opacity");return e===""?"1"↵:e}else{return bw.style.opacity}}}, cssNumber:{fillOpacity:true, fontWeight:true, lineHeight:true, opacity:true, orphans:true, widows:true, zIndex:true, zoom:true}, cssProps:{"float":b.support.cssFloat?"cssFloat":"styleFloat"}, style:function(bx, bw, bD, by){if(!bx||bx.nodeType===3||bx.nodeType===8||!bx.style){return}var bB, bC, bz=b.camelCase(bw), bv=bx.style, bE=b.cssHooks[bz];bw=b.cssProps[bz]||bz;if(bD===L){bC=typeof bD;if(bC==="string"&&(bB=l.exec(bD))){bD=+(bB[1]+1)\*+bB[2]+parseFloat(b.css(bx, bw));bC="number"}if(bD===null||bC==="number"&&isNaN(bD)){return}if(bC==="number"&&b.css↵Number[bz]){bD+="px"}if(!bE||!("set" in bE)||("set" in bE)&&(bD=bE.set(bx, bD))!==L){try{bv[bw]=bD}catch(bA){}}else{if(bE &&"get" in bE &&(bB=bE.get(bx, false, by))!==L){return bB}return bv[bw]}}, css:function(by, bx, bv){var bw, e;bx=b.camelCase(bx);e=b.cssHooks[bx];bx=b.cssProps[bx]||bx;if(bx==="cssFloat"){bx="float"}if(e &&"get" in e &&(bw=e.get(by, true, bv))!==L){return bw}else{if(Z){return Z(by, bx)}}, swap:function(bx, bw, by){var e={};for(var bv in bw){e[bv]=bx.style[bv];bx.style[bv]=bw[bv]}by.call(bx);for(bv in bw){bx.style[bv]=e[bv]} } )**

30.803.1.4 **if ( av.documentElement. currentStyle )**

Definition at line 23 of file jquery.js.

30.803.1.5 **if ( av.defaultView &&av.defaultView. getComputedStyle )**

Definition at line 23 of file jquery.js.

30.803.1.6 `if ( b.expr && b.expr.filters )`

Definition at line 23 of file jquery.js.

30.803.1.7 `if ( !b.support.opacity )`

Definition at line 23 of file jquery.js.

30.803.1.8 `function p ( by, bw, bv )`

Definition at line 23 of file jquery.js.

## 30.803.2 Variable Documentation

30.803.2.1 `var aD =/(.*?):[\t]*([\r\n]*)\r?$/mg`

Definition at line 23 of file jquery.js.

30.803.2.2 `var aM =/^(?:about|app|app-storage|.+-extension|file|res|widget):$/`

Definition at line 23 of file jquery.js.

30.803.2.3 `var ap =/[\\]$/`

Definition at line 23 of file jquery.js.

30.803.2.4 `var aQ =/^(?:GET|HEAD)$/`

Definition at line 23 of file jquery.js.

30.803.2.5 `var au =/opacity=([\^]*),z=/([A-Z])^ms)/g,bc=/^-?\d+(?:px)?$/i,bn=/^-?\d/,l=/^([\+])=([\+.\de]+)/,a7={position↵  
:"absolute",visibility:"hidden",display:"block"},an=["Left","Right"],a1=["Top","Bottom"],Z,aI,aX`

Definition at line 23 of file jquery.js.

30.803.2.6 `var aZ =/^(?:color|date|datetime|datetime-  
local|email|hidden|month|number|password|range|search|tel|text|time|url|week)$/i`

Definition at line 23 of file jquery.js.

## 30.803.2.7 var b

## Initial value:

```

=(function() {var bF=function(b0,b1) {return new bF.fn.init(b0,b1,bD)},bU=bb.jquery,bH=
bb.$,bD,bY=/^(?:(?![^<]*<[\w\W]+>)[^>]*$|#[\w\W-]*$)/,bM=/\S/,bI=/^\s+/,bE=/\s+$/,bA=/^(?!(\w+)\s+\/?>(?<
\/>)?$/,bN=/^[\],:;]{s}*$|,bW=/\[\(?:[^\]\/bfnrt]|u[0-9a-fA-F]{4})/g,bP=/^[^\\\/n\r]*\|true|false|null|-?d
+\/?\.?(\.d*)?(?:[eE](+|-)?d+)?/g,bJ=/(?:^|:|,)(?:*|\(|/g,by=/ (webkit) [\\/] ([\w.]+) /,bR=/ (opera) (?:.*version
)? [\\/] ([\w.]+) /,bQ=/ (msie) ([\w.]+) /,bS=/ (mozilla) (?:.*? rv:([\w.]+)?)?/,bB=-([a-z]{0-9})/g,bZ=/^-ms-/,bT
=function(b0,b1) {return(b1+"").toUpperCase()}.push,bU.userAgent,bV,bC,e,bL=Object.prototype.toString,bG=Object
.prototype.hasOwnProperty,bz=Array.prototype.push,bK=Array.prototype.slice,bO=String.prototype.trim,bv=Array
.prototype.indexOf,bx={};bF.fn=bF.prototype={constructor:bF,init:function(b0,b4,b3) {var b2,b5,b1,b6;if(!b0) {
return this;if(b0.nodeType) {this.context=this[0]=b0;this.length=1;return this;if(b0=="body"&&!b4&&av.body) {
this.context=av;this[0]=av.body;this.selector=b0;this.length=1;return this;if(typeof b0=="string") {if(b0.
charAt(0)=="<"&&b0.charAt(b0.length-1)==">"&&b0.length>=3) {b2=[null,b0,null]}else{b2=bY.exec(b0)}if(b2&&(b2[
1]||b4)) {if(b2[1]) {b4=b4 instanceof bF?b4[0]:b4;b6=(b4?b4.ownerDocument||b4:av);b1=bA.exec(b2,b1,b2)}if(bF
.isPlainObject(b4)) {b0=[av.createElement(b1[1])];bF.fn.attr.call(b0,b4,true)}else{b0=[b6.createElement(b1[1]
)]}else{b1=bF.buildFragment([b2[1]],[b6]);b0=(b1.cacheable?bF.clone(b1.fragment):b1.fragment).childNodes
return bF.merge(this,b0)}else{b5=av.getElementById(b2[2]);if(b5&&b5.parentNode) {if(b5.id==b2[2]) {return b3.
find(b0)}this.length=1;this[0]=b5}this.context=av;this.selector=b0;return this}if(!b4||b4.jquery) {return
(b4||b3).find(b0)}else{return this.constructor(b4).find(b0)}else{if(bF.isFunction(b0)) {return b3.ready(b0)
}}if(b0.selector!=L) {this.selector=b0.selector;this.context=b0.context;return bF.makeArray(b0,this)},
selector:"",jquery:"1.7.1",length:0,size:function() {return this.length},toArray:function() {return bK.call(this,0)
},get:function(b0) {return b0==null?this.toArray():(b0<0?this[this.length+b0]:this[b0])},pushStack:function(
b1,b3,b0) {var b2=this.constructor();if(bF.isArray(b1)) {b2.apply(b2,b1)}else{bF.merge(b2,b1)}b2.prevObject=
this;b2.context=this.context;if(b3=="find") {b2.selector=this.selector+(this.selector?" ":"")+b0}else{if(b3) {b2
.selector=this.selector+"."+b3+"("+b0+")"}return b2},each:function(b1,b0) {return bF.each(this,b1,b0)},
ready:function(b0) {bF.bindReady();bC.add(b0);return this},eq:function(b0) {b0+=b0;return b0===-1?this.slice(b0
):this.slice(b0,b0+1)},first:function() {return this.eq(0)},last:function() {return this.eq(-1)},slice:
function() {return this.pushStack(bK.apply(this,arguments),"slice",bK.call(arguments).join(""));map:function(b0) {
return this.pushStack(bF.map(this,function(b2,b1) {return b0.call(b2,b1,b2)}),end:function() {return this.
prevObject||this.constructor(null)},push:bz,sort:[].sort,splice:[].splice);bF.fn.init.prototype=bF.fn;bF.extend
=bF.fn.extend=function() {var b9,b2,b0,b1,b6,b7,b5=arguments[0]||[],b4=1,b3=arguments.length,b8=false;if(
typeof b5=="boolean") {b8=b5;b5=arguments[1]||[];b4=2}if(typeof b5!="object"&&bF.isFunction(b5)) {b5=[];if(b3==
=b4) {b5=this;--b4}for(;b4<b3;b4++) {if((b9=arguments[b4])!=null) {for(b2 in b9) {b0=b5[b2];b1=b9[b2];if(b5==b1
) {continue}if(b8&&b1&&(bF.isPlainObject(b1)||b6=bF.isArray(b1))) {if(b6) {b6=false;b7=b0&&bF.isArray(b0)?b0:
[]};else{b7=b0&&bF.isPlainObject(b0)?b0:[]};b5[b2]=bF.extend(b8,b7,b1)}else{if(b1!=
L) {b5[b2]=b1}}}}return b5};bF.extend({noConflict:function(b0) {if(bb.$===bF) {
bb.$=bH}if(b0&&bb.jquery===bF) {bb.jquery=bU}return bF},isReady:false,readyWait:1,holdReady:function(
b0) {if(b0) {bF.readyWait++}else{bF.ready(true)}},ready:function(b0) {if(b0==true&&!-bF.readyWait)|| (b0!
=true&&!bF.isReady) {if(!av.body) {return setTimeout(bF.ready,1)}bF.isReady=true;if(b0!
=true&&-bF.readyWait>0) {return}bC.fireWith(av,(bF));if(bF.fn.trigger(bF(av).trigger("ready").off("ready"))},bindReady:function() {
if(bC) {return}bC=bF.Callbacks("once memory");if(av.readyState=="complete") {return setImmediate(bF.ready,1)}if(
av.addEventListener) {av.addEventListener("DOMContentLoaded",e,false);bb.addEventListener("load",bF.ready,
false)}else{if(av.attachEvent) {av.attachEvent("onreadystatechange",e);bb.attachEvent("onload",bF.ready);var
b0=false;try {b0=bb.frameElement==null}catch(b1) {}if(av.documentElement.doScroll&&b0) {bW()}}},isFunction:
function(b0) {return bF.type(b0)=="function"},isArray:Array.isArray||function(b0) {return bF.type(b0)=="
array"},isWindow:function(b0) {return b0&&typeof b0=="object"&&"setInterval" in b0},isNumeric:function(b0) {
return !isNaN(parseFloat(b0))&&isFinite(b0)},type:function(b0) {return b0==null?String(b0):bx[bL.call(b0)]||"
object"},isPlainObject:function(b2) {if(!b2||bF.type(b2)!="object"||b2.nodeType||bF.isWindow(b2)) {return false}
try{if(b2.constructor&&!bG.call(b2,"constructor")&&!bG.call(b2.constructor.prototype,"isPrototypeOf")) {return
false}}catch(b1) {return false}var b0;for(b0 in b2) {return b0==L||bG.call(b2,b0)},isEmptyObject:function(
b1) {for(var b0 in b1) {return false}return true},error:function(b0) {throw new Error(b0)},parseJSON:function(b0)
) {if(typeof b0!="string"||!b0) {return null}b0=bF.trim(b0);if(bb.JSON&&bb.JSON.parse) {return
bb.JSON.parse(b0)}if(bN.test(b0.replace(bW,"@").replace(bP,"")) {return new Function("
return "+b0)())}bF.error("Invalid JSON: "+b0)},parseXML:function(b2) {var b0,b1;try{if(
bb.DOMParser) {b1=new DOMParser();b0=b1.parseFromStream(b2,"text/xml")}else{b0=new ActiveXObject("
Microsoft.XMLDOM");b0.async="false";b0.loadXML(b2)}catch(b3) {b0=L}if(!b0||b0.documentElement||b0.
getElementsByTagName("parsererror").length) {bF.error("Invalid XML: "+b2)}return b0},noop:function() {},globalEval:functio
) {if(b0&&bM.test(b0)) {bb.execScript(function(b1) {bb["eval"].call(bb,b1)})(b0)},camelCase:function(
b0) {return b0.replace(bZ,"ms-").replace(bB,bT)},nodeName:function(b1,b0) {return b1.nodeName&&b1.nodeName.
toUpperCase()===b0.toUpperCase()},each:function(b3,b6,b2) {var b1,b4=0,b5=b3.length,b0=b5===
L||bF.isFunction(b3);if(b2) {if(b0) {for(b1 in b3) {if(b6.apply(b3[b1],b2)===false) {break}}else{for(;b4<b5;)
{if(b6.apply(b3[b4++],b2)===false) {break}}}}else{if(b0) {for(b1 in b3) {if(b6.call(b3[b1],b1,b3[b1])===false)
{break}}}}else{for(;b4<b5;) {if(b6.call(b3[b4],b4,b3[b4++])===false) {break}}}}return b3},trim:bO?function(b0) {
return b0==null?"":b0.call(b0):function(b0) {return b0==null?"":b0.toString().replace(bI,"").replace(bE,"")},
makeArray:function(b3,b1) {var b0=b1||[];if(b3!=null) {var b2=bF.type(b3);if(b3.length==null||b2=="string"||
b2=="function"||b2=="regexp"||bF.isWindow(b3)) {b2.call(b0,b3)}else{bF.merge(b0,b3)}return b0},inArray:
function(b2,b3,b1) {var b0;if(b3) {if(bv) {return bv.call(b3,b2,b1)}b0=b3.length;b1=b1?b1<0?Math.max(0,b0+b1):b1:0;
for(;b1<b0;b1++) {if(b1 in b3&&b3[b1]===b2) {return b1}}return -1},merge:function(b4,b2) {var b3=b4.length,b1=
0;if(typeof b2.length=="number") {for(var b0=b2.length;b1<b0;b1++) {b4[b3++] =b2[b1]}}else{while(b2[b1]!
=L) {b4[b3++] =b2[b1++]}b4.length=b3;return b4},grep:function(b1,b6,b0) {var b2=[],b5,b0=!b0;for(var b3=0,b4
=b1.length;b3<b4;b3++) {b5=!b6(b1[b3],b3);if(b0!
=b5) {b2.push(b1[b3])}return b2},map:function(b0,b7,b8) {var
b5,b6,b4=[],b2=0,b1=b0.length,b3=b0 instanceof bF||b1!=L&&typeof b1=="number"&&((b1>0&&b0[0]&&b0[b1-1])
||b1==0)||bF.isArray(b0);if(b3) {for(;b2<b1;b2++) {b5=b7(b0[b2],b2,b8);if(b5!=null) {b4[b4.length]=b5}}else{
for(b6 in b0) {b5=b7(b0[b6],b6,b8);if(b5!=null) {b4[b4.length]=b5}}return b4.concat.apply([],b4)},guid:1,proxy:
function(b4,b3) {if(typeof b3=="string") {var b2=b4[b3];b3=b4[b3];if(!bF.isFunction(b4)) {return
L}var b0=bK.call(arguments,2),b1=function() {return b4.apply(b0,b0.concat(bK.call(arguments)))};b1.guid=b4.
guid=b4.guid||b1.guid||bF.guid++;return b1},access:function(b0,b8,b6,b2,b5,b7) {var b1=b0.length;if(typeof b8
=="object") {for(var b3 in b8) {bF.access(b0,b3,b8[b3],b2,b5,b6)}return b0}if(b6!=
L) {b2=!b7&&b2&&bF.isFunction(b6);for(var b4=0;b4<b1;b4++) {b5(b0[b4],b8,b2?b6.call(b0[b4],b4,b5(b0[b4],b8)
):b6,b7)}return b0}return b1?b5(b0[0],b8):L},now:function() {return(new Date()).getTime()},uaMatch:function(
b1) {b1=b1.toLowerCase();var b0=by.exec(b1)||bR.exec(b1)||bQ.exec(b1)||b1.indexOf("compatible")<0&&bS.exec(b1)
||[];return{browser:b0[1]||"",version:b0[2]||"0"}},sub:function() {function b0(b3,b4) {return new b0.fn.init(

```

```

b3,b4)}bF.extend(true,b0,this);b0.superclass=this;b0.fn=b0.prototype=this();b0.fn.constructor=b0;b0.sub=this.
sub;b0.fn.init=function b2(b3,b4){if(b4&&b4 instanceof bF&&!b4 instanceof b0){b4=b0(b4)}return bF.fn.init.
call(this,b3,b4,b1)};b0.fn.init.prototype=b0.fn;var b1=b0(av);return b0};bF.each("Boolean
Number String Function Array Date RegExp Object".split(" "),function(b1,b0){bX["[object "+b0+"]"]=b0.toLowerCase(
)});bv=bF.uaMatch(bX);if(bv.browser){bF.browser[bv.browser]=true;bF.browser.version=bV.version;if(bF.browser.
webkit){bF.browser.safari=true}if(bM.test("\xA0+");bI=/^\s$/;bE=/\s$/;bD=bF(av);if(av.
addEventListener){e=function(){av.removeEventListener("DOMContentLoaded",e,false);bF.ready()}else{if(av.attachEvent
){e=function(){if(av.readyState==="complete"){av.detachEvent("onreadystatechange",e);bF.ready()}}function
bw(){if(bF.isReady){return}try{av.documentElement.doScroll("left")}catch(b0){setTimeout(bw,1);return}bF.
ready()}return bF}});var a2={};function X(e){var bv=a2[e]={},bw,bx;e=e.split(/\s+/);for(bw=0,bx=e.length;bw<bx;
bw++){bv[e[bw]]=true}return bv}.Callbacks=function(bw){bw=bw?{a2[bw]|X(bw)}:[];var bB=[],bC=[],bx,by,bv,
bz,bA,bE=function(bF){var bG,bJ,bI,bH,bK;for(bG=0,bJ=bF.length;bG<bJ;bG++){bI=bF[bG];bH=
b.type(bI);if(bH==="array"){bE(bI)}else{if(bH==="function"){if(!bw.unique||!bD.has(bI)){bB.push(bI)}}},e
=bFunction(bG,bF){bF=bF||[];bx=!bw.memory||[bG,bF];by=true;BA=bv||0;bv=0;bz=bB.length;for(;bB&&bA<bz;bA++){if
(bB[bA].apply(bG,bF)==false&&bw.stopOnFalse){bx=true;break};by=false;if(bB){if(!bw.once){if(bC&&bC.length){
bx=bC.shift();bD.fireWith(bx[0],bx[1])}else{if(bx===true){bD.disable()}else{bB=[]}}},bD={add:function(){if
(bB){var bB=bB.length;bE(arguments);if(by){bz=bB.length;else{if(bx&&bx!==true){bv=bF;e(bx[0],bx[1])}}return
this},remove:function(){if(bB){var bF=arguments,bH=0,bI=bF.length;for(;bH<bI;bH++){for(var bG=0;bG<bB.
length;bG++){if(bF[bH]==bB[bG]){if(by){if(bG<=bz){bz--;if(bG<=bA){bA--}}bB.splice(bG--,1);if(bw.unique){break}
}}return this},has:function(bG){if(bB){var bF=0,bH=bB.length;for(;bF<bH;bF++){if(bG==bB[bF]){return true}}
return false},empty:function(){bB=[];return this},disable:function(){bB=bC=bx=L;return this},disabled:
function(){return !bB},lock:function(){bC=L;if(!bx||bx===true){bD.disable()}return this},locked:function(){
return !bC},fireWith:function(bG,bF){if(bC){if(by){if(!bw.once){bC.push([bG,bF])}else{if(!bw.once&&bx)}(e(bG,
bF))}return this},fire:function(){bD.fireWith(this,arguments);return this},fired:function(){return !!bx};
return bD};var aJ=[];b.extend({Deferred:function(bY){var bx=b.Callbacks("once memory"),bw=
b.Callbacks("once memory"),bv=b.Callbacks("memory"),e="pending",bA={resolve:bx,reject:bw,notify:bv},bC={
done:bx.add,fail:bw.add,progress:bv.add,state:function(){return e},isResolved:bx.fired,isRejected:bv.fired,
then:function(bE,bD,bF){bB.done(bE).fail(bD).progress(bF);return this},always:function(){bB.done.apply(bB,
arguments).fail.apply(bB,arguments);return this},pipe:function(bF,bE,bD){return bF.Deferred(function(bG){
b.each({done:[bF,"resolve"],fail:[bE,"reject"],progress:[bD,"notify"]},function(bI,bL){var bH=bL[0],bK=bL[
1],bJ;if(b.isFunction(bH)){bB[bI](function(){bJ=bH.apply(this,arguments);if(bJ&&
b.isFunction(bJ.promise)){bJ.promise().then(bG.resolve,bG.reject,bG.notify)}else{bG[bK+"With"](this===bB?
bG:this,[bJ])}}else{bB[bI](bG[bK])}})).promise();},promise:function(){if(bE==null){bE=bC}else{for(var bD
in bC){bE[bD]=bC[bD]}}return bE},bB=bC.promise({}),bz;for(bz in bA){bB[bz]=bA[bz].fire;bB[bz+"With"]=bA[bz]
.fireWith}bB.done(function(){e="resolved"},bw.disable,bv.lock).fail(function(){e="rejected"},bx.disable,bv.
lock);if(by){by.call(bB,bB)}return bB},when:function(bA){var bx=aJ.call(arguments,0),bv=0,e=bx.length,bB=new
Array(e),bw=e,by=e,bC=e<=1&&bA&&b.isFunction(bA.promise)?bA.Deferred():bE=bC.promise();function bD(bF){
return function(bG){bx[bF]=arguments.length>1?aJ.call(arguments,0):bG;if(!(--bw)){bC.resolveWith(bC,bx)}}
function bz(bF){return function(bG){bB[bF]=arguments.length>1?aJ.call(arguments,0):bG;bC.notifyWith(bE,bB)}}if
(e>1){for(;bv<e;bv++){if(bx[bv]&&bx[bv].promise&&b.isFunction(bx[bv].promise)){bx[bv].promise().then(bD(bv),
bC.reject,bz(bv))}else{--bw}}if(!bw){bC.resolveWith(bC,bx)}}else{if(bC!=bA){bC.resolveWith(bC,e?[bA]:[])}}
return bE}});b.support=(function(){var bJ,bI,bF,bG,bx,bE,bA,bD,bz,bK,bB,by,bw,bv=av.createElement("div"),bH=
av.documentElement,bV.setAttribute("className","t");bv.innerHTML="<link/><table/><a href='/a'
style='top:1px;float:left;opacity:.55;'><a/><input type='checkbox'/'>;bI=bv.getElementsByTagName("*").getElementsByTagName("a")
[0];if(!bI||!bI.length||!bF){return}bJG=av.createElement("select");bJG=av.createElement("div");bv.appendChild
(av.createElement("option"));bE=bv.getElementsByTagName("input")[0];bJ=leadingWhitespace(bv.firstChild.
nodeType===3),tbody:!bv.getElementsByTagName("tbody").length,htmlSerialize:!bv.getElementsByTagName("link").
length,style/top/.test(bF.getAttribute("style")),hrefNormalized:(bF.getAttribute("href")==="/a"),opacity/^0.5/
/.test(bF.style.opacity),cssFloat:!bF.style.cssFloat,checkOn:(bE.value==="on"),optSelected:bx.selected,
getSetAttribute:bv.className==="t",enctype:!av.createElement("form").enctype,html5Clone:av.createElement("nav")
.cloneNode(true).outerHTML==="<nav></nav>",submitBubbles:true,changeBubbles:true,focusInBubbles:false,
deleteExpando:true,noCloneEvent:true,inlineBlockNeedsLayout:false,shrinkWrapBlocks:false,reliableMarginRight:
true);bE.checked=true;bJ.noCloneChecked=bE.cloneNode(true).checked;bG.disabled=true;bJ.optDisabled=!bx.
disabled;try{delete bv.test}catch(bC){bJ.deleteExpando=false}if(!bv.addEventListener&&bv.fireEvent){
bv.attachEvent("onclick",function(){bJ.noCloneEvent=false});bv.cloneNode(true).fireEvent("onclick")}bE=av.
createElement("input");bE.value="t";bE.setAttribute("type","radio");bJ.radioValue=bE.value==="t";bE.
setAttribute("checked","checked");bv.appendChild(bE);bD=av.createDocumentFragment();bD.appendChild(bv.lastChild);bJ.
checkClone=bD.cloneNode(true).cloneNode(true).lastChild.checked;bJ.appendChild(bE.checked);bD.removeChild(bE
);bD.appendChild(bv);bv.innerHTML="";if(bb.getComputedStyle){bA=av.createElement("div");bA.style.width="0"
;bA.style.marginRight="0";bv.style.width="2px";bv.appendChild(bA);bJ.reliableMarginRight=(parseInt((
bb.getComputedStyle(bA,null)||{marginRight:0}).marginRight,10)||0)===0;if(bv.attachEvent){for(by in {
submit:1,change:1,focusin:1}){bB="on"+by;bw=(bB in bv);if(!bw){bv.setAttribute(bB,"return");bw=(typeof bv[bB]==
"function")?bJ[by+"Bubbles"]=bw}bD.removeChild(bv);bD=bG=bx=bA=bv=bE=null;b(function){var bM,bU,bV,bT,bN
,bO,bL,bS,bR,e,bP,bQ=av.getElementsByTagName("body")[0];if(!bQ){return}bL=1;bS="
position:absolute;top:0;left:0;width:1px;height:1px;margin:0";bR="visibility:hidden;border:0";e="style='"+bS+"border:
#000;padding:0';bP="<div '+e+'><div/></div><table '+e+' cellpadding='0'
cellspacing='0'><tr><td></td></tr></table>";bM=av.createElement("div");bM.style.cssText=bR+width:0;height:0;position:
bL+"px";bQ.insertBefore(bM,bQ.firstChild);bv=av.createElement("div");bM.appendChild(bv);bv.innerHTML="
<table><tr><td style='padding:0;border:0;display:none'></td><td></td></tr></table>";bz=bv.getElementsByTagName("
td");bw=(bz[0].offsetHeight===0);bz[0].style.display="";bz[1].style.display="none";bJ.reliableHiddenOffsets=
bw&&(bz[0].offsetHeight===0);bv.innerHTML="";bv.style.width=bv.style.paddingLeft="1px";
b.boxModel=bJ.boxModel=bv.offsetWidth===2;if(typeof bv.style.zoom!=="undefined"){bv.style.display="inline"
;bv.style.zoom=1;bJ.inlineBlockNeedsLayout=(bv.offsetWidth===2);bv.style.display="";bv.innerHTML="<div
style='width:4px;'></div>";bJ.shrinkWrapBlocks=(bv.offsetWidth===2)}bv.style.cssText=bS+bR;bv.innerHTML=bP;bU=bv.
firstChild;bV=bU.firstChild;bN=bU.nextSibling.firstChild.firstChild;bO={doesNotAddBorder:(bv.offsetTop===5),
doesAddBorderForTableAndCells:(bN.offsetTop===5)};bv.style.position="fixed";bv.style.top="20px";bO.
fixedPosition=(bv.offsetTop===20||bv.offsetTop===15);bv.style.position=bV.style.top="";bU.style.overflow="hidden";bU.
style.position="relative";bO.subtractsBorderForOverflowNotVisible=(bv.offsetTop===-5);bO.
doesNotIncludeMarginInBodyOffset=(bQ.offsetTop===bL);bQ.removeChild(bM);bv=bM=null;b.extend(bJ,bO)});var a
:\(.*\)|\.[.*/S/,aA=/[A-Z]/g;b.extend({cache:{},uid:0,expando:"jQuery"+(b.fn.jquery+Math.random()).
replace(/\D/g,""),noData:{embed:true,object:"clsid:D27CDB6E-AE6D-11cf-96B8-444553540000",applet:true},hasData:
function(e){e=e.nodeType?b.cache[e[b.expando]]:e[b.expando];return !e&&!S(e)},data:function(bx,bv,bz,
by){if(!b.acceptData(bx)){return}var bG,bA,bD,bE=b.expando,bC=typeof bv==="string",bF=bx.nodeType,bE=bF?
b.cache[bx]:bv=bF?bx[bE]:bx[bE]&&bE,bB=bv==="events";if(!bI||!e[bw]||(!bB&&!by&&!e[bw].data)&&bC&&bz===
L){return}if(!bw){if(bF){bx[bE]=bw+++b.uid;else{bw=bE};if(!e[bw]){e[bw]={};if(!bF){e[bw].toJSON=
b.noop}}if(typeof bv==="object"||typeof bv==="function"){if(bI){e[bw][bI]=bv}else{e[bw].data=
b.extend(e[bw].data,bv)}}bG=bA=e[bw];if(!by){if(!bA.data){bA.data={}}bA=bA.data}if(bz!=="

```



```

L) {bA[b.camelCase(bv)] = bz; if (bB && !bA[bv]) {return bG.events} if (bC) {bD = bA[bv]; if (bD == null) {bD = bA[
b.camelCase(bv)]}} else {bD = bA} return bD; }, removeData: function (bx, bv, by) { if (!b.acceptData(bx)) {return} var bB
, bA, bz, bC = b.expando, bD = bx.nodeType, e = bD ? b.cache : bx, bw = bD ? bx[bC] : bC; if (!e[bw]) {return} if (bv) {bB = by ? e[bw] : e[
bw].data; if (bB) {if (!b.isArray(bv)) {if (bv in bB) {bv = [bv]} else {bv = b.camelCase(bv); if (bv in bB) {bv = [bv]} else {
bv = bv.split(" ")}} for (bA = 0, bz = bv.length; bA < bz; bA++) {delete bB[bv[bA]]} if (! (by ? S :
b.isEmptyObject(bB))) {if (!by) {delete e[bw].data; if (!S(e[bw])) {return} if (
b.support.deleteExpando || !e.setInterval) {delete e[bw]} else {e[bw] = null} if (bD) {if (
b.support.deleteExpando) {delete bx[bC]} else {if (bx.removeAttribute) {bx.removeAttribute(bC)} else {bx[bC] = null
}}}; }, _data: function (bv, e, bw) {return b.data(bv, e, bw, true)} }, acceptData: function (bv) {if (bv.nodeName) {var e =
b.noData[bv.nodeName.toLowerCase()]; if (e) {return ! (e === true || bv.getAttribute("classid") !== e)}} return true}
}); b.fn.extend({data: function (by, bA) {var bB, e, bw, bz = null; if (typeof by === "undefined") {if (this.length) {bz =
b.data(this[0], by); if (this[0].nodeType === 1 && !b._data(this[0], "parsedAttrs")) {e = this[0].attributes; for (var bx =
0, bv = e.length; bx < bv; bx++) {bw = e[bx].name; if (bw.indexOf("data-") === 0) {bw = b.camelCase(bw.substring(5)); a5(
this[0], bw, bz[bw])}} b._data(this[0], "parsedAttrs", true)} return bz} else {if (typeof by === "object") {return this.
each(function () {b.data(this, by)}); } bB = by.split("."); bB[1] = bB[1] ? "." + bB[1] : ""; if (bA ===
L) {bz = this.triggerHandler("getData" + bB[1] + "!", [bB[0]]); if (bz === L && this.length) {bz =
b.data(this[0], by); bz = a5(this[0], by, bz)} return bz === L && bB[1] ? this.data(bB[0]) : bz} else {return this.
each(function () {var bC = b(this), bD = [bB[0], bA]; bC.triggerHandler("setData" + bB[1] + "!", bD);
b.data(this, by, bA); bC.triggerHandler("changeData" + bB[1] + "!", bD)}); }, removeData: function (e) {return this.
each(function () {b.removeData(this, e)}); }); function a5 (bx, bw, by) {if (by === L && bx.nodeType === 1) {var bv =
data -> +bw.replace(aA, "-$1").toLowerCase(); bi = bx.getAttribute(bv); if (typeof bv === "string") {try {bv = bv === "true" ?
true : bv === "false" ? false : bv === "null" ? null : b.isNumeric(bv) ? parseFloat(bv) : aS.test(bv) ?
b.parseJSON(bv) : by} catch (bz) {} b.data(bx, bw, bv)} else {bv = by} return bv} function
S (bv) {for (var e in bv) {if (e === "data" && b.isEmptyObject(bv[e])) {continue} if (e !== "toJSON") {return false}
return true} function bi (by, bx, bA) {var bw = bx + "defer", bv = bx + "queue", e = bx + "mark", bz = b._data (by, bw); if (bz && (bA ===
queue || !b._data (by, bv)) && (bA === "mark" || !b._data (by, e))) {setTimeout (function () {if (!
b._data (by, bv) && !b._data (by, e)) {b.removeData (by, bw, true); b.fire ()}, 0)}; } b.extend ({_mark: function (bv, e)
{if (bv) {e = (e || "fx") + "mark"; b._data (bv, e, (b._data (bv, e) || 0) + 1)}}, _unmark: function (by, bx, bv) {if (bv !== true) {bv =
bx; bx = by; bv = false} if (bx) {bv = bv || "fx"; var e = bv + "mark", bw = bv ? ((b._data (bx, e) || 1) - 1); if (bw) {b._data (bx, e, bw)
} else {b.removeData (bx, e, true); bi (bx, bv, "mark")}}; queue: function (bv, e, bx) {var bw, bv; if (bv) {e = (e || "fx") + "queue";
bw = b._data (bv, e); if (bx) {if (!bv || b.isArray (bx)) {bw = b._data (bv, e, b.makeArray (bx))} else {bw.push (bx)} return
bw} || [], dequeue: function (by, bx) {bx = bx || "fx"; var bv = b.queue (by, bx), bw = bv.shift (), e = {}; if (bw === "inprogress")
{bw = bv.shift ()} if (bw) {if (bx === "fx") {bv.unshift ("inprogress")}} b._data (by, bx + ".run", "e"); bw.call (by, function () {
b.dequeue (by, bx)}, e)} if (!bv.length) {b.removeData (by, bx + "queue " + bx + ".run", true); bi (by, bx, "queue")}});
b.fn.extend ({queue: function (e, bv) {if (typeof e === "string") {bv = e; e = "fx"} if (bv === L) {return
b.queue (this[0], e)} return this.each (function () {var bw = b.queue (this, e, bv); if (e === "fx" && bw[0] !==
inprogress) {b.dequeue (this, e)}}, dequeue: function (e) {return this.each (function () {
b.dequeue (this, e)}); }, delay: function (bv, e) {bv = b.fx ? b.fx.speeds [bv] || bv : bv; e = e || "fx"; return this.queue (e,
function (bx, bv) {var by = setTimeout (bv, bv); bw.stop (function () {clearTimeout (by)}); }, clearQueue: function (e) {
return this.queue (e || "fx", []), promise: function (bD, bw) {if (typeof bD !== "string") {bw = bD; bD =
L} bD = bD || "fx"; var e = b.Deferred (), bv = this, by = bv.length, bB = 1, bz = bD + "defer", bA = bD + "queue", bC = bD + "mark", bx;
function bE () {if (! (--bB)) {e.resolveWith (bv, [bv])} while (by --) {if ((bx = b.data (bv [by], bz,
L, true)) || (b.data (bv [by], bA, L, true) || b.data (bv [by], bC, L, true)) && b.data (bv [by], bz,
b.callbacks ("once memory"), true)) {bB++; bx.add (bE)} bE (); return e.promise ()}}; var aP = /[\n\t\r]/g, aF = /\s+/
, aU = /\r/g, g = /(?:button|input)$/i, D = /(?:button|input|object|select|textarea)/i, l = /<a(?:rea)?$/i, aO = /(?:autofocus|autoplay|async|checked|controls|defer|disabled|hidden|loop|multiple|
open|readonly|required|scoped|selected)$/i, F = b.support.getSetAttribute, be, aY, aF;
b.fn.extend ({attr: function (e, bv) {return b.access (this, e, bv, true, b.attr)}, removeAttr: function (e) {return thi
s.each (function () {b.removeAttr (this, e)}), prop: function (e, bv) {return b.access (this, e, bv, true,
b.prop)}, removeProp: function (e) {e = b.propFix [e] || e; return this.each (function () {try {this [e] =
L} delete this [e]} catch (bv) {}))}, addClass: function (by) {var bA, bB, bv, bx, bz, bB, e; if (
b.isFunction (by)) {return this.each (function (bC) {b (this).addClass (by.call (this, bC, this.className))}})
if (by && typeof by === "string") {bA = by.split (aF); for (bw = 0, bv = this.length; bw < bv; bw++) {bx = this [bw]; if (bx.nodeType ===
1) {if (!bx.className && bA.length === 1) {bx.className = by} else {bz = " " + bx.className + " "; for (bB = 0, e = bA.length; bB < e; bB++) {if (!~bz.indexOf (" " + bA [bB] + " ")) {bz += bA [bB] + " ";} bx.className = b.trim (bz)}}} return this}, removeClass:
function (bz) {var bA, bB, bv, by, bx, bB, e; if (b.isFunction (bz)) {return this.each (function (bC) {
b (this).removeClass (bz.call (this, bC, this.className))}}) if ((bz && typeof bz === "string") || bz ===
L) {bA = (bz || "").split (aF); for (bw = 0, bv = this.length; bw < bv; bw++) {by = this [bw]; if (by.nodeType === 1 && by.className)
{if (bz) {bx = (" " + by.className + " ").replace (aP, " "); for (bB = 0, e = bA.length; bB < e; bB++) {bx = bx.replace (" " + bA [bB] +
" ", " "); by.className = b.trim (bx)} else {by.className = ""}} return this}, toggleClass: function (bx, bv) {var bw =
typeof bx, e = typeof bv === "boolean"; if (b.isFunction (bx)) {return this.each (function (by) {
b (this).toggleClass (bx.call (this, by, this.className, bv), bv)} return this} if (bw ===
"string") {var bA, bB = 0, by = b (this), bB = bv, bC = bx.split (aF); while ((bA = bC [bB++]) && !by.hasClass (bA)) {by [bB] =
addClass: "removeClass"} (bA)} else {if (bw === "undefined" || bw === "boolean") {if (this.className) {
b._data (this, "__className__", this.className)} this.className = this.className || bx === false ? "":
b._data (this, "__className__") || ""}}}, hasClass: function (e) {var bx = " " + e + " ", bw = 0, bv = this.length; for (; bw <
bv; bw++) {if (this [bw].nodeType === 1 && (" " + this [bw].className + " ").replace (aP, " ").indexOf (bx) > -1) {return true}}
return false}, val: function (bx) {var e, bv, by, bw = this[0]; if (!arguments.length) {if (bw) {e =
b.valHooks [bw.nodeName.toLowerCase ()] || b.valHooks [bw.type]; if (e && "get" in e && (bv = e.get (bw, "value")) !==
L) {return bv} bv = bw.value; return typeof bv === "string" ? bv.replace (aU, "") : bv === null ? "" : bv} return bv} by =
b.isFunction (bx); return this.each (function (bA) {var bz = b (this), bB; if (this.nodeType === 1) {return} if (by) {
bB = bx.call (this, bA, bv.val ())} else {bB = bx} if (bB === null) {bB = ""} else {if (typeof bB === "number") {bB += ""} else {if (
b.isArray (bB)) {bB = b.map (bB, function (bC) {return bC === null ? "" : bC + ""}})} e = b.valHooks [this.nodeName.
toLowerCase ()] || b.valHooks [this.type]; if (!e || ("set" in e) || e.set (this, bB, "value") === L) {this.value = bB}}});
b.extend ({valHooks: {option: {get: function (e) {var bv = e.attributes.value; return !bv || bv.specified ? e.value : e.
text}}, select: {get: function (e) {var bA, bv, bz, bx, by = e.selectedIndex, bB = [], bC = e.options, bw = e.type === "select-one"
; if (by < 0) {return null} bv = bw ? by : 0; bz = bw ? by + 1 : bC.length; for (; bv < bz; bv++) {bx = bC [bv]; if (bx.selected && (
b.support.optDisabled ? !bx.disabled : bx.getAttribute ("disabled") === null) && !bx.parentNode.disabled || !
b.nodeName (bx.parentNode, "optgroup")) {bA = b (bx).val (); if (bw) {return bA} bB.push (bA)} if (bw && !bB.length && bC
.length) {return b (bC [by]).val ()} return bB}, set: function (bv, bw) {var e = b.makeArray (bw);
b (bv).find ("option").each (function () {this.selected = b.inArray (b (this).val (), e) > 0}); if (!e.length) {bv.
selectedIndex = -1} return e}}, attrFn: {val: true, css: true, html: true, text: true, data: true, width: true, height: true,
offset: true}, attr: function (bA, bx, bB, bv) {var bw, e, by, bv = bA.nodeType; if (!bA [bA] | bv === 3 || bv === 8 || bv === 2) {return} if (
bz && bx in b.attrFn) {return b (bA) [bx] (bB)} if (typeof bA.getAttribute === "undefined") {return
b.prop (bA, bx, bB)} by = bv !== 1 || !b.isXMLDoc (bA); if (by) {bx = bx.toLowerCase (); e = b.attrHooks [bx] || (ao.test (bx) ?
aY : be)} if (bB === L) {if (bB === null) {b.removeAttr (bA, bx); return} else {if (e && "set" in e && !bB) {return b (bA, bx, bB)
} !== L) {return bw} else {bA.setAttribute (bx, "" + bB); return bB}} else {if (e && "get" in e && by && (bw = e.get (bA, bx)) !==

```

```

null) {return bw} else {bw=bA.getAttribute (bx); return bw===null?L:bw}}; removeAttr: function (bx, bz) {var by, bA, bv,
e, bw=0; if (bz&&bx.nodeType===1) {bA=bz.toLowerCase().split (af); e=bA.length; for (; bw<e; bw++) {bv=bA[bw]; if (bv) {by
=b.propFix [bv] || bv; b.attr (bx, bv, ""); bx.removeAttribute (F?bv:by); if (ao.test (bv) &&bv in bx) {bx [bv]=false}}
}, attrHooks: {type: {set: function (e, bv) {if (g.test (e.nodeName) &&e.parentNode) {b.error ("type property can't be
changed") } else {if (!b.support.radioValue &&bv=== "radio" &&b.nodeName (e, "input")) {var bw=e.value; e.
setAttribute ("type", bv); if (bw) {e.value=bw} return bv}}}; value: {get: function (bv, e) {if (be&&b.nodeName (bv, "button")) {
return be.get (bv, e)} return e in bv?vv.value:null}, set: function (bv, bw, e) {if (be&&b.nodeName (bv, "button")) {return
be.set (bv, bw, e) | bv.value=bw}}}, propFix: {tabindex: "tabIndex", readonly: "readOnly", "for": "htmlFor", "class": "
className", maxLength: "maxLength", cellSpacing: "cellSpacing", cellPadding: "cellPadding", rowspan: "rowSpan", colspan:
"colSpan", useMap: "useMap", frameborder: "frameBorder", contentEditable: "contentEditable"}, prop: function (bz, bx,
bA) {var bw, e, by, bv=bz.nodeType; if (!bz || bv===3 | bv===8 | bv===2) {return} by=bv! =1 || !
b.isXMLDoc (bz); if (by) {bx=b.propFix [bx] || bx; e=b.propHooks [bx] | if (e&&"set" in e && (bw=e.set (bz,
bA, bx)) !=L) {return bw} else {return (bz [bx]=bA)} } else {if (e&&"get" in e && (bw=e.get (bz, bx)) !=null) {return bw}
else {return bz [bx]}}}, propHooks: {tabIndex: {get: function (bv) {var e=bv.getAttributeNode ("tabindex"); return e&&
e.specified?parseInt (e.value, 10) : D.test (bv.nodeName) | | L.test (bv.nodeName) &&bv.href??:
L}}}; b.attrHooks.tabindex=b.propHooks.tabIndex; aY={get: function (bv, e) {var bx, bw=
b.prop (bv, e); return bw===true | | typeof bw!="boolean" && (bx=bv.getAttributeNode (e)) &&bx.nodeValue!==false?e.
toLowerCase () : L; set: function (bv, bx, e) {var bw; if (bx===false) {b.removeAttr (bv, e) } else {bw=
b.propFix [e] | | e; if (bw in bv) {bv [bw]=true} bv.setAttribute (e, e.toLowerCase ()) } return e}; if (!F) {aF={name:
true, id: true}; be=b.valHooks.button={get: function (bw, bv) {var e; e=bw.getAttributeNode (bv); return e&&(aF [bv]?e.
nodeValue!="": e.specified?e.nodeValue:L); set: function (bw, bx, bv) {var e=bw.getAttributeNode (bv); if (!e) {e=av.
createAttribute (bv); bw.setAttributeNode (e) } return (e.nodeValue=bx+"")}; b.attrHooks.tabindex.set=be.set;
b.each (["width", "height", "function (bv, e) {b.attrHooks [e]=b.extend (b.attrHooks [e], {set: function (bw, bx) {if
(bx=== "auto") {bw.setAttribute (e, "auto"); return bx}}}; b.attrHooks.contentEditable={get: be.get, set: function (bv,
bw, e) {if (bw=== "false") {be.set (bv, bw, e) } } if (!b.support.hrefNormalized) {b.each (["href", "src", "width",
"height"], function (bv, e) {b.attrHooks [e]=b.extend (b.attrHooks [e], {get: function (bx) {var bw=bx.getAttribute
(e, 2); return bw===null?L:bw}})} } if (!b.support.style) {b.attrHooks.style={get: function (bx) {return e.style.
cssText.toLowerCase () | | L; set: function (e, bv) {return (e.style.cssText=" "+bv) } } if (!b.support.optSelected) {
b.propHooks.selected=b.extend (b.propHooks.selected, {get: function (bv) {var e=bv.parentNode; if (e) {e.selecte
dIndex; if (e.parentNode) {e.parentNode.selectedIndex} return null}}); if (!b.support.enctype) {
b.propFix.enctype="encoding"} if (!b.support.checkOn) {b.each (["radio", "checkbox"], function () {
b.valHooks [this]={get: function (e) {return e.getAttribute ("value") ===null?"on": e.value}})} }
b.each (["radio", "checkbox"], function () {b.valHooks [this]=b.extend (b.valHooks [this], {set: function (e, bv) {if (b
.isArray (bv)) {return (e.checked=b.inArray (b (e).val (), bv) >=0) } } }); var bd=/^ (?:text|input|select)$/
i, n=/^ ([^\.]*)? (?:\ ((+))? $/, J=/\bhover (\ . S+)? \b/, aO=/^key/, bF=/^ (?:mouse|contextmenu) |click/,
T=/^ (? :focus|focusout|blur) $/, U=/^ ([w*])? (?:# ([w-]+))? (?:\ ([w-]+))? (?:\ ([w-]+))? $/, Y=function (e) {var bv=U.exec
(e); if (bv) {bv [1]= (bv [1] || "").toLowerCase (); bv [3]=bv [3] &&new RegExp (" (? : ^ [\\s] "+bv [3] +" (? : \\s | $) ") } return bv}
, j=function (bw, e) {var bv=bw.attributes | | {}; return (!e [1] | | bv.nodeName.toLowerCase () ===e [1]) && (!e [2] | | (bv.id
| | {}).value===e [2]) && (!e [3] | | e [3].test ((bv [1] | | "class" | | {}).value)) } }, bt=function (e) {return
b.event.special.hover?e.e.replace (J, "mouseenter$1 mouseleave$1"); b.event={add: function (bx, bC, bJ, bA, by) {
var bD, bB, bK, bI, bH, bF, e, bG, bv, bz, bw, bE; if (bx.nodeType===3 | | bx.nodeType===8 | | bC | | bJ | | (bD=
b._data (bx))) {return} if (bJ.handler) {bv=bJ; bJ=b.handler} if (!bJ.guid) {bJ.guid=bD.events; if (!bK
) {bD.events=bK={}; bB=bD.handle; if (!bB) {bD.handle=bB=function (bL) {return typeof bL!="undefined" && (!bL |
| b.event.triggered)! =bL.type} b.event.dispatch.apply (bB.elem, arguments) : L; bB.elem=bx} bC=
b.trim (bt (bC)).split (" "); for (bI=0; bI<bC.length; bI++) {bH=n.exec (bC [bI] | | {}); bF=bH [2] | | {} ; bE=bH [3] | | "").split (
" ").sort (); bE=b.event.special [bF] | | {} ; bF=(by?bE.delegateType:bE.bindType) | | bF; bE=
b.event.special [bF] | | {} ; bG=b.extend ({type: bF, origType: bH [1], data: bA, handler: bJ, guid: bJ, selector: by,
quick: Y (by) }, namespace: e.join (" "), bv, bw=bK [bF] | | {}); if (!bv) {bw=bK [bF]=[]; bw.delegateCount=0; if (!bE.setup | | bE.
setup.call (bx, bA, e, bB) ===false) {if (bx.addEventListener) {bx.addEventListener (bF, bB, false) } else {if (bx.attachEvent
) {bx.attachEvent ("on"+bF, bB) } } if (bE.add) {bE.add.call (bx, bG); if (!bG.handler.guid) {bG.handler.guid=bJ.guid}
if (by) {bw.splice (bw.delegateCount++, 0, bG) } else {bw.push (bG) } b.event.global [bF]=bx>null, global: {} ,
remove: function (bJ, bE, bv, bH, bB) {var bI=b.hasData (bJ) &&b._data (bJ), bF, bx, bz, bL, bC, bA, bG, bw, by, bK, bD, e; if (!bI | | (
bw=bI.events)) {return} bE=b.trim (bt (bE | | "")).split (" "); for (bF=0; bF<bE.length; bF++) {bx=
n.exec (bE [bF] | | {}); bz=bL=bx [1]; bC=bx [2]; if (!bz) {for (bz in bw) {b.event.remove (bJ, bz+bE [bF], bv, bH, true) }
continue} by=b.event.special [bz] | | {} ; bz=(bH?by.delegateType:by.bindType) | | bz; bD=bw [bz] | | {} ; bA=bD.length; bC=bC?
new RegExp (" (^ [\\s] "+bC.split (" ").sort ().join (" \\ . (?: . * \\ .)? ") + "\\ . | $) ") : null; for (bG=0; bG<bD.length; bG++) {e
=bD [bG]; if ((bB | | bL===e.origType) && (!bv | | bv.guid===e.guid) && (!bC | | bC.test (e.namespace) && (!bH | | bH===e.
selector | | bH=== "*" &&e.selector)) {bD.splice (bG--, 1); if (e.selector) {bD.delegateCount--} if (by.remove) {by.remove.call (
bJ, e) } } if (bD.length===0 &&bA! =bD.length) {if (!by.teardown | | by.teardown.call (bJ, bC) ===false) {
b.removeEvent (bJ, bz, bI.handle) } delete bw [bz] } } if (bI.isEmptyObject (bw)) {bK=bI.handle; if (bK) {bK.elem=null}
b.removeData (bJ, ["events", "handle"], true) }, customEvent: {getData: true, setData: true, changedData: true}
}, trigger: function (bv, bD, bA, bJ) {if (bA&&(bA.nodeType===3 | | bA.nodeType===8)) {return} var bG=bv.type | | bv, bx=[], e,
bw, bC, bH, bz, by, bF, bE, bB, bI; if (T.test (bG+b.event.triggered)) {return} if (bG.indexOf (" ") >=0) {bG=bG.slice (0, -1
); bw=true} if (bG.indexOf (".") >=0) {bx=bG.split (" "); bG=bx.shift (); bx.sort () } if (!bA | |
b.event.customEvent [bG]) &&!b.event.global [bG]) {return} bv=typeof bv=="object"?bv |
b.expand?bv:new b.Event (bG, bv) : new b.Event (bG); bv.type=bG; bv.isTrigger=true; bv.exclusive=bw; bv.
namespace=bx.join (" "); bv.namespace_re=bv.namespace?new RegExp (" (^ [\\s] "+bx.join (" \\ . (?: . * \\ .)? ") + "\\ . | $) ") : null;
by=bG.indexOf (":") <0?"on"+bG:""; if (!bA) {e=b.cache; for (bC in e) {if (e [bC].events &&e [bC].events [bG]) {
b.event.trigger (bv, bD, e [bC].handle.elem, true) } } return} bv.result=L; if (!bv.target) {bv.target=bA} bD!=null
?b.makeArray (bD) : []; bD.unshift (bv); bF=b.event.special [bG] | | {} ; if (bF.trigger &&bF.trigger.apply (bA, bD) ===
false) {return} bB=[bA, bF.bindType | | bG]; if (!bJ && !bF.noBubble && !b.isWindow (bA)) {bI=bF.delegateType | | bG; bH=
T.test (bI+bG)?bA.parentNode: bz=null; for (; bH && bH=bH.parentNode) {bB.push ([bH, bI]); bz=bH} if (bz && bz===bA.
ownerDocument) {bB.push ([bz.defaultView | | bz.parentNode | | bb, bI]) } for (bC=0; bC<bB.length && !bv.
isPropagationStopped () ; bC++) {bH=bB [bC] [0]; bv.type=bB [bC] [1]; bE=(b._data (bH, "events") | | {}) [bv.type] &&
b._data (bH, "handle"); if (bE) {bE.apply (bH, bD) } bE=by && bH [by]; if (bE && bE.acceptData (bH) && bE.apply (bH, bD) ===
false) {bv.preventDefault () } } bv.type=bG; if (!bJ && !bv.isDefaultPrevented ()) {if (!bF._default | | bF._default.apply (bA.
ownerDocument, bD) ===false) && ! (bG=== "click" && bB.nodeName (bA, "a") && bB.acceptData (bA)) } {if (by && bA [bG] && ((bG! = "
focus" && bG! = "blur") | | bv.target.offsetWidth! =0) &&!b.isWindow (bA)) {bz=bA [by]; if (bz) {bA [by]=null}
b.event.triggered=bG; bA [bG] (); b.event.triggered=L; if (bz) {bA [by]=bz} } } return bv.result; dispatch:
function (e) {e=b.event.fix (e | | bb.event); var bz=(b._data (this, "events") | | {}) [e.type] | | {} ; bA=bz.delegateCount; bG
=[] .slice.call (arguments, 0), by=!e.exclusive && !e.namespace, bH=[], bC, bB, bK, bx, bF, bE, bv, bD, bI, bw, bJ; bG [0]=e; e.
delegateTarget=this; if (bA && !e.target.disabled && ! (e.button && e.type=== "click")) {bx=b (this); bx.context=this.
ownerDocument | | this; for (bK=e.target; bK! =this; bK=bK.parentNode) {this [bE]=[]; bx [0]=bK; for (bC=0; bC<bA; bC++) {
bI=bz [bC]; bw=bI.selector; if (bE [bw] ===L) {bE [bw]=bI.quick?j (bK, bI.quick) : bx.is (bw) }
if (bE [bw]) {bD.push (bI) } } if (bD.length) {bH.push ({elem: bK, matches: bD}) } } if (bz.length > bA) {bH.push ({elem: this
, matches: bz.slice (bA)) } } for (bC=0; bC<bH.length && !e.isPropagationStopped () ; bC++) {bv=bH [bC]; e.currentTarget=bv.
elem; for (bB=0; bB<bv.matches.length && !e.isImmediatePropagationStopped () ; bB++) {bI=bv.matches [bB]; if (by | | !e.

```



```

namespace&&!bI.namespace)||e.namespace_re&&e.namespace_re.test(bI.namespace))){e.data=bI.data;e.handleObj=bI;bf=
=((bI.event.special[bI.origType]||{}).handle||bI.handler).apply(bv.elem,bG);if(bf!=
L){e.preventDefault;if(bf===false){e.stopPropagation();}return e.result},props:"
attrChange attrName relatedNode srcElement altKey bubbles cancelable ctrlKey currentTarget eventPhase metaKey
relatedTarget shiftKey target timeStamp view which".split(" "),fixHooks:{},keyHooks:{props:"char charCode key
keyCode".split(" "),filter:function(bv,e){if(bv.which==null){bv.which=e.charCode!=null?e.charCode:e.keyCode}
return bv}},mouseHooks:{props:"button buttons clientX clientY fromElement offsetX offsetY pageX pageY screenX
screenY toElement".split(" "),filter:function(bx,bw){var by,bz,e,bv=bw.button,ba=bw.fromElement;if(bx.pageX==
null&&bw.clientX!=null){by=bx.target.ownerDocument||av;e=by.body;bx.pageX=bw.clientX+(bz
&&bz.scrollLeft||e&&e.scrollLeft||0)-(bz&&bz.clientLeft||e&&e.clientLeft||0);bx.pageY=bw.clientY+(bz&&bz.
scrollTop||e&&e.scrollTop||0)-(bz&&bz.clientTop||e&&e.clientTop||0)}if(!bx.relatedTarget&&ba){bx.relatedTarget=
ba===bx.target?bw.toElement:ba;if(!bx.which&&bv!=L){bx.which=(bv&1?1:(bv&2?3:(bv&4?2:0))}return bx}},fix:
function(bw){if(bw[b.expando]){return bw}var bv,bz,e=bw,bx=b.event.fixHooks[bw.type]||{};by=bx.props?this.
props.concat(bx.props):this.props;bw=b.Event(e);for(bv=by.length;bv;){bz=by[--bv];bw[bz]=e[bz]}if(!bw.
target){bw.target=e.srcElement||av}if(bw.target.nodeType===3){bw.target=bw.target.parentNode}if(bw.metaKey===
L){bw.metaKey=bw.ctrlKey}return bx.filter?bx.filter(bw,e):bw},special:{ready:{setup:
b.bindReady},load:{noBubble:true},focus:{delegateType:"focus"},blur:{delegateType:"mouseout"},
beforeunload:{setup:function(bw,bv,e){if(b.isWindow(this)){this.onbeforeunload=e};teardown:function(bv,e){if(this.
onbeforeunload===e){this.onbeforeunload=null}}}},simulate:function(bw,by,bx,bv){var bz=
b.extend(new b.Event(),bx,{type:bw,isSimulated:true,originalEvent:{}});if(bv){
b.event.trigger(bz,null,by)}else{b.event.dispatch.call(by,bz)}if(bz.isDefaultPrevented()){bx.
preventDefault();}b.event.handle=b.event.dispatch;b.removeEvent=av.removeEventListener?function(bv,e,bw){if(bv.
removeEventListener){bv.removeEventListener(e,bw,false)}:function(bv,e,bw){if(bv.detachEvent){bv.detachEvent("
on"+e,bw)};b.Event=Function(bv,e){if(!this instanceof b.Event){return new b.Event(bv,e)}if(bv&&bv.type)
{this.originalEvent=bv;this.type=bv.type;this.isDefaultPrevented=(bv.defaultPrevented||bv.returnValue===
false)||bv.getPreventDefault&&bv.getPreventDefault();}else{this.type=bv;if(e){b.extend(this,e)}this.
timeStamp=bv&&bv.timeStamp||b.now();}this[b.expando]=true;function bk(){return false}function
i(){return true}b.Event.prototype={preventDefault:function(){this.isDefaultPrevented=
i;var bv=this.originalEvent;if(!bv){return}if(bv.preventDefault){bv.preventDefault()}else{bv.returnValue=
false}},stopPropagation:function(){this.isPropagationStopped=i;var bv=this.originalEvent;if(!bv){return}if(
bv.stopPropagation){bv.stopPropagation()}bv.cancelBubble=true,stopImmediatePropagation:function(){this.
isImmediatePropagationStopped=i;this.stopPropagation();isDefaultPrevented=bk,isPropagationStopped=bk,
isImmediatePropagationStopped=bk};b.each({mouseenter:"mouseover",mouseleave:"mouseout"},function(bv,e){
b.event.special[bv]={delegateType:e,bindType:e,handle:function(bz){var bB=this,bA=bz.relatedTarget,by=bz.
handleObj,bw=by.selector,bx;if(!bA||(!bA===bB&&!bB.contains(bB,bA))){bz.type=by.origType;bx=by.handler.apply(
this,arguments);bz.type=e}return bx}});if(!b.support.submitBubbles){b.event.special.submit={setup:function(
){if(b.nodeName(this,"form")){return false}b.event.add(this,"click._submit keypress._submit",function(bx)
{var bw=bx.target,bv=b.nodeName(bw,"input")||b.nodeName(bw,"button")?bw.form:L;if(bv&&!bv.
_submit_attached){bv.event.add(bv,"submit._submit",function(e){if(this.parentNode&&e.isTrigger){b.event.simulate("submit"
this.parentNode,e,true)};bv._submit_attached=true}}),teardown:function(){if(b.nodeName(this,"form")){
return false}b.event.remove(this,"._submit")}}if(!b.support.changeBubbles){b.event.special.change={setup:
function(){if(bd.test(this.nodeName)){if(this.type=="checkbox"||this.type=="radio"){
b.event.add(this,"propertychange._change",function(e){if(e.originalEvent.propertyName=="checked"){this._j
ust_changed=true}});b.event.add(this,"click._change",function(e){if(this._just_changed&&!e.isTrigger){this.
_just_changed=false;b.event.simulate("change",this,e,true)}return false}b.event.add(this,
beforeactivate._change",function(bw){var bv=bw.target;if(bd.test(bv.nodeName)&&bv._change_attached){b.event.add(bv,
change._change",function(e){if(this.parentNode&&!e.isSimulated&&!e.isTrigger){b.event.simulate("change",this.pare
ntNode,e,true)};bv._change_attached=true}})},handle:function(bw){var e=bv.target;if(this!==e||bv.
isSimulated||bv.isTrigger||e.type!="radio"&&e.type!="checkbox")){return bv.handleObj.handler.apply(this,arguments)
}},teardown:function(){b.event.remove(this,"._change");return bd.test(this.nodeName)});if(!
b.support.focusinBubbles){b.each({focus:"focusin",blur:"focusout"},function(bx,e){var bv=0,bw=function(by
){b.event.simulate(e,by.target,b.event.fix(by),true)};b.event.special[e]={setup:function(){if(bv===0){
av.addEventListener(bx,bw,true)};teardown:function(){if(--bv===0){av.removeEventListener(bx,bw,true)}}}})
b.fn.extend({on:function(bw,e,bz,by,bv){var bA,bx;if(typeof bw=="object"){if(typeof e!="string"){bz=e;e=
L}for(bx in bw){this.on(bw,e,bz,bw[bx],bv)}return this}if(bz==null&&by==null){by=e;bz=e=L}else{if(by==null){
if(typeof e=="string"){by=bz;bz=L}else{by=bz;bz=e=L}}if(by===false){by=bk}else{if(!by){return this}if(b
v===1){ba=by;by=function(bB){b().off(bB);return bA.apply(this,arguments)};by.guid=bA.guid||bA.guid+b.guid+
)}return this.each(function(){b.event.add(this,bw,by,bz,e)}),one:function(bv,e,bx,bw){return this.on.call(t
his,bv,e,bx,bw,1)},off:function(bw,e,by){if(bw&&bw.preventDefault&&bw.handleObj){var bv=bw.handleObj;b(bw.de
legateTarget).off(bv.namespace?bv.type+"."+bv.namespace:bv.type,bv.selector,bv.handler);return this}if(
typeof bw=="object"){for(var bx in bw){this.off(bx,e,bw[bx])}return this}if(e===false||typeof e=="function"){by
=e;e=L}if(by===false){by=bk}return this.each(function(){b.event.remove(this,bw,by,e)}),bind:function(
e,bw,bv){return this.on(e,null,bw,bv)},unbind:function(e,bv){return this.off(e,null,bv)},live:function(e,bw,
bv){b(this.context).on(e,this.selector,bw,bv);return this},die:function(e,bv){b(this.context).off(e,this.
selector||"*",bv);return this},delegate:function(e,bv,bx,bw){return this.on(e,bv,bx,bw)},undelegate:function(
e,bv,bw){return arguments.length==1?this.off(e,"*"):this.off(bv,e,bw)},trigger:function(e,bv){return this.
each(function(){b.event.trigger(e,bv,this)}),triggerHandler:function(e,bv){if(this[0]){return
b.event.trigger(e,bv,this[0],true)};toggle:function(bx){var bv=arguments,e=bx.guid|
b.guid++,bw=0,by=function(bz){var bA=(b._data(this,"lastToggle"+bx.guid)|0)%bw;
b._data(this,"lastToggle"+bx.guid,bA+1);bz.preventDefault();return bv[bA].apply(this,arguments)||false};by
.guid=e;while(bw<bv.length){bv[bw++].guid=e}return this.click(bv)},hover:function(e,bv){return this.
mouseenter(e).mouseleave(bv|e)};b.each({"blur focus focusin focusout load resize scroll unload click dblclick
mousedown mouseup mousemove mouseover mouseout mouseenter mouseleave change select submit keydown keypress
keyup error contextmenu".split(" "),function(bv,e){b.fn[e]=function(bx,bw){return this}if(bw==null){bw=bx;bx=null}return
arguments.length>0?this.on(e,null,bx,bw):this.trigger(e)};if(b.attrFn){b.attrFn[e]=true}if(aO.test(e)){
b.event.fixHooks[e]=b.event.keyHooks}if(bf.test(e)){b.event.fixHooks[e]=b.event.mouseHooks}});
(function(){var bh=/((?:\((?:\(|\)|[\^()]+\)|\[(?:\]|\\[\]\[\]\]*\)|'["]*'|"'"')*\|\/g,bO=/
>+~,(\[\]\+)|\+~+)(\s*,\s*)?(?:\.[\]|n
)+/g,bC="sizzleCache"+(Math.random()+"").replace(".", "");bI=0,bL=Object.prototype.toString,bB=false,bA=true,bK=/\|\/g,bO=
0);var by=function(bV,e,bY,bZ){bY=bY||[];e=e||av;var
b1=e;if(e.nodeType===1&&e.nodeType!==9){return[]}if(!bV||typeof bV!="string"){return bY}var
bS,b3,b6,bR,b2,b5,b4,bX,bU=true,bT=by.isXML(e),bW=[],b0=bV;do{bH.exec(" ");
bS=bH.exec(b0);if(bS){b0=bS[3];bW.push(bS[1]);if(bS[2]){bR=bS[3];break}}while(bS);if(bW.length>1&&bD.exec(b
V)){if(bW.length===2&&bE.relative[bW[0]]){b3=bM(bW[0]+bW[1],e,bZ)}else{b3=bE.relative[bW[0]]?e:by(bW.shift(
)),e}while(bW.length){bv=bW.shift();if(bE.relative[bv]){bv+=bv.W.length}b3=bM(bv,b3,bZ)};else{if(!bZ&&bW.l
ngth>1&&e.nodeType===9&&!bT&&bE.match.ID.test(bW[0])&&!bE.match.ID.test(bW[bW.length-1])}{b2=by.find(bW.shif

```

```

t(),e,bT);e=b2.expr?by.filter(b2.expr,b2.set)[0]:b2.set[0]if(e){b2=bZ?{expr:bW.pop(),set:bF(bZ)}:by.find(bW
.pop(),bW.length==1&&(bW[0]===~"|bW[0]===~"+)}&&.parentNode?.parentNode:e,bT);b3=b2.expr?by.filter(b2.ex
pr,b2.set):b2.set;if(bW.length>0){b6=bF(b3)}else{bU=false}while(bW.length){b5=bW.pop();b4=b5;if(!bE.relative
[b5]){b5=""}else{b4=bW.pop()}if(b4==null){b4=e}bE.relative[b5](b6,b4,bT)}else{b6=bW[]}if(!b6){b6=b3}if(!b6){by.error
Array"}if(!bU){bY.push.apply(bY,b6)}else{if(e&&.nodeType===
1){for(bX=0;b6[bX]!-null;bX++){if(b6[bX]&&(b6[bX]===true||b6[bX].nodeType===1&&by.contains(e,b6[bX]))){bY.pu
sh(b3[bX])}}else{for(bX=0;b6[bX]!-null;bX++){if(b6[bX]&&b6[bX].nodeType===1){bY.push(b3[bX])}}}}else{bF(b6,bY)}if(bR
bY);by.uniqueSort=function(bR){if(bJ){bB=bA;bR.sort(bJ);if(bB){for(var e=1;e<bR.length;e++){if(bR[e]===bR[e-1]){bR.splice
bR};by.matches=function(e,bR){return by(e,null,null,bR)};by.matchesSelector=function(e,bR){return
by(bR,null,null,[e]).length>0};by.find=function(bX,e,bY){var
bW,bS,bU,bT,bV,bR;if(!bX){return[]}for(bS=0,bU=bE.order.length;bS<bU;bS++){bV=bE.order[bS];if((bT=bE.leftMatch[bV].exec
")){bT[1]
)=(bT[1]||"").replace(bK,"");bW=bE.find[bV](bT,e,bY);if(bW!=null){bX=bX.replace(bE.match[bV],"");break}}}}if(!bW){bW=typ
e.getElementsByTagName!="undefined"?e.getElementsByTagName("*"):[]return(set;bW,expr;bX)};by.filter=function(b1,b0,b
bW,e,bZ,b6,b3,bR,bT,bV,b2,bS=b1,b5=[],bY=b0,bX=b0&&b0[0]&&by.isXML(b0[0]);while(b1&&b0.length){for(bZ in
bE.filter){if((bW=bE.leftMatch[bZ].exec(b1))!=null&&bW[2]){bR=bE.filter[bZ];bT=bW[1];e=false;bW.splice(1,1);if(bT.subst
")}{continue}if(bY===b5){b5=[]}if(bE.preFilter
[bZ]){bW=bE.preFilter[bZ](bW,bY,b4,b5,bU,bX);if(!bW){e=b6=true}else{if(bW===true){continue}}if(bW){for(bV=0
;b3=bY[bV])!=null;bV++){if(b3){b6=bR(b3,bW,bV,bY);b2=bU^b6;if(b4&&b6!=null){if(b2){e=true}else{bY[bV]=false
}}else{if(b2){b5.push(b3);e=true}}}}if(b6!=l){if(!b4){bY=b5};b1=b1.replace(bE.match[bZ],"");if(!e){return[]}}break}}if
Error{"Syntax error, unrecognized expression: "+e};var bw=by.getText=function(bU){var
bS,bT,e=bU.nodeType,bR="";if(e){if(e===1||e===9){if(typeof bU.textContent==="string"){return bU.textContent}else{if(ty
bU.innerHTML==="string"){return
bU.innerText.replace(bO,"")}}else{for(bU=bU.firstChild;bU=bU.nextSibling){bR+=bw(bU)}}}}else{if(e===3||e===4){return
bU.nodeValue}}else{for(bS=0;(bT=bU[bS]);bS++){if(bT.nodeType===8){bR+=bw(bT)}}return bR};var bE=by.selectors={order:
.}),CLASS:/^\.(?:[\w\u00c0-\uFFFF-]|\\.|.+)/,NAME:/^name=[^']*|(?:[\w\u00c0-\uFFFF-]|\\.|.+)/,ATTR:/^[\w\u00c0-\uFFFF-]|\\.|.*/s
(?:[\w\u00c0-\uFFFF-]|\\.|.)+/s*(?:\{\s*(?:\@?=\s*(?:\{|\}|\.|.)*\}\s*(?:\|[\w\u00c0-\uFFFF-]|\\.|.)*\})\s*/
/,TAG:/^([\w\u00c0-\uFFFF-]|\\.|.)/,CHILD:/(only|nth|last|first)-child(?:\(\s*(even|odd|(?:[+\-]?\d
+|\(\s*(?:[+\-]?\d*)\n\s*(?:[+\-]?\s*(d+)?\s*)\)?\s*\))\s*(?:\|([\w\u00c0-\uFFFF-]|\\.|.)*\})\s*\)\s*(?:\(|\s*(\{|\}|\.|.)*\})\s*\)?/
),leftMatch:({}),attrMap:{"class":"className","for":"htmlFor"},attrHandle:{href:function(e){return e.getAttribute("href:
")},type:function(e){return e.getAttribute("type")}},relative:{"+":function(bW,bR){var bT=typeof bR==="string"
,bV=bT&&!bQ.test(bR)||bU&&!bV;if(bV){bR=bR.toLowerCase();for(var
bS=0,e=bW.length,bU;bS<e;bS++){if((bU=bW[bS])){while
((bU=bU.previousSibling)&&bU.nodeType!==1){bW[bS]=bX||bU&&bU.nodeName.toLowerCase()===bR?bU||false:bU===bR}}if(bX){by.
,bS=0,e=bW.length;if(bU&&!bQ.test(bR)){bR=bR.toLowerCase();for(bS=0;bS<e;bS++){bV=bW[bS];if(bV){var
bT=bV.parentNode;bW[bS]=bT.nodeName.toLowerCase()
LowerCase()===bR?bT:false}}else{for(bS=0;bS++){bV=bW[bS];if(bV){bW[bS]=bU?bV.parentNode:bV.parentNode===bR}}if(bU){by
&&!bQ.test(bR)}(bR=bR.toLowerCase()){bU=bR;e=bV}("parentNode",bR,bS,bT,bU,bV)","~":function(bT,bR,bV){var
bU,bS=bT+1,e=bN;if(typeof bR==="string"&&!bQ.test(bR)){bR=bR.toLowerCase();bU=bR;e=bV}("previousSibling"
,bR,bS,bT,bU,bV)}},find:{ID:function(bR,bS,bT){if(typeof bS.getElementById==="undefined"&&!bT){var
e=bS.getElementById(bR[1]);return e&&.parentNode[e]:[]},NAME:function(bS,bV){if(typeof bV.getElementsByName!="unde
bR=[],bU=bV.getElementsByName(bS[1]);for(var bT=0,e=bU.length;bT<e;bT++){if(bU[bT].getAttribute("name"
)===bS[1]){bR.push(bU[bT])}}return bR.length===0?null:bR};TAG:function(e,bR){if(typeof bR.getElementsByTagName!="
undefined"){return bR.getElementsByTagName(e[1])};preFilter:{CLASS:function(bT,bR,bS,e,bW,bX){bT=" "
+bT[1].replace(bK,"")+";if(bX){return bT}for(var bU=0,bV;(bV=bR[bU])!=null;bU++){if(bV){if(bW^(bV.className&&"
"+bV.className+" ").replace(/\t|\n|\r/g," "
).indexOf(bT)>0)}(if(!bS){e.push(bV)}else{if(bS){bR[bU]=false}}}}return false},ID:function(e){return e[1].replace(bK,
).toLowerCase();CHILD:function(e){if(e[1]===nth)"(if(!e[2]){by.error(e[0]);e[2]=e[2].replace(/\/+|\s/g,""
);var bR=/(?-)(\d*)(?:[+\-]?\d*)?/.exec(e[2])===even"&&"2n"||e[2]===odd"&&"2n+1"||!/\/.test(e[2])&&"0
n"+e[2]||e[2]);e[2]=(bR[1]+(bR[2]||1))-0;e[3]=bR[3]-0}else{if(e[2]){by.error(e[0]);e[0]=bI+};return
e},ATTR:function(bU,bR,bS,e,bV,bW){var bT=bU[1]=bU[1].replace(bK,""
);if(!bW&&bE.attrMap[bT]){bU[1]=bE.attrMap[bT]}bU[4]=(bU[4]||bU[5]||"").replace(bK,"");if(bU[2]===~)=){bU[4]=" "+bU[4]
bU},PSEUDO:function(bU,bR,bS,e,bV){if(bU[1]===not)"(if((bH.exec(bU[3])||"").length>1|^\/\w
\/.test(bU[3])){bU[3]=by(bU[3],null,null,bR)}else{var bT=by.filter(bU[3],bR,bS,true^bV);if(!bS){e.push.apply(e,bT)}return
false}else{if(bE.match.POS.test(bU[0])||bE.match.CHILD.test(bU[0])){return true}}return bU,POS:function(e){e.unshift
e}),filters:{enabled:function(e){return e.disabled===false&&e.type!=="hidden"},disabled:function(e){return
e.disabled===true},checked:function(e){return
e.checked===true},selected:function(e){if(e.parentNode){e.parentNode.selectedIndex}return e.selected===true},parent:fur
!e.firstChild},empty:function(e){return !e.firstChild},has:function(bS,bR,e){return !by(e[3],bS).length},header:funct
/i).test(e.nodeName)},text:function(bS){var e=bS.getAttribute("type"),bR=bS.type;return bS.nodeName.toLowerCase()===
input"&&text"===bR&&(e===bR||e===null)},radio:function(e){return e.nodeName.toLowerCase()===input"&&radio
===e.type},checkbox:function(e){return e.nodeName.toLowerCase()===input"&&checkbox"
===e.type},file:function(e){return e.nodeName.toLowerCase()===input"&&file"===e.type},password:function(e){return
e.nodeName.toLowerCase()===input"&&password"===e.type},submit:function(bR){var e=bR.nodeName.toLowerCase();return(e=
input"||e===button)"&&submit"===bR.type},image:function(e){return e.nodeName.toLowerCase()===input"&&
image"===e.type},reset:function(bR){var e=bR.nodeName.toLowerCase();return(e===input"||e===button)"&&reset"
===bR.type},button:function(bR){var e=bR.nodeName.toLowerCase();return e===input"&&button"===bR.type||e===
button"}},input:function(e){return (/input|select|textarea|button/i).test(e.nodeName)},focus:function(e){return
e===e.ownerDocument.activeElement}},setFilters:{first:function(bR,e){return
e===0},last:function(bS,bR,e,bT){return bR===bT.length-1},even:function(bR,e){return e%2===0},odd:function(bR,e){return
e%2===1},lt:function(bS,bR,e){return bR<e[3]-0},gt:function(bS,bR,e){return bR>e[3]-0},nth:function(bS,bR,e){return
e[3]-0===bR},eq:function(bS,bR,e){return e[3]-0===bR}}},filter:{PSEUDO:function(bS,bX,bW,bY){var
e=bX[1],bR=bE.filters[e];if(bR){return bR(bS,bW,bX,bY)}else{if(e==="contains"){return(bS.textContent||bS.innerText||bw
).indexOf(bX[3])>=0}else{if(e===not)"(var bT=bW[3];for(var
bV=0,bU=bT.length;bV<bU;bV++){if(bT[bV]===bX){return true}}return false}else{if(bE.error(e))}}},CHILD:function(bS,bU){var
bT,b0,bW,bZ,e,bV,bY,bX=bU[1],bR=bS;switch(bX){case"only":case"first":while((bR=bR.previousSibling)){if(bR.nodeType===1
)}(return true)bR=bS;case"last":while((bR=bR.nextSibling)){if(bR.nodeType===1){return false}}return
true;case"nth":bT=bU[2];b0=bU[3];if(bT===1&&b0===0){return
true}bW=bU[0];bZ=bS.parentNode;if(bZ&&(bZ[bC]!-bW)!|bS.no
deIndex){bV=0;for(bR=bZ.firstChild;bR=bR.nextSibling){if(bR.nodeType===1){bR.nodeTypeIndex++bV}}bZ[bC]=bW}bY=bS.nodeIno
bR.nodeType===1&&bR.getAttribute("id")===e),TAG:function(bR,e){return(e===*
&&bR.nodeType===1)||bR.nodeName&&bR.nodeName.toLowerCase()===e),CLASS:function(bR,e){return(" "+(bR.className||bR.getI
).indexOf(e)>-1),ATTR:function(bV,bT){var
bS=bT[1],e=by.attr?by.attr(bV,bS):bE.attrHandle[bS]?bE.attrHandle[bS](bV):bV[bS]!-null?bV[bS]:bV.getAttribute(bS),bW=e-
!bU&&by.attr?e!-null:bU===~=""?bW===bR:bU===~="*"?bW.indexOf(bR)>0:bU===~="~"?(" "+bW+" "
).indexOf(bR)>0:!(bR?bW&&e!-false:bU===~="!"?bW!-bR:bU===~="^"?bW.indexOf(bR)===0:bU===~="$"?bW.substr(bW.length-bR.length

```

```

?bW===bR||bW.substr(0,bR.length+1)===bR+"-":false),POS:function(bU,bR,bS,bV){var
e=bR[2],bT=bE.setFilters[e];if(bT){return bT(bU,bS,bR,bV)}};var bD=bE.match.POS,bx=function(bR,e){return"\\"+(e-0+1)
bE.match){bE.match[bz]=new RegExp(bE.match[bz].source+/(?!\[*\])(?!\[*\])
)/.source);bE.leftMatch[bz]=new RegExp(/^(?:.|[\r\n])*\/.source+bE.match[bz].source.replace(/\\(\\d+)/g,bx)};var
bF=function(bR,e){bR=Array.prototype.slice.call(bR,0);if(e){e.push.apply(e,bR);return e}return
bR};try(Array.prototype.slice.call(av.documentElement.childNodes,0)[0].nodeType)catch(bP){bF=function(bU,bT){var bS=0,
[object Array]}(Array.prototype.push.apply(bR,bU))else{if(typeof bU.length==="number"){for(var
e=bU.length;bS<e;bS++){bR.push(bU[bS])}else{for(;bU[bS];bS++){bR.push(bU[bS])}}return bR}}var
bJ,bG;if(av.documentElement.compareDocumentPosition){bJ=function(bR,e){if(bR===e){bB=true;return
0}if(!bR.compareDocumentPosition||!e.compareDocumentPosition){return bR.compareDocumentPosition?-1:1}return
bR.compareDocumentPosition(e)&4?-1:1}}else{bJ=function(bY,bX){if(bY===bX){bB=true;return 0}else{if(bY.sourceIndex&&bX.
Y.sourceIndex<bX.sourceIndex)}var bV,bR,bS=[],e=[],bU=bY.parentNode,bW=bX.parentNode,bZ=bU;if(bU===bW){return
bG(bY,bX)}else{if(!bU){return -1}else{if(!bW){return
1}}while(bZ){bS.unshift(bZ);bZ=bZ.parentNode}bZ=bW;while(bZ){e.unshift(bZ);bZ=bZ.parentNode}bV=bS.length;bR=e.length;
bT=0;bT<bV&&bT<bR;bT++){if(bS[bT]!==e[bT]){return bG(bS[bT],e[bT])}return
bT===bV?bG(bY,e[bT],-1):bG(bS[bT],bX,1)};bG=function(bR,e,bS){if(bR===e){return bS}var bT=bR.nextSibling;while(bT){if(
bR=av.createElement("div"),bS="script"+(new Date()).getTime(),e=av.documentElement;bR.innerHTML="<a name='
"+bS+"'>";e.insertBefore(bR,e.firstChild);if(av.getElemenById(bS)){bE.find.ID=function(bU,bV,bW){if(typeof
bV.getElemenById!=="undefined"&&!bW){var bT=bV.getElemenById(bU[1]);return bT?bT.id===bU[1]||typeof
bT.getAttributeNode!=="undefined"&&bT.getAttributeNode("id"
.nodeValue===bU[1]?[bT]:L:[])};bE.filter.ID=function(bV,bT){var bU=typeof bV.getAttributeNode!=="undefined"&&bV.getAttr
bV.nodeType===1&&bU&&bU.nodeValue===bT)}e.removeChild(bR);e=bR=null)}(function(){var e=av.createElement("div"
);e.appendChild(av.createComment(""));if(e.getElemenByTagName("").length>0){bE.find.TAG=function(bR,bV){var
bU=bV.getElemenByTagName(bR[1]);if(bR[1]===)*"}(var bT=[];for(var
bS=0;bU[bS];bS++){if(bU[bS].nodeType===1){bT.push(bU[bS])}bU=bT}return bU)}e.innerHTML=";if(e.firstCh
undefined"&&e.firstChild.getAttribute("href")!=="#")bE.attrHandle.href=function(bR){return bR.getAttribute("
href",2)};e=null)}(function(){if(av.querySelectorAll){function(){var e=by,bT=av.createElement("div"),bS="__sizzle__"
;bT.innerHTML="<p class='TEST'></p>;if(bT.querySelectorAll&&bT.querySelectorAll(".TEST"
.length===0){return}by=function(b4,bV,bZ,b3){bV=bV||av;if(!b3&&!by.isXML(bV)){var b2=/"^(\w+$)|\^\.([\w-]+)
+$/g.exec(b4);if(b2&&(bV.nodeType===1|bV.nodeType===9)){if(b2[1]){return
bF(bV.getElemenByTagName(b4),bZ)}else{if(b2[2]&&bE.find.CLASS&&bV.getElemenByClassName){return
bF(bV.getElemenByClassName(b2[2]),bZ)}if(bV.nodeType===9){if(b4==="body"&&bV.body){return bF([bV.body],bZ)}else{if
bY=bV.getElemenById(b2[3]);if(bY&&bY.parentNode){if(bY.id===b2[3]){return bF([bY],bZ)}else{return bF([],bZ)}}}try{ret
bF(bV.querySelectorAll(b4),bZ)}catch(b0){}else{if(bV.nodeType===1&&bV.nodeName.toLowerCase()!="object"){var
bW=bV,bX=bV.getAttribute("id"),bU=bX|bS,b6=bV.parentNode,b5=/"^s*[*+]/.test(b4);if(!bX){bV.setAttribute("id"
,bU)}else{bU=bU.replace(/"/g,"\\$&")}if(b5&&b6){bV=bV.parentNode;try{if(!b5|b6){return bF(bV.querySelectorAll("[id='
"+bU+"' "+b4),bZ)}catch(b1){}finally{if(!bX){bW.removeAttribute("id")}}}}return e(b4,bV,bZ,b3)};for(var
bR in e){by[bR]=e[bR]}bT=null)}(function(){var
e=av.documentElement,bS=e.matchesSelector||e.mozMatchesSelector||e.webkitMatchesSelector||e.msMatchesSelector;if(bS){v
),bR=false;try{bS.call(av.documentElement,"[test='']:sizzle"
)}catch(bT){bR=true}by.matchesSelector=function(bW,bY){bY=bY.replace(/\\s*([^\s])\s*/g,"='$1'");if(!by.isXML(bW))
!/=/\.test(bY){var bU=bS.call(bW,bY);if(bV||!bU|bW.document&&bW.document.nodeType===1){return bV}}catch(
bX){}return by(bY,null,null,[bW]).length>0}}(function(){var e=av.createElement("div");e.innerHTML="<div
class='test e'></div><div class='test'></div>;if(!e.getElemenByClassName|e.getElemenByClassName("e").
length===0){return}e.lastChild.className="e";if(e.getElemenByClassName("e").length===1){return}bE.order.
splice(1,0,["CLASS"]);bE.find.CLASS=function(bR,bS,bT){if(typeof bS.getElemenByClassName!=="undefined"&&!bT){
return bS.getElemenByClassName(bR[1]);e=null)}(function(){var bR,bW,bV,bZ,bX,bY){for(var bT=0,bS=bZ.length
;bT<bS;bT++){var e=bZ[bT];if(e){var bU=false;e=e[bR];while(e){if(e[bC]===bV){bU=bZ[e.sizset];break}if(e.
nodeType===1&&!bY){e[bC]=bV;e.sizset=bT;if(e.nodeName.toLowerCase()===bW){bU=e;break}e=e[bR];bZ[bT]=bU}}
function bN(bR,bW,bV,bZ,bX,bY){for(var bT=0,bS=bZ.length;bT<bS;bT++){var e=bZ[bT];if(e){var bU=false;e=e[bR];while(
e){if(e[bC]===bV){bU=bZ[e.sizset];break}if(e.nodeType===1){if(!bY){e[bC]=bV;e.sizset=bT;if(typeof bW!=="
string"){if(e===bW){bU=true;break}else{if(by.filter(bW,[e]).length>0){bU=e;break}}e=e[bR];bZ[bT]=bU}}if(av.
documentElement.contains){by.contains=function(bR,e){return bR===e&&(bR.contains?bR.contains(e):true)}else{if
(av.documentElement.compareDocumentPosition){by.contains=function(bR,e){return !(bR.compareDocumentPosition
(e)&16)}else{by.contains=function(){return false}}by.isXML=function(e){var bR=(e?e.ownerDocument||e:0)
.documentElement;return bR?bR.nodeName!="HTML":false};var bM=function(bS,e,bW){var bV,bX=[],bU="",bY=e.nodeType
?e:e;while((bV=bE.match.PSEUDO.exec(bS))){bU+=bV[0];bS=bS.replace(bE.match.PSEUDO,"")}bS=bE.relative[bS]?
bS+"*"+bS;for(var bT=0,bR=bY.length;bT<bR;bT++){by(bS,bY[bT],bX,bW)}return by.filter(bU,bX)};by.attr=
b.attr;by.selectors.attrMap={};b.find=by;b.expr=by.selectors;b.expr[":"]=b.expr.filters;
b.unique=by.uniqueSort;b.text=by.getText;b.isXMLDoc=by.isXML;b.contains=by.contains)}(function(){var ab=/Until$/
,aq="/^?(?:parents|prevUntil|prevAll)/,a9=/,/,bp="/^.[^#\\|.|.]*$/;P=Array.prototype.slice,H=
b.expr.match.POS,ay={children:true,contents:true,next:true,prev:true};b.fn.extend({find:function(e){var
bw=this,by,bv;if(typeof e!="string"){return b(e).filter(function(){for(by=0,bv=bw.length;by<bv;by++){if(b.c
ontains(bw[by],this){return true}}}}var bx=this.pushStack("",,"find",e),bA,bB,bz;for(by=0,bv=this.length;by
<bv;by++){bA=bx.length;b.find(e,this[by],bx);if(by>0){for(bB=bA;bB<bx.length;bB++){for(bz=0;bz<bA;bz++){if(
bx[bz]===bx[bB]){bx.splice(bB--,1);break}}}}return bx};has=function(bv){var e=b(bv);return this.filter(
function(){for(var bx=0,bw=e.length;bx<bw;bx++){if(b.c.contains(this,e[bx])){return true}}},not:function(e){
return this.pushStack(aG(this,e,false),"not",e)},filter:function(e){return this.pushStack(aG(this,e,true),"
filter",e)},is:function(e){return !e&&(typeof e=="string"?H.test(e)?b(e,this.context).index(this[0])>=0:
b.filter(e,this).length>0:this.filter(e).length>0)},closest:function(by,bx){var bv=[],bw,e,bz=this[0];if(
b.isArray(by)){var bB=1;while(bz&&bz.ownerDocument&&bz!==bx){for(bw=0;bw<by.length;bw++){if(
b(bz).is(by[bw])){bv.push({selector:by[bw],elem:bz,level:bB})}bz=bz.parentNode;bB++}return bv}var bA=H.
test(bv)||typeof by!="string"?b(by,bx|this.context):0;for(bw=0,e=this.length;bw<e;bw++){bz=this[bw];while(
bz){if(bA?bA.index(bz)>-1:b.find.matchesSelector(bz,by)){bv.push(bz);break}else{bz=bz.parentNode;if(!bz||!b
.ownerDocument||bz===bx|bz.nodeType===11){break}}}}bv=bv.length>1?b.unique(bv):bv;return this.pushStack(bv
,"closest",by)},index:function(e){if(!e){return this[0]&&this[0].parentNode?this.prevAll().length:-1}if(
typeof e=="string"){return b.inArray(this[0],b(e))}return b.inArray(e.jquery?e[0]:e,this),add:function(e,
bv){var bx=typeof e=="string"?b(e,bv):b.makeArray(e&&e.nodeType?[e]:e),bw=b.merge(this.get(0),bx);return
this.pushStack(C(bx[0])|C(bw[0])?bw:b.unique(bw)),andSelf:function(){return this.add(this.prevObject)});
function C(e){return e||e.parentNode|e.parentNode.nodeType===11?b.each({parent:function(bv){var e=bv.
parentNode;return e&&e.nodeType===11?e:null},parents:function(e){return b.dir(e,"parentNode"),parentsUntil:
function(bv,e,bw){return b.dir(bv,"parentNode",bw)},next:function(e){return b.nth(e,2,"nextSibling")},prev:
function(e){return b.nth(e,2,"previousSibling")},nextAll:function(e){return b.dir(e,"nextSibling")},prevAll:
function(e){return b.dir(e,"previousSibling")},nextUntil:function(bv,e,bw){return
b.dir(bv,"nextSibling",bw)},prevUntil:function(bv,e,bw){return b.dir(bv,"previousSibling",bw)},siblings:
function(e){return b.sibling(e.parentNode.firstChild,e)},children:function(e){return
b.sibling(e.firstChild)},contents:function(e){return b.nodeName(e,"iframe")?e.contentDocument||e.

```



```
contentWindow.document:b.makeArray(e.childNodes)},function(e,bv){b.fn[e]=function(by,bw){var bx=
b.map(this,bv,by);if(!b.ab.test(e){bw=by}if(bw&&typeof bw=="string"){bx=bw.filter(bw,bx)}bx=this.length>1&
&lay[e]?b.unique(bx):bx;if(this.length>1||a9.test(bw)&&aq.test(e)){bx=bw.reverse()}return this.pushStack(
bx,e,P.call(arguments).join(","))};b.extend({filter:function(bw,e,bv){if(bv){bw=":"+"bw+"}return e.
length===1?b.find.matchesSelector(e[0],bw)?[e[0]]:[]:b.find.matches(bw,e)},dir:function(bw,bv,by){var e=[]
,bx=bw[bv];while(bx&&bx.nodeType!==9&&(by===L||bx.nodeType===1||!b(bx).is(by))){if(bx.nodeType===1){e.push
(bx)}bx=bx[bv]}return e},nth:function(by,e,bw,bx){e=e||1;var bv=0;for(;by;by=by[bw]){if(by.nodeType===1&&+
bv===e){break}return by},sibling:function(bw,bv){var e=[];for(;bw;bw=bw.nextSibling){if(bw.nodeType===1&&bw
!==(bv){e.push(bw)}return e}};function aG(bx,bw,e){bw=bw||0;if(b.isFunction(bw)){return
b.grep(bx,function(bz,by){var bA=!bw.call(bz,by,bz);return bA===e})}else{if(bw.nodeType){return
b.grep(bx,function(bz,by){return(bz===bw)===e})}else{if(typeof bw=="string"){var bv=
b.grep(bx,function(bz,by){return by.nodeType===1});if(bp.test(bw)){return b.filter(bw,bv,!e)}else{bw=
b.filter(bw,bv)}}}return b.grep(bx,function(bz,by){return(b.inArray(bz,bw)>=0)===e})}function a(e){var
bw=aR.split("|"),bv=e.createDocumentFragment();if(bv.createElement){while(bw.length){bv.createElement(bw.pop
())}return bv}var aR="
abbr|article|aside|audio|canvas|datalist|details|figcaption|figure|footer|header|hgroup|mark|meter|nav|output|progress|
(?:area|br|col|embed|hr|img|input|link|meta|param)(([w:]|+)[^>]*\s*/>|ig,d=/<([w:]|+)[^>]*\s*/>|ig,w=/<tbody/
|W/<|<#?<w|/ae/<(?<script|style)/i,O=/<(?<script|object|embed|option|style/
/i,ah=new RegExp("<(?:'+aR+'|'i'|o'/checked\\s*(?:'=[^=]|\\s*.checked.)'/i,bm=/\\(java|ecma)script/
i,aN=/^\\s*<!(?:[CDATA|\\|\\-|/ax=(option:[1,"<select multiple='multiple'>","</select>"],legend:[1,"
fieldset>","</fieldset>"],thead:[1,"<table>"],tr:[2,"<tbody><tbody>","<tbody></tbody>"],td:[3,"
<table><tbody><tr>","</tr></tbody></table>"],col:[2,"<table><tbody></tbody><colgroup>","</colgroup></table>"],
area:[1,"<map>","</map>"],_default:[0,"",""],aca=(av);ax.optgroup=ax.option;ax.tbody=ax.tfoot=ax.
colgroup=ax.caption=ax.thead;ax.th=ax.td;if(!b.support.htmlSerialize){ax._default=[1,"div<div>","</div>"]}
b.fn.extend({text:function(e){if(b.isFunction(e)){return this.each(function(bw){var bv=
b(this);bv.text(e.call(this,bw,bv.text()))}}if(typeof e!="object"&&e!=L){return this.empty().append((
this[0]&&this[0].ownerDocument||av).createTextNode(e))}return b.text(this),wrapAll:function(e){if(
b.isFunction(e)){return this.each(function(bw){b(this).wrapAll(e.call(this,bw))}}if(this[0]){var bv=
b(e,this[0].ownerDocument).eq(0).clone(true);if(this[0].parentNode){bv.insertBefore(this[0])}bv.map(
function(){var bw=this;while(bw.firstChild&&bw.firstChild.nodeType===1){bw=bw.firstChild}return bv}).append(this)
return this},wrapInner:function(e){if(b.isFunction(e)){return this.each(function(bv){
b(this).wrapInner(e.call(this,bv))}}return this.each(function(){var bv=b(this),bw=bv.contents();if(
bw.length){bw.wrapAll(e)}else{bv.append(e)}}},wrap:function(e){var bv=b.isFunction(e);return this.
each(function(bw){b(this).wrapAll(bv?e.call(this,bw):e)}),unwrap:function(){return this.parent().each
(function(){if(!b.nodeName(this,"body")){b(this).replaceWith(this.childNodes)}.end(),append:function(){
return this.domManip(arguments,true,function(e){if(this.nodeType===1){this.appendChild(e)}},prepend:
function(){return this.domManip(arguments,true,function(e){if(this.nodeType===1){this.insertBefore(e,this.
firstChild)}},before:function(){if(this[0]&&this[0].parentNode){return this.domManip(arguments,false,function(bv){
this.parentNode.insertBefore(bv,this)}},else{if(arguments.length){var e=b.clean(arguments);e.push.apply(e,
this.toArray());return this.pushStack(e,"before",arguments)}},after:function(){if(this[0]&&this[0].
parentNode){return this.domManip(arguments,false,function(bv){this.parentNode.insertBefore(bv,this.nextSibling)})
else{if(arguments.length){var e=this.pushStack(this,"after",arguments);e.push.apply(e,
b.clean(arguments));return e}},remove:function(e,bx){for(var bv=0,bw;(bw=this[bv])!=null;bv++){if(!e||
b.filter(e,[bw]).length){if(!bx&&bw.nodeType===1){b.cleanData(bw.getElementsByTagName("*"))};
b.cleanData([bw])}if(bw.parentNode){bw.parentNode.removeChild(bw)}return this}{for(var
e=0,bv;(bv=this[e])!=null;e++){if(bv.nodeType===1){b.cleanData(bv.getElementsByTagName("*"))}while(bv.
firstChild){bv.removeChild(bv.firstChild)}return this},clone:function(bv,e){bv=bv==null?false:bv;e=e==null?bv:e;
return this.map(function(){return b.clone(this,bv,e)}),html:function(bx){if(bx===
L){return this[0]&&this[0].nodeType===1?this[0].innerHTML.replace(ag,""):null}else{if(typeof bx=="string"
&&!ae.test(bx)&&(b.support.leadingWhitespace||!ar.test(bx)&&!ax[(d.exec(bx)||["",""])[1].toLowerCase()])
bx=bx.replace(R,"<$1></$2>");try{for(var bw=0,bv=this.length;bw<bv;bw++){if(this[bw].nodeType===1){
b.cleanData(this[bw].getElementsByTagName("*"));this[bw].innerHTML=bx}}catch(by){this.empty().append(bx)
}}else{if(b.isFunction(bx)){this.each(function(bz){var e=b(this);e.html(bx.call(this,bz,e.html()))}
else{this.empty().append(bx)}return this},replaceWith:function(e){if(this[0]&&this[0].parentNode){if(
b.isFunction(e)){return this.each(function(bx){var bw=b(this),bv=bw.html();bw.replaceWith(e.call(this
,bx,bv))}if(typeof e!="string"){e=b(e).detach()}return this.each(function(){var bw=this.nextSibling,
bv=this.parentNode;b(this).remove();if(bw){b(bw).before(e)}else{b(bv).append(e)}}}else{return this.
length?this.pushStack(b(b.isFunction(e)?e():e),"replaceWith",e):this},detach:function(e){return this.remove(e,
true)},domManip:function(bB,bF,bE){var bx,by,bA,bD,bC=bB[0],bv=[];if(!b.support.checkClone&&arguments.
length===3&&typeof bC=="string"&&o.test(bC)){return this.each(function(){b(this).domManip(bB,bF,bE,true)})
if(b.isFunction(bC)){return this.each(function(bH){var bG=b(this);bB[0]=bC.call(this,bH,bF?bG.html():
L);bG.domManip(bB,bF,bE)}if(this[0]){bD=bC&&bC.parentNode;if(b.support.parentNode&&bD&&bD.nodeType===1&&
&bD.childNodes.length===this.length){bx={fragment:bD}else{bx=b.buildFragment(bB,this,bv)}bA=bx.fragment;
if(bA.childNodes.length===1){by=bA=bA.firstChild}else{by=bA.firstChild}if(by){bF=bF&&
b.nodeName(by,"tr");for(var bw=0,e=this.length,bz=e-1;bw<e;bw++){bE.call(bF?ba(this[bw],by):this[bw],bx,
cacheable||e>1&&bw<bz)?b.clone(bA,true,true):bA}}if(bv.length){b.each(bv,bv){function ba(e
,bv){return b.nodeName(e,"table")?(e.getElementsByTagName("tbody")[0]||e.appendChild(e.ownerDocument.
createElement("tbody")):e}function t(bB,bv){if(bv.nodeType===1||!b.hasData(bB)){return}var by,bx,e,bA=
b._data(bB),bz=b._data(bv,bA),bw=bA.events;if(bw){delete bz.handle;bz.events={};for(by in bw){for(bx=0,e=
bw[by].length;bx<e;bx++){b.event.add(bv,by+(bw[by][bx].namespace?"":"")+bw[by][bx].namespace,bw[by][bx],bw
[by][bx].data)}if(bz.data){bz.data=b.extend({},bz.data)}function ai(bv,e){var bw;if(bv.nodeType===1){
return}if(e.clearAttributes){e.clearAttributes()}if(e.mergeAttributes){e.mergeAttributes(bv)}bw=e.nodeName.
toLowerCase();if(bw=="object"){e.outerHTML=bv.outerHTML}else{if(bw=="input"&&(bv.type=="checkbox"||bv.type=="
radio")){if(bv.checked){e.defaultChecked=e.checked=bv.checked}if(e.value!=bv.value){e.value=bv.value}}else{
if(bw=="option"){e.selected=bv.defaultSelected}else{if(bw=="input"||bw=="textarea"){e.defaultValue=bv.
defaultValue}}e.removeAttribute(b.expando)}b.buildFragment(function(bz,bx,bv){var by,e,bw,bA,bB=bz[0];if(bx
&&bx[0]){bA=bx[0].ownerDocument||bx[0]}if(!bA.createDocumentFragment){bA=av}if(bz.length===1&&typeof bB=="
string"&&bB.length<512&&bA===av&&bB.charAt(0)!="&&!o.test(bB)&&(b.support.checkClone||!o.test(bB))&&(
b.support.html5Clone||!ah.test(bB))){e=true;bw=b.fragments[bB];if(bw&&bw===1){by=bw}}if(!by){by=bA.
createDocumentFragment();b.clean(bz,bA,by,bv)}if(e){b.fragments[bB]=bw?by:1}return{fragment:by,cacheable:e}};
b.fragments={};b.each({appendTo:"append",prependTo:"prepend",insertBefore:"before",insertAfter:"after",
replaceAll:"replaceWith"},function(e,bv){b.fn[e]=function(bw){var bz=[],bC=b(bw),bB=this.length===1&&this[0].
parentNode;if(bB&&bB.nodeType===11&&bB.childNodes.length===1&&bC.length===1){bC[bv](this[0]);return this
else{for(var bA=0,bx=bC.length;bA<bx;bA++){var by=(bA>0?this.clone(true):this).get();
b(bC[bA])[bv](by);bz=bz.concat(by)}return this.pushStack(bz,e,bC.selector)}};function bg(e){if(typeof e.
getElementsByTagName!="undefined"){return e.getElementsByTagName("*")}else{if(typeof e.querySelectorAll!="
undefined"){return e.querySelectorAll("*")}else{return []}}function az(e){if(e.type=="checkbox"||e.type=="
```

```

radio") {e.defaultChecked=e.checked}}function E(e) {var bv=(e.nodeName|| "").toLowerCase(); if (bv=="input") {az
(e) }else{if (bv!="script"&&typeof e.getElementsByTagName!="undefined") {b.grep(e.getElementsByTagName("
input"), az)}}function al(e) {var bv=av.createElement("div"); ac.appendChild(bv);bv.innerHTML=e.outerHTML;return
bv.firstChild}b.extend({clone:function(by, bA, bw) {var e, bv, bx, bz=b.support.html5Clone|| !ah.test("<" +by.
nodeName)?by.cloneNode(true):al(by); if ((!b.support.noCloneEvent|| !b.support.noCloneChecked)&& (by.nodeType===1||
by.nodeType===11)&& !b.isXMLDoc(by)) {ai(by, bz); e=bg(by); bv=bg(bz); for (bx=0; e[bx]; ++bx) {if (bv[bx]) {ai(e[bx],
bv[bx])}} }if (bA) {t(by, bz); if (bw) {e=bg(by); bv=bg(bz); for (bx=0; e[bx]; ++bx) {t(e[bx], bv[bx])}} }e=bv=null; return
bz}, clean:function(bw, by, bH, bA) {var bF; by=by|| av; if (typeof by.createElement=="undefined") {by=by.
ownerDocument|| by[0]&&by[0].ownerDocument|| av}var bI=[], bB; for (var bE=0, bz; (bz=bw[bE]) !=null; bE++) {if (typeof bz=="
number") {bz+=""; if (!bz) {continue}if (typeof bz=="string") {if (!W.test(bz)) {bz=by.createTextNode(bz)} else {bz=bz.
replace(R, "<$1></$2>"); var bK=(d.exec(bz)|| ["", ""]) [1].toLowerCase(), bx=ax[bK]|| ax._default, bD=bx[0], bv=by.
createElement("div"); if (by==av) {ac.appendChild(bv)} else {a(by).appendChild(bv)}bv.innerHTML=bx[1]+bz+bx[2]; while
(bD--) {bv=bv.lastChild}if (!b.support.tbody) {var e=w.test(bz), bC=bK=="table"&&!e?bv.firstChild&&bv.
firstChild.childNodes[bx[1]]=="<table>"&&!e?bv.childNodes[0]; for (bB=bC.length-1; bB>=0; --bB) {if (
b.nodeName(bC[bB], "tbody")&&!bC[bB].childNodes.length) {bC[bB].parentNode.removeChild(bC[bB])}} }if (!
b.support.leadingWhitespace&&ar.test(bz)) {bv.insertBefore(by.createTextNode(ar.exec(bz)[0]), bv.firstChild)
}bz=bv.childNodes}var bG; if (!b.support.appendChecked) {if (bz[0]&&typeof (bG=bz.length)=="number") {for (bB=0;
bB<bG; bB++) {E(bz[bB])}} else {E(bz)}}if (bz.nodeType) {bI.push(bz)} else {bI=b.merge(bI, bz)}}if (bH) {bF=function
(bL) {return !bL.type||bm.test(bL.type)}; for (bE=0; bI[bE]; bE++) {if (bA&&bI[bE].nodeName(bI[bE], "script")&&!bI[bE].
type|| bI[bE].type.toLowerCase()=="text/javascript") {bA.push(bI[bE].parentNode?bI[bE].parentNode.
removeChild(bI[bE]):bI[bE])} else {if (bI[bE].nodeType===1) {var bJ=b.grep(bI[bE].getElementsByTagName("script"), bF); bI.
splice.apply(bI, [bE+1, 0].concat(bJ))} bH.appendChild(bI[bE])}} }return bI}, cleanData:function(bv) {var by, bw, e=
b.cache, bB=b.event.special, bA=b.support.deleteExpando; for (var bz=0, bx; (bx=bv[bz]) !=null; bz++) {if (bx.
nodeName&&b.noData[bx.nodeName.toLowerCase()]) {continue}bw=bx[b.expando]; if (bw) {by=e[bw]; if (by&&by.events) {for
(var bC in by.events) {if (bB[bC]) {b.event.remove(bx, bC)} else {b.removeEvent(bx, bC, by.handle)}} }if (by.handle) {
by.handle.elem=null}} if (bA) {delete bx[b.expando]} else {if (bx.removeAttribute) {bx.removeAttribute(
b.expando)} }delete e[bw]}}}; function bo(e, bv) {if (bv.src) {b.ajax({url:bv.src, async:false, dataType:"
script"})} else {b.globalEval((bv.text||bv.textContent||bv.innerHTML|| "").replace(an, "/*$0*/"))} if (bv.parentNode) {
bv.parentNode.removeChild(bv)} }var ak=/alpha\(|\^|\)*\)/i

```

Definition at line 16 of file jquery.js.

### 30.803.2.8 function bb

Definition at line 16 of file jquery.js.

### 30.803.2.9 var bq =/#.\*\$/

Definition at line 23 of file jquery.js.

### 30.803.2.10 var bs =/r?\n/g

Definition at line 23 of file jquery.js.

### 30.803.2.11 var c

**Initial value:**

```

=/^\\\/\
* jQuery UI 1.8.18
*
* Copyright 2011

```

Definition at line 23 of file jquery.js.

### 30.803.2.12 b fn css =function(e,bv){if(arguments.length===2&&bv===L){return this}return b.access(this,e,bv,true,function(bx,bw,by){return by!==L?b.style(bx,bw,by):b.css(bx,bw)}}}

Definition at line 23 of file jquery.js.

30.803.2.13 **b curCSS =b.css**

Definition at line 23 of file jquery.js.

30.803.2.14 **var k =/%20/g**

Definition at line 23 of file jquery.js.

30.803.2.15 **function L {var av=bb.document,bu=bb.navigator,bl=bb.location**

Definition at line 16 of file jquery.js.

30.803.2.16 **Z =a||aX**

Definition at line 23 of file jquery.js.

## 30.804 gotools-core/doc/html/jquery.js File Reference

### Functions

- **b extend** ({cssHooks:{opacity:{get:function(bw, bv){if(bv){var e=Z(bw,"opacity","opacity");return e===""?"1"↵:e}else{return bw.style.opacity}}}}, cssNumber:{fillOpacity:true, fontWeight:true, lineHeight:true, opacity↵:true, orphans:true, widows:true, zIndex:true, zoom:true}, cssProps:{float:"b.support.cssFloat?"css↵Float":"styleFloat"}, style:function(bx, bw, bD, by){if(!bx||bx.nodeType===3||bx.nodeType===8||bx.↵style){return}var bB, bC, bz=b.camelCase(bw), bv=bx.style, bE=b.cssHooks[bz];bw=b.cssProps[bz]||bz;if(b↵D!==L){bC=typeof bD;if(bC==="string"&&(bB=l.exec(bD))){bD=(+(bB[1]+1)\*+bB[2])+parseFloat(b.css(bx, bw));bC="number"}if(bD===null||bC==="number"&&isNaN(bD)){return}if(bC==="number"&&!b.cssNumber[bz]){bD+="px"}if(!bE↵|bE){bD=bE.set(bx, bD)}!==L){try{bv[bw]=bD}catch(bA){}}else{if(bE &&"get" in bE &&(bB=bE.get(bx, false, by))!==L){return bB}return bv[bw]}}, css:function(by, bx, bv){var bw, e;bx=b.camelCase(bx);e=b.css↵Hooks[bx];bx=b.cssProps[bx]||bx;if(bx==="cssFloat"){bx="float"}if(e &&"get" in e &&(bw=e.get(by, true, bv))!==L){return bw}else{if(Z){return Z(by, bx)}}}, swap:function(bx, bw, by){var e={};for(var bv in bw){e[bv]=bx.style[bv];bx.style[bv]=bw[bv]}by.call(bx);for(bv in bw){bx.style[bv]=e[bv]}}
- **b each** (["height","width"], function(bv, e){b.cssHooks[e]={get:function(by, bx, bw){var bz;if(bx){if(by.offset↵Width!==(0){return p(by, e, bw)}else{b.swap(by, a7, function(){bz=p(by, e, bw)}})}return bz}}, set:function(bw, bx){if(bc.test(bx)){bx=parseFloat(bx);if(bx >=0){return bx+"px"}}else{return bx}}})
- **if** (!b.support.opacity)
- **b** (function(){if(!b.support.reliableMarginRight){b.cssHooks.marginRight={get:function(bw, bv){var e;b.↵swap(bw,{display:"inline-block"}, function(){if(bv){e=Z(bw,"margin-right","marginRight")}else{e=bw.style.↵marginRight}});return e}}})
- **if** (av.defaultView &&av.defaultView.getComputedStyle)
- **if** (av.documentElement.currentStyle)
- **function p** (by, bw, bv)
- **if** (b.expr &&b.expr.filters)

## Variables

- function **bb**
- function **L** {var av=bb.document,bu=bb.navigator,bl=bb.location
- var **b**
- var **au** =/opacity=(^[^)]\*)/;z=/([A-Z]|^ms)/g;bc=/^-?\d+(?:px)?\$/i;bn=/^-?\d/;l=/^([\+])=([\+.\de]+)/;a7={position↵:"absolute",visibility:"hidden",display:"block"},an=["Left","Right"],a1=["Top","Bottom"],Z,aI,aX
- **b fn css** =function(e,bv){if(arguments.length===2&&bv===L){return this}return b.access(this,e,bv,true,function(bx,bw,by){return by!==L?b.style(bx,bw,by):b.css(bx,bw)})}
- **b curCSS** =b.css
- **Z** =aI||aX
- var **k** =/%20/g
- var **ap** =/[\]\$/
- var **bs** =/r?\n/g
- var **bq** =/#.\*\$/
- var **aD** =/^(.\*?)[\t]\*([^\r\n]\*)r?\$/mg
- var **aZ** =/^(?:color|date|datetime|datetime-local|email|hidden|month|number|password|range|search|tel|text|time|url|week)\$/i
- var **aM** =/^(?:about|app|app-storage|.+\-extension|file|res|widget):\$/
- var **aQ** =/^(?:GET|HEAD)\$/
- var **c**

## 30.804.1 Function Documentation

- 30.804.1.1 **b ( function(){if(!b.support.reliableMarginRight){b.cssHooks.marginRight={get:function(bw, bv){var e;b.swap(bw,{display:"inline-block"}, function(){if(bv){e=Z(bw,"margin-right","marginRight")}else{e=bw.style.↵marginRight}};return e}} } )**
- 30.804.1.2 **b each ( function(bv, e){b.cssHooks[e]={get:function(by, bx, bw){var bz;if(bx){if(by.offsetWidth!==0){return p(by, e, bw)}else{b.swap(by, a7, function(){bz=p(by, e, bw)}}}return bz}}, set:function(bw, bx){if(bc.test(bx)){bx=parseFloat(bx);if(bx >=0){return bx+"px"}}else{return bx}} } )**
- 30.804.1.3 **b extend ( {cssHooks:{opacity:{get:function(bw, bv){if(bv){var e=Z(bw,"opacity","opacity");return e===""?"1"↵:e}else{return bw.style.opacity}}}, cssNumber:{fillOpacity:true, fontWeight:true, lineHeight:true, opacity:true, orphans:true, widows:true, zIndex:true, zoom:true}, cssProps:{"float":b.support.cssFloat?"cssFloat":"styleFloat"}, style:function(bx, bw, bD, by){if(!bx||bx.nodeType===3||bx.nodeType===8||!bx.style){return}var bB, bC, bz=b.camelCase(bw), bv=bx.style, bE=b.cssHooks[bz];bw=b.cssProps[bz]||bz;if(bD===L){bC=typeof bD;if(bC==="string"&&(bB=l.exec(bD))){bD=+(bB[1]+1)\*+bB[2]+parseFloat(b.css(bx, bw));bC="number"}if(bD===null||bC==="number"&&isNaN(bD)){return}if(bC==="number"&&b.css↵Number[bz]){bD+="px"}if(!bE||!("set" in bE)||("set" in bE)&&(bD=bE.set(bx, bD))!==L){try{bv[bw]=bD}catch(bA){}}else{if(bE &&"get" in bE &&(bB=bE.get(bx, false, by))!==L){return bB}return bv[bw]}; css:function(by, bx, bv){var bw, e;bx=b.camelCase(bx);e=b.cssHooks[bx];bx=b.cssProps[bx]||bx;if(bx==="cssFloat"){bx="float"}if(e &&"get" in e &&(bw=e.get(by, true, bv))!==L){return bw}else{if(Z){return Z(by, bx)}}}, swap:function(bx, bw, by){var e={};for(var bv in bw){e[bv]=bx.style[bv];bx.style[bv]=bw[bv]}by.call(bx);for(bv in bw){bx.style[bv]=e[bv]}} } )**
- 30.804.1.4 **if ( av.documentElement. currentStyle )**

Definition at line 23 of file jquery.js.

- 30.804.1.5 **if ( av.defaultView &&av.defaultView. getComputedStyle )**

Definition at line 23 of file jquery.js.

30.804.1.6 `if ( b.expr && b.expr.filters )`

Definition at line 23 of file jquery.js.

30.804.1.7 `if ( !b.support.opacity )`

Definition at line 23 of file jquery.js.

30.804.1.8 `function p ( by, bw, bv )`

Definition at line 23 of file jquery.js.

## 30.804.2 Variable Documentation

30.804.2.1 `var aD =/(.*?):[\t]*([\r\n]*)\r?$/mg`

Definition at line 23 of file jquery.js.

30.804.2.2 `var aM =/^(?:about|app|app-storage|.+|-extension|file|res|widget):$/`

Definition at line 23 of file jquery.js.

30.804.2.3 `var ap =/[\\]$/`

Definition at line 23 of file jquery.js.

30.804.2.4 `var aQ =/^(?:GET|HEAD)$/`

Definition at line 23 of file jquery.js.

30.804.2.5 `var au =/opacity=([\^]*),z=/([A-Z])^ms)/g,bc=/^-?\d+(?:px)?$/i,bn=/^-?\d/,l=/^([\+])=([\+.\de]+),a7={position↵  
:"absolute",visibility:"hidden",display:"block"},an=["Left","Right"],a1=["Top","Bottom"],Z,aI,aX`

Definition at line 23 of file jquery.js.

30.804.2.6 `var aZ =/^(?:color|date|datetime|datetime-  
local|email|hidden|month|number|password|range|search|tel|text|time|url|week)$/i`

Definition at line 23 of file jquery.js.



## 30.804.2.7 var b

## Initial value:

```

=(function() {var bF=function(b0,b1) {return new bF.fn.init(b0,b1,bD)},bU=bb.jquery,bH=
bb.$,bD,bY=/^(?![^<]*<([\w\W]+>)[^>]*$|#([\w\W]+)\/,bM=\/S/,bI=\/\s+/,bE=\/\s+$/,bA=\/^(\/\w+)\s*\/(?<
\/\>)?$/,bN=\/^\[\],:([\s]*$|,bW=\/\([\s]*$|,bFnrj=[u0-9a-fA-F]{4})/g,bP=\/"^[^\\n\r]*"truefalse|null|-?d
+(?!\.\\d*)?(?:[eE](+|-)\?d+)?/g,bJ=\/(?:^|:|,)(?:\s*\|)/g,by=\/(webkit)[\\/]([\w.]+)\/,bR=\/(opera)(?!.version
)?[\\/]([\w.]+)\/,bQ=\/(msie) ([\w.]+)\/,bS=\/(mozilla)(?:.*? rv:([\w.]+))?/,bB=\/-([a-z]{0-9})/ig,bZ=\/^ms-/,bT
=function(b0,b1) {return(b1+"").toUpperCase()}.push,bX=bu.userAgent,bV,bC,e,bL=Object.prototype.toString,bG=Object
.prototype.hasOwnProperty,bz=Array.prototype.push,bK=Array.prototype.slice,bO=String.prototype.trim,bv=Array
.prototype.indexOf,bx={};bF.fn=bF.prototype={constructor:bF,init:function(b0,b4,b3) {var b2,b5,b1,b6;if(!b0) {
return this;if(b0.nodeType) {this.context=this[0]=b0;this.length=1;return this;if(b0=="body"&&!b4&&av.body) {
this.context=av;this[0]=av.body;this.selector=b0;this.length=1;return this;if(typeof b0=="string") {if(b0.
charAt(0)=="<"&&b0.charAt(b0.length-1)==">"&&b0.length>=3) {b2=[null,b0,null]}else{b2=bY.exec(b0)}if(b2&&(b2[
1]||b4)) {if(b2[1]) {b4=b4 instanceof bF?b4[0]:b4;b6=(b4?b4.ownerDocument||b4:av);b1=bA.exec(b2,b1,b2)}if(bF
.isPlainObject(b4)) {b0=[av.createElement(b1[1])];bF.fn.attr.call(b0,b4,true)}else{b0=[b6.createElement(b1[1]
)]}else{b1=bF.buildFragment([b2[1]],[b6]);b0=(b1.cacheable?bF.clone(b1.fragment):b1.fragment).childNodes
return bF.merge(this,b0)}else{b5=av.getElementById(b2[2]);if(b5&&b5.parentNode) {if(b5.id==b2[2]) {return b3.
find(b0)}this.length=1;this[0]=b5}this.context=av;this.selector=b0;return this}if(!b4||b4.jquery) {return
(b4||b3).find(b0)}else{return this.constructor(b4).find(b0)}else{if(bF.isFunction(b0)) {return b3.ready(b0)
}}if(b0.selector!=L) {this.selector=b0.selector;this.context=b0.context;return bF.makeArray(b0,this)},
selector:"",jquery:"1.7.1",length:0,size:function() {return this.length},toArray:function() {return bK.call(this,0)
},get:function(b0) {return b0==null?this.toArray(): (b0<0?this[this.length+b0]:this[b0])},pushStack:function(
b1,b3,b0) {var b2=this.constructor();if(bF.isArray(b1)) {b2.apply(b2,b1)}else{bF.merge(b2,b1)}b2.prevObject=
this;b2.context=this.context;if(b3=="find") {b2.selector=this.selector+(this.selector?" ":"")+b0}else{if(b3) {b2
.selector=this.selector+"."+b3+"("+b0+")"}return b2},each:function(b1,b0) {return bF.each(this,b1,b0)},
ready:function(b0) {bF.bindReady();bC.add(b0);return this},eq:function(b0) {b0+=b0;return b0===-1?this.slice(b0
):this.slice(b0,b0+1)},first:function() {return this.eq(0)},last:function() {return this.eq(-1)},slice:
function() {return this.pushStack(bK.apply(this,arguments),"slice",bK.call(arguments).join(","))},map:function(b0) {
return this.pushStack(bF.map(this,function(b2,b1) {return b0.call(b2,b1,b2)}),end:function() {return this.
prevObject||this.constructor(null)},push:bz,sort:[].sort,splice:[].splice);bF.fn.init.prototype=bF.fn;bF.extend
=bF.fn.extend=function() {var b9,b2,b0,b1,b6,b7,b5=arguments[0]||{};b4=1,b3=arguments.length,b8=false;if(
typeof b5=="boolean") {b8=b5;b5=arguments[1]||{};b4=2;if(typeof b5!="object"&&bF.isFunction(b5)) {b5={};if(b3==
=b4) {b5=this;--b4}for(;b4<b3;b4++) {if((b9=arguments[b4])!=null) {for(b2 in b9) {b0=b5[b2];b1=b9[b2];if(b5==b1
) {continue}if(b8&&b1&&(bF.isPlainObject(b1)||b6=bF.isArray(b1))) {if(b6) {b6=false;b7=b0&&bF.isArray(b0)?b0:
[]};else{b7=b0&&bF.isPlainObject(b0)?b0:{};b5[b2]=bF.extend(b8,b7,b1)}else{if(b1!=
L) {b5[b2]=b1}}}}return b5};bF.extend({noConflict:function(b0) {if(bb.$===bF) {
bb.$=bH}if(b0&&bb.jquery===bF) {bb.jquery=bU}return bF},isReady:false,readyWait:1,holdReady:function(
b0) {if(b0) {bF.readyWait++}else{bF.ready(true)}},ready:function(b0) {if(b0==true&&!-bF.readyWait)|| (b0!
=true&&!bF.isReady) {if(!av.body) {return setTimeout(bF.ready,1)}bF.isReady=true;if(b0!
=true&&-bF.readyWait>0) {return}bC.fireWith(av,(bF));if(bF.fn.trigger(bF(av).trigger("ready").off("ready"))},bindReady:function() {
if(bC) {return}bC=bF.Callbacks("once memory");if(av.readyState=="complete") {return setImmediate(bF.ready,1)}if(
av.addEventListener) {av.addEventListener("DOMContentLoaded",e,false);bb.addEventListener("load",bF.ready,
false)}else{if(av.attachEvent) {av.attachEvent("onreadystatechange",e);bb.attachEvent("onload",bF.ready);var
b0=false;try {b0=bb.frameElement==null}catch(b1) {}if(av.documentElement.doScroll&&b0) {bW()}}},isFunction:
function(b0) {return bF.type(b0)=="function"},isArray:Array.isArray||function(b0) {return bF.type(b0)=="
array"},isWindow:function(b0) {return b0&&typeof b0=="object"&&"setInterval" in b0},isNumeric:function(b0) {
return !isNaN(parseFloat(b0))&&isFinite(b0)},type:function(b0) {return b0==null?String(b0):bx[bL.call(b0)]||"
object"},isPlainObject:function(b2) {if(!b2||bF.type(b2)!="object"||b2.nodeType||bF.isWindow(b2)) {return false}
try{if(b2.constructor&&!bG.call(b2,"constructor")&&!bG.call(b2.constructor.prototype,"isPrototypeOf")) {return
false}}catch(b1) {return false}var b0;for(b0 in b2) {return b0==L||bG.call(b2,b0)},isEmptyObject:function(
b1) {for(var b0 in b1) {return false}return true},error:function(b0) {throw new Error(b0)},parseJSON:function(b0)
) {if(typeof b0!="string"||!b0) {return null}b0=bF.trim(b0);if(bb.JSON&&bb.JSON.parse) {return
bb.JSON.parse(b0)}if(bN.test(b0.replace(bW,"@").replace(bP,""))) .replace(bJ,"")) {return new Function("
return "+b0)())}bF.error("Invalid JSON: "+b0)},parseXML:function(b2) {var b0,b1;try{if(
bb.DOMParser) {b1=new DOMParser();b0=b1.parseFromStream(b2,"text/xml")}else{b0=new ActiveXObject("
Microsoft.XMLDOM");b0.async="false";b0.loadXML(b2)}catch(b3) {b0=L}if(!b0||b0.documentElement||b0.
getElementsByTagName("parsererror").length) {bF.error("Invalid XML: "+b2)}return b0},noop:function() {},globalEval:functio
) {if(b0&&bM.test(b0)) {bb.execScript(function(b1) {bb["eval"].call(bb,b1)})(b0)},camelCase:function(
b0) {return b0.replace(bZ,"ms-").replace(bB,bT)},nodeName:function(b1,b0) {return b1.nodeName&&b1.nodeName.
toUpperCase()===b0.toUpperCase()},each:function(b3,b6,b2) {var b1,b4=0,b5=b3.length,b0=b5===
L||bF.isFunction(b3);if(b2) {if(b0) {for(b1 in b3) {if(b6.apply(b3[b1],b2)===false) {break}}else{for(;b4<b5;)
{if(b6.apply(b3[b4++],b2)===false) {break}}}}else{if(b0) {for(b1 in b3) {if(b6.call(b3[b1],b1,b3[b1])===false)
{break}}}}else{for(;b4<b5;) {if(b6.call(b3[b4],b4,b3[b4++])===false) {break}}}}return b3},trim:b0?function(b0) {
return b0==null?"":b0.call(b0):function(b0) {return b0==null?"":b0.toString().replace(bI,"").replace(bE,"")},
makeArray:function(b3,b1) {var b0=b1||[];if(b3!=null) {var b2=bF.type(b3);if(b3.length==null||b2=="string"||
b2=="function"||b2=="regexp"||bF.isWindow(b3)) {b2.call(b0,b3)}else{bF.merge(b0,b3)}return b0},inArray:
function(b2,b3,b1) {var b0;if(b3) {if(bv) {return bv.call(b3,b2,b1)}b0=b3.length;b1=b1?b1<0?Math.max(0,b0+b1):b1:0;
for(;b1<b0;b1++) {if(b1 in b3&&b3[b1]===b2) {return b1}}return -1},merge:function(b4,b2) {var b3=b4.length,b1=
0;if(typeof b2.length=="number") {for(var b0=b2.length;b1<b0;b1++) {b4[b3++] =b2[b1]}}else{while(b2[b1]!
=L) {b4[b3++] =b2[b1++]}b4.length=b3;return b4},grep:function(b1,b6,b0) {var b2=[],b5,b0=!b0;for(var b3=0,b4
=b1.length;b3<b4;b3++) {b5=!b6(b1[b3],b3);if(b0!
=b5) {b2.push(b1[b3])}return b2},map:function(b0,b7,b8) {var
b5,b6,b4=[],b2=0,b1=b0.length,b3=b0 instanceof bF||b1!=L&&typeof b1=="number"&&(b1>0&&b0[0]&&b0[b1-1]
)||b1==0||bF.isArray(b0);if(b3) {for(;b2<b1;b2++) {b5=b7(b0[b2],b2,b8);if(b5!=null) {b4[b4.length]=b5}}else{
for(b6 in b0) {b5=b7(b0[b6],b6,b8);if(b5!=null) {b4[b4.length]=b5}}return b4.concat.apply([],b4)},guid:1,proxy:
function(b4,b3) {if(typeof b3=="string") {var b2=b4[b3];b3=b4[b3];if(!bF.isFunction(b4)) {return
L}}var b0=bK.call(arguments,2),b1=function() {return b4.apply(b0,b0.concat(bK.call(arguments)))};b1.guid=b4.
guid=b4.guid||b1.guid||bF.guid++;return b1},access:function(b0,b8,b6,b2,b5,b7) {var b1=b0.length;if(typeof b8
=="object") {for(var b3 in b8) {bF.access(b0,b3,b8[b3],b2,b5,b6)}return b0}if(b6!=
L) {b2=!b7&&b2&&bF.isFunction(b6);for(var b4=0;b4<b1;b4++) {b5(b0[b4],b8,b2?b6.call(b0[b4],b4,b5(b0[b4],b8)
):b6,b7)}return b0}return b1?b5(b0[0],b8):L},now:function() {return(new Date()).getTime()},uaMatch:function(
b1) {b1=b1.toLowerCase();var b0=by.exec(b1)||bR.exec(b1)||bQ.exec(b1)||b1.indexOf("compatible")<0&&bS.exec(b1)
||[];return{browser:b0[1]||"",version:b0[2]||"0"}},sub:function() {function b0(b3,b4) {return new b0.fn.init(

```

```

b3,b4)}bF.extend(true,b0,this);b0.superclass=this;b0.fn=b0.prototype=this();b0.fn.constructor=b0;b0.sub=this.
sub;b0.fn.init=function b2(b3,b4){if(b4&&b4 instanceof bF&&!b4 instanceof b0){b4=b0(b4)}return bF.fn.init.
call(this,b3,b4,b1)};b0.fn.init.prototype=b0.fn;var b1=b0(av);return b0};bF.each("Boolean
Number String Function Array Date RegExp Object".split(" "),function(b1,b0){bX["[object "+b0+"]"]=b0.toLowerCase(
)});bv=bF.uaMatch(bX);if(bv.browser){bF.browser[bv.browser]=true;bF.browser.version=bv.version;if(bF.browser.
webkit){bF.browser.safari=true}if(bM.test("\xA0+");bI=/^\[\s\xA0+\$/;bE=/\[\s\xA0+\$/;bD=bF(av);if(av.
addEventListener){e=function(){av.removeEventListener("DOMContentLoaded",e,false);bF.ready()}else{if(av.attachEvent
){e=function(){if(av.readyState==="complete"){av.detachEvent("onreadystatechange",e);bF.ready()}}function
bw(){if(bF.isReady){return}try{av.documentElement.doScroll("left")}catch(b0){setTimeout(bw,1);return}bF.
ready()}return bF}});var a2={};function X(e){var bv=a2[e]={},bw,bx;e=e.split(/\s+/);for(bw=0,bx=e.length;bw<bx;
bw++){bv[e[bw]]=true}return bv}.Callbacks=function(bw){bw=bw?(a2[bw]||X(bw)):{};var bB=[],bC=[],bx,by,bv,
bz,bA,bE=function(bF){var bG,bJ,bI,bH,bK;for(bG=0,bJ=bF.length;bG<bJ;bG++){bI=bF[bG];bH=
b.type(bI);if(bH==="array"){bE(bI)}else{if(bH==="function"){if(!bw.unique||!bD.has(bI)){bB.push(bI)}}},e
=bFunction(bG,bF){bF=bF||[];bx=!bw.memory||[bG,bF];by=true;BA=bv||0;bv=0;bz=bB.length;for(;bB&&bA<bz;bA++){if
(bB[bA].apply(bG,bF)==false&&bw.stopOnFalse){bx=true;break};by=false;if(bB){if(!bw.once){if(bC&&bC.length){
bx=bC.shift();bD.fireWith(bx[0],bx[1])}else{if(bx===true){bD.disable()}else{bB=[]}}},bD={add:function(){if
(bB){var bB=bB.length;bE(arguments);if(by){bz=bB.length;else{if(bx&&bx!==true){bv=bF;e(bx[0],bx[1])}}return
this},remove:function(){if(bB){var bF=arguments,bH=0,bI=bF.length;for(;bH<bI;bH++){if(bC&&bC.length){
length;bG++}{if(bF[bH]==bB[bG]){if(by){if(bG<=bz){bz--;if(bG<=bA){bA--}}bB.splice(bG--,1);if(bw.unique){break}
}}return this},has:function(bG){if(bB){var bF=0,bH=bB.length;for(;bF<bH;bF++){if(bG===bB[bF]){return true}}
return false},empty:function(){bB=[];return this},disable:function(){bB=bC=bx=L;return this},disabled:
function(){return !bB},lock:function(){bC=L;if(!bx||bx===true){bD.disable()}return this},locked:function(){
return !bC},fireWith:function(bG,bF){if(bC){if(by){if(!bw.once){bC.push([bG,bF])}else{if(!bw.once&&bx)}(e(bG,
bF))}return this},fire:function(){bD.fireWith(this,arguments);return this},fired:function(){return !!bx}};
return bD};var aJ=[];b.extend({Deferred:function(bv){var bx=b.Callbacks("once memory"),bw=
b.Callbacks("once memory"),bv=b.Callbacks("memory"),e="pending",bA={resolve:bx,reject:bw,notify:bv},bC={
done:bx.add,fail:bw.add,progress:bv.add,state:function(){return e},isResolved:bx.fired,isRejected:bv.fired,
then:function(bE,bD,bF){bB.done(bE).fail(bD).progress(bF);return this},always:function(){bB.done.apply(bB,
arguments).fail.apply(bB,arguments);return this},pipe:function(bF,bE,bD){return bF.Deferred(function(bG){
b.each({done:[bF,"resolve"],fail:[bE,"reject"],progress:[bD,"notify"]},function(bI,bL){var bH=bL[0],bK=bL[
1],bJ;if(b.isFunction(bH)){bB[bI](function(){bJ=bH.apply(this,arguments);if(bJ&&
b.isFunction(bJ.promise)){bJ.promise().then(bG.resolve,bG.reject,bG.notify)}else{bG[bK+"With"](this===bB?
bG:this,[bJ])}}else{bB[bI](bG[bK])}})).promise();},promise:function(){if(bE===null){bE=bC}else{for(var bD
in bC){bE[bD]=bC[bD]}}return bE},bB=bC.promise({}),bz;for(bz in bA){bB[bz]=bA[bz].fire;bB[bz+"With"]=bA[bz]
.fireWith}bB.done(function(){e="resolved"},bw.disable,bv.lock).fail(function(){e="rejected"},bx.disable,bv.
lock);if(by){by.call(bB,bB)}return bB},when:function(bA){var bx=aJ.call(arguments,0),bv=0,e=bx.length,bB=new
Array(e),bw=e,by=e,bC=e<=1&&bA&&b.isFunction(bA.promise)?bA.Deferred():bE=bC.promise();function bD(bF){
return function(bG){bx[bF]=arguments.length>1?aJ.call(arguments,0):bG;if(!(--bw)){bC.resolveWith(bC,bx)}}
function bz(bF){return function(bG){bB[bF]=arguments.length>1?aJ.call(arguments,0):bG;bC.notifyWith(bE,bB)}}if
(e>1){for(;bv<e;bv++){if(bx[bv]&&bx[bv].promise&&b.isFunction(bx[bv].promise)){bx[bv].promise().then(bD(bv),
bC.reject,bz(bv))}else{--bw}}if(!bw){bC.resolveWith(bC,bx)}}else{if(bC!==bA){bC.resolveWith(bC,e?[bA]:[])}}
return bE}});b.support=(function(){var bJ,bI,bF,bG,bx,bE,bA,bD,bz,bK,bB,by,bw,bv=av.createElement("div"),bH=
av.documentElement,bv.setAttribute("className","t");bv.innerHTML="<link/><table/><a href='/a'
style='top:1px;float:left;opacity:.55;'><a/><input type='checkbox'/'>;bI=bv.getElementsByTagName("*");bF=bv.
getElementsByName("a")[0];if(!bI||!bI.length||!bF){return}bJG=av.createElement("select");bJG.appendChild(
av.createElement("option"));bE=bv.getElementsByTagName("input")[0];bJ=leadingWhitespace(bv.firstChild.
nodeType===3),tbody:!bv.getElementsByTagName("tbody").length,htmlSerialize:!bv.getElementsByTagName("link").
length,style/top/.test(bF.getAttribute("style")),hrefNormalized:(bF.getAttribute("href")==="/a"),opacity/^0.5
5/.test(bF.style.opacity),cssFloat:!bF.style.cssFloat,checkOn:(bE.value==="on"),optSelected:bx.selected,
getSetAttribute:bv.className==="t",enctype:!av.createElement("form").enctype,html5Clone:av.createElement("nav"
).cloneNode(true).outerHTML==="<nav></nav>",submitBubbles:true,changeBubbles:true,focusInBubbles:false,
deleteExpando:true,noCloneEvent:true,inlineBlockNeedsLayout:false,shrinkWrapBlocks:false,reliableMarginRight:
true);bE.checked=true;bJ.noCloneChecked=bE.cloneNode(true).checked;bG.disabled=true;bJ.optDisabled=!bx.
disabled;try{delete bv.test}catch(bC){bJ.deleteExpando=false}if(!bv.addEventListener&&bv.fireEvent){
bv.attachEvent("onclick",function(){bJ.noCloneEvent=false});bv.cloneNode(true).fireEvent("onclick")}bE=av.
createElement("input");bE.value="t";bE.setAttribute("type","radio");bJ.radioValue=bE.value==="t";bE.
setAttribute("checked","checked");bv.appendChild(bE);bD=av.createDocumentFragment();bD.appendChild(bv.lastChild);bJ.
checkClone=bD.cloneNode(true).cloneNode(true).lastChild.checked;bJ.appendChild(bE.checked);bD.removeChild(bE
);bD.appendChild(bv);bv.innerHTML="";if(bb.getComputedStyle){bA=av.createElement("div");bA.style.width="0"
;bA.style.marginRight="0";bv.style.width="2px";bv.appendChild(bA);bJ.reliableMarginRight=(parseInt((
bb.getComputedStyle(bA,null)||{marginRight:0}).marginRight,10)||0)===0}if(bv.attachEvent){for(by in {
submit:1,change:1,focusin:1}){bB="on"+by;bw=(bB in bv);if(!bw){bv.setAttribute(bB,"return");bw=(typeof bv[bB]==
"function")?bJ[by+"Bubbles"]=bw}bD.removeChild(bv);bD.removeChild(bv);b(function(){var bM,bU,bV,bT,bN
,bO,bL,bS,bR,e,bP,bQ=av.getElementsByTagName("body")[0];if(!bQ){return}bL=1;bS="
position:absolute;top:0;left:0;width:1px;height:1px;margin:0";bR="visibility:hidden;border:0";e="style='"+bS+"border:
#000;padding:0';bP="<div "+e+"><div></div></table><table "+e+" cellpadding='0'
cellspacing='0'><tr><td></td></tr></table>";bM=av.createElement("div");bM.style.cssText=bR+width:0;height:0;position:
bL+"px";bQ.insertBefore(bM,bQ.firstChild);bv=av.createElement("div");bM.appendChild(bv);bv.innerHTML="
<table><tr><td style='padding:0;border:0;dislay:none'></td><td></td></tr></table>";bz=bv.getElementsByTagName("
td");bw=(bz[0].offsetHeight===0);bz[0].style.display="";bz[1].style.display="none";bJ.reliableHiddenOffsets=
bw&&(bz[0].offsetHeight===0);bv.innerHTML="";bv.style.width=bv.style.paddingLeft="1px";
b.boxModel=bJ.boxModel=bv.offsetWidth===2;if(typeof bv.style.zoom!=="undefined"){bv.style.display="inline"
};bv.style.zoom=1;bJ.inlineBlockNeedsLayout=(bv.offsetWidth===2);bv.style.display="";bv.innerHTML="<div
style='width:4px;'></div>";bJ.shrinkWrapBlocks=(bv.offsetWidth===2)}bv.style.cssText=bS+bR;bv.innerHTML=bP;bU=bv.
firstChild;bV=bU.firstChild;bN=bU.nextSibling.firstChild.firstChild;bO={doesNotAddBorder:(bv.offsetTop===5),
doesAddBorderForTableAndCells:(bN.offsetTop===5)};bv.style.position="fixed";bv.style.top="20px";bO.
fixedPosition=(bv.offsetTop===20)||bv.offsetTop===15);bv.style.position=bV.style.top="";bU.style.overflow="hidden";bU.
style.position="relative";bO.subtractsBorderForOverflowNotVisible=(bv.offsetTop===-5);bO.
doesNotIncludeMarginInBodyOffset=(bQ.offsetTop===bL);bQ.removeChild(bM);bv=bM=null;b.extend(bJ,bO)});var a
:\(.*)|[\s/]{.+/S/,aA=/[A-Z]/g;b.extend({cache:{},uid:0,expando:"jQuery"+(b.fn.jquery+Math.random()).
replace(/\D/g,""),noData:{embed:true,object:"clsid:D27CDB6E-AE6D-11cf-96B8-444553540000",applet:true},hasData:
function(e){e=e.nodeType?b.cache[e[b.expando]]:[e[b.expando]];return !e&&!S(e)},data:function(bx,bv,bz,
by){if(!b.acceptData(bx)){return}var bG,bA,bD,bE=b.expando,bC=typeof bv==="string",bF=bx.nodeType,bF=bF?
b.cache[bx]:bv=bF?bx[bE]:bx[bE]&&bE,bB=bv==="events";if(!bI||!e[bw]||(!bB&&!by&&!e[bw].data)&&bC&&bz===
L){return}if(!bw){if(bF){bx[bE]=bw+++b.uid++;e[bw]=e[bw]||{};if(!bF[e[bw]].toJSON=
b.noop){if(typeof bv==="object"||typeof bv==="function"){if(bI){e[bw][e[bw],bv]}else{e[bw].data=
b.extend(e[bw].data,bv)}}bG=bA=e[bw];if(!by){if(!bA.data){bA.data={}}bA=bA.data}if(bz!=""

```

```

L) {bA[b.camelCase(bv)] = bz; if (bB && !bA[bv]) {return bG.events} if (bC) {bD = bA[bv]; if (bD == null) {bD = bA[
b.camelCase(bv)]}} else {bD = bA} return bD; }, removeData: function (bx, bv, by) { if (!b.acceptData(bx)) {return} var bB
, bA, bz, bC = b.expando, bD = bx.nodeType, e = bD ? b.cache : bx, bw = bD ? bx[bC] : bC; if (!e[bw]) {return} if (bv) {bB = by ? e[bw] : e[
bw].data; if (bB) {if (!b.isArray(bv)) {if (bv in bB) {bv = [bv]} else {bv = b.camelCase(bv); if (bv in bB) {bv = [bv]} else {
bv = bv.split(" ")}} for (bA = 0, bz = bv.length; bA < bz; bA++) {delete bB[bv[bA]]} if (! (by ? S :
b.isEmptyObject(bB))) {if (!by) {delete e[bw].data; if (!S(e[bw])) {return} if (
b.support.deleteExpando || !e.setInterval) {delete e[bw]} else {e[bw] = null} if (bD) {if (
b.support.deleteExpando) {delete bx[bC]} else {if (bx.removeAttribute) {bx.removeAttribute(bC)} else {bx[bC] = null
}}}; }, _data: function (bv, e, bw) {return b.data(bv, e, bw, true)}}, acceptData: function (bv) {if (bv.nodeName) {var e =
b.noData[bv.nodeName.toLowerCase()]; if (e) {return ! (e === true || bv.getAttribute("classid") !== e)}}} return true}
}); b.fn.extend({data: function (by, bA) {var bB, e, bw, bz = null; if (typeof by === "undefined") {if (this.length) {bz =
b.data(this[0], by); if (this[0].nodeType === 1 && !b._data(this[0], "parsedAttrs", true)) {return bz} else {if (typeof by === "object") {return this.
each(function () {b.data(this, by)}); } bB = by.split("."); bB[1] = bB[1] ? "." + bB[1] : ""; if (bA ===
L) {bz = this.triggerHandler("getData" + bB[1] + "!", [bB[0]]); if (bz === L && this.length) {bz =
b.data(this[0], by); bz = a5(this[0], by, bz)} return bz === L && bB[1] ? this.data(bB[0]) : bz} else {return this.
each(function () {var bC = b(this), bD = [bB[0], bA]; bC.triggerHandler("setData" + bB[1] + "!", bD);
b.data(this, by, bA); bC.triggerHandler("changeData" + bB[1] + "!", bD)}); }, removeData: function (e) {return this.
each(function () {b.removeData(this, e)}); }); function a5 (bx, bw, by) {if (by === L && bx.nodeType === 1) {var bv =
data -> +bw.replace(aA, "-$1").toLowerCase(); bv = bx.getAttribute(bv); if (typeof bv === "string") {try {bv = by === "true" ?
true : by === "false" ? false : by === "null" ? null : b.isNumeric(bv) ? parseFloat(bv) : aS.test(bv) ?
b.parseJSON(bv) : by} catch (bz) {} b.data(bx, bw, bv)} else {bv = by} return bv} function
S (bv) {for (var e in bv) {if (e === "data" && b.isEmptyObject(bv[e])) {continue} if (e !== "toJSON") {return false}}
return true} function bi (by, bx, bA) {var bw = bx + "defer", bv = bx + "queue", e = bx + "mark", bz = b._data (by, bw); if (bz && (bA ===
queue || !b._data (by, bv)) && (bA === "mark" || !b._data (by, e))) {setTimeout (function () {if (!
b._data (by, bv) && !b._data (by, e)) {b.removeData (by, bw, true); b.fire ()}, 0)}; } b.extend ({_mark: function (bv, e)
{if (bv) {e = (e || "fx") + "mark"; b._data (bv, e, (b._data (bv, e) || 0) + 1)}}, _unmark: function (by, bx, bv) {if (bv !== true) {bv =
bx; bx = by; by = false} if (bx) {bv = bv || "fx"; var e = bv + "mark", bw = bv ? ((b._data (bx, e) || 1) - 1); if (bw) {b._data (bx, e, bw)
} else {b.removeData (bx, e, true); bi (bx, bv, "mark")}}; queue: function (bv, e, bx) {var bw, bv; if (e || "fx" + "queue";
bw = b._data (bv, e); if (bx) {if (!bw || b.isArray (bx)) {bw = b._data (bv, e, b.makeArray (bx))} else {bw.push (bx)} return
bw} || [], dequeue: function (by, bx) {bx = bx || "fx"; var bv = b.queue (by, bx), bw = bv.shift (), e = {}; if (bw === "inprogress")
{bw = bv.shift ()} if (bw) {if (bx === "fx") {bv.unshift ("inprogress")}; b._data (by, bx + ".run", "e"); bw.call (by, function () {
b.dequeue (by, bx)}, e)} if (!bv.length) {b.removeData (by, bx + "queue" + "run", true); bi (by, bx, "queue")}});
b.fn.extend ({queue: function (e, bv) {if (typeof e === "string") {bv = e; e = "fx"} if (bv === L) {return
b.queue (this[0], e)} if (! (e === "fx")) {return this.each (function () {var bw = b.queue (this, e, bv); if (e === "fx" && bw[0] !==
inprogress") {b.dequeue (this, e)}); }, dequeue: function (e) {return this.each (function () {
b.dequeue (this, e)}); }, delay: function (bv, e) {bv = b.fx ? b.fx.speeds [bv] || bv : bv; e = e || "fx"; return this.queue (e,
function (bx, bv) {var by = setTimeout (bv, e); bw.stop (function () {clearTimeout (by)}); }, clearQueue: function (e) {
return this.queue (e || "fx", []), promise: function (bD, bw) {if (typeof bD !== "string") {bw = bD; bD =
L} bD = bD || "fx"; var e = b.Deferred (), bv = this, by = bv.length, bB = 1, bz = bD + "defer", bA = bD + "queue", bC = bD + "mark", bx;
function bE () {if (! (e === "fx")) {e.resolveWith (bv, [bv])} while (by--) {if ((bx = b.data (bv [by], bz,
L, true) || (b.data (bv [by], bA, L, true) || b.data (bv [by], bC, L, true)) && b.data (bv [by], bz,
b.callbacks ("once memory"), true)) {bB++; bx.add (bE)} return e.promise ()}; var aP = /\n\t\r/g, aF = /\s+/
, aU = /\r/g, g = /(?:button|input)$/i, D = /(?:button|input|object|select|textarea)$/i, l = /a(?:rea)?$/i, aO = /(?:autofocus|autoplay|async|checked|controls|defer|disabled|hidden|loop|multiple|
open|readonly|required|scoped|selected)$/i, F = b.support.getSetAttribute, be, aY, aF;
b.fn.extend ({attr: function (e, bv) {return b.access (this, e, bv, true, b.attr)}, removeAttr: function (e) {return thi
s.each (function () {b.removeAttr (this, e)}), prop: function (e, bv) {return b.access (this, e, bv, true,
b.prop)}, removeProp: function (e) {e = b.propFix [e] || e; return this.each (function () {try {this [e] =
L} delete this [e]} catch (bv) {}))}, addClass: function (by) {var bA, bB, bv, bx, bz, bB, e; if (
b.isFunction (by)) {return this.each (function (bC) {b (this).addClass (by.call (this, bC, this.className))}})
if (by && typeof by === "string") {bA = by.split (aF); for (bB = 0, bv = this.length; bB < bv; bB++) {bx = this [bB]; if (bx.nodeType ===
1) {if (!bx.className && bA.length === 1) {bx.className = by} else {bz = " " + bx.className + " "; for (bB = 0, e = bA.length; bB < e; bB++) {bz += bA [bB] + " ";} bx.className = b.trim (bz)}}} return this}, removeClass:
function (bz) {var bA, bB, bv, by, bx, bB, e; if (b.isFunction (bz)) {return this.each (function (bC) {
b (this).removeClass (bz.call (this, bC, this.className))}}) if (bz && typeof bz === "string") || bz ===
L) {bA = (bz || "").split (aF); for (bB = 0, bv = this.length; bB < bv; bB++) {by = this [bB]; if (by.nodeType === 1 && by.className)
{if (bz) {bx = (" " + by.className + " ").replace (aP, " "); for (bB = 0, e = bA.length; bB < e; bB++) {bx = bx.replace (" " + bA [bB] +
" ", " "); by.className = b.trim (bx)} else {by.className = ""}} return this}, toggleClass: function (bx, bv) {var bw =
typeof bx, e = typeof bv === "boolean"; if (b.isFunction (bx)) {return this.each (function (by) {
b (this).toggleClass (bx.call (this, by, this.className, bv), bv)}); } return this.each (function () {if (bv ===
"string") {var bA, bB = 0, by = b (this), bB = bv, bC = bx.split (aF); while (bA = bC [bB++]) {bB = e ? bB : !by.hasClass (bA); by [bB] ?
addClass : "removeClass"} (bA)} else {if (bv === "undefined" || bv === "boolean") {if (this.className) {
b._data (this, "__className__", this.className)} this.className = this.className || bx === false ? "" :
b._data (this, "__className__") || ""}}}, hasClass: function (e) {var bx = " " + e + " ", bw = 0, bv = this.length; for (; bw <
bv; bw++) {if (this [bw].nodeType === 1 && (" " + this [bw].className + " ").replace (aP, " ").indexOf (bx) > -1) {return true}}
return false}, val: function (bx) {var e, bv, by, bw = this[0]; if (!arguments.length) {if (bw) {e =
b.valHooks [bw.nodeName.toLowerCase ()] || b.valHooks [bw.type]; if (e && "get" in e && (bv = e.get (bw, "value")) !==
L) {return bv} bv = bw.value; return typeof bv === "string" ? bv.replace (aU, "") : bv === null ? "" : bv} return bv; }
if (b.isFunction (bx); return this.each (function (bA) {var bz = b (this), bB; if (this.nodeType === 1) {return} if (by) {
bB = bx.call (this, bA, bv.val ())} else {bB = bx} if (bB === null) {bB = ""} else {if (typeof bB === "number") {bB += ""} else {if (
b.isArray (bB)) {bB = b.map (bB, function (bC) {return bC === null ? "" : bC + ""}})} e = b.valHooks [this.nodeName.
toLowerCase ()] || b.valHooks [this.type]; if (!e || ("set" in e) || e.set (this, bB, "value") === L) {this.value = bB}}});
b.extend ({valHooks: {option: {get: function (e) {var bv = e.attributes.value; return !bv || bv.specified ? e.value : e.
text}}, select: {get: function (e) {var bA, bv, bz, bx, by = e.selectedIndex, bB = [], bC = e.options, bw = e.type === "select-one"
; if (by < 0) {return null} bv = bw ? by : 0; bz = bw ? by + 1 : bC.length; for (; bv < bz; bv++) {bx = bC [bv]; if (bx.selected && (
b.support.optDisabled ? !bx.disabled : bx.getAttribute ("disabled") === null) && !bx.parentNode.disabled || !
b.nodeName (bx.parentNode, "optgroup")) {bA = b (bx).val (); if (bw) {return bA} bB.push (bA)} if (bw && !bB.length && bC
.length) {return b (bC [by]).val ()} return bB}, set: function (bv, bw) {var e = b.makeArray (bw);
b (bv).find ("option").each (function () {this.selected = b.inArray (b (this).val (), e) > 0}); if (!e.length) {bv.
selectedIndex = -1} return e}}, attrFn: {val: true, css: true, html: true, text: true, data: true, width: true, height: true,
offset: true}, attr: function (bA, bx, bB, bv) {var bw, e, by, bv = bA.nodeType; if (!bA [bA] | bv === 3 || bv === 8 || bv === 2) {return} if (
bz && bx in b.attrFn) {return b (bA) [bx] (bB)} if (typeof bA.getAttribute === "undefined") {return
b.prop (bA, bx, bB)} by = bv !== 1 || !b.isXMLDoc (bA); if (by) {bx = bx.toLowerCase (); e = b.attrHooks [bx] || (ao.test (bx) ?
aY : be)} if (bv = L) {if (bB === null) {b.removeAttr (bA, bx); return} else {if (e && "set" in e && "set" (bA, bB, bv)
) !== L) {return bw} else {bA.setAttribute (bx, "" + bB); return bB}} else {if (e && "get" in e && by && (bw = e.get (bA, bx)) !==

```



```

null) {return bw} else {bw=bA.getAttribute(bx); return bw===null?L:bw}}, removeAttr: function(bx, bz) {var by, bA, bv,
e, bw=0; if (bz && bx.nodeType===1) {bA=bz.toLowerCase().split(af); e=bA.length; for (; bw<e; bw++) {bv=bA[bw]; if (bv) {by
=b.propFix[bv] || bv; b.attr(bx, bv, ""); bx.removeAttribute(b?vv:by); if (ao.test(bv) && by in bx) {bx[by]=false}}
}, attrHooks: {type: {set: function(e, bv) {if (g.test(e.nodeName) && e.parentNode) {b.error("type property can't be
changed")}; else {if (!b.support.radioValue && bv=== "radio" && b.nodeName(e, "input")) {var bw=e.value; e.
setAttribute("type", bv); if (bw) {e.value=bw} return bv}}}, value: {get: function(bv, e) {if (be && b.nodeName(bv, "button")) {
return be.get(bv, e)} return e in bv?vv.value: null}}, set: function(bv, bw, e) {if (be && b.nodeName(bv, "button")) {return
be.set(bv, bw, e) | bv.value=bw}}, propFix: {tabindex: "tabIndex", readonly: "readOnly", "for": "htmlFor", "class": "
className", maxLength: "maxLength", cellSpacing: "cellSpacing", cellPadding: "cellPadding", rowspan: "rowSpan", colspan:
"colSpan", useMap: "useMap", frameborder: "frameBorder", contentEditable: "contentEditable"}, prop: function(bz, bx,
bA) {var bw, e, by, bv=bz.nodeType; if (!bz || bv===3 || bv===8 || bv===2) {return} by=bv===1 || !
b.isXMLDoc(bz); if (by) {bx=b.propFix[bz] || bx; e=b.propHooks[bz]}; if (bA !== L) {if (e && "set" in e && (bw=e.set(bz,
bA, bx)) !== L) {return bw} else {return (bz[bz]=bA)} } else {if (e && "get" in e && (bw=e.get(bz, bx)) !== null) {return bw}
else {return bz[bz]}}}, propHooks: {tabIndex: {get: function(bv, e) {var e=bv.getAttributeNode("tabindex"); return e &&
e.specified? parseInt(e.value, 10) : D.test(bv.nodeName) || L.test(bv.nodeName) && bv.href??:
L}}}; b.attrHooks.tabIndex=b.propHooks.tabIndex; aY={get: function(bv, e) {var bx, bw=
b.prop(bv, e); return bw===true || typeof bw!=="boolean" && (bx=bv.getAttributeNode(e)) && bx.nodeValue!==false?e.
toLowerCase(): L}; set: function(bv, bx, e) {var bw; if (bx===false) {b.removeAttribute(bv, e)} else {bw=
b.propFix[e] || e; if (bw in bv) {bv[bw]=true} bv.setAttribute(e, e.toLowerCase())} return e}; if (!F) {aF={name:
true, id: true}; be=b.valHooks.button={get: function(bw, bv) {var e; e=bw.getAttributeNode(bv); return e && (aF[bv]?e.
nodeValue!=="": e.specified?e.nodeValue: L}; set: function(bw, bx, bv) {var e=bw.getAttributeNode(bv); if (!e) {e=av.
createAttribute(bv); bw.setAttributeNode(e)} return (e.nodeValue=bx+"")}; b.attrHooks.tabIndex.set=be.set;
b.each(["width", "height", "function(bv, e) {b.attrHooks[e]=b.extend(b.attrHooks[e], {set: function(bw, bx) {if
(bx=== "auto") {bw.setAttribute(e, "auto"); return bx}}}; b.attrHooks.contentEditable={get: be.get, set: function(bv,
bw, e) {if (bw=== "false") {be.set(bv, bw, e)} } if (!b.support.hrefNormalized) {b.each(["href", "src", "width",
"height"], function(bv, e) {b.attrHooks[e]=b.extend(b.attrHooks[e], {get: function(bx) {var bw=bx.getAttribute
(e, 2); return bw===null?L:bw}})} } if (!b.support.style) {b.attrHooks.style={get: function() {return e.style.
cssText.toLowerCase() || L}; set: function(e, bv) {return (e.style.cssText=" "+bv)}} } if (!b.support.optSelected) {
b.propHooks.selected=b.extend(b.propHooks.selected, {get: function(bv) {var e=bv.parentNode; if (e) {e.selecte
dIndex; if (e.parentNode) {e.parentNode.selectedIndex} return null}}); if (!b.support.enctype) {
b.propFix.enctype="encoding"} if (!b.support.checkOn) {b.each(["radio", "checkbox"], function() {
b.valHooks[this]={get: function(e) {return e.getAttribute("value")===null?"on": e.value}}});
b.each(["radio", "checkbox"], function() {b.valHooks[this]=b.extend(b.valHooks[this], {set: function(e, bv) {if (b
.isArray(bv)) {return (e.checked=b.inArray(b(e).val(), bv)>=0)} } }); var bd=/^(?:text|input|select)$/
i, n=/(^[\s]*)(?:\s+)?$/i, J=/\bhover(\s+)?\b/, aO=/^key/, bF=/^(?:mouse|contextmenu|click/,
T=/(?!\s):focus(?:\s|outblurr)$/i, U=/^(w*)$/i, W=/(?!\s|+)$/i, Y=function(e) {var bv=U.exec
(e); if (bv) {bv[1]=(bv[1] || "").toLowerCase(); bv[3]=bv[3] && new RegExp("(?:^|\\s)"+bv[3]+"(?:\\s|$)")} return bv
}; j=function(bw, e) {var bv=bw.attributes || {}; return (!e[1] || bv.nodeName.toLowerCase()===e[1]) && (!e[2] || (bv.id
|| {})).value===e[2]) && (!e[3] || e[3].test((bv["class"] || {})).value)}; bt=function(e) {return
b.event.special.hover?e: e.replace(J, "mouseenter$1 mouseleave$1"); b.event={add: function(bx, bC, bJ, bA, by) {
var bD, bB, bK, bI, bH, bF, e, bG, bv, bz, bw, bE; if (bx.nodeType===3 || bx.nodeType===8 || !bC || !bJ || !
(bD=b._data(bx))) {return} if (bJ.handler) {bv=bJ.handler} if (bJ.guid) {bJ.guid=b.guid+bxK=bd.events; if (!bK
) {bD.events=bK={}; bB=bD.handle; if (!bB) {bD.handle=bB=function(bL) {return typeof bL!="undefined" && (!bL ||
b.event.triggered!==bL.type)} b.event.dispatch.apply(bB.elem, arguments): L}; bB.elem=bx} bC=
b.trim(bC).split(" "); for (bI=0; bI<bC.length; bI++) {bH=n.exec(bC[bI]) || []; bF=bH[1] || []; bE=bH[2] || ""}.split(
".").sort(); bE=b.event.special[bF] || {}; bF=(by?bE.delegateType: bE.bindType) || bF; bE=
b.event.special[bF] || {}; bG=b.extend({type: bF, origType: bH[1], data: bA, handler: bJ, guid: bJ, selector: by,
quick: Y(bY)}, {namespace: e.join(" ")}, bv); bw=bK[bF] || []; if (!bw) {bw=bK[bF]=[]; bw.delegateCount=0; if (!bE.setup || bE.
setup.call(bx, bA, e, bB)===false) {if (bx.addEventListener) {bx.addEventListener(bF, bB, false)} else {if (bx.attachEvent
) {bx.attachEvent("on"+bF, bB)}}} if (bE.add) {bE.add.call(bx, bG); if (!bG.handler.guid) {bG.handler.guid=bJ.guid}}
if (by) {bw.splice(bw.delegateCount++, 0, bG)} else {bw.push(bG)} b.event.global[bF]=bx=null, global: {},
remove: function(bJ, bE, bv, bH, bB) {var bI=b.hasData(bJ) && b._data(bJ), bF, bx, bz, bL, bC, bA, bG, bw, by, bK, bD, e; if (!bI || (
bw=bI.events)) {return} bE=b.trim(bE || "").split(" "); for (bF=0; bF<bE.length; bF++) {bx=
n.exec(bE[bF]) || []; bz=bL=bx[1]; bC=bx[2]; if (!bz) {for (bz in bw) {b.event.remove(bJ, bz+bE[bF], bv, bH, true)}
continue} by=b.event.special[bz] || {}; bz=(bH?by.delegateType: by.bindType) || bz; bD=bw[bz] || []; bA=bD.length; bC=bC?
new RegExp("(^|\\s)"+bC.split(" ").sort().join("\\.?(?:.*\\s)?")+"(\\s|$)": null; for (bG=0; bG<bD.length; bG++) {e
=bD[bG]; if ((bH || bL===e.origType) && (!bv || bv.guid===e.guid) && (!bC || bC.test(e.namespace) && (!bH || bH===e.
selector || bH=== "*" && e.selector)) {bD.splice(bG--, 1); if (e.selector) {bD.delegateCount--} if (by.remove) {by.remove.call(
bJ, e)} } if (bD.length===0 && bA!===bD.length) {if (!by.teardown || by.teardown.call(bJ, bC)===false) {
b.removeEvent(bJ, bz, bI.handle)} delete bw[bz] } if (bI.isEmptyObject(bw)) {bK=bI.handle; if (bK) {bK.elem=null}
b.removeData(bJ, ["events", "handle"], true)}, customEvent: {getData: true, setData: true, changedData: true}
}, trigger: function(bv, bD, bA, bJ) {if (bA && (bA.nodeType===3 || bA.nodeType===8)) {return} var bG=bv.type || bv, bx=[], e,
bw, bC, bH, bz, by, bF, bE, bB, bI; if (T.test(bG+bv.event.triggered)) {return} if (bG.indexOf(".")>=0) {bG=bG.slice(0, -1
); bw=true} if (bG.indexOf(".")>=0) {bx=bG.split("."); bG=bx.shift(); bx.sort()} if (!bA ||
b.event.customEvent[bG]) && !b.event.global[bG]) {return} bv=typeof bv=="object"?bv:
b.expand?bv: new b.Event(bG, bv) | bG; bv.type=bG; bv.isTrigger=true; bv.exclusive=bw; bv.
namespace=bx.join(" "); bv.namespace_re=bv.namespace?new RegExp("(^|\\s)"+bx.join("\\.?(?:.*\\s)?")+"(\\s|$)": null;
by=bG.indexOf(".")<0?"on"+bG:""; if (!bA) {e=b.cache; for (bC in e) {if (e[bC].events && e[bC].events[bG]) {
b.event.trigger(bv, bD, e[bC].handle.elem, true)} } return} bv.result=L; if (!bv.target) {bv.target=bA} bD!=null
?b.makeArray(bD) : []; bD.unshift(bv); bF=b.event.special[bG] || {}; if (bF.trigger && bF.trigger.apply(bA, bD)===
false) {return} bB=[bA, bF.bindType | bG]; if (!bJ && !bF.noBubble && !b.isWindow(bA)) {bI=bF.delegateType | bG; bH=
T.test(bI+bG)?bA.parentNode: bz=null; for (; bH; bH=bH.parentNode) {bB.push([bH, bI]); bz=bH} if (bz && bz===bA.
ownerDocument) {bB.push([bz.defaultView | bz.parentWindow | bb, bI])} for (bC=0; bC<bB.length && !bv.
isPropagationStopped() && bC++) {bH=bB[bC][0]; bv.type=bB[bC][1]; bE=(b._data(bH, "events") || {})[bv.type] &&
b._data(bH, "handle"); if (bE) {bE.apply(bH, bD)} bE=by && bH[by]; if (bE && bE.acceptData(bH) && bE.apply(bH, bD)===
false) {bv.preventDefault()} } bv.type=bG; if (!bJ && !bv.isDefaultPrevented()) {if (!bF.default || bF.default.apply(bA.
ownerDocument, bD)===false) && !bG=== "click" && bB.nodeName(bA, "a") && bB.acceptData(bA)) {if (by && bA[bG] && (bG!=="
focus" && bG!=="blur")) | bv.target.offsetWidth!==0 && !b.isWindow(bA)) {bz=bA[by]; if (bz) {bA[by]=null}
b.event.triggered=bG; bA[bG](); b.event.triggered=L; if (bz) {bA[by]=bz}} } return bv.result, dispatch:
function(e) {e=b.event.fix(e | bb.event); var bz=(e._data(this, "events") || {})[e.type] || []; bA=bz.delegateCount; bG
=[] .slice.call(arguments, 0), by=!e.exclusive && !e.namespace, bH=[], bC, bB, bK, bx, bF, bE, bv, bD, bI, bw, bJ; bG[0]=e; e.
delegateTarget=this; if (bA && !e.target.disabled && !e.button && e.type=== "click") {bx=b(this); bx.context=this.
ownerDocument | this; for (bK=e.target; bK!="" && bK!="" && bK.parentNode | this) {bE={}; bx[0]=bK; for (bC=0; bC<bA; bC++) {
bI=bz[bC]; bw=bI.selector; if (bE[bw]===L) {bE[bw]=bI.quick?j(bK, bI.quick): bx.is(bw)}
if (bE[bw]) {bD.push(bI)} } if (bD.length) {bH.push({elem: bK, matches: bD})} } if (bz.length > bA) {bH.push({elem: this
, matches: bz.slice(bA)})} } for (bC=0; bC<bH.length && !e.isPropagationStopped() && bC++) {bv=bH[bC]; e.currentTarget=bv.
elem; for (bB=0; bB<bv.matches.length && !e.isImmediatePropagationStopped() && bB++) {bI=bv.matches[bB]; if (by || (!e.

```

```

namespace&&!bI.namespace)||e.namespace_re&&e.namespace_re.test(bI.namespace))){e.data=bI.data;e.handleObj=bI;bf=
=((b.event.special[bI.origType]||{}).handle||bI.handler).apply(bv.elem,bG);if(bf!=
L){e.preventDefault;if(bf===false){e.stopPropagation()}};return e.result},props:"
attrChange attrName relatedNode srcElement altKey bubbles cancelable ctrlKey currentTarget eventPhase metaKey
relatedTarget shiftKey target timeStamp view which".split(" "),fixHooks:{},keyHooks:{props:"Char charCode key
keyCode".split(" "),filter:function(bv,e){if(bv.which===null){bv.which=e.charCode!=null?e.charCode:e.keyCode}
return bv}},mouseHooks:{props:"button buttons clientX clientY fromElement offsetX offsetY pageX pageY screenX
screenY toElement".split(" "),filter:function(bx,bw){var by,bz,e,bv=bw.button,bA=bw.fromElement;if(bx.pageX==
null&&bw.clientX!=null){by=bx.target.ownerDocument||av;bz=by.documentElement;e=by.body;bx.pageX=bw.clientX+(bz
&&bz.scrollLeft||e&&e.scrollLeft||0)-(bz&&bz.clientLeft||e&&e.clientLeft||0);bx.pageY=bw.clientY+(bz&&bz.
scrollTop||e&&e.scrollTop||0)-(bz&&bz.clientTop||e&&e.clientTop||0)}if(!bx.relatedTarget&&bA){bx.relatedTarget=
bA===bx.target?bA.toElement:bA}if(!bx.which&&bv===L){bx.which=(bv&1?1:(bv&2?3:(bv&4?2:0))}return bx}},fix:
function(bw){if(bw[b.expando]){return bw}var bv,bz,e=bw,bx=b.event.fixHooks[bw.type]||{};by=bx.props?this.
props.concat(bx.props):this.props;bw=b.Event(e);for(bv=by.length;bv;){bz=by[--bv];bw[bz]=e[bz]}if(!bw.
target){bw.target=e.srcElement||av}if(bw.target.nodeType===3){bw.target=bw.target.parentNode}if(bw.metaKey===
L){bw.metaKey=bw.ctrlKey}return bx.filter?bx.filter(bw,e):bw},special:{ready:{setup:
b.bindReady},load:{noBubble:true},focus:{delegateType:"focus"},blur:{delegateType:"mouseout"},
beforeunload:{setup:function(bw,bv,e){if(b.isWindow(this)){this.onbeforeunload=e};teardown:function(bv,e){if(this.
onbeforeunload===e){this.onbeforeunload=null}}}},simulate:function(bw,by,bx,bv){var bz=
b.extend(new b.Event(),bx,{type:bw,isSimulated:true,originalEvent:{}});if(bv){
b.event.trigger(bz,null,by)}else{b.event.dispatch.call(by,bz)}if(bz.isDefaultPrevented()){bx.
preventDefault()};b.event.handle=b.event.dispatch;b.removeEvent=av.removeEventListener?function(bv,e,bw){if(bv.
removeEventListener){bv.removeEventListener(e,bw,false)}:function(bv,e,bw){if(bv.detachEvent){bv.detachEvent("
on"+e,bw)};b.Event=function(bv,e){if(!this instanceof b.Event){return new b.Event(bv,e)}if(bv&&bv.type){
this.originalEvent=bv;this.type=bv.type;this.isDefaultPrevented=(bv.defaultPrevented||bv.returnValue===
false)||bv.getPreventDefault&&bv.getPreventDefault()};if(bz){b.extend(this,e)}this.
timeStamp=bv&&bv.timeStamp||b.now();if(this[b.expando]==true);function bk(){return false}function
i(){return true}b.Event.prototype={preventDefault:function(){this.isDefaultPrevented=
i;var bv=this.originalEvent;if(!bv){return}if(bv.preventDefault){bv.preventDefault()}else{bv.returnValue=
false}};stopPropagation:function(){this.isPropagationStopped=i;var bv=this.originalEvent;if(!bv){return}if(
bv.stopPropagation){bv.stopPropagation()}bv.cancelBubble=true;stopImmediatePropagation:function(){this.
isImmediatePropagationStopped=i;this.stopPropagation()};isDefaultPrevented:bk,isPropagationStopped:bk,
isImmediatePropagationStopped:bk};b.each({mouseenter:"mouseover",mouseleave:"mouseout"},function(bv,e){
b.event.special[bv]={delegateType:e,bindType:e,handle:function(bz){var bB=this,bA=bz.relatedTarget,by=bz.
handleObj,bw=by.selector,bx;if(!bA||!(bA===bB&&!bB.contains(bB,bA))){bz.type=by.origType;bx=by.handler.apply(
this,arguments);bz.type=e}return bx}});if(!b.support.submitBubbles){b.event.special.submit={setup:function(
){if(b.nodeName(this,"form")){return false}b.event.add(this,"click._submit keypress._submit",function(bx)
{var bw=bx.target,bv=b.nodeName(bw,"input")||b.nodeName(bw,"button")?bw.form:L;if(bv&&!bv.
_submit_attached){bv.event.add(bv,"submit._submit",function(e){if(this.parentNode&&e.isTrigger){b.event.simulate("submit
this.parentNode,e,true)});bv._submit_attached=true}});teardown:function(){if(b.nodeName(this,"form")){
return false}b.event.remove(this,"._submit")}}if(!b.support.changeBubbles){b.event.special.change={setup:
function(){if(bd.test(this.nodeName)){if(this.type=="checkbox"||this.type=="radio"){
b.event.add(this,"propertychange._change",function(e){if(e.originalEvent.propertyName=="checked"){this._j
ust_changed=true}});b.event.add(this,"click._change",function(e){if(this._just_changed&&!e.isTrigger){this.
_just_changed=false;b.event.simulate("change",this,e,true)})}return false}b.event.add(this,
beforeactivate._change",function(bw){var bv=bw.target;if(bd.test(bv.nodeName)&&bv._change_attached){b.event.add(bv,
change._change",function(e){if(this.parentNode&&e.isSimulated&&!e.isTrigger){b.event.simulate("change",this.pare
ntNode,e,true)});bv._change_attached=true}});handle:function(bw){var e=bv.target;if(this===e||bv.
isSimulated||bv.isTrigger||e.type!="radio"&&e.type!="checkbox")}{return bv.handleObj.handler.apply(this,arguments)
}};teardown:function(){b.event.remove(this,"._change");return bd.test(this.nodeName)});if(!
b.support.focusinBubbles){b.each({focus:"focusin",blur:"focusout"},function(bx,e){var bv=0,bw=function(by
){b.event.simulate(e,by.target,b.event.fix(by),true)};b.event.special[e]={setup:function(){if(bv===0){
av.addEventListener(bx,bw,true)};teardown:function(){if(--bv===0){av.removeEventListener(bx,bw,true)}}}}
b.fn.extend({on:function(bw,e,bz,by,bv){var bA,bx;if(typeof bw=="object"){if(typeof e!="string"){bz=e;e=
L}for(bx in bw){this.on(bw,e,bz,bw[bx],bv)}return this}if(bz===null&&by===null){by=e;e=L}else if(bz===null){
if(typeof e=="string"){by=bz;e=L}else{by=bz;e=L}}if(bz===false){by=bk}else if(!by){return this}if(b
v===1){bA=by;by=function(bB){b().off(bB);return bA.apply(this,arguments)};by.guid=bA.guid||bA.guid+
)}return this.each(function(){b.event.add(this,bw,by,bz,e)});one:function(bv,e,bx,bw){return this.on.call(t
his,bv,e,bx,bw,1)};off:function(bw,e,by){if(bw&&bw.preventDefault&&bw.handleObj){var bv=bw.handleObj;b(bw.de
legateTarget).off(bv.namespace?bv.type+"."+bv.namespace:bv.type,bv.selector,bv.handler);return this}if(
typeof bw=="object"){for(var bx in bw){this.off(bx,e,bw[bx])}return this}if(e===false||typeof e=="function"){by
=e;e=L}if(by===false){by=bk}return this.each(function(){b.event.remove(this,bw,by,e)});bind:function(
e,bw,bv){return this.on(e,null,bw,bv)};unbind:function(e,bv){return this.off(e,null,bv)};live:function(e,bw,
bv){b(this.context).on(e,this.selector,bw,bv);return this};die:function(e,bv){b(this.context).off(e,this.
selector||"*",bv);return this};delegate:function(e,bv,bx,bw){return this.on(e,bv,bx,bw)};undelegate:function(
e,bv,bw){return arguments.length==1?this.off(e,"*"):this.off(bv,e,bw)};trigger:function(e,bv){return this.
each(function(){b.event.trigger(e,bv,this)});triggerHandler:function(e,bv){if(this[0]){return
b.event.trigger(e,bv,this[0],true)};toggle:function(bx){var bv=arguments,e=bx.guid|
b.guid++;bw=0;by=function(bz){var bA=(b._data(this,"lastToggle"+bx.guid)||0)%bw;
b._data(this,"lastToggle"+bx.guid,bA+1);bz.preventDefault();return bv[bA].apply(this,arguments)||false};by
.guid=e;while(bw<bv.length){bv[bw++].guid=e}return this.click(bv)};hover:function(e,bv){return this.
mouseenter(e).mouseleave(bv||e)};each(function(e){"blur focus focusin focusout load resize scroll unload click dblclick
mousedown mouseup mousemove mouseover mouseout mouseenter mouseleave change select submit keydown keypress
keyup error contextmenu").split(" ").function(bv,e){b.fn[e]=function(bx,bw){return this;if(bw===null){bw=bx;bx=null}return
arguments.length>0?this.on(e,null,bx,bw):this.trigger(e)};if(b.attrFn){b.attrFn[e]=true}if(aO.test(e)){
b.event.fixHooks[e]=b.event.keyHooks}if(bf.test(e)){b.event.fixHooks[e]=b.event.mouseHooks}});
(function(){var bH=/((?:\((?:\([^()]+\)|[^()]+)+\)|\[(?:\[[^\[\]]*\]|"[^"]*"|'[^']*'+\)|[^\[\]]+\)|\]|\.|\[
>+~,(\[\\]+)+|[>+~])|\s*\s*?|(?:\.[\d]+\s*)?|(?:\.[\d]+\s*)?|(?:\.[\d]+\s*)?|(?:\.[\d]+\s*)?|(?:\.[\d]+\s*)?|
)+/g,bC="sizzleCache"+(Math.random()+"");replace(" ","");bI=0,bL=Object.prototype.toString,bB=false,bA=true,bK=/\|/g,bO=/
0);var by=function(bV,e,bY,bZ){bY=bY||[];e=e||av;var
b1=e;if(e.nodeType===1&&e.nodeType!==9){return[]}if(!bV||typeof bV!="string"){return bY}var
bS,b3,b6,bR,b2,b5,b4,bX,bU=bR,bT=by.isXML(e),bW=[],b0=bV;do{bH.exec(" ");
bS=bH.exec(b0);if(bS){b0=bS[3];bW.push(bS[1]);if(bS[2]){bR=bS[3];break}}while(bS);if(bW.length>1&&bD.exec(b
V)){if(bW.length===2&&bE.relative[bW[0]]){b3=bM(bW[0]+bW[1],e,bZ)}else{b3=bE.relative[bW[0]]?e:by(bW.shift(
)),e}while(bW.length){bv=bW.shift();if(bE.relative[bv]){bv+=bW.shift()}b3=bM(bv,b3,bZ)};else if(!bZ&&bW.le
ngth>1&&e.nodeType===9&&!bT&&bE.match.ID.test(bW[0])&&!bE.match.ID.test(bW[bW.length-1])}{b2=by.find(bW.shif

```





```

?bW===bR||bW.substr(0,bR.length+1)===bR+"-":false),POS:function(bU,bR,bS,bV){var
e=bR[2],bT=bE.setFilters(e);if(bT){return bT(bU,bS,bR,bV)}};var bD=bE.match.POS,bx=function(bR,e){return""+(e-0+1)
bE.match[!bE.match[bz]=new RegExp(bE.match[bz].source+/(?![^*]*)?(?![^\s]*\s)/.source));bE.leftMatch[bz]=new RegExp(/^(?:.|[\r\n])*\/.source+bE.match[bz].source.replace(/\\((\d+)/g,bx))};var
bF=function(bR,e){bR=Array.prototype.slice.call(bR,0);if(e){e.push.apply(e,bR);return e}return
bR};try(Array.prototype.slice.call(av.documentElement.childNodes,0)[0].nodeType).catch(bP){bF=function(bU,bT){var bS=0,
[object Array]}(Array.prototype.push.apply(bR,bU))else{if(typeof bU.length==="number"){for(var
e=bU.length;bS<e;bS++){bR.push(bU[bS])}else{for(;bU[bS];bS++){bR.push(bU[bS])}}return bR}}var
bJ,bG;if(av.documentElement.compareDocumentPosition){bJ=function(bR,e){if(bR===e){bB=true;return
0}if(!bR.compareDocumentPosition||!e.compareDocumentPosition){return bR.compareDocumentPosition?-1:1}return
bR.compareDocumentPosition(e)&4?-1:1}}else{bJ=function(bY,bX){if(bY===bX){bB=true;return 0}else{if(bY.sourceIndex&&bX
bY.sourceIndex<bX.sourceIndex)}var bV,bR,bS=[],e=[];bU=bY.parentNode,bW=bX.parentNode,bZ=bU;if(bU===bW){return
bG(bY,bX)}else{if(!bU){return -1}else{if(!bW){return
1}}while(bZ){bS.unshift(bZ);bZ=bZ.parentNode}bZ=bW;while(bZ){e.unshift(bZ);bZ=bZ.parentNode}bV=bS.length;bR=e.length;
bT=0;bT<bV&&bT<bR;bT++;if(bS[bT]!==e[bT]){return bG(bS[bT],e[bT])}return
bT===bV?bG(bY,e[bT],-1):bG(bS[bT],bX,1)};bG=function(bR,e,bS){if(bR===e){return bS}var bT=bR.nextSibling;while(bT){if(
bR=av.createElement("div"),bS="script"+(new Date()).getTime(),e=av.documentElement;bR.innerHTML="<a name='
"+bS+"'>";e.insertBefore(bR,e.firstChild);if(av.getElemenById(bS)){bE.find.ID=function(bU,bV,bW){if(typeof
bV.getElemenById!=="undefined"&&!bW){var bT=bV.getElemenById(bU[1]);return bT?bT.id===bU[1]||typeof
bT.getAttributeNode!=="undefined"&&bT.getAttributeNode("id")
.nodeValue===bU[1]?[bT]:L:[]};bE.filter.ID=function(bV,bT){var bU=typeof bV.getAttributeNode!=="undefined"&&bV.getAttr
bV.nodeType===1&&bU&&bU.nodeValue===bT)}e.removeChild(bR);e=bR=null)}(function(){var e=av.createElement("div"
);e.appendChild(av.createComment(""));if(e.getElemenByTagName("a").length>0){bE.find.TAG=function(bR,bV){var
bU=bV.getElemenByTagName(bR[1]);if(bR[1]===bT){var bT=[];for(var
bS=0;bU[bS];bS++){if(bU[bS].nodeType===1){bT.push(bU[bS])}}bU=bT}return bU}}e.innerHTML="";if(e.firstCh
undefined"&&e.firstChild.getAttribute("href")!=="#")bE.attrHandle.href=function(bR){return bR.getAttribute("
href",2)};e=null)}(function(){if(av.querySelectorAll){bE.find.ID=function(bU,bV,bT){var e=by,bT=av.createElement("div"),bS="__sizzle__
";bT.innerHTML="<p class='TEST'></p>";if(bT.querySelector&&bT.querySelectorAll(".TEST"
.length===0){return}by=function(b4,bV,bZ,b3){bV=bV||av;if(!b3&&!by.isXML(bV)){var b2=/"^(\w+$)|^\.(?!(\w|-
)+$)/.exec(b4);if(b2&&(bV.nodeType===1||bV.nodeType===9)){if(b2[1]){return
bF(bV.getElemenByTagName(b4),bZ)}else{if(b2[2]&&bE.find.CLASS&&bV.getElemenByClassName){return
bF(bV.getElemenByClassName(b2[2]),bZ)}}if(bV.nodeType===9){if(b4==="body"&&bV.body){return bF([bV.body],bZ)}else{if
bV.getElemenById(b2[3]);if(bY&&bY.parentNode){if(bY.id===b2[3]){return bF([bY],bZ)}else{return bF([],bZ)}}}try{ret
bF(bV.querySelector(b4),bZ)}catch(b0){}}else{if(bV.nodeType===1&&bV.nodeName.toLowerCase()!="object"){var
bW=bV,bX=bV.getAttribute("id"),bU=bX||bS,b6=bV.parentNode,b5=/"^s*[+*]/.test(b4);if(!bX){bV.setAttribute("id"
,bU)}else{bU=bU.replace(/"/g,"\\$&")}if(b5&&b6){bV=bV.parentNode;try{if(!b5||b6){return bF(bV.querySelectorAll("[id='
"+bU+"' "+b4),bZ)}catch(b1){}}finally{if(!bX){bW.removeAttribute("id")}}}}return e(b4,bV,bZ,b3)};for(var
bR in e){by[bR]=e[bR]}bT=null)}(function(){var
e=av.documentElement,bS=e.matchesSelector||e.mozMatchesSelector||e.webkitMatchesSelector||e.msMatchesSelector;if(bS){v
),bR=false;try{bS.call(av.documentElement,"[test!='']:sizzle"
)}catch(bT){bR=true}by.matchesSelector=function(bW,bY){bY=bY.replace(/\\/s*([^\s])\s*/g,"='$1'");if(!by.isXML(bW))
!/=/\s*\.test(bY)}(var e=bZ[bT];if(e){var bU=false;e=e[bR];while(e){if(e[bC]===bV){bU=bZ[e.sizset];break}if(e
nodeType===1&&!bY){e[bC]=bV;e.sizset=bT;if(e.nodeName.toLowerCase()===bW){bU=e;break}e=e[bR];bZ[bT]=bU}}
function bN(bR,bW,bV,bZ,bX,bY){for(var bT=0,bS=bZ.length;bT<bS;bT++){var e=bZ[bT];if(e){var bU=false;e=e[bR];while(
e){if(e[bC]===bV){bU=bZ[e.sizset];break}if(e.nodeType===1){if(!bY){e[bC]=bV;e.sizset=bT;if(typeof bW!=="
string"){if(e===bW){bU=true;break}else{if(by.filter(bW,[e]).length>0){bU=e;break}}e=e[bR];bZ[bT]=bU}}if(av
documentElement.contains){by.contains=function(bR,e){return bR===e&&(bR.contains?bR.contains(e):true)}}else{if
(av.documentElement.compareDocumentPosition){by.contains=function(bR,e){return !(bR.compareDocumentPosition
(e)&16)}}else{by.contains=function(){return false}}by.isXML=function(e){var bR=(e?e.ownerDocument||e:0)
.documentElement;return bR?bR.nodeName!="HTML":false};var bM=function(bS,e,bW){var bV,bX=[],bU="",bY=e.nodeType
?e:e;while((bV=bE.match.PSEUDO.exec(bS))){bU+=bV[0];bS=bS.replace(bE.match.PSEUDO,"")}bS=bE.relative[bS]?
bS+"*"+bS;for(var bT=0,bR=bY.length;bT<bR;bT++){by(bS,bY[bT],bX,bW)}return by.filter(bU,bX)};by.attr=
b.attr;by.selectors.attrMap={};b.find=by;b.expr=by.selectors;b.expr[":"]=b.expr.filters;
b.unique=by.uniqueSort;b.text=by.getText;b.isXMLDoc=by.isXML;b.contains=by.contains)}(var ab=/Until$/
,aq="/^?(?:parents|prevUntil|prevAll)/,a9=/,/ ,bp="/^.[^#\|\\.]*$/ ,P=Array.prototype.slice,H=
b.expr.match.POS,ay={children:true,contents:true,next:true,prev:true};b.fn.extend({find:function(e){var
bw=this,by,bv;if(typeof e!="string"){return b(e).filter(function(){for(by=0,bv=bw.length;by<bv;by++){if(b.c
ontains(bw[by],this){return true}}}}var bx=this.pushStack("", "find", e),bA,bB,bz;for(by=0,bv=this.length;by
<bv;by++){bA=bx.length;b.find(e,this[by],bx);if(by>0){for(bB=bA;bB<bx.length;bB++){for(bz=0;bz<bA;bz++){if(
bx[bz]===bx[bB]){bx.splice(bB--,1);break}}}}return bx};has=function(bv){var e=b(bv);return this.filter(
function(){for(var bx=0,bw=e.length;bx<bw;bx++){if(b.c.contains(this,e[bx])){return true}}},not:function(e){
return this.pushStack(aG(this,e,false),"not",e)},filter:function(e){return this.pushStack(aG(this,e,true),"
filter",e)},is:function(e){return !e&&(typeof e=="string"?H.test(e)?b(e,this.context).index(this[0])>=0:
b.filter(e,this).length>0:this.filter(e).length>0)},closest:function(by,bx){var bv=[],bw,e,bz=this[0];if(
b.isArray(by)){var bB=1;while(bz&&bz.ownerDocument&&bz!==bx){for(bw=0;bw<by.length;bw++){if(
b(bz).is(by[bw])){bv.push({selector:by[bw],elem:bz,level:bB})}bz=bz.parentNode;bB++}}return bv}var bA=H
.test(by)||typeof by!="string"?b(by,bx)[this.context]:0;for(bw=0,e=this.length;bw<e;bw++){bz=this[bw];while(
bz){if(bA?bA.index(bz)>-1:b.find.matchesSelector(bz,by)){bv.push(bz);break}else{bz=bz.parentNode;if(!bz||!b
.ownerDocument||bz===bx||bz.nodeType===11){break}}};bv=bv.length>1?b.unique(bv):bv;return this.pushStack(bv
,"closest",by)},index:function(e){if(!e){return this[0]&&this[0].parentNode?this.prevAll().length:-1}if(
typeof e=="string"){return b.inArray(this[0],b(e))}return b.inArray(e.jquery?e[0]:e,this),add:function(e,
bv){var bx=typeof e=="string"?b(e,bv):b.makeArray(e&&e.nodeType?[e]:e),bw=b.merge(this.get(0),bx);return
this.pushStack(C(bx[0])|C(bw[0])?bw:b.unique(bw)),andSelf:function(){return this.add(this.prevObject)});
function C(e){return e||e.parentNode||e.parentNode.nodeType===11?b.each({parent:function(bv){var e=bv
.parentNode;return e&&e.nodeType===11?e:null},parents:function(e){return b.dir(e,"parentNode")},parentsUntil:
function(bv,e,bw){return b.dir(bv,"parentNode",bw)},next:function(e){return b.nth(e,2,"nextSibling")},prev:
function(e){return b.nth(e,2,"previousSibling")},nextAll:function(e){return b.dir(e,"nextSibling")},prevAll:
function(e){return b.dir(e,"previousSibling")},nextUntil:function(bv,e,bw){return
b.dir(bv,"nextSibling",bw)},prevUntil:function(bv,e,bw){return b.dir(bv,"previousSibling",bw)},siblings:
function(e){return b.sibling(e.parentNode.firstChild,e)},children:function(e){return
b.sibling(e.firstChild)},contents:function(e){return b.nodeName(e,"iframe")?e.contentDocument||e.

```

```

contentWindow.document: b.makeArray(e.childNodes)}, function(e, bv) {b.fn[e]=function(by, bw) {var bx=
b.map(this, bv, by); if (!ab.test(e)) {bw=by} if (bw&&typeof bw=="string") {bx=b.filter(bw, bx)} bx=this.length>1&
&lay[e]?b.unique(bx):bx; if (this.length>1) |a9.test(bw) &&aq.test(e) {bx=bx.reverse()} return this.pushStack(
bx, e, P.call(arguments).join(", ")); b.extend({filter: function(bw, e, bv) {if (bv) {bw=":" +bw+""} return e.
length===1?b.find.matchesSelector(e[0], bw)?[e[0]]:[]:b.find.matches(bw, e)}, dir: function(bw, bv, by) {var e=[]
, bx=bw[bv]; while (bx&&bx.nodeType!=""&&(by===L || bx.nodeType===1 || !b(bx).is(by))) {if (bx.nodeType===1) {e.push
(bx)} bx=bw[bv]} return e}, nth: function(by, e, bw, bx) {e=e||1; var bv=0; for (; by; by=by[bw]) {if (by.nodeType===1&&+
bv===e) {break}} return by}, sibling: function(bw, bv) {var e=[]; for (; bw; bw=bw.nextSibling) {if (bw.nodeType===1&&bw
!=""&&bv) {e.push(bw)}} return e}}; function ag(bw, bw, e) {bw=bw||0; if (b.isFunction(bw)) {return
b.grep(bx, function(bz, by) {var ba=!bw.call(bz, by, bz); return ba===e})} else {if (bw.nodeType) {return
b.grep(bx, function(bz, by) {return (bz===bw)===e})} else {if (typeof bw=="string") {var bv=
b.grep(bx, function(bz, by) {return by.nodeType===1})} if (bv.test(bw)) {return b.filter(bw, bv, e)} else {bw=
b.filter(bw, bv)}}} return b.grep(bx, function(bz, by) {return (b.inArray(bz, bw)>=0)===e})} function a(e) {var
bw=aR.split("|"), bv=e.createDocumentFragment(); if (bv.createElement) {while (bw.length) {bv.createElement(bw.pop
())}} return bv} var aR="
abbr|article|aside|audio|canvas|datalist|details|figcaption|figure|footer|header|hgroup|mark|meter|nav|output|progress|
(?:|area|br|col|embed|hr|img|input|link|meta|param)(?![w:]|^|\s|\/>|ig,d=/<([w:]|+)+/, w=/<tbody/
|, W=/<|&#?|w;/, ae=/<(?:script|style)/i, O=/<(?:script|object|embed|option|style)/
i, ah=new RegExp("<(?:+aR+)", "i"), o=/checked\s*(?:[=]|\s|\s*.checked.)/i, bm=/\s\/(java|ecma) script/
i, aN=/^\

```



```

radio") {e.defaultChecked=e.checked}}function E(e) {var bv=(e.nodeName|| "").toLowerCase(); if (bv=="input") {az
(e) }else{if (bv!="script"&&typeof e.getElementsByTagName!="undefined") {b.grep(e.getElementsByTagName("
input"), az)}}function al(e) {var bv=av.createElement("div"); ac.appendChild(bv);bv.innerHTML=e.outerHTML;return
bv.firstChild}b.extend({clone:function(by, bA, bw) {var e, bv, bx, bz=b.support.html5Clone|| !ah.test("<" +by.
nodeName)?by.cloneNode(true):al(by); if ((!b.support.noCloneEvent|| !b.support.noCloneChecked) && (by.nodeType===1||
by.nodeType===11) && !b.isXMLDoc(by)) {ai(by, bz); e=bg(by); bv=bg(bz); for (bx=0; e[bx]; ++bx) {if (bv[bx]) {ai(e[bx],
bv[bx])}} }if (bA) {t(by, bz); if (bw) {e=bg(by); bv=bg(bz); for (bx=0; e[bx]; ++bx) {t(e[bx], bv[bx])}} }e=bv=null; return
bz}, clean:function(bw, by, bH, bA) {var bF; by=by|| av; if (typeof by.createElement=="undefined") {by=by.
ownerDocument|| by[0]&&by[0].ownerDocument|| av}var bI=[], bB; for (var bE=0, bz; (bz=bw[bE]) !=null; bE++) {if (typeof bz=="
number") {bz+=""; if (!bz) {continue}if (typeof bz=="string") {if (!W.test(bz)) {bz=by.createTextNode(bz)} else {bz=bz.
replace(R, "<$1></$2>"); var bK=(d.exec(bz)|| ["", ""])[1].toLowerCase(), bx=ax[bK]|| ax._default, bD=bx[0], bv=by.
createElement("div"); if (by==av) {ac.appendChild(bv)} else {a(by).appendChild(bv)}bv.innerHTML=bx[1]+bz+bx[2]; while
(bD--) {bv=bv.lastChild}if (!b.support.tbody) {var e=w.test(bz), bC=bK=="table"&&!e?bv.firstChild&&bv.
firstChild.childNodes:bx[1]=="<table>"&&!e?bv.childNodes:[]; for (bB=bC.length-1; bB>=0; --bB) {if (
b.nodeName(bC[bB], "tbody") && !bC[bB].childNodes.length) {bC[bB].parentNode.removeChild(bC[bB])}} }if (!
b.support.leadingWhitespace&&ar.test(bz)) {bv.insertBefore(by.createTextNode(ar.exec(bz)[0]), bv.firstChild)
}bz=bv.childNodes}var bG; if (!b.support.appendChecked) {if (bz[0]&&typeof (bG=bz.length)=="number") {for (bB=0;
bB<bG; bB++) {E(bz[bB])}} else {E(bz)}}if (bz.nodeType) {bI.push(bz)} else {bI=b.merge(bI, bz)}}if (bH) {bF=function
(bL) {return !bL.type||bm.test(bL.type)}; for (bE=0; bI[bE]; bE++) {if (bA&&bI[bE].nodeName(bI[bE], "script") && (!bI[bE].
type|| bI[bE].type.toLowerCase()=="text/javascript")) {bA.push(bI[bE].parentNode?bI[bE].parentNode.
removeChild(bI[bE]):bI[bE])} else {if (bI[bE].nodeType===1) {var bJ=b.grep(bI[bE].getElementsByTagName("script"), bF); bI.
splice.apply(bI, [bE+1, 0].concat(bJ))} bH.appendChild(bI[bE])}} }return bI}, cleanData:function(bv) {var by, bw, e=
b.cache, bB=b.event.special, bA=b.support.deleteExpando; for (var bz=0, bx; (bx=bv[bz]) !=null; bz++) {if (bx.
nodeName&&b.noData[bx.nodeName.toLowerCase()]) {continue}bw=bx[b.expando]; if (bw) {by=e[bw]; if (by&&by.events) {for
(var bC in by.events) {if (bB[bC]) {b.event.remove(bx, bC)} else {b.removeEvent(bx, bC, by.handle)}} }if (by.handle) {
by.handle.elem=null}} if (bA) {delete bx[b.expando]} else {if (bx.removeAttribute) {bx.removeAttribute(
b.expando)} delete e[bw]}}}; function bo(e, bv) {if (bv.src) {b.ajax({url:bv.src, async:false, dataType:"
script"})} else {b.globalEval((bv.text||bv.textContent||bv.innerHTML|| "").replace(aN, "/*$0*/"))} if (bv.parentNode) {
bv.parentNode.removeChild(bv)} }var ak=/alpha\(|\)|*\)/i

```

Definition at line 16 of file jquery.js.

### 30.804.2.8 function bb

Definition at line 16 of file jquery.js.

### 30.804.2.9 var bq =/#.\*\$/

Definition at line 23 of file jquery.js.

### 30.804.2.10 var bs =/r?\n/g

Definition at line 23 of file jquery.js.

### 30.804.2.11 var c

**Initial value:**

```

=/^\\\/\
* jQuery UI 1.8.18
*
* Copyright 2011

```

Definition at line 23 of file jquery.js.

### 30.804.2.12 b fn css =function(e,bv){if(arguments.length===2&&bv===L){return this}return b.access(this,e,bv,true,function(bx,bw,by){return by!==L?b.style(bx,bw,by):b.css(bx,bw)}}}

Definition at line 23 of file jquery.js.

**30.804.2.13** `b curCSS =b.css`

Definition at line 23 of file jquery.js.

**30.804.2.14** `var k =/%20/g`

Definition at line 23 of file jquery.js.

**30.804.2.15** `function L {var av=bb.document,bu=bb.navigator,bl=bb.location`

Definition at line 16 of file jquery.js.

**30.804.2.16** `Z =a||aX`

Definition at line 23 of file jquery.js.

**30.805** `doc/html/jquery_8js.js` File Reference**Variables**

- `var jquery_8js`

**30.805.1** Variable Documentation**30.805.1.1** `var jquery_8js`**Initial value:**

```
=
[
 ["b", "jquery_8js.html#a2fa551895933fae935a0a6b87282241d", null],
 ["each", "jquery_8js.html#a871ff39db627c54c710a3e9909b8234c", null],
 ["extend", "jquery_8js.html#a5fb206c91c64d1be35fde236706eab86", null],
 ["if", "jquery_8js.html#a2c54bd8ed7482e89d19331ba61fe221c", null],
 ["if", "jquery_8js.html#a30d3d2cd5b567c9f31b2aa30b9cb3bb8", null],
 ["if", "jquery_8js.html#a42cbfadee2b4749e8f699ea8d745a0e4", null],
 ["if", "jquery_8js.html#a9db6d45a025ad692282fe23e69eeba43", null],
 ["p", "jquery_8js.html#a2335e57f79b6acfb6de59c235dc8a83e", null],
 ["aD", "jquery_8js.html#ad223f5fba68c41c1236671ac5c5b0fcb", null],
 ["aM", "jquery_8js.html#a8cc6111a5def3ea889157d13fb9a9672", null],
 ["ap", "jquery_8js.html#a6ddf393cc7f9a8828e197bb0d9916c44", null],
 ["aQ", "jquery_8js.html#a79eb58dc6cdf0aef563d5dc1ded27df5", null],
 ["au", "jquery_8js.html#a4fd8ddfab07c8d7c7cae0ab0e052cad3", null],
 ["aZ", "jquery_8js.html#ac87125cdee1a5e57da4ef619af49bc7d", null],
 ["b", "jquery_8js.html#aa4026ad5544b958e54ce5e106fal805", null],
 ["bb", "jquery_8js.html#a1d6558865876e1c8cca029fce41a4bdb", null],
 ["bq", "jquery_8js.html#af6ee77c71b2c89bdb365145ac5ad1219", null],
 ["bs", "jquery_8js.html#ae77642f8ef73fb9c20c2a737d956acda", null],
 ["c", "jquery_8js.html#abce695e0af988ece0826d9ad59b8160d", null],
 ["css", "jquery_8js.html#a89ad527fcd82c01ebb587332f5b4fcd4", null],
 ["curCSS", "jquery_8js.html#a88b21f8ba3af86d6981b1da520ece33b", null],
 ["k", "jquery_8js.html#ab26645c014aa005ecedef329ecf58c99", null],
 ["L", "jquery_8js.html#a38ee4c0b5f4fe2a18d0c783af540d253", null],
 ["Z", "jquery_8js.html#adc18d83abfd9f87d396e8fd6b6ac0fe1", null]
]
```

Definition at line 1 of file jquery\_8js.js.

## 30.806 doc/html/main\_8cpp.js File Reference

### Variables

- var [main\\_8cpp](#)

#### 30.806.1 Variable Documentation

##### 30.806.1.1 var main\_8cpp

#### Initial value:

```
=
[
 ["eqPoints", "main_8cpp.html#a78f67d05b2d460e4479cbb45d364472", null],
 ["ltLexPoint", "main_8cpp.html#a3ad6217edc1c1d915556be09a7a6e512", null],
 ["main", "main_8cpp.html#ae66f6b31b5ad750f1fe042a706a4e3d4", null]
]
```

Definition at line 1 of file main\_8cpp.js.

## 30.807 doc/html/make3\_d\_8c.js File Reference

### Variables

- var [make3\\_d\\_8c](#)

#### 30.807.1 Variable Documentation

##### 30.807.1.1 var make3\_d\_8c

#### Initial value:

```
=
[
 ["MAKE3D", "make3_d_8c.html#af73dcd2ed246a565df1162b610461e64", null],
 ["make3D", "make3_d_8c.html#a96b282e7a516c8aad1cdcdaa8ca3240b", null]
]
```

Definition at line 1 of file make3\_d\_8c.js.

## 30.808 doc/html/makecvkreg\_8c.js File Reference

### Variables

- var [makecvkreg\\_8c](#)

### 30.808.1 Variable Documentation

#### 30.808.1.1 var makecvkreg\_8c

**Initial value:**

```
=
[
 ["MAKE_CV_KREG", "makecvkreg_8c.html#a3d78c8854b03c4e271f7fcdc2642aaaa", null],
 ["make_cv_kreg", "makecvkreg_8c.html#a192abaac4dcfb0ab13f98dafcea4b5fe", null]
]
```

Definition at line 1 of file makecvkreg\_8c.js.

## 30.809 doc/html/makesfkreg\_8c.js File Reference

### Variables

- var [makesfkreg\\_8c](#)

### 30.809.1 Variable Documentation

#### 30.809.1.1 var makesfkreg\_8c

**Initial value:**

```
=
[
 ["MAKE_SF_KREG", "makesfkreg_8c.html#ab2dd7beaefalb53540f56163621670cd", null],
 ["make_sf_kreg", "makesfkreg_8c.html#a4d929d52047fe5051ed490891f3275d9", null]
]
```

Definition at line 1 of file makesfkreg\_8c.js.

## 30.810 doc/html/maketracks\_8c.js File Reference

### Variables

- var [maketracks\\_8c](#)

### 30.810.1 Variable Documentation

#### 30.810.1.1 var maketracks\_8c

**Initial value:**

```
=
[
 ["MAKE_TRACKS", "maketracks_8c.html#af500037a7945500574264d2dda094220", null],
 ["make_tracks", "maketracks_8c.html#ad36cb40b8b93e0b6a8fa52234213ccb8", null]
]
```

Definition at line 1 of file maketracks\_8c.js.

## 30.811 doc/html/mk\_cv\_cycl\_8c.js File Reference

### Variables

- var [mk\\_cv\\_cycl\\_8c](#)

#### 30.811.1 Variable Documentation

##### 30.811.1.1 var mk\_cv\_cycl\_8c

###### Initial value:

```
=
[
 ["MAKE_CV_CYCLIC", "mk_cv_cycl_8c.html#afd0d637cd833f1e12447d183a72bf1ee", null],
 ["make_cv_cyclic", "mk_cv_cycl_8c.html#a0045019c9f5a8e5dfdd29a7cd029e99e", null]
]
```

Definition at line 1 of file mk\_cv\_cycl\_8c.js.

## 30.812 doc/html/modules.js File Reference

### Variables

- var [modules](#)

#### 30.812.1 Variable Documentation

##### 30.812.1.1 var modules

###### Initial value:

```
=
[
 ["RBD common library", "group_rbd_common.html", "group_rbd_common"]
]
```

Definition at line 1 of file modules.js.

## 30.813 doc/html/mouse\_8cpp.js File Reference

### Variables

- var [mouse\\_8cpp](#)

### 30.813.1 Variable Documentation

#### 30.813.1.1 var mouse\_8cpp

##### Initial value:

```
=
[
 ["GLUT_DISABLE_ATEXIT_HACK", "mouse_8cpp.html#ab07b3c93a1236fc83b162fb0a8945d72", null],
 ["PI", "mouse_8cpp.html#a598a3330b3c21701223ee0ca14316eca", null],
 ["draw_cursor", "mouse_8cpp.html#a108608b0f2d804c84a271547036b6ffd", null],
 ["Mouse", "mouse_8cpp.html#a5510b38c98574e8eae388f618f24f39f", null],
 ["MouseRotate", "mouse_8cpp.html#a240121d2c94e4f25abd36a26a76d48cf", null],
 ["MouseTranslate", "mouse_8cpp.html#a43fbe83a677284dcb5f401d73db80b7b", null],
 ["MouseZoom", "mouse_8cpp.html#a75f208ad1a6f0a1062c4b26ef1e13162", null],
 ["transversal_rotation", "mouse_8cpp.html#ac98380509e96a10b686a8a872a830c4a", null],
 ["allow_zrot", "mouse_8cpp.html#a506eb7695cd87667f501cfd89fe29839", null],
 ["curx", "mouse_8cpp.html#aee28fdc5f13f7a57d1996da267972350", null],
 ["cury", "mouse_8cpp.html#a19f72fc5f0f52553c05605db140d1e5a", null],
 ["curz", "mouse_8cpp.html#ad0deb24d97bde83be716f3bdd98621a2", null],
 ["last_x", "mouse_8cpp.html#abbf254766520cad4d7fe189eb184194c", null],
 ["last_x0", "mouse_8cpp.html#a697ce1bd6e962a31605d4672581067d5", null],
 ["last_y", "mouse_8cpp.html#a31e24c11fe7b55913887a78bff6ed30e", null],
 ["last_y0", "mouse_8cpp.html#ac9f8093288787fc838cca3e095567eb2", null],
 ["mouse_movement", "mouse_8cpp.html#abaf1e7e04c60ccfa143a1c5706d646d9", null]
]
```

Definition at line 1 of file mouse\_8cpp.js.

## 30.814 doc/html/mouse\_8h.js File Reference

### Variables

- var [mouse\\_8h](#)

### 30.814.1 Variable Documentation

#### 30.814.1.1 var mouse\_8h

##### Initial value:

```
=
[
 ["MOUSE_H_INCLUDED", "mouse_8h.html#aaffe8d332d06f58a994aed49ecc20415a", null],
 ["Mouse", "mouse_8h.html#a5510b38c98574e8eae388f618f24f39f", null],
 ["MouseRotate", "mouse_8h.html#a240121d2c94e4f25abd36a26a76d48cf", null],
 ["MouseTranslate", "mouse_8h.html#a43fbe83a677284dcb5f401d73db80b7b", null],
 ["MouseZoom", "mouse_8h.html#a75f208ad1a6f0a1062c4b26ef1e13162", null],
 ["transversal_rotation", "mouse_8h.html#ac98380509e96a10b686a8a872a830c4a", null],
 ["allow_zrot", "mouse_8h.html#a506eb7695cd87667f501cfd89fe29839", null],
 ["last_x", "mouse_8h.html#abbf254766520cad4d7fe189eb184194c", null],
 ["last_x0", "mouse_8h.html#a697ce1bd6e962a31605d4672581067d5", null],
 ["last_y", "mouse_8h.html#a31e24c11fe7b55913887a78bff6ed30e", null],
 ["last_y0", "mouse_8h.html#ac9f8093288787fc838cca3e095567eb2", null],
 ["mouse_movement", "mouse_8h.html#abaf1e7e04c60ccfa143a1c5706d646d9", null]
]
```

Definition at line 1 of file mouse\_8h.js.

## 30.815 doc/html/myexcept\_8cpp.js File Reference

### Variables

- var [myexcept\\_8cpp](#)

#### 30.815.1 Variable Documentation

##### 30.815.1.1 var myexcept\_8cpp

###### Initial value:

```
=
[
 ["WANT_STREAM", "myexcept_8cpp.html#a6ed6c2d6e68d8f0e7900326b4e30850c", null],
 ["WANT_STRING", "myexcept_8cpp.html#a981403274df661407a2ed2b71f7f2e29", null],
 ["Terminate", "myexcept_8cpp.html#a9b848654502024dde7e46ab002f2c9dd", null]
]
```

Definition at line 1 of file myexcept\_8cpp.js.

## 30.816 doc/html/myexcept\_8h.js File Reference

### Variables

- var [myexcept\\_8h](#)

#### 30.816.1 Variable Documentation

##### 30.816.1.1 var myexcept\_8h

###### Initial value:

```
=
[
 ["Tracer", "class_r_b_d___c_o_m_m_o_n_l_l_tracer.html", "class_r_b_d___c_o_m_m_o_n_l_l_tracer"],
 ["BaseException", "class_r_b_d___c_o_m_m_o_n_l_l_base_exception.html", "
 class_r_b_d___c_o_m_m_o_n_l_l_base_exception"],
 ["Janitor", "class_r_b_d___c_o_m_m_o_n_l_l_janitor.html", "class_r_b_d___c_o_m_m_o_n_l_l_janitor"],
 ["Logic_error", "class_r_b_d___c_o_m_m_o_n_l_l_logic_error.html", "
 class_r_b_d___c_o_m_m_o_n_l_l_logic_error"],
 ["Runtime_error", "class_r_b_d___c_o_m_m_o_n_l_l_runtime_error.html", "
 class_r_b_d___c_o_m_m_o_n_l_l_runtime_error"],
 ["Domain_error", "class_r_b_d___c_o_m_m_o_n_l_l_domain_error.html", "
 class_r_b_d___c_o_m_m_o_n_l_l_domain_error"],
 ["Invalid_argument", "class_r_b_d___c_o_m_m_o_n_l_l_invalid_argument.html", "
 class_r_b_d___c_o_m_m_o_n_l_l_invalid_argument"],
 ["Length_error", "class_r_b_d___c_o_m_m_o_n_l_l_length_error.html", "
 class_r_b_d___c_o_m_m_o_n_l_l_length_error"],
 ["Out_of_range", "class_r_b_d___c_o_m_m_o_n_l_l_out_of_range.html", "
 class_r_b_d___c_o_m_m_o_n_l_l_out_of_range"],
 ["Range_error", "class_r_b_d___c_o_m_m_o_n_l_l_range_error.html", "
 class_r_b_d___c_o_m_m_o_n_l_l_range_error"],
 ["Overflow_error", "class_r_b_d___c_o_m_m_o_n_l_l_overflow_error.html", "
 class_r_b_d___c_o_m_m_o_n_l_l_overflow_error"],
 ["Bad_alloc", "class_r_b_d___c_o_m_m_o_n_l_l_bad_alloc.html", "
 class_r_b_d___c_o_m_m_o_n_l_l_bad_alloc"],
 ["Catch", "myexcept_8h.html#af19f2de9b4bccab916654c258aa6f2d4", null],
]
```

```
["CatchAll", "myexcept_8h.html#a549ceb8bbab061440f4698483d6ad179", null],
["CatchAndThrow", "myexcept_8h.html#a3a7fd3e342806240062de1b14ff917ca", null],
["FREE_CHECK", "myexcept_8h.html#a38496bbc880a5b060716ea39a9e67990", null],
["MONITOR_INT_DELETE", "myexcept_8h.html#a0e12927a29ed5ea7b344106ac3a914b6", null],
["MONITOR_INT_NEW", "myexcept_8h.html#a124780b20c1f4395226be43e322456c1", null],
["MONITOR_REAL_DELETE", "myexcept_8h.html#af06beffef7dc5c1f4a48faf3a8b1b904", null],
["MONITOR_REAL_NEW", "myexcept_8h.html#a607c47d26b6401eb36af406cac59ff81", null],
["NEW_DELETE", "myexcept_8h.html#a7814d199e1dale1a6255fa85f8e8c320", null],
["ReThrow", "myexcept_8h.html#af3461fc02f3e0b6bc70e6c310cef1424", null],
["Throw", "myexcept_8h.html#a7554028bfd929fe4391cb3f9ed7b9da1", null],
["Try", "myexcept_8h.html#a7a2b8ccf41cff483bac8e86ac7feff9b", null],
["Exception", "myexcept_8h.html#abbff87b4655088a6fd923859d2a6a8d2", null],
["Terminate", "myexcept_8h.html#a6afccc25f7fade97471bae8d72cbb931", null]
]
```

Definition at line 1 of file `myexcept_8h.js`.

## 30.817 doc/html/namespace\_go.js File Reference

### Variables

- var [namespace\\_go](#)

### 30.817.1 Variable Documentation

#### 30.817.1.1 var namespace\_go

Definition at line 1 of file `namespace_go.js`.

## 30.818 doc/html/namespace\_go\_1\_1\_coons\_patch\_gen.js File Reference

### Variables

- var [namespace\\_go\\_1\\_1\\_coons\\_patch\\_gen](#)

### 30.818.1 Variable Documentation

#### 30.818.1.1 var namespace\_go\_1\_1\_coons\_patch\_gen

#### Initial value:

```
=
[
["UnknownError", "class_go_1_1_coons_patch_gen_1_1_un_known_error.html", null]
]
```

Definition at line 1 of file `namespace_go_1_1_coons_patch_gen.js`.



## 30.819 doc/html/namespace\_go\_1\_1\_l\_r\_spline\_utils.js File Reference

### Variables

- var [namespace\\_go\\_1\\_1\\_l\\_r\\_spline\\_utils](#)

#### 30.819.1 Variable Documentation

##### 30.819.1.1 var namespace\_go\_1\_1\_l\_r\_spline\_utils

###### Initial value:

```
=
[
 ["support_compare", "struct_go_1_1_l_r_spline_utils_1_1support__compare.html", "
 struct_go_1_1_l_r_spline_utils_1_1support__compare"]
]
```

Definition at line 1 of file namespace\_go\_1\_1\_l\_r\_spline\_utils.js.

## 30.820 doc/html/namespace\_go\_1\_1\_box\_structuring.js File Reference

### Variables

- var [namespace\\_go\\_1\\_1\\_box\\_structuring](#)

#### 30.820.1 Variable Documentation

##### 30.820.1.1 var namespace\_go\_1\_1\_box\_structuring

###### Initial value:

```
=
[
 ["BoundingBoxStructure", "class_go_1_1_box_structuring_1_1_bounding_box_structure.html", "
 class_go_1_1_box_structuring_1_1_bounding_box_structure"],
 ["SubSurfaceBoundingBox", "class_go_1_1_box_structuring_1_1_sub_surface_bounding_box.html", "
 class_go_1_1_box_structuring_1_1_sub_surface_bounding_box"],
 ["SurfaceData", "class_go_1_1_box_structuring_1_1_surface_data.html", "
 class_go_1_1_box_structuring_1_1_surface_data"]
]
```

Definition at line 1 of file namespace\_go\_1\_1\_box\_structuring.js.

## 30.821 doc/html/namespace\_n\_e\_w\_m\_a\_t.js File Reference

### Variables

- var [namespace\\_n\\_e\\_w\\_m\\_a\\_t](#)

### 30.821.1 Variable Documentation

#### 30.821.1.1 var namespace\_n\_e\_w\_m\_a\_t

Definition at line 1 of file namespace\_n\_e\_w\_m\_a\_t.js.

## 30.822 doc/html/namespace\_r\_b\_d\_\_\_c\_o\_m\_m\_o\_n.js File Reference

### Variables

- var [namespace\\_r\\_b\\_d\\_\\_\\_c\\_o\\_m\\_m\\_o\\_n](#)

### 30.822.1 Variable Documentation

#### 30.822.1.1 var namespace\_r\_b\_d\_\_\_c\_o\_m\_m\_o\_n

#### Initial value:

```
=
[
 ["Bad_alloc", "class_r_b_d___c_o_m_m_o_n_l_l_bad_alloc.html", "
 class_r_b_d___c_o_m_m_o_n_l_l_bad_alloc"],
 ["BaseException", "class_r_b_d___c_o_m_m_o_n_l_l_base_exception.html", "
 class_r_b_d___c_o_m_m_o_n_l_l_base_exception"],
 ["Domain_error", "class_r_b_d___c_o_m_m_o_n_l_l_domain_error.html", "
 class_r_b_d___c_o_m_m_o_n_l_l_domain_error"],
 ["Invalid_argument", "class_r_b_d___c_o_m_m_o_n_l_l_invalid_argument.html", "
 class_r_b_d___c_o_m_m_o_n_l_l_invalid_argument"],
 ["Janitor", "class_r_b_d___c_o_m_m_o_n_l_l_janitor.html", "class_r_b_d___c_o_m_m_o_n_l_l_janitor"],
 ["Length_error", "class_r_b_d___c_o_m_m_o_n_l_l_length_error.html", "
 class_r_b_d___c_o_m_m_o_n_l_l_length_error"],
 ["Logic_error", "class_r_b_d___c_o_m_m_o_n_l_l_logic_error.html", "
 class_r_b_d___c_o_m_m_o_n_l_l_logic_error"],
 ["OneDimSolve", "class_r_b_d___c_o_m_m_o_n_l_l_one_dim_solve.html", "
 class_r_b_d___c_o_m_m_o_n_l_l_one_dim_solve"],
 ["Out_of_range", "class_r_b_d___c_o_m_m_o_n_l_l_out_of_range.html", "
 class_r_b_d___c_o_m_m_o_n_l_l_out_of_range"],
 ["Overflow_error", "class_r_b_d___c_o_m_m_o_n_l_l_overflow_error.html", "
 class_r_b_d___c_o_m_m_o_n_l_l_overflow_error"],
 ["R1_R1", "class_r_b_d___c_o_m_m_o_n_l_l_r1_r1.html", "class_r_b_d___c_o_m_m_o_n_l_l_r1_r1"],
 ["Range_error", "class_r_b_d___c_o_m_m_o_n_l_l_range_error.html", "
 class_r_b_d___c_o_m_m_o_n_l_l_range_error"],
 ["Runtime_error", "class_r_b_d___c_o_m_m_o_n_l_l_runtime_error.html", "
 class_r_b_d___c_o_m_m_o_n_l_l_runtime_error"],
 ["SolutionException", "class_r_b_d___c_o_m_m_o_n_l_l_solution_exception.html", "
 class_r_b_d___c_o_m_m_o_n_l_l_solution_exception"],
 ["Tracer", "class_r_b_d___c_o_m_m_o_n_l_l_tracer.html", "class_r_b_d___c_o_m_m_o_n_l_l_tracer"]
]
```

Definition at line 1 of file namespace\_r\_b\_d\_\_\_c\_o\_m\_m\_o\_n.js.

## 30.823 doc/html/namespacehed.js File Reference

### Variables

- var [namespacehed](#)

### 30.823.1 Variable Documentation

#### 30.823.1.1 var namespacehed

**Initial value:**

```
=
[
 ["Dart", "classhed_1_1_dart.html", "classhed_1_1_dart"],
 ["Edge", "classhed_1_1_edge.html", "classhed_1_1_edge"],
 ["Node", "classhed_1_1_node.html", "classhed_1_1_node"],
 ["Triangulation", "classhed_1_1_triangulation.html", "classhed_1_1_triangulation"],
 ["TTLtraits", "structhed_1_1_t_t_ltraits.html", "structhed_1_1_t_t_ltraits"]
]
```

Definition at line 1 of file namespacehed.js.

## 30.824 doc/html/namespacehetriang.js File Reference

### Variables

- var [namespacehetriang](#)

### 30.824.1 Variable Documentation

#### 30.824.1.1 var namespacehetriang

**Initial value:**

```
=
[
 ["Dart", "classhetriang_1_1_dart.html", "classhetriang_1_1_dart"],
 ["Edge", "classhetriang_1_1_edge.html", "classhetriang_1_1_edge"],
 ["Node", "classhetriang_1_1_node.html", "classhetriang_1_1_node"],
 ["Triangulation", "classhetriang_1_1_triangulation.html", "classhetriang_1_1_triangulation"],
 ["TTLtraits", "structhetriang_1_1_t_t_ltraits.html", "structhetriang_1_1_t_t_ltraits"]
]
```

Definition at line 1 of file namespacehetriang.js.

## 30.825 doc/html/namespacemembers\_dup.js File Reference

### Variables

- var [namespacemembers\\_dup](#)

### 30.825.1 Variable Documentation

#### 30.825.1.1 var namespacemembers\_dup

##### Initial value:

```
=
[
 ["_", "namespacemembers.html", null],
 ["a", "namespacemembers_a.html", null],
 ["b", "namespacemembers_b.html", null],
 ["c", "namespacemembers_c.html", null],
 ["d", "namespacemembers_d.html", null],
 ["e", "namespacemembers_e.html", null],
 ["f", "namespacemembers_f.html", null],
 ["g", "namespacemembers_g.html", null],
 ["h", "namespacemembers_h.html", null],
 ["i", "namespacemembers_i.html", null],
 ["j", "namespacemembers_j.html", null],
 ["k", "namespacemembers_k.html", null],
 ["l", "namespacemembers_l.html", null],
 ["m", "namespacemembers_m.html", null],
 ["n", "namespacemembers_n.html", null],
 ["o", "namespacemembers_o.html", null],
 ["p", "namespacemembers_p.html", null],
 ["q", "namespacemembers_q.html", null],
 ["r", "namespacemembers_r.html", null],
 ["s", "namespacemembers_s.html", null],
 ["t", "namespacemembers_t.html", null],
 ["u", "namespacemembers_u.html", null],
 ["v", "namespacemembers_v.html", null],
 ["w", "namespacemembers_w.html", null],
 ["x", "namespacemembers_x.html", null],
 ["y", "namespacemembers_y.html", null],
 ["z", "namespacemembers_z.html", null]
]
```

Definition at line 1 of file namespacemembers\_dup.js.

## 30.826 doc/html/namespacemembers\_eval.js File Reference

### Variables

- var [namespacemembers\\_eval](#)

### 30.826.1 Variable Documentation

#### 30.826.1.1 var namespacemembers\_eval

##### Initial value:

```
=
[
 ["a", "namespacemembers_eval.html", null],
 ["b", "namespacemembers_eval_b.html", null],
 ["c", "namespacemembers_eval_c.html", null],
 ["d", "namespacemembers_eval_d.html", null],
 ["e", "namespacemembers_eval_e.html", null],
 ["f", "namespacemembers_eval_f.html", null],
 ["g", "namespacemembers_eval_g.html", null],
 ["h", "namespacemembers_eval_h.html", null],
 ["i", "namespacemembers_eval_i.html", null],
 ["j", "namespacemembers_eval_j.html", null],
 ["k", "namespacemembers_eval_k.html", null],
]
```

```

["l", "namespacemembers_eval_l.html", null],
["m", "namespacemembers_eval_m.html", null],
["n", "namespacemembers_eval_n.html", null],
["o", "namespacemembers_eval_o.html", null],
["p", "namespacemembers_eval_p.html", null],
["r", "namespacemembers_eval_r.html", null],
["s", "namespacemembers_eval_s.html", null],
["t", "namespacemembers_eval_t.html", null],
["u", "namespacemembers_eval_u.html", null],
["v", "namespacemembers_eval_v.html", null],
["x", "namespacemembers_eval_x.html", null],
["y", "namespacemembers_eval_y.html", null],
["z", "namespacemembers_eval_z.html", null]
]

```

Definition at line 1 of file namespacemembers\_eval.js.

## 30.827 doc/html/namespacemembers\_func.js File Reference

### Variables

- var [namespacemembers\\_func](#)

### 30.827.1 Variable Documentation

#### 30.827.1.1 var namespacemembers\_func

#### Initial value:

```

=
[
["_", "namespacemembers_func.html", null],
["a", "namespacemembers_func_a.html", null],
["b", "namespacemembers_func_b.html", null],
["c", "namespacemembers_func_c.html", null],
["d", "namespacemembers_func_d.html", null],
["e", "namespacemembers_func_e.html", null],
["f", "namespacemembers_func_f.html", null],
["g", "namespacemembers_func_g.html", null],
["h", "namespacemembers_func_h.html", null],
["i", "namespacemembers_func_i.html", null],
["j", "namespacemembers_func_j.html", null],
["k", "namespacemembers_func_k.html", null],
["l", "namespacemembers_func_l.html", null],
["m", "namespacemembers_func_m.html", null],
["n", "namespacemembers_func_n.html", null],
["o", "namespacemembers_func_o.html", null],
["p", "namespacemembers_func_p.html", null],
["q", "namespacemembers_func_q.html", null],
["r", "namespacemembers_func_r.html", null],
["s", "namespacemembers_func_s.html", null],
["t", "namespacemembers_func_t.html", null],
["u", "namespacemembers_func_u.html", null],
["v", "namespacemembers_func_v.html", null],
["w", "namespacemembers_func_w.html", null]
]

```

Definition at line 1 of file namespacemembers\_func.js.

## 30.828 doc/html/namespaces.js File Reference

### Variables

- var [namespaces](#)

## 30.828.1 Variable Documentation

### 30.828.1.1 var namespaces

#### Initial value:

```
=
[
 ["Go", "namespace_go.html", "namespace_go"],
 ["hed", "namespacehed.html", null],
 ["hetriang", "namespacehetriang.html", null],
 ["NEWMAT", "namespace_n_e_w_m_a_t.html", null],
 ["RBD_COMMON", "namespace_r_b_d_c_o_m_m_o_n.html", null],
 ["RBD_LIBRARIES", "namespace_r_b_d_l_i_b_r_a_r_i_e_s.html", null],
 ["std", "namespacestd.html", null],
 ["ttl", "namespacettl.html", null],
 ["ttl_constr", "namespacettl_constr.html", null],
 ["ttl_util", "namespacettl_util.html", null]
]
```

Definition at line 1 of file namespaces.js.

## 30.829 doc/html/navtree.js File Reference

### Functions

- function [getData](#) (varName)
- function [stripPath](#) (uri)
- function [stripPath2](#) (uri)
- function [hashValue](#) ()
- function [hashUrl](#) ()
- function [pathName](#) ()
- function [localStorageSupported](#) ()
- function [storeLink](#) (link)
- function [deleteLink](#) ()
- function [cachedLink](#) ()
- function [getScript](#) (scriptName, func, show)
- function [createIndent](#) (o, domNode, node, level)
- function [gotoAnchor](#) (anchor, aname, updateLocation)
- function [newNode](#) (o, po, text, link, childrenData, lastNode)
- function [showRoot](#) ()
- function [expandNode](#) (o, node, imm, [showRoot](#))
- function [glowEffect](#) (n, duration)
- function [highlightAnchor](#) ()
- function [selectAndHighlight](#) (hash, n)
- function [showNode](#) (o, node, index, hash)
- function [removeToInsertLater](#) (element)
- function [getNode](#) (o, po)
- function [gotoNode](#) (o, subIndex, root, hash, relpath)
- function [navTo](#) (o, root, hash, relpath)
- function [showSyncOff](#) (n, relpath)
- function [showSyncOn](#) (n, relpath)
- function [toggleSyncButton](#) (relpath)
- function [initNavTree](#) (toroot, relpath)

## Variables

- var `navTreeSubIndices` = new Array()
- var `animationInProgress` = false

### 30.829.1 Function Documentation

#### 30.829.1.1 function `cachedLink ( )`

Definition at line 63 of file `navtree.js`.

#### 30.829.1.2 function `createIndent ( o, domNode, node, level )`

Definition at line 91 of file `navtree.js`.

#### 30.829.1.3 function `deleteLink ( )`

Definition at line 56 of file `navtree.js`.

#### 30.829.1.4 function `expandNode ( o, node, imm, showRoot )`

Definition at line 253 of file `navtree.js`.

#### 30.829.1.5 function `getData ( varName )`

Definition at line 3 of file `navtree.js`.

#### 30.829.1.6 function `getNode ( o, po )`

Definition at line 384 of file `navtree.js`.

#### 30.829.1.7 function `getScript ( scriptName, func, show )`

Definition at line 72 of file `navtree.js`.

#### 30.829.1.8 function `glowEffect ( n, duration )`

Definition at line 281 of file `navtree.js`.

#### 30.829.1.9 function `gotoAnchor ( anchor, aname, updateLocation )`

Definition at line 129 of file `navtree.js`.

30.829.1.10 `function gotoNode ( o, subIndex, root, hash, relpath )`

Definition at line 397 of file `navtree.js`.

30.829.1.11 `function hashUrl ( )`

Definition at line 28 of file `navtree.js`.

30.829.1.12 `function hashValue ( )`

Definition at line 23 of file `navtree.js`.

30.829.1.13 `function highlightAnchor ( )`

Definition at line 288 of file `navtree.js`.

30.829.1.14 `function initNavTree ( toroot, relpath )`

Definition at line 466 of file `navtree.js`.

30.829.1.15 `function localStorageSupported ( )`

Definition at line 38 of file `navtree.js`.

30.829.1.16 `function navTo ( o, root, hash, relpath )`

Definition at line 412 of file `navtree.js`.

30.829.1.17 `function newNode ( o, po, text, link, childrenData, lastNode )`

Definition at line 157 of file `navtree.js`.

30.829.1.18 `function pathName ( )`

Definition at line 33 of file `navtree.js`.

30.829.1.19 `function removeToInsertLater ( element )`

Definition at line 371 of file `navtree.js`.



30.829.1.20 `function selectAndHighlight ( hash, n )`

Definition at line 307 of file navtree.js.

30.829.1.21 `function showNode ( o, node, index, hash )`

Definition at line 330 of file navtree.js.

30.829.1.22 `function showRoot ( )`

Definition at line 238 of file navtree.js.

30.829.1.23 `function showSyncOff ( n, relpath )`

Definition at line 442 of file navtree.js.

30.829.1.24 `function showSyncOn ( n, relpath )`

Definition at line 447 of file navtree.js.

30.829.1.25 `function storeLink ( link )`

Definition at line 49 of file navtree.js.

30.829.1.26 `function stripPath ( uri )`

Definition at line 10 of file navtree.js.

30.829.1.27 `function stripPath2 ( uri )`

Definition at line 15 of file navtree.js.

30.829.1.28 `function toggleSyncButton ( relpath )`

Definition at line 452 of file navtree.js.

## 30.829.2 Variable Documentation

30.829.2.1 `var animationInProgress = false`

Definition at line 127 of file navtree.js.

30.829.2.2 `var navTreeSubIndices = new Array()`

Definition at line 1 of file `navtree.js`.

## 30.830 `doc/html/navtreedata.js` File Reference

### Variables

- var [NAVTREE](#)
- var [NAVTREEINDEX](#)
- var [SYNCONMSG](#) = 'click to disable panel synchronisation'
- var [SYNCOFFMSG](#) = 'click to enable panel synchronisation'

### 30.830.1 Variable Documentation

30.830.1.1 `var NAVTREE`

Definition at line 1 of file `navtreedata.js`.

30.830.1.2 `var NAVTREEINDEX`

Definition at line 175 of file `navtreedata.js`.

30.830.1.3 `var SYNCOFFMSG = 'click to enable panel synchronisation'`

Definition at line 244 of file `navtreedata.js`.

30.830.1.4 `var SYNCONMSG = 'click to disable panel synchronisation'`

Definition at line 243 of file `navtreedata.js`.

## 30.831 `doc/html/navtreeindex0.js` File Reference

### Variables

- var [NAVTREEINDEX0](#)

### 30.831.1 Variable Documentation

30.831.1.1 `var NAVTREEINDEX0`

Definition at line 1 of file `navtreeindex0.js`.

## 30.832 doc/html/navtreeindex1.js File Reference

### Variables

- var [NAVTREEINDEX1](#)

#### 30.832.1 Variable Documentation

##### 30.832.1.1 var NAVTREEINDEX1

Definition at line 1 of file navtreeindex1.js.

## 30.833 doc/html/navtreeindex10.js File Reference

### Variables

- var [NAVTREEINDEX10](#)

#### 30.833.1 Variable Documentation

##### 30.833.1.1 var NAVTREEINDEX10

Definition at line 1 of file navtreeindex10.js.

## 30.834 doc/html/navtreeindex11.js File Reference

### Variables

- var [NAVTREEINDEX11](#)

#### 30.834.1 Variable Documentation

##### 30.834.1.1 var NAVTREEINDEX11

Definition at line 1 of file navtreeindex11.js.

## 30.835 doc/html/navtreeindex12.js File Reference

### Variables

- var [NAVTREEINDEX12](#)

### 30.835.1 Variable Documentation

#### 30.835.1.1 var NAVTREEINDEX12

Definition at line 1 of file navtreeindex12.js.

## 30.836 doc/html/navtreeindex13.js File Reference

### Variables

- var [NAVTREEINDEX13](#)

### 30.836.1 Variable Documentation

#### 30.836.1.1 var NAVTREEINDEX13

Definition at line 1 of file navtreeindex13.js.

## 30.837 doc/html/navtreeindex14.js File Reference

### Variables

- var [NAVTREEINDEX14](#)

### 30.837.1 Variable Documentation

#### 30.837.1.1 var NAVTREEINDEX14

Definition at line 1 of file navtreeindex14.js.

## 30.838 doc/html/navtreeindex15.js File Reference

### Variables

- var [NAVTREEINDEX15](#)

### 30.838.1 Variable Documentation

#### 30.838.1.1 var NAVTREEINDEX15

Definition at line 1 of file navtreeindex15.js.

## 30.839 doc/html/navtreeindex16.js File Reference

### Variables

- var [NAVTREEINDEX16](#)

#### 30.839.1 Variable Documentation

##### 30.839.1.1 var NAVTREEINDEX16

Definition at line 1 of file navtreeindex16.js.

## 30.840 doc/html/navtreeindex17.js File Reference

### Variables

- var [NAVTREEINDEX17](#)

#### 30.840.1 Variable Documentation

##### 30.840.1.1 var NAVTREEINDEX17

Definition at line 1 of file navtreeindex17.js.

## 30.841 doc/html/navtreeindex18.js File Reference

### Variables

- var [NAVTREEINDEX18](#)

#### 30.841.1 Variable Documentation

##### 30.841.1.1 var NAVTREEINDEX18

Definition at line 1 of file navtreeindex18.js.

## 30.842 doc/html/navtreeindex19.js File Reference

### Variables

- var [NAVTREEINDEX19](#)

### 30.842.1 Variable Documentation

#### 30.842.1.1 var NAVTREEINDEX19

Definition at line 1 of file navtreeindex19.js.

## 30.843 doc/html/navtreeindex2.js File Reference

### Variables

- var [NAVTREEINDEX2](#)

### 30.843.1 Variable Documentation

#### 30.843.1.1 var NAVTREEINDEX2

Definition at line 1 of file navtreeindex2.js.

## 30.844 doc/html/navtreeindex20.js File Reference

### Variables

- var [NAVTREEINDEX20](#)

### 30.844.1 Variable Documentation

#### 30.844.1.1 var NAVTREEINDEX20

Definition at line 1 of file navtreeindex20.js.

## 30.845 doc/html/navtreeindex21.js File Reference

### Variables

- var [NAVTREEINDEX21](#)

### 30.845.1 Variable Documentation

#### 30.845.1.1 var NAVTREEINDEX21

Definition at line 1 of file navtreeindex21.js.

## 30.846 doc/html/navtreeindex22.js File Reference

### Variables

- var [NAVTREEINDEX22](#)

#### 30.846.1 Variable Documentation

##### 30.846.1.1 var NAVTREEINDEX22

Definition at line 1 of file navtreeindex22.js.

## 30.847 doc/html/navtreeindex23.js File Reference

### Variables

- var [NAVTREEINDEX23](#)

#### 30.847.1 Variable Documentation

##### 30.847.1.1 var NAVTREEINDEX23

Definition at line 1 of file navtreeindex23.js.

## 30.848 doc/html/navtreeindex24.js File Reference

### Variables

- var [NAVTREEINDEX24](#)

#### 30.848.1 Variable Documentation

##### 30.848.1.1 var NAVTREEINDEX24

Definition at line 1 of file navtreeindex24.js.

## 30.849 doc/html/navtreeindex25.js File Reference

### Variables

- var [NAVTREEINDEX25](#)

### 30.849.1 Variable Documentation

#### 30.849.1.1 var NAVTREEINDEX25

Definition at line 1 of file navtreeindex25.js.

## 30.850 doc/html/navtreeindex26.js File Reference

### Variables

- var [NAVTREEINDEX26](#)

### 30.850.1 Variable Documentation

#### 30.850.1.1 var NAVTREEINDEX26

Definition at line 1 of file navtreeindex26.js.

## 30.851 doc/html/navtreeindex27.js File Reference

### Variables

- var [NAVTREEINDEX27](#)

### 30.851.1 Variable Documentation

#### 30.851.1.1 var NAVTREEINDEX27

Definition at line 1 of file navtreeindex27.js.

## 30.852 doc/html/navtreeindex28.js File Reference

### Variables

- var [NAVTREEINDEX28](#)

### 30.852.1 Variable Documentation

#### 30.852.1.1 var NAVTREEINDEX28

Definition at line 1 of file navtreeindex28.js.



## 30.853 doc/html/navtreeindex29.js File Reference

### Variables

- var [NAVTREEINDEX29](#)

### 30.853.1 Variable Documentation

#### 30.853.1.1 var NAVTREEINDEX29

Definition at line 1 of file navtreeindex29.js.

## 30.854 doc/html/navtreeindex3.js File Reference

### Variables

- var [NAVTREEINDEX3](#)

### 30.854.1 Variable Documentation

#### 30.854.1.1 var NAVTREEINDEX3

Definition at line 1 of file navtreeindex3.js.

## 30.855 doc/html/navtreeindex30.js File Reference

### Variables

- var [NAVTREEINDEX30](#)

### 30.855.1 Variable Documentation

#### 30.855.1.1 var NAVTREEINDEX30

Definition at line 1 of file navtreeindex30.js.

## 30.856 doc/html/navtreeindex31.js File Reference

### Variables

- var [NAVTREEINDEX31](#)

### 30.856.1 Variable Documentation

#### 30.856.1.1 var NAVTREEINDEX31

Definition at line 1 of file navtreeindex31.js.

## 30.857 doc/html/navtreeindex32.js File Reference

### Variables

- var [NAVTREEINDEX32](#)

### 30.857.1 Variable Documentation

#### 30.857.1.1 var NAVTREEINDEX32

Definition at line 1 of file navtreeindex32.js.

## 30.858 doc/html/navtreeindex33.js File Reference

### Variables

- var [NAVTREEINDEX33](#)

### 30.858.1 Variable Documentation

#### 30.858.1.1 var NAVTREEINDEX33

Definition at line 1 of file navtreeindex33.js.

## 30.859 doc/html/navtreeindex34.js File Reference

### Variables

- var [NAVTREEINDEX34](#)

### 30.859.1 Variable Documentation

#### 30.859.1.1 var NAVTREEINDEX34

Definition at line 1 of file navtreeindex34.js.

## 30.860 doc/html/navtreeindex35.js File Reference

### Variables

- var [NAVTREEINDEX35](#)

### 30.860.1 Variable Documentation

#### 30.860.1.1 var NAVTREEINDEX35

Definition at line 1 of file navtreeindex35.js.

## 30.861 doc/html/navtreeindex36.js File Reference

### Variables

- var [NAVTREEINDEX36](#)

### 30.861.1 Variable Documentation

#### 30.861.1.1 var NAVTREEINDEX36

Definition at line 1 of file navtreeindex36.js.

## 30.862 doc/html/navtreeindex37.js File Reference

### Variables

- var [NAVTREEINDEX37](#)

### 30.862.1 Variable Documentation

#### 30.862.1.1 var NAVTREEINDEX37

Definition at line 1 of file navtreeindex37.js.

## 30.863 doc/html/navtreeindex38.js File Reference

### Variables

- var [NAVTREEINDEX38](#)

### 30.863.1 Variable Documentation

#### 30.863.1.1 var NAVTREEINDEX38

Definition at line 1 of file navtreeindex38.js.

## 30.864 doc/html/navtreeindex39.js File Reference

### Variables

- var [NAVTREEINDEX39](#)

### 30.864.1 Variable Documentation

#### 30.864.1.1 var NAVTREEINDEX39

Definition at line 1 of file navtreeindex39.js.

## 30.865 doc/html/navtreeindex4.js File Reference

### Variables

- var [NAVTREEINDEX4](#)

### 30.865.1 Variable Documentation

#### 30.865.1.1 var NAVTREEINDEX4

Definition at line 1 of file navtreeindex4.js.

## 30.866 doc/html/navtreeindex40.js File Reference

### Variables

- var [NAVTREEINDEX40](#)

### 30.866.1 Variable Documentation

#### 30.866.1.1 var NAVTREEINDEX40

Definition at line 1 of file navtreeindex40.js.

## 30.867 doc/html/navtreeindex41.js File Reference

### Variables

- var [NAVTREEINDEX41](#)

### 30.867.1 Variable Documentation

#### 30.867.1.1 var NAVTREEINDEX41

Definition at line 1 of file navtreeindex41.js.

## 30.868 doc/html/navtreeindex42.js File Reference

### Variables

- var [NAVTREEINDEX42](#)

### 30.868.1 Variable Documentation

#### 30.868.1.1 var NAVTREEINDEX42

Definition at line 1 of file navtreeindex42.js.

## 30.869 doc/html/navtreeindex43.js File Reference

### Variables

- var [NAVTREEINDEX43](#)

### 30.869.1 Variable Documentation

#### 30.869.1.1 var NAVTREEINDEX43

Definition at line 1 of file navtreeindex43.js.

## 30.870 doc/html/navtreeindex44.js File Reference

### Variables

- var [NAVTREEINDEX44](#)

### 30.870.1 Variable Documentation

#### 30.870.1.1 var NAVTREEINDEX44

Definition at line 1 of file navtreeindex44.js.

## 30.871 doc/html/navtreeindex45.js File Reference

### Variables

- var [NAVTREEINDEX45](#)

### 30.871.1 Variable Documentation

#### 30.871.1.1 var NAVTREEINDEX45

Definition at line 1 of file navtreeindex45.js.

## 30.872 doc/html/navtreeindex46.js File Reference

### Variables

- var [NAVTREEINDEX46](#)

### 30.872.1 Variable Documentation

#### 30.872.1.1 var NAVTREEINDEX46

Definition at line 1 of file navtreeindex46.js.

## 30.873 doc/html/navtreeindex47.js File Reference

### Variables

- var [NAVTREEINDEX47](#)

### 30.873.1 Variable Documentation

#### 30.873.1.1 var NAVTREEINDEX47

Definition at line 1 of file navtreeindex47.js.

## 30.874 doc/html/navtreeindex48.js File Reference

### Variables

- var [NAVTREEINDEX48](#)

#### 30.874.1 Variable Documentation

##### 30.874.1.1 var NAVTREEINDEX48

Definition at line 1 of file navtreeindex48.js.

## 30.875 doc/html/navtreeindex49.js File Reference

### Variables

- var [NAVTREEINDEX49](#)

#### 30.875.1 Variable Documentation

##### 30.875.1.1 var NAVTREEINDEX49

Definition at line 1 of file navtreeindex49.js.

## 30.876 doc/html/navtreeindex5.js File Reference

### Variables

- var [NAVTREEINDEX5](#)

#### 30.876.1 Variable Documentation

##### 30.876.1.1 var NAVTREEINDEX5

Definition at line 1 of file navtreeindex5.js.

## 30.877 doc/html/navtreeindex50.js File Reference

### Variables

- var [NAVTREEINDEX50](#)

### 30.877.1 Variable Documentation

#### 30.877.1.1 var NAVTREEINDEX50

Definition at line 1 of file navtreeindex50.js.

## 30.878 doc/html/navtreeindex51.js File Reference

### Variables

- var [NAVTREEINDEX51](#)

### 30.878.1 Variable Documentation

#### 30.878.1.1 var NAVTREEINDEX51

Definition at line 1 of file navtreeindex51.js.

## 30.879 doc/html/navtreeindex52.js File Reference

### Variables

- var [NAVTREEINDEX52](#)

### 30.879.1 Variable Documentation

#### 30.879.1.1 var NAVTREEINDEX52

Definition at line 1 of file navtreeindex52.js.

## 30.880 doc/html/navtreeindex53.js File Reference

### Variables

- var [NAVTREEINDEX53](#)

### 30.880.1 Variable Documentation

#### 30.880.1.1 var NAVTREEINDEX53

Definition at line 1 of file navtreeindex53.js.



## 30.881 doc/html/navtreeindex54.js File Reference

### Variables

- var [NAVTREEINDEX54](#)

### 30.881.1 Variable Documentation

#### 30.881.1.1 var NAVTREEINDEX54

Definition at line 1 of file navtreeindex54.js.

## 30.882 doc/html/navtreeindex55.js File Reference

### Variables

- var [NAVTREEINDEX55](#)

### 30.882.1 Variable Documentation

#### 30.882.1.1 var NAVTREEINDEX55

Definition at line 1 of file navtreeindex55.js.

## 30.883 doc/html/navtreeindex56.js File Reference

### Variables

- var [NAVTREEINDEX56](#)

### 30.883.1 Variable Documentation

#### 30.883.1.1 var NAVTREEINDEX56

Definition at line 1 of file navtreeindex56.js.

## 30.884 doc/html/navtreeindex57.js File Reference

### Variables

- var [NAVTREEINDEX57](#)

### 30.884.1 Variable Documentation

#### 30.884.1.1 var NAVTREEINDEX57

Definition at line 1 of file navtreeindex57.js.

## 30.885 doc/html/navtreeindex58.js File Reference

### Variables

- var [NAVTREEINDEX58](#)

### 30.885.1 Variable Documentation

#### 30.885.1.1 var NAVTREEINDEX58

Definition at line 1 of file navtreeindex58.js.

## 30.886 doc/html/navtreeindex59.js File Reference

### Variables

- var [NAVTREEINDEX59](#)

### 30.886.1 Variable Documentation

#### 30.886.1.1 var NAVTREEINDEX59

Definition at line 1 of file navtreeindex59.js.

## 30.887 doc/html/navtreeindex6.js File Reference

### Variables

- var [NAVTREEINDEX6](#)

### 30.887.1 Variable Documentation

#### 30.887.1.1 var NAVTREEINDEX6

Definition at line 1 of file navtreeindex6.js.

## 30.888 doc/html/navtreeindex60.js File Reference

### Variables

- var [NAVTREEINDEX60](#)

### 30.888.1 Variable Documentation

#### 30.888.1.1 var NAVTREEINDEX60

Definition at line 1 of file navtreeindex60.js.

## 30.889 doc/html/navtreeindex61.js File Reference

### Variables

- var [NAVTREEINDEX61](#)

### 30.889.1 Variable Documentation

#### 30.889.1.1 var NAVTREEINDEX61

Definition at line 1 of file navtreeindex61.js.

## 30.890 doc/html/navtreeindex62.js File Reference

### Variables

- var [NAVTREEINDEX62](#)

### 30.890.1 Variable Documentation

#### 30.890.1.1 var NAVTREEINDEX62

Definition at line 1 of file navtreeindex62.js.

## 30.891 doc/html/navtreeindex63.js File Reference

### Variables

- var [NAVTREEINDEX63](#)

### 30.891.1 Variable Documentation

#### 30.891.1.1 var NAVTREEINDEX63

Definition at line 1 of file navtreeindex63.js.

## 30.892 doc/html/navtreeindex7.js File Reference

### Variables

- var [NAVTREEINDEX7](#)

### 30.892.1 Variable Documentation

#### 30.892.1.1 var NAVTREEINDEX7

Definition at line 1 of file navtreeindex7.js.

## 30.893 doc/html/navtreeindex8.js File Reference

### Variables

- var [NAVTREEINDEX8](#)

### 30.893.1 Variable Documentation

#### 30.893.1.1 var NAVTREEINDEX8

Definition at line 1 of file navtreeindex8.js.

## 30.894 doc/html/navtreeindex9.js File Reference

### Variables

- var [NAVTREEINDEX9](#)

### 30.894.1 Variable Documentation

#### 30.894.1.1 var NAVTREEINDEX9

Definition at line 1 of file navtreeindex9.js.

## 30.895 doc/html/newfft\_8cpp.js File Reference

### Variables

- var [newfft\\_8cpp](#)

### 30.895.1 Variable Documentation

#### 30.895.1.1 var newfft\_8cpp

##### Initial value:

```
=
[
 ["REPORT", "newfft_8cpp.html#a787099914a94ed31fa544b985b55752f", null],
 ["WANT_MATH", "newfft_8cpp.html#a8335b327d416f961a5dbaa0e058cc515", null],
 ["WANT_STREAM", "newfft_8cpp.html#a6ed6c2d6e68d8f0e7900326b4e30850c", null],
 ["square", "newfft_8cpp.html#ae0c42a7620957500708a95cbb31e4260", null],
 ["square", "newfft_8cpp.html#ab12b3cad66448e2bfc8c590d002052da", null]
]
```

Definition at line 1 of file newfft\_8cpp.js.

## 30.896 doc/html/newknots\_8c.js File Reference

### Variables

- var [newknots\\_8c](#)

### 30.896.1 Variable Documentation

#### 30.896.1.1 var newknots\_8c

##### Initial value:

```
=
[
 ["NEWKNOTS", "newknots_8c.html#a2cfe4b9fd93d72364a6776e3f87406b9", null],
 ["newknots", "newknots_8c.html#abc755a018e609869e18ffac75568f040", null]
]
```

Definition at line 1 of file newknots\_8c.js.

## 30.897 doc/html/newmat1\_8cpp.js File Reference

### Variables

- var [newmat1\\_8cpp](#)

### 30.897.1 Variable Documentation

#### 30.897.1.1 var newmat1\_8cpp

**Initial value:**

```
=
[
 ["REPORT", "newmat1_8cpp.html#a787099914a94ed31fa544b985b55752f", null],
 ["Rectangular", "newmat1_8cpp.html#a858f15c5cad7d2271c38bc54fbca9b5c", null]
]
```

Definition at line 1 of file newmat1\_8cpp.js.

## 30.898 doc/html/newmat2\_8cpp.js File Reference

### Variables

- var [newmat2\\_8cpp](#)

### 30.898.1 Variable Documentation

#### 30.898.1.1 var newmat2\_8cpp

**Initial value:**

```
=
[
 ["MONITOR", "newmat2_8cpp.html#a152473011fe478405090e77cd65667fb", null],
 ["REPORT", "newmat2_8cpp.html#a787099914a94ed31fa544b985b55752f", null],
 ["WANT_MATH", "newmat2_8cpp.html#a8335b327d416f961a5dbaa0e058cc515", null],
 ["DotProd", "newmat2_8cpp.html#ad4a6e1e55a32b648fc6564538f1cfc35", null]
]
```

Definition at line 1 of file newmat2\_8cpp.js.

## 30.899 doc/html/newmat3\_8cpp.js File Reference

### Variables

- var [newmat3\\_8cpp](#)

### 30.899.1 Variable Documentation

#### 30.899.1.1 var newmat3\_8cpp

**Initial value:**

```
=
[
 ["MONITOR", "newmat3_8cpp.html#a152473011fe478405090e77cd65667fb", null],
 ["REPORT", "newmat3_8cpp.html#a787099914a94ed31fa544b985b55752f", null]
]
```

Definition at line 1 of file newmat3\_8cpp.js.

## 30.900 doc/html/newmat4\_8cpp.js File Reference

### Variables

- var [newmat4\\_8cpp](#)

#### 30.900.1 Variable Documentation

##### 30.900.1.1 var newmat4\_8cpp

###### Initial value:

```
=
[
 ["DO_SEARCH", "newmat4_8cpp.html#a46095a903a9295f8efd055f5c72f4dac", null],
 ["REPORT", "newmat4_8cpp.html#a787099914a94ed31fa544b985b55752f", null],
 ["Compare", "newmat4_8cpp.html#afad59b9d5b29862e68144236e760ccb8", null]
]
```

Definition at line 1 of file newmat4\_8cpp.js.

## 30.901 doc/html/newmat5\_8cpp.js File Reference

### Variables

- var [newmat5\\_8cpp](#)

#### 30.901.1 Variable Documentation

##### 30.901.1.1 var newmat5\_8cpp

###### Initial value:

```
=
[
 ["REPORT", "newmat5_8cpp.html#a787099914a94ed31fa544b985b55752f", null]
]
```

Definition at line 1 of file newmat5\_8cpp.js.

## 30.902 doc/html/newmat6\_8cpp.js File Reference

### Variables

- var [newmat6\\_8cpp](#)

### 30.902.1 Variable Documentation

#### 30.902.1.1 var newmat6\_8cpp

**Initial value:**

```
=
[
 ["REPORT", "newmat6_8cpp.html#a787099914a94ed31fa544b985b55752f", null],
 ["KP", "newmat6_8cpp.html#ae3166f1201bb79865ac1945a9ebc64d6", null],
 ["operator-", "newmat6_8cpp.html#a46268a98e860120bbd9f0dd63aff0da5", null],
 ["SP", "newmat6_8cpp.html#a12f8dde3a19dd650078a3243e0517e84", null]
]
```

Definition at line 1 of file newmat6\_8cpp.js.

## 30.903 doc/html/newmat7\_8cpp.js File Reference

### Variables

- var [newmat7\\_8cpp](#)

### 30.903.1 Variable Documentation

#### 30.903.1.1 var newmat7\_8cpp

**Initial value:**

```
=
[
 ["REPORT", "newmat7_8cpp.html#a787099914a94ed31fa544b985b55752f", null],
 ["IsZero", "newmat7_8cpp.html#a3038afe6bd8d1b109230942ebfdb58cf", null],
 ["operator==", "newmat7_8cpp.html#adedal12c50558644158778d1e55d325f", null],
 ["operator=", "newmat7_8cpp.html#a28893be0b55f3b6e534c15d8d7a7e0cf", null]
]
```

Definition at line 1 of file newmat7\_8cpp.js.

## 30.904 doc/html/newmat8\_8cpp.js File Reference

### Variables

- var [newmat8\\_8cpp](#)



### 30.904.1 Variable Documentation

#### 30.904.1.1 var newmat8\_8cpp

**Initial value:**

```
=
[
 ["REPORT", "newmat8_8cpp.html#a787099914a94ed31fa544b985b55752f", null],
 ["WANT_MATH", "newmat8_8cpp.html#a8335b327d416f961a5dbaa0e058cc515", null],
 ["DotProduct", "newmat8_8cpp.html#a9e8ce3d9f963c09ee55fd8e84d540973", null],
 ["square", "newmat8_8cpp.html#ae0c42a7620957500708a95cbb31e4260", null]
]
```

Definition at line 1 of file newmat8\_8cpp.js.

## 30.905 doc/html/newmat9\_8cpp.js File Reference

### Variables

- var [newmat9\\_8cpp](#)

### 30.905.1 Variable Documentation

#### 30.905.1.1 var newmat9\_8cpp

**Initial value:**

```
=
[
 ["ios_format_flags", "newmat9_8cpp.html#a0876a6126359b29117790265d60d034d", null],
 ["REPORT", "newmat9_8cpp.html#a787099914a94ed31fa544b985b55752f", null],
 ["WANT_FSTREAM", "newmat9_8cpp.html#a37d625862c54954e237886ad4bb5e75f", null],
 ["operator<<", "newmat9_8cpp.html#a766fb839d2cb6052d302c75e6254f4e3", null],
 ["operator<<<", "newmat9_8cpp.html#a027ba0d51e59dba3c06b646673c8f3a0", null]
]
```

Definition at line 1 of file newmat9\_8cpp.js.

## 30.906 doc/html/newmat\_8h.js File Reference

### Variables

- var [newmat\\_8h](#)

### 30.906.1 Variable Documentation

#### 30.906.1.1 var newmat\_8h

Definition at line 1 of file newmat\_8h.js.

## 30.907 doc/html/newmatap\_8h.js File Reference

### Variables

- var [newmatap\\_8h](#)

### 30.907.1 Variable Documentation

#### 30.907.1.1 var newmatap\_8h

Definition at line 1 of file newmatap\_8h.js.

## 30.908 doc/html/newmatex\_8cpp.js File Reference

### Variables

- var [newmatex\\_8cpp](#)

### 30.908.1 Variable Documentation

#### 30.908.1.1 var newmatex\_8cpp

#### Initial value:

```
=
[
 ["WANT_STREAM", "newmatex_8cpp.html#a6ed6c2d6e68d8f0e7900326b4e30850c", null],
 ["MatrixErrorNoSpace", "newmatex_8cpp.html#ab44cf78e8368f61a521f79abbd4ae2b7", null]
]
```

Definition at line 1 of file newmatex\_8cpp.js.

## 30.909 doc/html/newmatio\_8h.js File Reference

### Variables

- var [newmatio\\_8h](#)

### 30.909.1 Variable Documentation

#### 30.909.1.1 var newmatio\_8h

#### Initial value:

```
=
[
 ["NEWMATIO_LIB", "newmatio_8h.html#a65fb8e2c45ceffd1871256a348617cbf", null],
 ["operator<<", "newmatio_8h.html#aac312f0f56489c9a1f829f5f9bbd17b8", null],
 ["operator<<", "newmatio_8h.html#adc370b27ca69af82de17f158a4973cec", null]
]
```

Definition at line 1 of file newmatio\_8h.js.

## 30.910 doc/html/newmatnl\_8cpp.js File Reference

### Variables

- var [newmatnl\\_8cpp](#)

### 30.910.1 Variable Documentation

#### 30.910.1.1 var newmatnl\_8cpp

##### Initial value:

```
=
[
 ["WANT_MATH", "newmatnl_8cpp.html#a8335b327d416f961a5dbaa0e058cc515", null],
 ["WANT_STREAM", "newmatnl_8cpp.html#a6ed6c2d6e68d8f0e7900326b4e30850c", null]
]
```

Definition at line 1 of file newmatnl\_8cpp.js.

## 30.911 doc/html/newmatnl\_8h.js File Reference

### Variables

- var [newmatnl\\_8h](#)

### 30.911.1 Variable Documentation

#### 30.911.1.1 var newmatnl\_8h

##### Initial value:

```
=
[
 ["FindMaximum2", "class_new_mat_l_l_find_maximum2.html", "class_new_mat_l_l_find_maximum2"],
 ["R1_Col_ID", "class_new_mat_l_l_rl_col_id.html", "class_new_mat_l_l_rl_col_id"],
 ["NonLinearLeastSquares", "class_new_mat_l_l_non_linear_least_squares.html", "class_new_mat_l_l_non_linear_least_squares"],
 ["LL_D_FI", "class_new_mat_l_l_l_d_fi.html", "class_new_mat_l_l_l_d_fi"],
 ["MLE_D_FI", "class_new_mat_l_l_m_l_e_d_fi.html", "class_new_mat_l_l_m_l_e_d_fi"],
 ["NEWMATNL_LIB", "newmatnl_8h.html#aed6d29a446241bb9799d3e281972c9c1", null]
]
```

Definition at line 1 of file newmatnl\_8h.js.

## 30.912 doc/html/newmatrc\_8h.js File Reference

### Variables

- var [newmatrc\\_8h](#)

### 30.912.1 Variable Documentation

#### 30.912.1.1 var newmatrc\_8h

##### Initial value:

```
=
[
 ["LoadAndStoreFlag", "class_load_and_store_flag.html", "class_load_and_store_flag"],
 ["MatrixRowCol", "class_matrix_row_col.html", "class_matrix_row_col"],
 ["MatrixRow", "class_matrix_row.html", "class_matrix_row"],
 ["MatrixCol", "class_matrix_col.html", "class_matrix_col"],
 ["MatrixColX", "class_matrix_col_x.html", "class_matrix_col_x"],
 ["NEWMATRC_LIB", "newmatrc_8h.html#a96d76d6bee402806b06fccc48adf34e0", null],
 ["LSF", "newmatrc_8h.html#ae75c691eb2643cc7fe952b12f074040c", [
 ["LoadOnEntry", "newmatrc_8h.html#ae75c691eb2643cc7fe952b12f074040caf82aaa0389ad880499aa17ee995acd62", null],
 ["StoreOnExit", "newmatrc_8h.html#ae75c691eb2643cc7fe952b12f074040cae09c50f27d849b518368992b0f092916", null],
 ["DirectPart", "newmatrc_8h.html#ae75c691eb2643cc7fe952b12f074040cac1dd577ab18fd18fcbe43d9654721c8f", null],
 ["StoreHere", "newmatrc_8h.html#ae75c691eb2643cc7fe952b12f074040cab2a94c88732f5f171000bbfcafcb98d", null],
 ["HaveStore", "newmatrc_8h.html#ae75c691eb2643cc7fe952b12f074040ca53133c14cb06987614a1d83f0c44700c", null]
]
]]
```

Definition at line 1 of file newmatrc\_8h.js.

## 30.913 doc/html/newmatrm\_8cpp.js File Reference

### Variables

- var [newmatrm\\_8cpp](#)

### 30.913.1 Variable Documentation

#### 30.913.1.1 var newmatrm\_8cpp

##### Initial value:

```
=
[
 ["REPORT", "newmatrm_8cpp.html#a787099914a94ed31fa544b985b55752f", null],
 ["ComplexScale", "newmatrm_8cpp.html#ae7d0eadc62c8b79047f0d82cb2addb5a", null],
 ["Rotate", "newmatrm_8cpp.html#ab7b780d91409fd2f1e11b6d07789c231", null]
]
```

Definition at line 1 of file newmatrm\_8cpp.js.

## 30.914 doc/html/newmatrm\_8h.js File Reference

### Variables

- var [newmatrm\\_8h](#)

### 30.914.1 Variable Documentation

#### 30.914.1.1 var newmatrm\_8h

##### Initial value:

```
=
[
 ["RectMatrixRowCol", "class_rect_matrix_row_col.html", "class_rect_matrix_row_col"],
 ["RectMatrixRow", "class_rect_matrix_row.html", "class_rect_matrix_row"],
 ["RectMatrixCol", "class_rect_matrix_col.html", "class_rect_matrix_col"],
 ["RectMatrixDiag", "class_rect_matrix_diag.html", "class_rect_matrix_diag"],
 ["NEWMATRM_LIB", "newmatrm_8h.html#a57c1191464ba3297be3ac91c6a339c60", null],
 ["sign", "newmatrm_8h.html#a405e11394e2ab3803e14552eb1f7b1d4", null],
 ["square", "newmatrm_8h.html#ae0c42a7620957500708a95cbb31e4260", null]
]
```

Definition at line 1 of file newmatrm\_8h.js.

## 30.915 doc/html/nl\_\_ex\_8cpp.js File Reference

### Variables

- var [nl\\_\\_ex\\_8cpp](#)

### 30.915.1 Variable Documentation

#### 30.915.1.1 var nl\_\_ex\_8cpp

##### Initial value:

```
=
[
 ["Model_3pe", "class_model__3pe.html", "class_model__3pe"],
 ["WANT_MATH", "nl__ex_8cpp.html#a8335b327d416f961a5dbaa0e058cc515", null],
 ["WANT_STREAM", "nl__ex_8cpp.html#a6ed6c2d6e68d8f0e7900326b4e30850c", null],
 ["main", "nl__ex_8cpp.html#ae66f6b31b5ad750f1fe042a706a4e3d4", null]
]
```

Definition at line 1 of file nl\_\_ex\_8cpp.js.

## 30.916 doc/html/orient\_curves\_8h.js File Reference

### Variables

- var [orient\\_curves\\_8h](#)

### 30.916.1 Variable Documentation

#### 30.916.1.1 var orient\_curves\_8h

**Initial value:**

```
=
[
 ["orientCurves", "orient_curves_8h.html#a5bd2ee4daeb32bba01486884a638e5eb", null]
]
```

Definition at line 1 of file orient\_curves\_8h.js.

## 30.917 doc/html/pickcrvsf\_8c.js File Reference

### Variables

- var [pickcrvsf\\_8c](#)

### 30.917.1 Variable Documentation

#### 30.917.1.1 var pickcrvsf\_8c

**Initial value:**

```
=
[
 ["PICK_CRV_SF", "pickcrvsf_8c.html#a155aaf976a37fafa8e7bb3d1c973d64c", null],
 ["pick_crv_sf", "pickcrvsf_8c.html#a1ec8aa653acdb0bfd2a176460f6b9833", null]
]
```

Definition at line 1 of file pickcrvsf\_8c.js.

## 30.918 doc/html/pocrvtang\_8c.js File Reference

### Variables

- var [pocrvtang\\_8c](#)

### 30.918.1 Variable Documentation

#### 30.918.1.1 var pocrvtang\_8c

**Initial value:**

```
=
[
 ["PO_CRV_TANG", "pocrvtang_8c.html#af76534eda9b944a2b8b2c6ef042eddb9", null],
 ["po_crv_tang", "pocrvtang_8c.html#a6e8a1352a942920cdd5e77ddaa8aaf3a", null]
]
```

Definition at line 1 of file pocrvtang\_8c.js.

## 30.919 doc/html/precisio\_8h.js File Reference

### Variables

- var [precisio\\_8h](#)

#### 30.919.1 Variable Documentation

##### 30.919.1.1 var precisio\_8h

#### Initial value:

```
=
[
 ["FloatingPointPrecision", "class_n_e_w_m_a_t_l_1_floating_point_precision.html", null],
 ["PRECISION_LIB", "precisio_8h.html#a3ab28e0c843e28acdfaa488761772d4b", null],
 ["WANT_MATH", "precisio_8h.html#a8335b327d416f961a5dbaa0e058cc515", null]
]
```

Definition at line 1 of file precisio\_8h.js.

## 30.920 doc/html/randomnoise\_8h.js File Reference

### Variables

- var [randomnoise\\_8h](#)

#### 30.920.1 Variable Documentation

##### 30.920.1.1 var randomnoise\_8h

#### Initial value:

```
=
[
 ["normalNoise", "randomnoise_8h.html#a676090feb61df5841159959dd8a2e773", null],
 ["uniformNoise", "randomnoise_8h.html#a5c4a39c5c7ee9e89ae32946275e646ce", null]
]
```

Definition at line 1 of file randomnoise\_8h.js.

## 30.921 doc/html/raster\_8h.js File Reference

### Variables

- var [raster\\_8h](#)

### 30.921.1 Variable Documentation

#### 30.921.1.1 var raster\_8h

**Initial value:**

```
=
[
 ["RASTERUTILS_H_INCLUDED", "raster_8h.html#ae6d95c22bae73000aac0efdc2997c55d", null],
 ["read_ppm_file", "raster_8h.html#ab0cd289afbca1d817808954b3b7b7abc", null],
 ["write_ppm_file", "raster_8h.html#a9695a5e9660225deeed83e08f077a4f2", null]
]
```

Definition at line 1 of file raster\_8h.js.

## 30.922 doc/html/refine\_\_all\_8c.js File Reference

### Variables

- var [refine\\_\\_all\\_8c](#)

### 30.922.1 Variable Documentation

#### 30.922.1.1 var refine\_\_all\_8c

**Initial value:**

```
=
[
 ["REFINE_ALL", "refine__all_8c.html#a7021f2e1b451a7d88475a011da9cffe9", null],
 ["refine_all", "refine__all_8c.html#a09d880f4de7830b355b4e88ce89a199c", null]
]
```

Definition at line 1 of file refine\_\_all\_8c.js.

## 30.923 doc/html/resize.js File Reference

### Functions

- function [readCookie](#) (cookie)
- function [writeCookie](#) (cookie, val, expiration)
- function [resizeWidth](#) ()
- function [restoreWidth](#) (navWidth)
- function [resizeHeight](#) ()
- function [initResizable](#) ()



## Variables

- var `cookie_namespace` = 'doxygen'
- var `sidenav`
- var `navtree`
- var `content`
- var `header`

## 30.923.1 Function Documentation

### 30.923.1.1 function `initResizable ( )`

Definition at line 62 of file `resize.js`.

### 30.923.1.2 function `readCookie ( cookie )`

Definition at line 4 of file `resize.js`.

### 30.923.1.3 function `resizeHeight ( )`

Definition at line 52 of file `resize.js`.

### 30.923.1.4 function `resizeWidth ( )`

Definition at line 37 of file `resize.js`.

### 30.923.1.5 function `restoreWidth ( navWidth )`

Definition at line 45 of file `resize.js`.

### 30.923.1.6 function `writeCookie ( cookie, val, expiration )`

Definition at line 25 of file `resize.js`.

## 30.923.2 Variable Documentation

### 30.923.2.1 var `content`

Definition at line 2 of file `resize.js`.

### 30.923.2.2 var `cookie_namespace` = 'doxygen'

Definition at line 1 of file `resize.js`.

### 30.923.2.3 var header

Definition at line 2 of file resize.js.

### 30.923.2.4 var navtree

Definition at line 2 of file resize.js.

### 30.923.2.5 var sidenav

Definition at line 2 of file resize.js.

## 30.924 doc/html/s1001\_8c.js File Reference

### Variables

- var [s1001\\_8c](#)

### 30.924.1 Variable Documentation

#### 30.924.1.1 var s1001\_8c

#### Initial value:

```
=
[
 ["s1001", "s1001_8c.html#a56ccf110d6c3ffd6ec1bc77ba9bf017f", null],
 ["s1001", "s1001_8c.html#a80164f5f99c5cfb032d8384a3dc986ba", null]
]
```

Definition at line 1 of file s1001\_8c.js.

## 30.925 doc/html/s1011\_8c.js File Reference

### Variables

- var [s1011\\_8c](#)

### 30.925.1 Variable Documentation

#### 30.925.1.1 var s1011\_8c

**Initial value:**

```
=
[
 ["s1011", "s1011_8c.html#a2450ea7c588d7669544dd8f787498d80", null],
 ["s1011", "s1011_8c.html#ad410d6930a817ffac1b37386954c46a1", null]
]
```

Definition at line 1 of file s1011\_8c.js.

## 30.926 doc/html/s1012\_8c.js File Reference

### Variables

- var [s1012\\_8c](#)

### 30.926.1 Variable Documentation

#### 30.926.1.1 var s1012\_8c

**Initial value:**

```
=
[
 ["s1012", "s1012_8c.html#a6271f794dd9fabab2eac8e3053de0319", null],
 ["s1012", "s1012_8c.html#a976d969f65fd46bba600e032927c02c0", null]
]
```

Definition at line 1 of file s1012\_8c.js.

## 30.927 doc/html/s1013\_8c.js File Reference

### Variables

- var [s1013\\_8c](#)

### 30.927.1 Variable Documentation

#### 30.927.1.1 var s1013\_8c

**Initial value:**

```
=
[
 ["s1013", "s1013_8c.html#aa28f965565254269350f5017d414b42a", null],
 ["s1013", "s1013_8c.html#a728ddfbcb68d2a36a9d89b2042da2ac1d", null]
]
```

Definition at line 1 of file s1013\_8c.js.

## 30.928 doc/html/s1014\_8c.js File Reference

### Variables

- var [s1014\\_8c](#)

### 30.928.1 Variable Documentation

#### 30.928.1.1 var s1014\_8c

##### Initial value:

```
=
[
 ["s1014", "s1014_8c.html#a6387307dd5d73e0625ba7692f19936d2", null],
 ["s1014", "s1014_8c.html#afdde53e01257120ed8dff6eaea809e27", null]
]
```

Definition at line 1 of file s1014\_8c.js.

## 30.929 doc/html/s1015\_8c.js File Reference

### Variables

- var [s1015\\_8c](#)

### 30.929.1 Variable Documentation

#### 30.929.1.1 var s1015\_8c

##### Initial value:

```
=
[
 ["s1015", "s1015_8c.html#aac8add6ae31d5bf0b48e9e638e5f0016", null],
 ["s1015", "s1015_8c.html#a7c7cb84a3a10e0ba1a3282a81d12b3a9", null]
]
```

Definition at line 1 of file s1015\_8c.js.

## 30.930 doc/html/s1016\_8c.js File Reference

### Variables

- var [s1016\\_8c](#)

### 30.930.1 Variable Documentation

#### 30.930.1.1 var s1016\_8c

**Initial value:**

```
=
[
 ["s1016", "s1016_8c.html#ae9abb35161fc73b35ad6a7d187928232", null],
 ["s1016", "s1016_8c.html#a0c6d91061697e2998e64585592fefaf4", null]
]
```

Definition at line 1 of file s1016\_8c.js.

## 30.931 doc/html/s1017\_8c.js File Reference

### Variables

- var [s1017\\_8c](#)

### 30.931.1 Variable Documentation

#### 30.931.1.1 var s1017\_8c

**Initial value:**

```
=
[
 ["s1017", "s1017_8c.html#a5259eaf0d4050357b7138c0f0a3183ab", null],
 ["s1017", "s1017_8c.html#a30761a4ad3ec2cfc9cd26a38f7c52d6c", null]
]
```

Definition at line 1 of file s1017\_8c.js.

## 30.932 doc/html/s1018\_8c.js File Reference

### Variables

- var [s1018\\_8c](#)

### 30.932.1 Variable Documentation

#### 30.932.1.1 var s1018\_8c

**Initial value:**

```
=
[
 ["s1018", "s1018_8c.html#af72f29e0e0f70c80f0a0393e6d1a0dc8", null],
 ["s1018", "s1018_8c.html#a1c00f59e472b28f899644d258992859e", null]
]
```

Definition at line 1 of file s1018\_8c.js.

## 30.933 doc/html/s1021\_8c.js File Reference

### Variables

- var [s1021\\_8c](#)

### 30.933.1 Variable Documentation

#### 30.933.1.1 var s1021\_8c

##### Initial value:

```
=
[
 ["s1021", "s1021_8c.html#a2a8a8b124088a4401508238e97705ca8", null],
 ["s1021", "s1021_8c.html#a9c29a340f1991e0d5bd3625fb8861ae9", null]
]
```

Definition at line 1 of file s1021\_8c.js.

## 30.934 doc/html/s1022\_8c.js File Reference

### Variables

- var [s1022\\_8c](#)

### 30.934.1 Variable Documentation

#### 30.934.1.1 var s1022\_8c

##### Initial value:

```
=
[
 ["s1022", "s1022_8c.html#a1a1336f406499846d8524b12e2049ed3", null],
 ["s1022", "s1022_8c.html#a50db2aed3ebc0f36aa4bf2aaf3c7592a", null]
]
```

Definition at line 1 of file s1022\_8c.js.

## 30.935 doc/html/s1023\_8c.js File Reference

### Variables

- var [s1023\\_8c](#)

### 30.935.1 Variable Documentation

#### 30.935.1.1 var s1023\_8c

**Initial value:**

```
=
[
 ["s1023", "s1023_8c.html#a2319b7b895af87056c0dbbac2f8c1db7", null],
 ["s1023", "s1023_8c.html#a1397ccd0fbaa0287d0522c7ec60cd84e", null]
]
```

Definition at line 1 of file s1023\_8c.js.

## 30.936 doc/html/s1024\_8c.js File Reference

### Variables

- var [s1024\\_8c](#)

### 30.936.1 Variable Documentation

#### 30.936.1.1 var s1024\_8c

**Initial value:**

```
=
[
 ["s1024", "s1024_8c.html#a5ccd291a4e3f4bc7fb2a19eadd3932b", null],
 ["s1024", "s1024_8c.html#a053e206fdca7490b98d5af6691908037", null]
]
```

Definition at line 1 of file s1024\_8c.js.

## 30.937 doc/html/s1025\_8c.js File Reference

### Variables

- var [s1025\\_8c](#)

### 30.937.1 Variable Documentation

#### 30.937.1.1 var s1025\_8c

**Initial value:**

```
=
[
 ["s1025", "s1025_8c.html#a83f0f0cacd1261f3061f555f9141ac5c", null],
 ["s1025", "s1025_8c.html#ab781e32a5221879d484d15db903a0b42", null]
]
```

Definition at line 1 of file s1025\_8c.js.

## 30.938 doc/html/s1119\_8c.js File Reference

### Variables

- var [s1119\\_8c](#)

### 30.938.1 Variable Documentation

#### 30.938.1.1 var s1119\_8c

##### Initial value:

```
=
[
 ["s1119", "s1119_8c.html#acf43354b6dbc649264e8cfd7ab88fa38", null],
 ["s1119", "s1119_8c.html#a425aa27d6333f7fb2588c8dffeda922b", null]
]
```

Definition at line 1 of file s1119\_8c.js.

## 30.939 doc/html/s1161\_8c.js File Reference

### Variables

- var [s1161\\_8c](#)

### 30.939.1 Variable Documentation

#### 30.939.1.1 var s1161\_8c

##### Initial value:

```
=
[
 ["s1161", "s1161_8c.html#a2b8cba67ceaf34cb0164b9c7b91fd794", null],
 ["s1161", "s1161_8c.html#a1468d5cd03f78b73b532c5ea9e764283", null]
]
```

Definition at line 1 of file s1161\_8c.js.

## 30.940 doc/html/s1162\_8c.js File Reference

### Variables

- var [s1162\\_8c](#)



### 30.940.1 Variable Documentation

#### 30.940.1.1 var s1162\_8c

**Initial value:**

```
=
[
 ["s1162", "s1162_8c.html#af082be7a7b2fec444bd7b5c9c7ccfa28", null],
 ["s1162", "s1162_8c.html#a963ee9680ed373de27d6469ddd515d76", null]
]
```

Definition at line 1 of file s1162\_8c.js.

## 30.941 doc/html/s1172\_8c.js File Reference

### Variables

- var [s1172\\_8c](#)

### 30.941.1 Variable Documentation

#### 30.941.1.1 var s1172\_8c

**Initial value:**

```
=
[
 ["s1172", "s1172_8c.html#a4f05d96507f55416f094fca8580a39d4", null],
 ["s1172", "s1172_8c.html#afbdb2cd45b33c12c544718d9a1a6edc6", null]
]
```

Definition at line 1 of file s1172\_8c.js.

## 30.942 doc/html/s1173\_8c.js File Reference

### Variables

- var [s1173\\_8c](#)

### 30.942.1 Variable Documentation

#### 30.942.1.1 var s1173\_8c

**Initial value:**

```
=
[
 ["s1173", "s1173_8c.html#a2e08429081bab8c16a1c8687443129b7", null],
 ["s1173", "s1173_8c.html#adfe997d7c6353c6fef25e5b6c554a995", null]
]
```

Definition at line 1 of file s1173\_8c.js.

## 30.943 doc/html/s1174\_8c.js File Reference

### Variables

- var [s1174\\_8c](#)

### 30.943.1 Variable Documentation

#### 30.943.1.1 var s1174\_8c

##### Initial value:

```
=
[
 ["s1174", "s1174_8c.html#a0e6e7478e43a2fffaa212d45095e4f0d", null],
 ["s1174", "s1174_8c.html#af6e3d6bac38979775cbc65e7c67a726e", null]
]
```

Definition at line 1 of file s1174\_8c.js.

## 30.944 doc/html/s1190\_8c.js File Reference

### Variables

- var [s1190\\_8c](#)

### 30.944.1 Variable Documentation

#### 30.944.1.1 var s1190\_8c

##### Initial value:

```
=
[
 ["s1190", "s1190_8c.html#a2e828756d2f79a7814e435fc7b67af8a", null],
 ["s1190", "s1190_8c.html#a571ab89bc8fd94a46cdde612db5d6be8", null]
]
```

Definition at line 1 of file s1190\_8c.js.

## 30.945 doc/html/s1192\_8c.js File Reference

### Variables

- var [s1192\\_8c](#)

### 30.945.1 Variable Documentation

#### 30.945.1.1 var s1192\_8c

**Initial value:**

```
=
[
 ["s1192", "s1192_8c.html#a63274164e81b43fdf3245113f8157d6b", null],
 ["s1192", "s1192_8c.html#ae71c8e60d97baa45efea8c3512005528", null]
]
```

Definition at line 1 of file s1192\_8c.js.

## 30.946 doc/html/s1219\_8c.js File Reference

### Variables

- var [s1219\\_8c](#)

### 30.946.1 Variable Documentation

#### 30.946.1.1 var s1219\_8c

**Initial value:**

```
=
[
 ["s1219", "s1219_8c.html#a2be917f34d9f27fb61514adcc63f1e63", null],
 ["s1219", "s1219_8c.html#a0b95d504d0e6678c9f49cbc44de8e196", null]
]
```

Definition at line 1 of file s1219\_8c.js.

## 30.947 doc/html/s1220\_8c.js File Reference

### Variables

- var [s1220\\_8c](#)

### 30.947.1 Variable Documentation

#### 30.947.1.1 var s1220\_8c

**Initial value:**

```
=
[
 ["s1220", "s1220_8c.html#a6683cab3e26a6eedbdc4d1f866a7be08", null],
 ["s1220", "s1220_8c.html#a7920b9717becd2b896bb3c4de0ce0e09", null]
]
```

Definition at line 1 of file s1220\_8c.js.

## 30.948 doc/html/s1221\_8c.js File Reference

### Variables

- var [s1221\\_8c](#)

### 30.948.1 Variable Documentation

#### 30.948.1.1 var s1221\_8c

##### Initial value:

```
=
[
 ["s1221", "s1221_8c.html#a4c9acc62a212403e6165d58f107f2e69", null],
 ["s1221", "s1221_8c.html#ad18ef4c36086f4e0505e7f8ad543eddd", null]
]
```

Definition at line 1 of file s1221\_8c.js.

## 30.949 doc/html/s1222\_8c.js File Reference

### Variables

- var [s1222\\_8c](#)

### 30.949.1 Variable Documentation

#### 30.949.1.1 var s1222\_8c

##### Initial value:

```
=
[
 ["s1222", "s1222_8c.html#a598c40da485810552a54da20c31bc771", null],
 ["s1222", "s1222_8c.html#ae0ff05bc204ef58cbc447e15da65104f", null]
]
```

Definition at line 1 of file s1222\_8c.js.

## 30.950 doc/html/s1223\_8c.js File Reference

### Variables

- var [s1223\\_8c](#)

### 30.950.1 Variable Documentation

#### 30.950.1.1 var s1223\_8c

**Initial value:**

```
=
[
 ["s1223", "s1223_8c.html#a0931d678f6ca74583e9266b5c0cbd11d", null],
 ["s1223", "s1223_8c.html#a7f266f4175fdf0623bd7218f1fca8082", null]
]
```

Definition at line 1 of file s1223\_8c.js.

## 30.951 doc/html/s1224\_8c.js File Reference

### Variables

- var [s1224\\_8c](#)

### 30.951.1 Variable Documentation

#### 30.951.1.1 var s1224\_8c

**Initial value:**

```
=
[
 ["s1224", "s1224_8c.html#a70e0a3c2bad9c05be84ac9de899208a7", null],
 ["s1224", "s1224_8c.html#a51d86bfe293f9c52a01089d5bef1cbe3", null]
]
```

Definition at line 1 of file s1224\_8c.js.

## 30.952 doc/html/s1225\_8c.js File Reference

### Variables

- var [s1225\\_8c](#)

### 30.952.1 Variable Documentation

#### 30.952.1.1 var s1225\_8c

**Initial value:**

```
=
[
 ["s1225", "s1225_8c.html#af20713c1c01491193949f485f1c190b7", null],
 ["s1225", "s1225_8c.html#a694d9795b276fc5eae6d3ef1d663803d", null]
]
```

Definition at line 1 of file s1225\_8c.js.

## 30.953 doc/html/s1226\_8c.js File Reference

### Variables

- var [s1226\\_8c](#)

### 30.953.1 Variable Documentation

#### 30.953.1.1 var s1226\_8c

##### Initial value:

```
=
[
 ["s1226", "s1226_8c.html#a81711abdea8a89b41af77cf3548424c6", null],
 ["s1226", "s1226_8c.html#af8b1be094720c25ec8e221dd2c03c6f9", null]
]
```

Definition at line 1 of file s1226\_8c.js.

## 30.954 doc/html/s1227\_8c.js File Reference

### Variables

- var [s1227\\_8c](#)

### 30.954.1 Variable Documentation

#### 30.954.1.1 var s1227\_8c

##### Initial value:

```
=
[
 ["s1227", "s1227_8c.html#af10330a648416d74712ce2fcd36c74b", null],
 ["s1227", "s1227_8c.html#a44b462edc43a3570a149b5d10a0ecd38", null]
]
```

Definition at line 1 of file s1227\_8c.js.

## 30.955 doc/html/s1231\_8c.js File Reference

### Variables

- var [s1231\\_8c](#)

### 30.955.1 Variable Documentation

#### 30.955.1.1 var s1231\_8c

**Initial value:**

```
=
[
 ["s1231", "s1231_8c.html#a258195a1741bbcae24f4fe54ab3124a0", null],
 ["s1231", "s1231_8c.html#a5cb87e14c129d1250113814231c7584b", null]
]
```

Definition at line 1 of file s1231\_8c.js.

## 30.956 doc/html/s1232\_8c.js File Reference

### Variables

- var [s1232\\_8c](#)

### 30.956.1 Variable Documentation

#### 30.956.1.1 var s1232\_8c

**Initial value:**

```
=
[
 ["s1232", "s1232_8c.html#a7cc3b5081ff3055e00d5a13f0a8e9b3d", null],
 ["s1232", "s1232_8c.html#abcc863a3de7f2bc6e38852ef06f0e29d", null]
]
```

Definition at line 1 of file s1232\_8c.js.

## 30.957 doc/html/s1233\_8c.js File Reference

### Variables

- var [s1233\\_8c](#)

### 30.957.1 Variable Documentation

#### 30.957.1.1 var s1233\_8c

**Initial value:**

```
=
[
 ["s1233", "s1233_8c.html#a7f743335f0b13e160336e69707be1d65", null],
 ["s1233", "s1233_8c.html#a891bf909d3d7bc948b1d29c2d642d46a", null]
]
```

Definition at line 1 of file s1233\_8c.js.

## 30.958 doc/html/s1235\_8c.js File Reference

### Variables

- var [s1235\\_8c](#)

### 30.958.1 Variable Documentation

#### 30.958.1.1 var s1235\_8c

##### Initial value:

```
=
[
 ["s1235", "s1235_8c.html#aa10161c73a5093ff0e3badb9487ef944", null],
 ["s1235", "s1235_8c.html#ae1473bb1de379da575f1168a08224bec", null]
]
```

Definition at line 1 of file s1235\_8c.js.

## 30.959 doc/html/s1236\_8c.js File Reference

### Variables

- var [s1236\\_8c](#)

### 30.959.1 Variable Documentation

#### 30.959.1.1 var s1236\_8c

##### Initial value:

```
=
[
 ["s1236", "s1236_8c.html#aa33fe1bb24997baf0a196e0222f92b4e", null],
 ["s1236", "s1236_8c.html#aff93eacbd6d08855e7c84fac0fc0fd29", null]
]
```

Definition at line 1 of file s1236\_8c.js.

## 30.960 doc/html/s1237\_8c.js File Reference

### Variables

- var [s1237\\_8c](#)



### 30.960.1 Variable Documentation

#### 30.960.1.1 var s1237\_8c

**Initial value:**

```
=
[
 ["s1237", "s1237_8c.html#a06f4b329ebd9478a3e737eedfa42463c", null],
 ["s1237", "s1237_8c.html#af8462ab7f71fdbe9fc94f1c838cf75b4", null]
]
```

Definition at line 1 of file s1237\_8c.js.

## 30.961 doc/html/s1238\_8c.js File Reference

### Variables

- var [s1238\\_8c](#)

### 30.961.1 Variable Documentation

#### 30.961.1.1 var s1238\_8c

**Initial value:**

```
=
[
 ["s1238", "s1238_8c.html#a39df6fe62fd3385b8c7d7cf0a41e8f68", null],
 ["s1238", "s1238_8c.html#af479669cbd3b3f5b6704c9f2052668ed", null]
]
```

Definition at line 1 of file s1238\_8c.js.

## 30.962 doc/html/s1239\_8c.js File Reference

### Variables

- var [s1239\\_8c](#)

### 30.962.1 Variable Documentation

#### 30.962.1.1 var s1239\_8c

**Initial value:**

```
=
[
 ["s1239", "s1239_8c.html#a61bcd3745ed140f9b036175a1033435b", null],
 ["s1239", "s1239_8c.html#a57b5d3c1718ae8ecb51f5d4ddf095862", null]
]
```

Definition at line 1 of file s1239\_8c.js.

## 30.963 doc/html/s1240\_8c.js File Reference

### Variables

- var [s1240\\_8c](#)

### 30.963.1 Variable Documentation

#### 30.963.1.1 var s1240\_8c

##### Initial value:

```
=
[
 ["s1240", "s1240_8c.html#a0971794330c111b76321c140e59b70a3", null],
 ["s1240", "s1240_8c.html#a8533b0342d8758533728a812bc7dab94", null]
]
```

Definition at line 1 of file s1240\_8c.js.

## 30.964 doc/html/s1241\_8c.js File Reference

### Variables

- var [s1241\\_8c](#)

### 30.964.1 Variable Documentation

#### 30.964.1.1 var s1241\_8c

##### Initial value:

```
=
[
 ["s1241", "s1241_8c.html#a4a8781d94e744a9829374d742bdf66f4", null],
 ["s1241", "s1241_8c.html#af9417a5b1b1fa9dcd4a1783856c23ec3", null]
]
```

Definition at line 1 of file s1241\_8c.js.

## 30.965 doc/html/s1243\_8c.js File Reference

### Variables

- var [s1243\\_8c](#)

### 30.965.1 Variable Documentation

#### 30.965.1.1 var s1243\_8c

**Initial value:**

```
=
[
 ["s1243", "s1243_8c.html#ac5dc9fa40ebdff3add1ac4f522be390a", null],
 ["s1243", "s1243_8c.html#a1ace4d974a3f96d73e85bb76248503fb", null]
]
```

Definition at line 1 of file s1243\_8c.js.

## 30.966 doc/html/s1244\_8c.js File Reference

### Variables

- var [s1244\\_8c](#)

### 30.966.1 Variable Documentation

#### 30.966.1.1 var s1244\_8c

**Initial value:**

```
=
[
 ["s1244", "s1244_8c.html#af14104d9a52102a29bdb5a23d9d8b27e", null],
 ["s1244", "s1244_8c.html#ac8fdb8e67a138a4882e76025b24ebbd8", null]
]
```

Definition at line 1 of file s1244\_8c.js.

## 30.967 doc/html/s1245\_8c.js File Reference

### Variables

- var [s1245\\_8c](#)

### 30.967.1 Variable Documentation

#### 30.967.1.1 var s1245\_8c

**Initial value:**

```
=
[
 ["s1245", "s1245_8c.html#a1dba5d931d965b7106f4f737881d6f6d", null],
 ["s1245", "s1245_8c.html#ad54b9d68117a6e670b1daba3e1f29a5c", null]
]
```

Definition at line 1 of file s1245\_8c.js.

## 30.968 doc/html/s1251\_8c.js File Reference

### Variables

- var [s1251\\_8c](#)

### 30.968.1 Variable Documentation

#### 30.968.1.1 var s1251\_8c

##### Initial value:

```
=
[
 ["s1251", "s1251_8c.html#a4f87e50a7e57af5dd9ed6371bc37f778", null],
 ["s1251", "s1251_8c.html#a95695088f1d3f5de4fde290ce14edb9b", null]
]
```

Definition at line 1 of file s1251\_8c.js.

## 30.969 doc/html/s1252\_8c.js File Reference

### Variables

- var [s1252\\_8c](#)

### 30.969.1 Variable Documentation

#### 30.969.1.1 var s1252\_8c

##### Initial value:

```
=
[
 ["s1252", "s1252_8c.html#a4a658f051bab6ff33e018eff459dd4e5", null],
 ["s1252", "s1252_8c.html#a4150f0a2cb6340c3ef266a5899bde984", null]
]
```

Definition at line 1 of file s1252\_8c.js.

## 30.970 doc/html/s1301\_8c.js File Reference

### Variables

- var [s1301\\_8c](#)

### 30.970.1 Variable Documentation

#### 30.970.1.1 var s1301\_8c

**Initial value:**

```
=
[
 ["s1301", "s1301_8c.html#a87664c4ab275003faddf52da4b19be0c", null],
 ["s1301", "s1301_8c.html#a53e991bcbe9eb302e837d1ac46213c35", null]
]
```

Definition at line 1 of file s1301\_8c.js.

## 30.971 doc/html/s1302\_8c.js File Reference

### Variables

- var [s1302\\_8c](#)

### 30.971.1 Variable Documentation

#### 30.971.1.1 var s1302\_8c

**Initial value:**

```
=
[
 ["s1302", "s1302_8c.html#a9d9f159ab859926dad2112a952b8b6cd", null],
 ["s1302", "s1302_8c.html#a09aeadc91df21fe24abbe76da2abdd9", null]
]
```

Definition at line 1 of file s1302\_8c.js.

## 30.972 doc/html/s1303\_8c.js File Reference

### Variables

- var [s1303\\_8c](#)

### 30.972.1 Variable Documentation

#### 30.972.1.1 var s1303\_8c

**Initial value:**

```
=
[
 ["s1303", "s1303_8c.html#ab98372d457f4093a1a00049ff30968ac", null],
 ["s1303", "s1303_8c.html#ab095f44eb1cbbd4b0182b746799b71c9", null]
]
```

Definition at line 1 of file s1303\_8c.js.

## 30.973 doc/html/s1304\_8c.js File Reference

### Variables

- var [s1304\\_8c](#)

### 30.973.1 Variable Documentation

#### 30.973.1.1 var s1304\_8c

##### Initial value:

```
=
[
 ["s1304", "s1304_8c.html#a04dcd6d004b9864307ed1341b3afef7e", null],
 ["s1304", "s1304_8c.html#a9072f2da05219f3fd0c4cc6f969ade81", null]
]
```

Definition at line 1 of file s1304\_8c.js.

## 30.974 doc/html/s1305\_8c.js File Reference

### Variables

- var [s1305\\_8c](#)

### 30.974.1 Variable Documentation

#### 30.974.1.1 var s1305\_8c

##### Initial value:

```
=
[
 ["s1305", "s1305_8c.html#adc0ad0abd85cfe7e0a7a9c5dfe26f4db", null],
 ["s1305", "s1305_8c.html#a8f1e1f9e1b0605a58187c77e96c6da63", null]
]
```

Definition at line 1 of file s1305\_8c.js.

## 30.975 doc/html/s1306\_8c.js File Reference

### Variables

- var [s1306\\_8c](#)

### 30.975.1 Variable Documentation

#### 30.975.1.1 var s1306\_8c

**Initial value:**

```
=
[
 ["s1306", "s1306_8c.html#a4ac3aaf0daf9041de15f0b6eb0f187", null],
 ["s1306", "s1306_8c.html#a2f55a277175f71fbb649d5fcba8c1314", null]
]
```

Definition at line 1 of file s1306\_8c.js.

## 30.976 doc/html/s1307\_8c.js File Reference

### Variables

- var [s1307\\_8c](#)

### 30.976.1 Variable Documentation

#### 30.976.1.1 var s1307\_8c

**Initial value:**

```
=
[
 ["s1307", "s1307_8c.html#a6a91db92a736628bc5caba4593000c6f", null],
 ["s1307", "s1307_8c.html#a42661bbffbd78086c9eee3737f0193cb", null]
]
```

Definition at line 1 of file s1307\_8c.js.

## 30.977 doc/html/s1308\_8c.js File Reference

### Variables

- var [s1308\\_8c](#)

### 30.977.1 Variable Documentation

#### 30.977.1.1 var s1308\_8c

**Initial value:**

```
=
[
 ["s1308", "s1308_8c.html#a0d8f2e6ff587aede5779e675e8fb34aa", null],
 ["s1308", "s1308_8c.html#a6cb6311342a40f856ca9f54d73eae953", null]
]
```

Definition at line 1 of file s1308\_8c.js.

## 30.978 doc/html/s1309\_8c.js File Reference

### Variables

- var [s1309\\_8c](#)

### 30.978.1 Variable Documentation

#### 30.978.1.1 var s1309\_8c

##### Initial value:

```
=
[
 ["s1309", "s1309_8c.html#a885cc4c255c051fcb8c39a59b605d516", null],
 ["s1309", "s1309_8c.html#a8a13ea00a9578bb0a141d9c7329212a4", null]
]
```

Definition at line 1 of file s1309\_8c.js.

## 30.979 doc/html/s1310\_8c.js File Reference

### Variables

- var [s1310\\_8c](#)

### 30.979.1 Variable Documentation

#### 30.979.1.1 var s1310\_8c

##### Initial value:

```
=
[
 ["s1310", "s1310_8c.html#a66176bb507ea977fb1c75fa658f97176", null],
 ["s1310", "s1310_8c.html#aa1ac9d7cc59026d2125f2feb608704b0", null]
]
```

Definition at line 1 of file s1310\_8c.js.

## 30.980 doc/html/s1311\_8c.js File Reference

### Variables

- var [s1311\\_8c](#)



## 30.980.1 Variable Documentation

### 30.980.1.1 var s1311\_8c

#### Initial value:

```
=
[
 ["s1311", "s1311_8c.html#a9e92923a88642f0abccff8b8ae036f78a", null],
 ["s1311", "s1311_8c.html#a5b85914b02c70d710b06912207ed284e", null]
]
```

Definition at line 1 of file s1311\_8c.js.

## 30.981 doc/html/s1312\_8c.js File Reference

### Variables

- var [s1312\\_8c](#)

## 30.981.1 Variable Documentation

### 30.981.1.1 var s1312\_8c

#### Initial value:

```
=
[
 ["s1312", "s1312_8c.html#a655e10996b8fc3d9ebe2cc1f975ed8fb", null],
 ["s1312", "s1312_8c.html#abf550a849865b9c274ada136ae344ca0", null]
]
```

Definition at line 1 of file s1312\_8c.js.

## 30.982 doc/html/s1313\_8c.js File Reference

### Variables

- var [s1313\\_8c](#)

## 30.982.1 Variable Documentation

### 30.982.1.1 var s1313\_8c

#### Initial value:

```
=
[
 ["s1313", "s1313_8c.html#aed8010dcef179e760efeac56226813c3", null],
 ["s1313", "s1313_8c.html#a05fa7bb93e5285c716c05297a6212829", null]
]
```

Definition at line 1 of file s1313\_8c.js.

## 30.983 doc/html/s1314\_8c.js File Reference

### Variables

- var [s1314\\_8c](#)

### 30.983.1 Variable Documentation

#### 30.983.1.1 var s1314\_8c

##### Initial value:

```
=
[
 ["s1314", "s1314_8c.html#afbf43a189d5b7f43d4c17c5fadcf35e8", null],
 ["s1314", "s1314_8c.html#a33b2926e4ada0e5bd11039dad0b00b16", null]
]
```

Definition at line 1 of file s1314\_8c.js.

## 30.984 doc/html/s1315\_8c.js File Reference

### Variables

- var [s1315\\_8c](#)

### 30.984.1 Variable Documentation

#### 30.984.1.1 var s1315\_8c

##### Initial value:

```
=
[
 ["s1315", "s1315_8c.html#ad0d68d829cb23aeadcff787f6acc2c70", null],
 ["s1315", "s1315_8c.html#a478596052c3b1bcd110b3a7444725602", null]
]
```

Definition at line 1 of file s1315\_8c.js.

## 30.985 doc/html/s1316\_8c.js File Reference

### Variables

- var [s1316\\_8c](#)

### 30.985.1 Variable Documentation

#### 30.985.1.1 var s1316\_8c

**Initial value:**

```
=
[
 ["s1316", "s1316_8c.html#a8426c1ccdc6e04844f4354f57ca07187", null],
 ["s1316", "s1316_8c.html#a161cd04bd67e22cb56002a28c07497c3", null]
]
```

Definition at line 1 of file s1316\_8c.js.

## 30.986 doc/html/s1317\_8c.js File Reference

### Variables

- var [s1317\\_8c](#)

### 30.986.1 Variable Documentation

#### 30.986.1.1 var s1317\_8c

**Initial value:**

```
=
[
 ["s1317", "s1317_8c.html#acbe916c08812c9332a5f1fd86ff77c1e", null],
 ["s1317", "s1317_8c.html#a5e5c86b111e0200098efd7030696ac54", null]
]
```

Definition at line 1 of file s1317\_8c.js.

## 30.987 doc/html/s1318\_8c.js File Reference

### Variables

- var [s1318\\_8c](#)

### 30.987.1 Variable Documentation

#### 30.987.1.1 var s1318\_8c

**Initial value:**

```
=
[
 ["s1318", "s1318_8c.html#a1aa2d44d42ecc555f813f719e7ed0b82", null],
 ["s1318", "s1318_8c.html#ac742fee572c83d9aaa24ea7cb01cd9e5", null]
]
```

Definition at line 1 of file s1318\_8c.js.

## 30.988 doc/html/s1319\_8c.js File Reference

### Variables

- var [s1319\\_8c](#)

### 30.988.1 Variable Documentation

#### 30.988.1.1 var s1319\_8c

##### Initial value:

```
=
[
 ["s1319", "s1319_8c.html#aa4aafb12fc1d4099f478a5f6ea53f0f6", null],
 ["s1319", "s1319_8c.html#acbd0469d864acc9844b3c26abadd5e0", null]
]
```

Definition at line 1 of file s1319\_8c.js.

## 30.989 doc/html/s1320\_8c.js File Reference

### Variables

- var [s1320\\_8c](#)

### 30.989.1 Variable Documentation

#### 30.989.1.1 var s1320\_8c

##### Initial value:

```
=
[
 ["s1320", "s1320_8c.html#a0d50afc02292ea6110f40dc0730641ef", null],
 ["s1320", "s1320_8c.html#aabeeb642be67520195f27aed9eee6878", null]
]
```

Definition at line 1 of file s1320\_8c.js.

## 30.990 doc/html/s1321\_8c.js File Reference

### Variables

- var [s1321\\_8c](#)

### 30.990.1 Variable Documentation

#### 30.990.1.1 var s1321\_8c

**Initial value:**

```
=
[
 ["s1321", "s1321_8c.html#af3e6bd48bd83df0e604ff195ae6350e9", null],
 ["s1321", "s1321_8c.html#abc423ced6c7a58871452be43e7503f33", null]
]
```

Definition at line 1 of file s1321\_8c.js.

## 30.991 doc/html/s1322\_8c.js File Reference

### Variables

- var [s1322\\_8c](#)

### 30.991.1 Variable Documentation

#### 30.991.1.1 var s1322\_8c

**Initial value:**

```
=
[
 ["s1322", "s1322_8c.html#a2e1e6c409b0b2a8d76bb05ae0b6b23cf", null],
 ["s1322", "s1322_8c.html#ad263986de488331eb092675ae103fe50", null]
]
```

Definition at line 1 of file s1322\_8c.js.

## 30.992 doc/html/s1323\_8c.js File Reference

### Variables

- var [s1323\\_8c](#)

### 30.992.1 Variable Documentation

#### 30.992.1.1 var s1323\_8c

**Initial value:**

```
=
[
 ["s1323", "s1323_8c.html#a4c79b67625296717ee27eb5e6227abb1", null],
 ["s1323", "s1323_8c.html#aa0e01c7cc57461a140df1ae95958bd1a", null]
]
```

Definition at line 1 of file s1323\_8c.js.

## 30.993 doc/html/s1324\_8c.js File Reference

### Variables

- var [s1324\\_8c](#)

### 30.993.1 Variable Documentation

#### 30.993.1.1 var s1324\_8c

##### Initial value:

```
=
[
 ["s1324", "s1324_8c.html#a4c8d4e636dd1cd805c3421054787246f", null],
 ["s1324", "s1324_8c.html#ad6177f844ae19bf4cea1336d62418c1e", null]
]
```

Definition at line 1 of file s1324\_8c.js.

## 30.994 doc/html/s1325\_8c.js File Reference

### Variables

- var [s1325\\_8c](#)

### 30.994.1 Variable Documentation

#### 30.994.1.1 var s1325\_8c

##### Initial value:

```
=
[
 ["s1325", "s1325_8c.html#a67ee413bed57ae251ee2c3b878a60396", null],
 ["s1325", "s1325_8c.html#a99bf06beb880cfb2682cec203197fa9e", null]
]
```

Definition at line 1 of file s1325\_8c.js.

## 30.995 doc/html/s1327\_8c.js File Reference

### Variables

- var [s1327\\_8c](#)

### 30.995.1 Variable Documentation

#### 30.995.1.1 var s1327\_8c

**Initial value:**

```
=
[
 ["s1327", "s1327_8c.html#a72b0aeb9475fd2ac4ffb0b40c3149d3e", null],
 ["s1327", "s1327_8c.html#ab70f32af5a77f85182f6452e497d2465", null]
]
```

Definition at line 1 of file s1327\_8c.js.

## 30.996 doc/html/s1328\_8c.js File Reference

### Variables

- var [s1328\\_8c](#)

### 30.996.1 Variable Documentation

#### 30.996.1.1 var s1328\_8c

**Initial value:**

```
=
[
 ["s1328", "s1328_8c.html#ade58f92e451c261fbc2ee684affd5a61", null],
 ["s1328", "s1328_8c.html#a590cbe13f313c9b1e72fe2f1203fe4d1", null]
]
```

Definition at line 1 of file s1328\_8c.js.

## 30.997 doc/html/s1329\_8c.js File Reference

### Variables

- var [s1329\\_8c](#)

### 30.997.1 Variable Documentation

#### 30.997.1.1 var s1329\_8c

**Initial value:**

```
=
[
 ["s1329", "s1329_8c.html#ab0d54ae747a497889d937dc352bd6744", null],
 ["s1329", "s1329_8c.html#a2a13f555b330c163192a01a870aed06a", null]
]
```

Definition at line 1 of file s1329\_8c.js.

## 30.998 doc/html/s1330\_8c.js File Reference

### Variables

- var [s1330\\_8c](#)

### 30.998.1 Variable Documentation

#### 30.998.1.1 var s1330\_8c

##### Initial value:

```
=
[
 ["s1330", "s1330_8c.html#a7e167cdade1254b94a0d3823f9d5a66c", null],
 ["s1330", "s1330_8c.html#a1fae0075f1afac55aad1c49989015e9a", null]
]
```

Definition at line 1 of file s1330\_8c.js.

## 30.999 doc/html/s1331\_8c.js File Reference

### Variables

- var [s1331\\_8c](#)

### 30.999.1 Variable Documentation

#### 30.999.1.1 var s1331\_8c

##### Initial value:

```
=
[
 ["s1331", "s1331_8c.html#ab307ba5ca24ab4f531485754617d67af", null],
 ["s1331", "s1331_8c.html#a00724236770bc18f6bb3f74b84815f88", null]
]
```

Definition at line 1 of file s1331\_8c.js.

## 30.1000 doc/html/s1332\_8c.js File Reference

### Variables

- var [s1332\\_8c](#)



### 30.1000.1 Variable Documentation

#### 30.1000.1.1 var s1332\_8c

**Initial value:**

```
=
[
 ["s1332", "s1332_8c.html#ad7e16210f93feaaafb84da623e172ba6", null],
 ["s1332", "s1332_8c.html#ac8e33e2d2c107bfbbe0745ac1d88d5c0", null]
]
```

Definition at line 1 of file s1332\_8c.js.

## 30.1001 doc/html/s1333\_8c.js File Reference

### Variables

- var [s1333\\_8c](#)

### 30.1001.1 Variable Documentation

#### 30.1001.1.1 var s1333\_8c

**Initial value:**

```
=
[
 ["s1333", "s1333_8c.html#acb2745d58a1db1daa188c1aca382561c", null],
 ["s1333", "s1333_8c.html#a09094b83efe45bf371981a53f8a1e0e4", null]
]
```

Definition at line 1 of file s1333\_8c.js.

## 30.1002 doc/html/s1333count\_8c.js File Reference

### Variables

- var [s1333count\\_8c](#)

### 30.1002.1 Variable Documentation

#### 30.1002.1.1 var s1333count\_8c

**Initial value:**

```
=
[
 ["s1333_COUNT", "s1333count_8c.html#ab94d9a438ee6c843fc55791b07454488", null],
 ["s1333_count", "s1333count_8c.html#ae0243effd266804446667fd437e94057", null]
]
```

Definition at line 1 of file s1333count\_8c.js.

## 30.1003 doc/html/s1333cycli\_8c.js File Reference

### Variables

- var [s1333cycli\\_8c](#)

### 30.1003.1 Variable Documentation

#### 30.1003.1.1 var s1333cycli\_8c

##### Initial value:

```
=
[
 ["s1333_CYCLIC", "s1333cycli_8c.html#aa4c035acfd165d3f513dab0223260b25", null],
 ["s1333_cyclic", "s1333cycli_8c.html#aefaeba47a818bc9b44884a9ec82e6c0c", null]
]
```

Definition at line 1 of file s1333cycli\_8c.js.

## 30.1004 doc/html/s1334\_8c.js File Reference

### Variables

- var [s1334\\_8c](#)

### 30.1004.1 Variable Documentation

#### 30.1004.1.1 var s1334\_8c

##### Initial value:

```
=
[
 ["s1334", "s1334_8c.html#a307f364fe6dc547ea2793b91096a1105", null],
 ["s1334", "s1334_8c.html#ad41dd1d9bcfe75753e66b14918f01539", null]
]
```

Definition at line 1 of file s1334\_8c.js.

## 30.1005 doc/html/s1340\_8c.js File Reference

### Variables

- var [s1340\\_8c](#)

### 30.1005.1 Variable Documentation

#### 30.1005.1.1 var s1340\_8c

**Initial value:**

```
=
[
 ["s1340", "s1340_8c.html#ac59747c754c2ded945a7c3b7f5d0f7fb", null],
 ["s1340", "s1340_8c.html#af1459d0de194b29c4fc669aa56385637", null]
]
```

Definition at line 1 of file s1340\_8c.js.

## 30.1006 doc/html/s1341\_8c.js File Reference

### Variables

- var [s1341\\_8c](#)

### 30.1006.1 Variable Documentation

#### 30.1006.1.1 var s1341\_8c

**Initial value:**

```
=
[
 ["s1341", "s1341_8c.html#ac3b880ff5beac915a167507d9b8b5f76", null],
 ["s1341", "s1341_8c.html#a741deaab564540cf800100a6c1c47c3a", null]
]
```

Definition at line 1 of file s1341\_8c.js.

## 30.1007 doc/html/s1342\_8c.js File Reference

### Variables

- var [s1342\\_8c](#)

### 30.1007.1 Variable Documentation

#### 30.1007.1.1 var s1342\_8c

**Initial value:**

```
=
[
 ["s1342", "s1342_8c.html#a63f3de47618467cc1f3b26d356b2f356", null],
 ["s1342", "s1342_8c.html#af20707cee7c04d498088f1631036b1e6", null]
]
```

Definition at line 1 of file s1342\_8c.js.

## 30.1008 doc/html/s1343\_8c.js File Reference

### Variables

- var [s1343\\_8c](#)

### 30.1008.1 Variable Documentation

#### 30.1008.1.1 var s1343\_8c

##### Initial value:

```
=
[
 ["s1343", "s1343_8c.html#a5b5e3dbbf447d87facfddfe208dba3a2", null],
 ["s1343", "s1343_8c.html#ad070d9110d9e4120a3307f4ea6f87904", null]
]
```

Definition at line 1 of file s1343\_8c.js.

## 30.1009 doc/html/s1345\_8c.js File Reference

### Variables

- var [s1345\\_8c](#)

### 30.1009.1 Variable Documentation

#### 30.1009.1.1 var s1345\_8c

##### Initial value:

```
=
[
 ["s1345", "s1345_8c.html#a193a55148d292106b60cd669a7094c6d", null],
 ["s1345", "s1345_8c.html#aca7a1c3bda7399de78ed1f68af109eef", null]
]
```

Definition at line 1 of file s1345\_8c.js.

## 30.1010 doc/html/s1346\_8c.js File Reference

### Variables

- var [s1346\\_8c](#)

### 30.1010.1 Variable Documentation

#### 30.1010.1.1 var s1346\_8c

**Initial value:**

```
=
[
 ["s1346", "s1346_8c.html#a660a4ae237160c28e84f937afa893297", null],
 ["s1346", "s1346_8c.html#ad634967ff422a1f5aa90a6437dd18b10", null]
]
```

Definition at line 1 of file s1346\_8c.js.

## 30.1011 doc/html/s1347\_8c.js File Reference

### Variables

- var [s1347\\_8c](#)

### 30.1011.1 Variable Documentation

#### 30.1011.1.1 var s1347\_8c

**Initial value:**

```
=
[
 ["s1347", "s1347_8c.html#abdddbdf8b3d65acaaa83bd280797ea36e", null],
 ["s1347", "s1347_8c.html#a2cb26213ffe7003af077b195d9054ebc", null]
]
```

Definition at line 1 of file s1347\_8c.js.

## 30.1012 doc/html/s1348\_8c.js File Reference

### Variables

- var [s1348\\_8c](#)

### 30.1012.1 Variable Documentation

#### 30.1012.1.1 var s1348\_8c

**Initial value:**

```
=
[
 ["s1348", "s1348_8c.html#a985fc5c863c4c8f032393347605dfe07", null],
 ["s1348", "s1348_8c.html#a0ad0beb48f57f5de5ece8de2fc59cb89", null]
]
```

Definition at line 1 of file s1348\_8c.js.

## 30.1013 doc/html/s1349\_8c.js File Reference

### Variables

- var [s1349\\_8c](#)

### 30.1013.1 Variable Documentation

#### 30.1013.1.1 var s1349\_8c

##### Initial value:

```
=
[
 ["s1349", "s1349_8c.html#a2a2a0da550da46571545631505f85ab5", null],
 ["s1349", "s1349_8c.html#a2d8ef743d9282816f9b051a7553e3a4a", null]
]
```

Definition at line 1 of file s1349\_8c.js.

## 30.1014 doc/html/s1350\_8c.js File Reference

### Variables

- var [s1350\\_8c](#)

### 30.1014.1 Variable Documentation

#### 30.1014.1.1 var s1350\_8c

##### Initial value:

```
=
[
 ["s1350", "s1350_8c.html#a768c827ac3ac13ae07d6b6f9f39ab418", null],
 ["s1350", "s1350_8c.html#a69772baebcf7fab66d959aa216d5b96e", null]
]
```

Definition at line 1 of file s1350\_8c.js.

## 30.1015 doc/html/s1351\_8c.js File Reference

### Variables

- var [s1351\\_8c](#)

### 30.1015.1 Variable Documentation

#### 30.1015.1.1 var s1351\_8c

**Initial value:**

```
=
[
 ["s1351", "s1351_8c.html#a071aa51592ac3e321e06e6d52ae36a56", null],
 ["s1351", "s1351_8c.html#affa858eea360871fba19e48ab23621d3", null]
]
```

Definition at line 1 of file s1351\_8c.js.

## 30.1016 doc/html/s1352\_8c.js File Reference

### Variables

- var [s1352\\_8c](#)

### 30.1016.1 Variable Documentation

#### 30.1016.1.1 var s1352\_8c

**Initial value:**

```
=
[
 ["s1352", "s1352_8c.html#aace167c5bca01666b56379f8e52cf7fd", null],
 ["s1352", "s1352_8c.html#a1f28435cc74c5e4a5d4a61085968bb18", null]
]
```

Definition at line 1 of file s1352\_8c.js.

## 30.1017 doc/html/s1353\_8c.js File Reference

### Variables

- var [s1353\\_8c](#)

### 30.1017.1 Variable Documentation

#### 30.1017.1.1 var s1353\_8c

**Initial value:**

```
=
[
 ["s1353", "s1353_8c.html#ab0d54c8c594a05e74562b723a54afce7", null],
 ["s1353", "s1353_8c.html#a31c4aef7b440763d22258dae47bc52ea", null]
]
```

Definition at line 1 of file s1353\_8c.js.

## 30.1018 doc/html/s1354\_8c.js File Reference

### Variables

- var [s1354\\_8c](#)

### 30.1018.1 Variable Documentation

#### 30.1018.1.1 var s1354\_8c

##### Initial value:

```
=
[
 ["s1354", "s1354_8c.html#a25b3231763fd33d5432eccbb7b39595f", null],
 ["s1354", "s1354_8c.html#a707af475e779ee837238cdf8960813b0", null]
]
```

Definition at line 1 of file s1354\_8c.js.

## 30.1019 doc/html/s1355\_8c.js File Reference

### Variables

- var [s1355\\_8c](#)

### 30.1019.1 Variable Documentation

#### 30.1019.1.1 var s1355\_8c

##### Initial value:

```
=
[
 ["s1355", "s1355_8c.html#a98f387a10bdc941cc4ee1949d1b70b2d", null],
 ["s1355", "s1355_8c.html#ab8c94157c0227992970fc1cae84f1312", null]
]
```

Definition at line 1 of file s1355\_8c.js.

## 30.1020 doc/html/s1356\_8c.js File Reference

### Variables

- var [s1356\\_8c](#)



### 30.1020.1 Variable Documentation

#### 30.1020.1.1 var s1356\_8c

**Initial value:**

```
=
[
 ["s1356", "s1356_8c.html#a88953576d012723c97c5b8373bb6211d", null],
 ["s1356", "s1356_8c.html#a175a32cb1d645c160faa4d3d92a355ce", null]
]
```

Definition at line 1 of file s1356\_8c.js.

## 30.1021 doc/html/s1357\_8c.js File Reference

### Variables

- var [s1357\\_8c](#)

### 30.1021.1 Variable Documentation

#### 30.1021.1.1 var s1357\_8c

**Initial value:**

```
=
[
 ["s1357", "s1357_8c.html#a344246c9d63e828b5d00dd9ba44e1bbd", null],
 ["s1357", "s1357_8c.html#aa1d0940fdb2942d0cca2104da77a759b", null]
]
```

Definition at line 1 of file s1357\_8c.js.

## 30.1022 doc/html/s1358\_8c.js File Reference

### Variables

- var [s1358\\_8c](#)

### 30.1022.1 Variable Documentation

#### 30.1022.1.1 var s1358\_8c

**Initial value:**

```
=
[
 ["s1358", "s1358_8c.html#a648a2c60b1e464e248a02c74aae55ed3", null],
 ["s1358", "s1358_8c.html#aa7ab0c3daaf200bcc57dacb001213bfc", null]
]
```

Definition at line 1 of file s1358\_8c.js.

## 30.1023 doc/html/s1359\_8c.js File Reference

### Variables

- var [s1359\\_8c](#)

### 30.1023.1 Variable Documentation

#### 30.1023.1.1 var s1359\_8c

##### Initial value:

```
=
[
 ["s1359", "s1359_8c.html#aab086b4078a68bf044e51878b7af9a72", null],
 ["s1359", "s1359_8c.html#a4847037081587fbf7c42899bc9112e03", null]
]
```

Definition at line 1 of file s1359\_8c.js.

## 30.1024 doc/html/s1360\_8c.js File Reference

### Variables

- var [s1360\\_8c](#)

### 30.1024.1 Variable Documentation

#### 30.1024.1.1 var s1360\_8c

##### Initial value:

```
=
[
 ["s1360", "s1360_8c.html#a3697612bca77b715fc76fbef535c0e6a", null],
 ["s1360", "s1360_8c.html#adfe3c417084afbb277f4d239a8c24d80", null]
]
```

Definition at line 1 of file s1360\_8c.js.

## 30.1025 doc/html/s1361\_8c.js File Reference

### Variables

- var [s1361\\_8c](#)

### 30.1025.1 Variable Documentation

#### 30.1025.1.1 var s1361\_8c

**Initial value:**

```
=
[
 ["s1361", "s1361_8c.html#ab4c1204945bc84b95d6b3918b3e78b14", null],
 ["s1361", "s1361_8c.html#a83fba16e124ba3c7c23d41a560afaabf", null]
]
```

Definition at line 1 of file s1361\_8c.js.

## 30.1026 doc/html/s1362\_8c.js File Reference

### Variables

- var [s1362\\_8c](#)

### 30.1026.1 Variable Documentation

#### 30.1026.1.1 var s1362\_8c

**Initial value:**

```
=
[
 ["s1362", "s1362_8c.html#af6b028df1316ed63742b12ec8430b347", null],
 ["s1362", "s1362_8c.html#acc973e63174affe23c51f48b892b08c1", null]
]
```

Definition at line 1 of file s1362\_8c.js.

## 30.1027 doc/html/s1363\_8c.js File Reference

### Variables

- var [s1363\\_8c](#)

### 30.1027.1 Variable Documentation

#### 30.1027.1.1 var s1363\_8c

**Initial value:**

```
=
[
 ["s1363", "s1363_8c.html#a0bd8483141a72072291e4cba70edf1ad", null],
 ["s1363", "s1363_8c.html#ac2be086fc1de20dc8a00d7e6f1b428e1", null]
]
```

Definition at line 1 of file s1363\_8c.js.

## 30.1028 doc/html/s1364\_8c.js File Reference

### Variables

- var [s1364\\_8c](#)

### 30.1028.1 Variable Documentation

#### 30.1028.1.1 var s1364\_8c

##### Initial value:

```
=
[
 ["s1364", "s1364_8c.html#a7dd392b4c514f869575e7d07eb39715a", null],
 ["s1364", "s1364_8c.html#ab3f1874bb3643ea37433a63aee0d01fa", null]
]
```

Definition at line 1 of file s1364\_8c.js.

## 30.1029 doc/html/s1365\_8c.js File Reference

### Variables

- var [s1365\\_8c](#)

### 30.1029.1 Variable Documentation

#### 30.1029.1.1 var s1365\_8c

##### Initial value:

```
=
[
 ["s1365", "s1365_8c.html#ab8b1b2eca2af9dd9dca62cd6cc02da7e", null],
 ["s1365", "s1365_8c.html#afad4ba49070fbff41b1fa8203bc13b4c", null]
]
```

Definition at line 1 of file s1365\_8c.js.

## 30.1030 doc/html/s1366\_8c.js File Reference

### Variables

- var [s1366\\_8c](#)

### 30.1030.1 Variable Documentation

#### 30.1030.1.1 var s1366\_8c

**Initial value:**

```
=
[
 ["s1366", "s1366_8c.html#a2e1ce87fe9d084069151fa8995aa7fac", null],
 ["s1366", "s1366_8c.html#a766ef57616413b5ed07c6d6fcdbe61f", null]
]
```

Definition at line 1 of file s1366\_8c.js.

## 30.1031 doc/html/s1367\_8c.js File Reference

### Variables

- var [s1367\\_8c](#)

### 30.1031.1 Variable Documentation

#### 30.1031.1.1 var s1367\_8c

**Initial value:**

```
=
[
 ["s1366", "s1367_8c.html#a2e1ce87fe9d084069151fa8995aa7fac", null],
 ["s1367", "s1367_8c.html#aaff59efbce3b52296d98653bd0648f19", null]
]
```

Definition at line 1 of file s1367\_8c.js.

## 30.1032 doc/html/s1369\_8c.js File Reference

### Variables

- var [s1369\\_8c](#)

### 30.1032.1 Variable Documentation

#### 30.1032.1.1 var s1369\_8c

**Initial value:**

```
=
[
 ["s1369", "s1369_8c.html#a5452d9beb150cfb7ffac2c6aeebfff36a", null],
 ["s1369", "s1369_8c.html#a243118f384fd093632a8f78a8a30b853", null]
]
```

Definition at line 1 of file s1369\_8c.js.

## 30.1033 doc/html/s1370\_8c.js File Reference

### Variables

- var [s1370\\_8c](#)

### 30.1033.1 Variable Documentation

#### 30.1033.1.1 var s1370\_8c

##### Initial value:

```
=
[
 ["s1370", "s1370_8c.html#a93b454b31e1de3f73de7e2117f6330aa", null],
 ["s1370", "s1370_8c.html#a746010f9f90a1608bfff1e1efe059f79", null]
]
```

Definition at line 1 of file s1370\_8c.js.

## 30.1034 doc/html/s1371\_8c.js File Reference

### Variables

- var [s1371\\_8c](#)

### 30.1034.1 Variable Documentation

#### 30.1034.1.1 var s1371\_8c

##### Initial value:

```
=
[
 ["s1371", "s1371_8c.html#a3f1b0494f9be05bfff28bbfbdc1eb9bb6", null],
 ["s1371", "s1371_8c.html#a5207fbd2ceb4f2d9f2824bf7a2cffd59", null]
]
```

Definition at line 1 of file s1371\_8c.js.

## 30.1035 doc/html/s1372\_8c.js File Reference

### Variables

- var [s1372\\_8c](#)

### 30.1035.1 Variable Documentation

#### 30.1035.1.1 var s1372\_8c

**Initial value:**

```
=
[
 ["s1372", "s1372_8c.html#a46a6b3e1c594558add03d4d3f8c1a20e", null],
 ["s1372", "s1372_8c.html#a84e94b6339f5e48a7157f8d16948797b", null]
]
```

Definition at line 1 of file s1372\_8c.js.

## 30.1036 doc/html/s1373\_8c.js File Reference

### Variables

- var [s1373\\_8c](#)

### 30.1036.1 Variable Documentation

#### 30.1036.1.1 var s1373\_8c

**Initial value:**

```
=
[
 ["s1373", "s1373_8c.html#a44c4888ec49f44093929ee5a482bc5de", null],
 ["s1373", "s1373_8c.html#ab63f9a66fb9094644dac1350f7eaf37c", null]
]
```

Definition at line 1 of file s1373\_8c.js.

## 30.1037 doc/html/s1374\_8c.js File Reference

### Variables

- var [s1374\\_8c](#)

### 30.1037.1 Variable Documentation

#### 30.1037.1.1 var s1374\_8c

**Initial value:**

```
=
[
 ["s1374", "s1374_8c.html#ab1882aa66a39365f05fb774d3572bd8c", null],
 ["s1374", "s1374_8c.html#aec48f0b2ea35a2a24975fe177a3037ca", null]
]
```

Definition at line 1 of file s1374\_8c.js.

## 30.1038 doc/html/s1375\_8c.js File Reference

### Variables

- var [s1375\\_8c](#)

### 30.1038.1 Variable Documentation

#### 30.1038.1.1 var s1375\_8c

##### Initial value:

```
=
[
 ["s1375", "s1375_8c.html#a223e2b9635bf9e8b4d29c81ec2c00d7f", null],
 ["s1375", "s1375_8c.html#a35a8380a308297346065b756767b9c14", null]
]
```

Definition at line 1 of file s1375\_8c.js.

## 30.1039 doc/html/s1376\_8c.js File Reference

### Variables

- var [s1376\\_8c](#)

### 30.1039.1 Variable Documentation

#### 30.1039.1.1 var s1376\_8c

##### Initial value:

```
=
[
 ["s1376", "s1376_8c.html#ae3bc4d271d9113fff3ac1f6d9a7882f9", null],
 ["s1376", "s1376_8c.html#a3dc11d99b2aab5c3999574018d8d771c", null]
]
```

Definition at line 1 of file s1376\_8c.js.

## 30.1040 doc/html/s1377\_8c.js File Reference

### Variables

- var [s1377\\_8c](#)



### 30.1040.1 Variable Documentation

#### 30.1040.1.1 var s1377\_8c

**Initial value:**

```
=
[
 ["s1377", "s1377_8c.html#a71f0bfbdb1f07ffa674d01faf44f4217f", null],
 ["s1377", "s1377_8c.html#af06c868a78283e6cdd0c4b4ccf58ebaf", null]
]
```

Definition at line 1 of file s1377\_8c.js.

## 30.1041 doc/html/s1378\_8c.js File Reference

### Variables

- var [s1378\\_8c](#)

### 30.1041.1 Variable Documentation

#### 30.1041.1.1 var s1378\_8c

**Initial value:**

```
=
[
 ["s1378", "s1378_8c.html#a1e1b11d2170ea4c036aafc01b7ad14d7", null],
 ["s1378", "s1378_8c.html#ae96b81135414a6a3eb145d014a1cd612", null]
]
```

Definition at line 1 of file s1378\_8c.js.

## 30.1042 doc/html/s1379\_8c.js File Reference

### Variables

- var [s1379\\_8c](#)

### 30.1042.1 Variable Documentation

#### 30.1042.1.1 var s1379\_8c

**Initial value:**

```
=
[
 ["s1379", "s1379_8c.html#a5b722eccc232c1c045f396a530f70487", null],
 ["s1379", "s1379_8c.html#aa381bd905be046678b669b0dfcd7213f", null]
]
```

Definition at line 1 of file s1379\_8c.js.

## 30.1043 doc/html/s1380\_8c.js File Reference

### Variables

- var [s1380\\_8c](#)

### 30.1043.1 Variable Documentation

#### 30.1043.1.1 var s1380\_8c

##### Initial value:

```
=
[
 ["s1380", "s1380_8c.html#afa01098127274a02fa967ca8fb0adeb2", null],
 ["s1380", "s1380_8c.html#ada04b92a0f0445889f8fcdda8f911d70", null]
]
```

Definition at line 1 of file s1380\_8c.js.

## 30.1044 doc/html/s1381\_8c.js File Reference

### Variables

- var [s1381\\_8c](#)

### 30.1044.1 Variable Documentation

#### 30.1044.1.1 var s1381\_8c

##### Initial value:

```
=
[
 ["s1381", "s1381_8c.html#a69a8954104ab7b61529ecfa6a224343e", null],
 ["s1381", "s1381_8c.html#ae2aef1b37e8f2a6e0fccaf2f348774f2", null]
]
```

Definition at line 1 of file s1381\_8c.js.

## 30.1045 doc/html/s1382\_8c.js File Reference

### Variables

- var [s1382\\_8c](#)

### 30.1045.1 Variable Documentation

#### 30.1045.1.1 var s1382\_8c

**Initial value:**

```
=
[
 ["s1382", "s1382_8c.html#aaccd5a5a0772c1e0ada51f6de2efb63f", null],
 ["s1382", "s1382_8c.html#a2edccf991b4a3a6ce9e6e66758e981a1", null]
]
```

Definition at line 1 of file s1382\_8c.js.

## 30.1046 doc/html/s1383\_8c.js File Reference

### Variables

- var [s1383\\_8c](#)

### 30.1046.1 Variable Documentation

#### 30.1046.1.1 var s1383\_8c

**Initial value:**

```
=
[
 ["s1383", "s1383_8c.html#aa1a11f8e113a1ec05ae8e9d9c551ab59", null],
 ["s1383", "s1383_8c.html#aed301e19e53a6d96c15d692421c593a7", null]
]
```

Definition at line 1 of file s1383\_8c.js.

## 30.1047 doc/html/s1384\_8c.js File Reference

### Variables

- var [s1384\\_8c](#)

### 30.1047.1 Variable Documentation

#### 30.1047.1.1 var s1384\_8c

**Initial value:**

```
=
[
 ["s1384", "s1384_8c.html#a5ad99e4bf03cba855fc7c5436ef152a9", null],
 ["s1384", "s1384_8c.html#af9d7ea3ce9cf9e5639fbd865af3d1c88", null]
]
```

Definition at line 1 of file s1384\_8c.js.

## 30.1048 doc/html/s1385\_8c.js File Reference

### Variables

- var [s1385\\_8c](#)

### 30.1048.1 Variable Documentation

#### 30.1048.1.1 var s1385\_8c

##### Initial value:

```
=
[
 ["s1385", "s1385_8c.html#ab2274200efdfdbbefe70f59e184d41db", null],
 ["s1385", "s1385_8c.html#a28c33bca6b87afd5f207435a45b50644", null]
]
```

Definition at line 1 of file s1385\_8c.js.

## 30.1049 doc/html/s1386\_8c.js File Reference

### Variables

- var [s1386\\_8c](#)

### 30.1049.1 Variable Documentation

#### 30.1049.1.1 var s1386\_8c

##### Initial value:

```
=
[
 ["s1386", "s1386_8c.html#ad83db55d9a94abc1c817811df1262f2a", null],
 ["s1386", "s1386_8c.html#a4beb6f6bfala38988db8c3a0dd62bcee", null]
]
```

Definition at line 1 of file s1386\_8c.js.

## 30.1050 doc/html/s1387\_8c.js File Reference

### Variables

- var [s1387\\_8c](#)

### 30.1050.1 Variable Documentation

#### 30.1050.1.1 var s1387\_8c

**Initial value:**

```
=
[
 ["s1387", "s1387_8c.html#a423983b8d38292dec54b6fefac1e142a", null],
 ["s1387", "s1387_8c.html#ad7dc082c10ce120f96ce6be869c50cdf", null]
]
```

Definition at line 1 of file s1387\_8c.js.

## 30.1051 doc/html/s1388\_8c.js File Reference

### Variables

- var [s1388\\_8c](#)

### 30.1051.1 Variable Documentation

#### 30.1051.1.1 var s1388\_8c

**Initial value:**

```
=
[
 ["s1388", "s1388_8c.html#ad1d74ab73d285526127f371002770c99", null],
 ["s1388", "s1388_8c.html#a89c207a6bfb91d893d4f85ce5f43739e", null]
]
```

Definition at line 1 of file s1388\_8c.js.

## 30.1052 doc/html/s1389\_8c.js File Reference

### Variables

- var [s1389\\_8c](#)

### 30.1052.1 Variable Documentation

#### 30.1052.1.1 var s1389\_8c

**Initial value:**

```
=
[
 ["s1389", "s1389_8c.html#a3fdf8286addc1a22f10ac7a5b4116c22", null],
 ["s1389", "s1389_8c.html#a3f330364138ddd588856c93822bf404", null]
]
```

Definition at line 1 of file s1389\_8c.js.

## 30.1053 doc/html/s1390\_8c.js File Reference

### Variables

- var [s1390\\_8c](#)

### 30.1053.1 Variable Documentation

#### 30.1053.1.1 var s1390\_8c

##### Initial value:

```
=
[
 ["s1390", "s1390_8c.html#ae6c31d34b1c8e2de697e810e40cda817", null],
 ["s1390", "s1390_8c.html#ad607a45da4118391c128f7de66d18565", null]
]
```

Definition at line 1 of file s1390\_8c.js.

## 30.1054 doc/html/s1391\_8c.js File Reference

### Variables

- var [s1391\\_8c](#)

### 30.1054.1 Variable Documentation

#### 30.1054.1.1 var s1391\_8c

##### Initial value:

```
=
[
 ["s1391", "s1391_8c.html#a22a8061c91f71c94d37fdedf332d2362", null],
 ["fshapeProc", "s1391_8c.html#a23a1402d8877b20ca4dec1197f85940c", null],
 ["s1391", "s1391_8c.html#a46706579d2366ef990f1874f849d894a", null]
]
```

Definition at line 1 of file s1391\_8c.js.

## 30.1055 doc/html/s1393\_8c.js File Reference

### Variables

- var [s1393\\_8c](#)

### 30.1055.1 Variable Documentation

#### 30.1055.1.1 var s1393\_8c

**Initial value:**

```
=
[
 ["s1393", "s1393_8c.html#a30c4519567589040d19eb4e73090e0e8", null],
 ["s1393", "s1393_8c.html#aafa7980f983e89ca6820ae78757a166e4", null]
]
```

Definition at line 1 of file s1393\_8c.js.

## 30.1056 doc/html/s1399\_8c.js File Reference

### Variables

- var [s1399\\_8c](#)

### 30.1056.1 Variable Documentation

#### 30.1056.1.1 var s1399\_8c

**Initial value:**

```
=
[
 ["s1399", "s1399_8c.html#aced37cc843b3553873764360452e9a71", null],
 ["s1399", "s1399_8c.html#a695a27f9339d4297b5848c30bca94b25", null]
]
```

Definition at line 1 of file s1399\_8c.js.

## 30.1057 doc/html/s1401\_8c.js File Reference

### Variables

- var [s1401\\_8c](#)

### 30.1057.1 Variable Documentation

#### 30.1057.1.1 var s1401\_8c

**Initial value:**

```
=
[
 ["s1401", "s1401_8c.html#a267d111f6d47bf39435b819bfe629629", null],
 ["s1401", "s1401_8c.html#aefca4b8131ccfc8cd4461b74837c97ea", null]
]
```

Definition at line 1 of file s1401\_8c.js.

## 30.1058 doc/html/s1421\_8c.js File Reference

### Variables

- var [s1421\\_8c](#)

### 30.1058.1 Variable Documentation

#### 30.1058.1.1 var s1421\_8c

##### Initial value:

```
=
[
 ["s1421", "s1421_8c.html#aff91803aeccc09a16ac8ad217e3fdf8e", null],
 ["s1421", "s1421_8c.html#a1d84bdeee41e7d1e179d8e5af81b01a7", null]
]
```

Definition at line 1 of file s1421\_8c.js.

## 30.1059 doc/html/s1422\_8c.js File Reference

### Variables

- var [s1422\\_8c](#)

### 30.1059.1 Variable Documentation

#### 30.1059.1.1 var s1422\_8c

##### Initial value:

```
=
[
 ["s1422", "s1422_8c.html#af2db7ff709e4b93cd4c4d78270a1317e", null],
 ["s1422", "s1422_8c.html#ae613095392eee8b7af89e94770535e79", null]
]
```

Definition at line 1 of file s1422\_8c.js.

## 30.1060 doc/html/s1424\_8c.js File Reference

### Variables

- var [s1424\\_8c](#)



### 30.1060.1 Variable Documentation

#### 30.1060.1.1 var s1424\_8c

**Initial value:**

```
=
[
 ["s1424", "s1424_8c.html#a23875afacb0f919a894f321157f24a74", null],
 ["s1424", "s1424_8c.html#a2d1687b79d1417f43d242da585b7523e", null]
]
```

Definition at line 1 of file s1424\_8c.js.

## 30.1061 doc/html/s1425\_8c.js File Reference

### Variables

- var [s1425\\_8c](#)

### 30.1061.1 Variable Documentation

#### 30.1061.1.1 var s1425\_8c

**Initial value:**

```
=
[
 ["s1425", "s1425_8c.html#a363a479b71dc362c4a260c13b1a5da3b", null],
 ["s1425", "s1425_8c.html#a146c184e4ab808d3edd09d830bf874a4", null]
]
```

Definition at line 1 of file s1425\_8c.js.

## 30.1062 doc/html/s1435\_8c.js File Reference

### Variables

- var [s1435\\_8c](#)

### 30.1062.1 Variable Documentation

#### 30.1062.1.1 var s1435\_8c

**Initial value:**

```
=
[
 ["s1435", "s1435_8c.html#ace36343bff396338ebe5fdf10201a1ed", null],
 ["s1435", "s1435_8c.html#a28ff92fb8a25444ef76aafe4dd8d264d", null]
]
```

Definition at line 1 of file s1435\_8c.js.

## 30.1063 doc/html/s1436\_8c.js File Reference

### Variables

- var [s1436\\_8c](#)

### 30.1063.1 Variable Documentation

#### 30.1063.1.1 var s1436\_8c

##### Initial value:

```
=
[
 ["s1436", "s1436_8c.html#a2379f39fc5e396ee4dc69d5e547528a0", null],
 ["s1436", "s1436_8c.html#a3573ac59b657ab2d26fca2a65009f01d", null]
]
```

Definition at line 1 of file s1436\_8c.js.

## 30.1064 doc/html/s1437\_8c.js File Reference

### Variables

- var [s1437\\_8c](#)

### 30.1064.1 Variable Documentation

#### 30.1064.1.1 var s1437\_8c

##### Initial value:

```
=
[
 ["s1437", "s1437_8c.html#a2f0e6d83f0324ebbb0405e8935a8fc13", null],
 ["s1437", "s1437_8c.html#a53c7b0fc0654f7760ec9665550c5cebb", null]
]
```

Definition at line 1 of file s1437\_8c.js.

## 30.1065 doc/html/s1438\_8c.js File Reference

### Variables

- var [s1438\\_8c](#)

### 30.1065.1 Variable Documentation

#### 30.1065.1.1 var s1438\_8c

**Initial value:**

```
=
[
 ["s1438", "s1438_8c.html#a0699da2fe06c3d70d72c80cb73af4812", null],
 ["s1438", "s1438_8c.html#ae188ebb527a630649c7b6ca4b088e55c", null]
]
```

Definition at line 1 of file s1438\_8c.js.

## 30.1066 doc/html/s1439\_8c.js File Reference

### Variables

- var [s1439\\_8c](#)

### 30.1066.1 Variable Documentation

#### 30.1066.1.1 var s1439\_8c

**Initial value:**

```
=
[
 ["s1439", "s1439_8c.html#aab597b90f15ac5585f94f96943f89447", null],
 ["s1439", "s1439_8c.html#abc8bed96e83ffdf3815d56429b62a0e7", null]
]
```

Definition at line 1 of file s1439\_8c.js.

## 30.1067 doc/html/s1440\_8c.js File Reference

### Variables

- var [s1440\\_8c](#)

### 30.1067.1 Variable Documentation

#### 30.1067.1.1 var s1440\_8c

**Initial value:**

```
=
[
 ["s1440", "s1440_8c.html#a0b27fa37dda40a4060307450d6d5d226", null],
 ["s1440", "s1440_8c.html#a892946d7d6996e99398f22e5d6111e8b", null]
]
```

Definition at line 1 of file s1440\_8c.js.

## 30.1068 doc/html/s1450\_8c.js File Reference

### Variables

- var [s1450\\_8c](#)

### 30.1068.1 Variable Documentation

#### 30.1068.1.1 var s1450\_8c

##### Initial value:

```
=
[
 ["s1450", "s1450_8c.html#a13e986b5135727102f6445457cbf675d", null],
 ["s1450", "s1450_8c.html#aaf364c274b9f654c16da906c82a22d9b", null]
]
```

Definition at line 1 of file s1450\_8c.js.

## 30.1069 doc/html/s1451\_8c.js File Reference

### Variables

- var [s1451\\_8c](#)

### 30.1069.1 Variable Documentation

#### 30.1069.1.1 var s1451\_8c

##### Initial value:

```
=
[
 ["s1451", "s1451_8c.html#aa9664a8c9a271228a83657af6569d0d2", null],
 ["s1451", "s1451_8c.html#a39d89e0c25330c6158ba136153d4ab7e", null]
]
```

Definition at line 1 of file s1451\_8c.js.

## 30.1070 doc/html/s1452\_8c.js File Reference

### Variables

- var [s1452\\_8c](#)

### 30.1070.1 Variable Documentation

#### 30.1070.1.1 var s1452\_8c

**Initial value:**

```
=
[
 ["s1452", "s1452_8c.html#ac5f5b3b65256b029067e792144d72590", null],
 ["s1452", "s1452_8c.html#a7a6f8f8465ec22bd06b5d645083c8a9e", null]
]
```

Definition at line 1 of file s1452\_8c.js.

## 30.1071 doc/html/s1500\_8c.js File Reference

### Variables

- var [s1500\\_8c](#)

### 30.1071.1 Variable Documentation

#### 30.1071.1.1 var s1500\_8c

**Initial value:**

```
=
[
 ["s1500", "s1500_8c.html#a4031c940ea0ebec17da0d0c29812ae4d", null],
 ["s1500", "s1500_8c.html#aa8d8348c0584de3227b445151d1af8f0", null]
]
```

Definition at line 1 of file s1500\_8c.js.

## 30.1072 doc/html/s1501\_8c.js File Reference

### Variables

- var [s1501\\_8c](#)

### 30.1072.1 Variable Documentation

#### 30.1072.1.1 var s1501\_8c

**Initial value:**

```
=
[
 ["s1501", "s1501_8c.html#ae55d6b7f7f11ee1913dd6f49b2ed3625", null],
 ["s1501", "s1501_8c.html#a0dc7b561fa684990dd1d3c3a4796535c", null]
]
```

Definition at line 1 of file s1501\_8c.js.

## 30.1073 doc/html/s1502\_8c.js File Reference

### Variables

- var [s1502\\_8c](#)

### 30.1073.1 Variable Documentation

#### 30.1073.1.1 var s1502\_8c

##### Initial value:

```
=
[
 ["s1502", "s1502_8c.html#a73d73094d094f6a4daa2b13ae9adc98d", null],
 ["s1502", "s1502_8c.html#a641edf5b161b1c82bda09397ade7eb05", null]
]
```

Definition at line 1 of file s1502\_8c.js.

## 30.1074 doc/html/s1503\_8c.js File Reference

### Variables

- var [s1503\\_8c](#)

### 30.1074.1 Variable Documentation

#### 30.1074.1.1 var s1503\_8c

##### Initial value:

```
=
[
 ["s1503", "s1503_8c.html#a20e8937a5d9e76562da1bfa4a27ca66a", null],
 ["s1503", "s1503_8c.html#a528a2821a28f09299bab45d9720dbcd1", null]
]
```

Definition at line 1 of file s1503\_8c.js.

## 30.1075 doc/html/s1504\_8c.js File Reference

### Variables

- var [s1504\\_8c](#)

### 30.1075.1 Variable Documentation

#### 30.1075.1.1 var s1504\_8c

**Initial value:**

```
=
[
 ["s1504", "s1504_8c.html#a2c0a0d623e26b63686dffec2a3b61085", null],
 ["s1504", "s1504_8c.html#ad5173328786b66f85b9bb21f30e67146", null]
]
```

Definition at line 1 of file s1504\_8c.js.

## 30.1076 doc/html/s1505\_8c.js File Reference

### Variables

- var [s1505\\_8c](#)

### 30.1076.1 Variable Documentation

#### 30.1076.1.1 var s1505\_8c

**Initial value:**

```
=
[
 ["s1505", "s1505_8c.html#a39acb508d084c04c130c4df7b5b0f966", null],
 ["s1505", "s1505_8c.html#ac2f92230bcddb5a8c1f74a3f2918835", null]
]
```

Definition at line 1 of file s1505\_8c.js.

## 30.1077 doc/html/s1506\_8c.js File Reference

### Variables

- var [s1506\\_8c](#)

### 30.1077.1 Variable Documentation

#### 30.1077.1.1 var s1506\_8c

**Initial value:**

```
=
[
 ["s1506", "s1506_8c.html#a88531f3bdfccd4f64af0cc496539e2df", null],
 ["s1506", "s1506_8c.html#a9e804b833c5d7d9f7e09a9f5e159e5df", null]
]
```

Definition at line 1 of file s1506\_8c.js.

## 30.1078 doc/html/s1507\_8c.js File Reference

### Variables

- var [s1507\\_8c](#)

### 30.1078.1 Variable Documentation

#### 30.1078.1.1 var s1507\_8c

##### Initial value:

```
=
[
 ["s1507", "s1507_8c.html#ae5b9efbd72688d47e5e892546771608d", null],
 ["s1507", "s1507_8c.html#a36481d7d53f952f53d0d9c00b10ede72", null]
]
```

Definition at line 1 of file s1507\_8c.js.

## 30.1079 doc/html/s1508\_8c.js File Reference

### Variables

- var [s1508\\_8c](#)

### 30.1079.1 Variable Documentation

#### 30.1079.1.1 var s1508\_8c

##### Initial value:

```
=
[
 ["s1508", "s1508_8c.html#a783beb8c157b47f8e9eed854bcf9c697", null],
 ["s1508", "s1508_8c.html#a50e25f2c58ba524f34cfa7e1880981", null]
]
```

Definition at line 1 of file s1508\_8c.js.

## 30.1080 doc/html/s1510\_8c.js File Reference

### Variables

- var [s1510\\_8c](#)



### 30.1080.1 Variable Documentation

#### 30.1080.1.1 var s1510\_8c

**Initial value:**

```
=
[
 ["s1510", "s1510_8c.html#a9f7f52b7e9dcd6cc44d50949a30e9ef7", null],
 ["s1510", "s1510_8c.html#a45ce40ef11ef21ba4d62dad4ae3d0f6a", null]
]
```

Definition at line 1 of file s1510\_8c.js.

## 30.1081 doc/html/s1511\_8c.js File Reference

### Variables

- var [s1511\\_8c](#)

### 30.1081.1 Variable Documentation

#### 30.1081.1.1 var s1511\_8c

**Initial value:**

```
=
[
 ["s1511", "s1511_8c.html#ae934f5b3048d0acfcfef77e628d8c7b7", null],
 ["s1511", "s1511_8c.html#a2d221716f7f7ef064466df8cef4041f5", null]
]
```

Definition at line 1 of file s1511\_8c.js.

## 30.1082 doc/html/s1512\_8c.js File Reference

### Variables

- var [s1512\\_8c](#)

### 30.1082.1 Variable Documentation

#### 30.1082.1.1 var s1512\_8c

**Initial value:**

```
=
[
 ["s1512", "s1512_8c.html#a40106d2e1df895759345540c23e28fce", null],
 ["s1512", "s1512_8c.html#a3312b4b7017be2a75ba73bb69488d191", null]
]
```

Definition at line 1 of file s1512\_8c.js.

## 30.1083 doc/html/s1513\_8c.js File Reference

### Variables

- var [s1513\\_8c](#)

### 30.1083.1 Variable Documentation

#### 30.1083.1.1 var s1513\_8c

##### Initial value:

```
=
[
 ["s1513", "s1513_8c.html#ab3ef93b65c975c171433bdeeeec4beaf", null],
 ["s1513", "s1513_8c.html#afdf45308ab45f5300e02027b0c28fc52", null]
]
```

Definition at line 1 of file s1513\_8c.js.

## 30.1084 doc/html/s1514\_8c.js File Reference

### Variables

- var [s1514\\_8c](#)

### 30.1084.1 Variable Documentation

#### 30.1084.1.1 var s1514\_8c

##### Initial value:

```
=
[
 ["s1514", "s1514_8c.html#a83ac8ac8d2c4e6150a7103ac64948bf9", null],
 ["s1514", "s1514_8c.html#aeb12428dd7ddb9c6c78b1d6df0a348f8", null]
]
```

Definition at line 1 of file s1514\_8c.js.

## 30.1085 doc/html/s1515\_8c.js File Reference

### Variables

- var [s1515\\_8c](#)

### 30.1085.1 Variable Documentation

#### 30.1085.1.1 var s1515\_8c

**Initial value:**

```
=
[
 ["s1515", "s1515_8c.html#ab5eee8762ab66175d0cfa6e77c7ae8d4", null],
 ["s1515", "s1515_8c.html#ac62e8d53b455285134bf444bcbfa8852", null]
]
```

Definition at line 1 of file s1515\_8c.js.

## 30.1086 doc/html/s1516\_8c.js File Reference

### Variables

- var [s1516\\_8c](#)

### 30.1086.1 Variable Documentation

#### 30.1086.1.1 var s1516\_8c

**Initial value:**

```
=
[
 ["s1516", "s1516_8c.html#ad1cd8f37659ea120c14d73db66ccad8a", null],
 ["s1516", "s1516_8c.html#a4ed22b9d3da9759b61a5f15fc78cd4a9", null]
]
```

Definition at line 1 of file s1516\_8c.js.

## 30.1087 doc/html/s1517\_8c.js File Reference

### Variables

- var [s1517\\_8c](#)

### 30.1087.1 Variable Documentation

#### 30.1087.1.1 var s1517\_8c

**Initial value:**

```
=
[
 ["s1517", "s1517_8c.html#a95d58e22a1b437ccb97da8cb1110f9df", null],
 ["s1517", "s1517_8c.html#a6a1207257941658272037af1840bd1ce", null]
]
```

Definition at line 1 of file s1517\_8c.js.

## 30.1088 doc/html/s1518\_8c.js File Reference

### Variables

- var [s1518\\_8c](#)

### 30.1088.1 Variable Documentation

#### 30.1088.1.1 var s1518\_8c

##### Initial value:

```
=
[
 ["s1518", "s1518_8c.html#acdd28d02400b6c6090451eb6a6e4a69f", null],
 ["s1518", "s1518_8c.html#a332b2b27b2619bc65b029009c0904b89", null]
]
```

Definition at line 1 of file s1518\_8c.js.

## 30.1089 doc/html/s1520\_8c.js File Reference

### Variables

- var [s1520\\_8c](#)

### 30.1089.1 Variable Documentation

#### 30.1089.1.1 var s1520\_8c

##### Initial value:

```
=
[
 ["s1520", "s1520_8c.html#acc4fea4f45afaf5586b8c8eb0f37fa8e", null],
 ["s1520", "s1520_8c.html#acb0823675c13bf78016c65c9eb74261b", null]
]
```

Definition at line 1 of file s1520\_8c.js.

## 30.1090 doc/html/s1521\_8c.js File Reference

### Variables

- var [s1521\\_8c](#)

### 30.1090.1 Variable Documentation

#### 30.1090.1.1 var s1521\_8c

**Initial value:**

```
=
[
 ["s1521", "s1521_8c.html#a8cd39404134a411bf62134e959f06904", null],
 ["s1521", "s1521_8c.html#a84356cb25a8649dab3dc218bacc251dc", null]
]
```

Definition at line 1 of file s1521\_8c.js.

## 30.1091 doc/html/s1522\_8c.js File Reference

### Variables

- var [s1522\\_8c](#)

### 30.1091.1 Variable Documentation

#### 30.1091.1.1 var s1522\_8c

**Initial value:**

```
=
[
 ["s1522", "s1522_8c.html#a0fe619b936f3c47b8257e8f905e9b427", null],
 ["s1522", "s1522_8c.html#a689aae5d08a076c4bb1fc58020519edc", null]
]
```

Definition at line 1 of file s1522\_8c.js.

## 30.1092 doc/html/s1528\_8c.js File Reference

### Variables

- var [s1528\\_8c](#)

### 30.1092.1 Variable Documentation

#### 30.1092.1.1 var s1528\_8c

**Initial value:**

```
=
[
 ["s1528", "s1528_8c.html#ad1c4db07c34f6f4435134b2e71a63712", null],
 ["s1528", "s1528_8c.html#a7e27f141786fb80a448ecdb19d38709b", null]
]
```

Definition at line 1 of file s1528\_8c.js.

## 30.1093 doc/html/s1529\_8c.js File Reference

### Variables

- var [s1529\\_8c](#)

### 30.1093.1 Variable Documentation

#### 30.1093.1.1 var s1529\_8c

##### Initial value:

```
=
[
 ["s1529", "s1529_8c.html#a6bbd251a67111324464676accf0c2934", null],
 ["s1529", "s1529_8c.html#a3b7dd6d38ebb34bbdedf68005565416e", null]
]
```

Definition at line 1 of file s1529\_8c.js.

## 30.1094 doc/html/s1530\_8c.js File Reference

### Variables

- var [s1530\\_8c](#)

### 30.1094.1 Variable Documentation

#### 30.1094.1.1 var s1530\_8c

##### Initial value:

```
=
[
 ["s1530", "s1530_8c.html#aa689db34fbc96ac8767550dfef3ca03b", null],
 ["s1530", "s1530_8c.html#a8d53e22e3dc47c27199a9e8b462ba5ea", null]
]
```

Definition at line 1 of file s1530\_8c.js.

## 30.1095 doc/html/s1531\_8c.js File Reference

### Variables

- var [s1531\\_8c](#)

### 30.1095.1 Variable Documentation

#### 30.1095.1.1 var s1531\_8c

**Initial value:**

```
=
[
 ["s1531", "s1531_8c.html#a5c2636ce8e4c57f7d077a99433d6b021", null],
 ["s1531", "s1531_8c.html#af9e6a78d3a08df396c03a37f6118574d", null]
]
```

Definition at line 1 of file s1531\_8c.js.

## 30.1096 doc/html/s1534\_8c.js File Reference

### Variables

- var [s1534\\_8c](#)

### 30.1096.1 Variable Documentation

#### 30.1096.1.1 var s1534\_8c

**Initial value:**

```
=
[
 ["s1534", "s1534_8c.html#a6d46e247f66948f4ce57ba02cb0e6299", null],
 ["s1534", "s1534_8c.html#aac6969bc32e0226f9310dfc113e42ddc", null]
]
```

Definition at line 1 of file s1534\_8c.js.

## 30.1097 doc/html/s1535\_8c.js File Reference

### Variables

- var [s1535\\_8c](#)

### 30.1097.1 Variable Documentation

#### 30.1097.1.1 var s1535\_8c

**Initial value:**

```
=
[
 ["s1535", "s1535_8c.html#a2cc843fc4f91f947e0f5531834b6ba0f", null],
 ["s1535", "s1535_8c.html#a3c9412d6b5458fedf81c1431a148bb21", null]
]
```

Definition at line 1 of file s1535\_8c.js.

## 30.1098 doc/html/s1536\_8c.js File Reference

### Variables

- var [s1536\\_8c](#)

### 30.1098.1 Variable Documentation

#### 30.1098.1.1 var s1536\_8c

##### Initial value:

```
=
[
 ["s1536", "s1536_8c.html#ac3dbc16cf1f4ea63fe8ff7a93e9685cc", null],
 ["s1536", "s1536_8c.html#ae2c1f52401a24b646e4717bddca2606", null]
]
```

Definition at line 1 of file s1536\_8c.js.

## 30.1099 doc/html/s1537\_8c.js File Reference

### Variables

- var [s1537\\_8c](#)

### 30.1099.1 Variable Documentation

#### 30.1099.1.1 var s1537\_8c

##### Initial value:

```
=
[
 ["s1537", "s1537_8c.html#a706bcf941cce8fd0cde16f264195196f", null],
 ["s1537", "s1537_8c.html#a8a59ceab2a71a300b6bd69c211481ed9", null]
]
```

Definition at line 1 of file s1537\_8c.js.

## 30.1100 doc/html/s1538\_8c.js File Reference

### Variables

- var [s1538\\_8c](#)



### 30.1100.1 Variable Documentation

#### 30.1100.1.1 var s1538\_8c

**Initial value:**

```
=
[
 ["s1538", "s1538_8c.html#a7183f7a144c5eac9e3b9c52d83b44113", null],
 ["s1538", "s1538_8c.html#a86a32c12907cc53dd75ce4ccee5c6e5", null]
]
```

Definition at line 1 of file s1538\_8c.js.

## 30.1101 doc/html/s1539\_8c.js File Reference

### Variables

- var [s1539\\_8c](#)

### 30.1101.1 Variable Documentation

#### 30.1101.1.1 var s1539\_8c

**Initial value:**

```
=
[
 ["s1539", "s1539_8c.html#aaf0337bb0984e297d10f5359b4e1e175", null],
 ["s1539", "s1539_8c.html#ab4cad17c2172a091775a454c285bb6e0", null]
]
```

Definition at line 1 of file s1539\_8c.js.

## 30.1102 doc/html/s1540\_8c.js File Reference

### Variables

- var [s1540\\_8c](#)

### 30.1102.1 Variable Documentation

#### 30.1102.1.1 var s1540\_8c

**Initial value:**

```
=
[
 ["s1540", "s1540_8c.html#a40360950b51b502e42a0ce1c245dfdbf", null],
 ["s1540", "s1540_8c.html#aec9a229152e7f56fad1eac83309b8fea", null]
]
```

Definition at line 1 of file s1540\_8c.js.

## 30.1103 doc/html/s1541\_8c.js File Reference

### Variables

- var [s1541\\_8c](#)

### 30.1103.1 Variable Documentation

#### 30.1103.1.1 var s1541\_8c

##### Initial value:

```
=
[
 ["s1541", "s1541_8c.html#a61caeb2c247865af3f1944eca833592d", null],
 ["s1541", "s1541_8c.html#a0cc1c23c9944ff6a6c8f5e35350be5d2", null]
]
```

Definition at line 1 of file s1541\_8c.js.

## 30.1104 doc/html/s1542\_8c.js File Reference

### Variables

- var [s1542\\_8c](#)

### 30.1104.1 Variable Documentation

#### 30.1104.1.1 var s1542\_8c

##### Initial value:

```
=
[
 ["s1542", "s1542_8c.html#a43ee1413e2e7fa57fa6328cdcbc88958", null],
 ["s1542", "s1542_8c.html#a771aeeecd58f39100a261c5d4e2843a6", null]
]
```

Definition at line 1 of file s1542\_8c.js.

## 30.1105 doc/html/s1600\_8c.js File Reference

### Variables

- var [s1600\\_8c](#)

### 30.1105.1 Variable Documentation

#### 30.1105.1.1 var s1600\_8c

**Initial value:**

```
=
[
 ["s1600", "s1600_8c.html#a0e086ef8c5b60aaf1e1cbe53bb909d15", null],
 ["s1600", "s1600_8c.html#a82d436f466dbc9dcae3ac2576b883234", null]
]
```

Definition at line 1 of file s1600\_8c.js.

## 30.1106 doc/html/s1601\_8c.js File Reference

### Variables

- var [s1601\\_8c](#)

### 30.1106.1 Variable Documentation

#### 30.1106.1.1 var s1601\_8c

**Initial value:**

```
=
[
 ["s1601", "s1601_8c.html#a58d1799a235fad5fc73211581a4f10bd", null],
 ["s1601", "s1601_8c.html#ae44e83cf53dde1b524244c75c1a54d18", null]
]
```

Definition at line 1 of file s1601\_8c.js.

## 30.1107 doc/html/s1602\_8c.js File Reference

### Variables

- var [s1602\\_8c](#)

### 30.1107.1 Variable Documentation

#### 30.1107.1.1 var s1602\_8c

**Initial value:**

```
=
[
 ["s1602", "s1602_8c.html#ae441b62d9e775eac0e226927beb2aa60", null],
 ["s1602", "s1602_8c.html#a5cf39e641ec3fa3f25ec10b2b4a6f372", null]
]
```

Definition at line 1 of file s1602\_8c.js.

## 30.1108 doc/html/s1603\_8c.js File Reference

### Variables

- var [s1603\\_8c](#)

### 30.1108.1 Variable Documentation

#### 30.1108.1.1 var s1603\_8c

##### Initial value:

```
=
[
 ["s1603", "s1603_8c.html#abab6abdb1eb459b3267cc303f4d0312a", null],
 ["s1603", "s1603_8c.html#af5cada6b407a1c48665f8d8ae469ba74", null]
]
```

Definition at line 1 of file s1603\_8c.js.

## 30.1109 doc/html/s1604\_8c.js File Reference

### Variables

- var [s1604\\_8c](#)

### 30.1109.1 Variable Documentation

#### 30.1109.1.1 var s1604\_8c

##### Initial value:

```
=
[
 ["s1604", "s1604_8c.html#a3aac309f8fd545c030cf6f6300089131", null],
 ["s1604", "s1604_8c.html#af9afa82adc59f7c7ae5e53b99c5165ae", null]
]
```

Definition at line 1 of file s1604\_8c.js.

## 30.1110 doc/html/s1605\_8c.js File Reference

### Variables

- var [s1605\\_8c](#)

### 30.1110.1 Variable Documentation

#### 30.1110.1.1 var s1605\_8c

**Initial value:**

```
=
[
 ["s1605", "s1605_8c.html#ab6751a36b7218303f52ed8e7a75b33b2", null],
 ["s1605", "s1605_8c.html#ad631f33ea34d60b5774a2f32c0fa1229", null]
]
```

Definition at line 1 of file s1605\_8c.js.

## 30.1111 doc/html/s1606\_8c.js File Reference

### Variables

- var [s1606\\_8c](#)

### 30.1111.1 Variable Documentation

#### 30.1111.1.1 var s1606\_8c

**Initial value:**

```
=
[
 ["s1606", "s1606_8c.html#ab01d8698a6b6b039469eaff89aa8e7ad", null],
 ["s1606", "s1606_8c.html#a1c50b95759b03dbe3d5dd524a69967e7", null]
]
```

Definition at line 1 of file s1606\_8c.js.

## 30.1112 doc/html/s1607\_8c.js File Reference

### Variables

- var [s1607\\_8c](#)

### 30.1112.1 Variable Documentation

#### 30.1112.1.1 var s1607\_8c

**Initial value:**

```
=
[
 ["s1607", "s1607_8c.html#a2010cb7acdc0f920fcf80a0300941ec", null],
 ["s1607", "s1607_8c.html#a4f5ea196a30c048609cdd11b07f5fa7d", null]
]
```

Definition at line 1 of file s1607\_8c.js.

## 30.1113 doc/html/s1608\_8c.js File Reference

### Variables

- var [s1608\\_8c](#)

#### 30.1113.1 Variable Documentation

##### 30.1113.1.1 var s1608\_8c

###### Initial value:

```
=
[
 ["s1608", "s1608_8c.html#a19347a30ca53e0f2795a2f582444b845", null],
 ["s1608", "s1608_8c.html#a19e84c9b8e255d300f0c9a1fa798ed93", null]
]
```

Definition at line 1 of file s1608\_8c.js.

## 30.1114 doc/html/s1609\_8c.js File Reference

### Variables

- var [s1609\\_8c](#)

#### 30.1114.1 Variable Documentation

##### 30.1114.1.1 var s1609\_8c

###### Initial value:

```
=
[
 ["s1609", "s1609_8c.html#af701b05874a679871299af201d4ed586", null],
 ["s1609", "s1609_8c.html#a2ade710773132adbbc2a84c30e3525b2", null]
]
```

Definition at line 1 of file s1609\_8c.js.

## 30.1115 doc/html/s1611\_8c.js File Reference

### Variables

- var [s1611\\_8c](#)

### 30.1115.1 Variable Documentation

#### 30.1115.1.1 var s1611\_8c

**Initial value:**

```
=
[
 ["s1611", "s1611_8c.html#a88beb8be5874c4052b0c09f4aad27862", null],
 ["s1611", "s1611_8c.html#ab16ca516eba69d9c97365ee429d0019d", null]
]
```

Definition at line 1 of file s1611\_8c.js.

## 30.1116 doc/html/s1612\_8c.js File Reference

### Variables

- var [s1612\\_8c](#)

### 30.1116.1 Variable Documentation

#### 30.1116.1.1 var s1612\_8c

**Initial value:**

```
=
[
 ["s1612", "s1612_8c.html#a7d718ebf13d4472371fc78d195d29f3c", null],
 ["s1612", "s1612_8c.html#a79b79b224bddd4b580ea4c874b8f166d", null]
]
```

Definition at line 1 of file s1612\_8c.js.

## 30.1117 doc/html/s1613\_8c.js File Reference

### Variables

- var [s1613\\_8c](#)

### 30.1117.1 Variable Documentation

#### 30.1117.1.1 var s1613\_8c

**Initial value:**

```
=
[
 ["s1613", "s1613_8c.html#a7f75b81162cdb9ed867db90bdf260365", null],
 ["s1613", "s1613_8c.html#a153900a2d14f50ebaaf8e67cd9e9aeed", null]
]
```

Definition at line 1 of file s1613\_8c.js.

## 30.1118 doc/html/s1613bez\_8c.js File Reference

### Variables

- var [s1613bez\\_8c](#)

### 30.1118.1 Variable Documentation

#### 30.1118.1.1 var s1613bez\_8c

##### Initial value:

```
=
[
 ["S1613BEZ", "s1613bez_8c.html#aadb5b964d3e0e032301d36be31302f5b", null],
 ["s1613bez", "s1613bez_8c.html#a8a7ee8fce499d06ce861100548c05e94", null]
]
```

Definition at line 1 of file s1613bez\_8c.js.

## 30.1119 doc/html/s1614\_8c.js File Reference

### Variables

- var [s1614\\_8c](#)

### 30.1119.1 Variable Documentation

#### 30.1119.1.1 var s1614\_8c

##### Initial value:

```
=
[
 ["S1614", "s1614_8c.html#a7cc6ee3cbf608efe862886b10d8b680a", null],
 ["s1614", "s1614_8c.html#aa6d6b843ac707ca90be34563f3174bfc", null]
]
```

Definition at line 1 of file s1614\_8c.js.

## 30.1120 doc/html/s1615\_8c.js File Reference

### Variables

- var [s1615\\_8c](#)



### 30.1120.1 Variable Documentation

#### 30.1120.1.1 var s1615\_8c

**Initial value:**

```
=
[
 ["s1615", "s1615_8c.html#a45bb4e1c39bc77a180710bb0db1167fb", null],
 ["s1615", "s1615_8c.html#a0aeb63abc64b157d9dc969b9dd7c0f22", null]
]
```

Definition at line 1 of file s1615\_8c.js.

## 30.1121 doc/html/s1616\_8c.js File Reference

### Variables

- var [s1616\\_8c](#)

### 30.1121.1 Variable Documentation

#### 30.1121.1.1 var s1616\_8c

**Initial value:**

```
=
[
 ["s1616", "s1616_8c.html#a95cc7e6a80da3caeb7ad771c9acc8a2b", null],
 ["s1616", "s1616_8c.html#a902a4c5938dc0867db13451e998a6734", null]
]
```

Definition at line 1 of file s1616\_8c.js.

## 30.1122 doc/html/s1617\_8c.js File Reference

### Variables

- var [s1617\\_8c](#)

### 30.1122.1 Variable Documentation

#### 30.1122.1.1 var s1617\_8c

**Initial value:**

```
=
[
 ["s1617", "s1617_8c.html#a741f5fd1e98c470cf1db99d6e6ad249b", null],
 ["s1617", "s1617_8c.html#aa0b7055f1aeb5314cafcbee63ab57e8a", null]
]
```

Definition at line 1 of file s1617\_8c.js.

## 30.1123 doc/html/s1618\_8c.js File Reference

### Variables

- var [s1618\\_8c](#)

#### 30.1123.1 Variable Documentation

##### 30.1123.1.1 var s1618\_8c

###### Initial value:

```
=
[
 ["s1618", "s1618_8c.html#ab85ceeeae837c525f258bb73446dbdbf", null],
 ["s1618", "s1618_8c.html#a32a13fecae69349237bbbb3ffaa311d73", null]
]
```

Definition at line 1 of file s1618\_8c.js.

## 30.1124 doc/html/s1619\_8c.js File Reference

### Variables

- var [s1619\\_8c](#)

#### 30.1124.1 Variable Documentation

##### 30.1124.1.1 var s1619\_8c

###### Initial value:

```
=
[
 ["s1619", "s1619_8c.html#a4bafbeb505ea8b9bb61e347b011550cd", null],
 ["s1619", "s1619_8c.html#acfa872673774e245ca9f9d1129d16572", null]
]
```

Definition at line 1 of file s1619\_8c.js.

## 30.1125 doc/html/s1620\_8c.js File Reference

### Variables

- var [s1620\\_8c](#)

### 30.1125.1 Variable Documentation

#### 30.1125.1.1 var s1620\_8c

**Initial value:**

```
=
[
 ["s1620", "s1620_8c.html#aad71d3eaa62385d792a54c11971a244e", null],
 ["s1620", "s1620_8c.html#a71d58a8d850db4d06df449c2e18461b6", null]
]
```

Definition at line 1 of file s1620\_8c.js.

## 30.1126 doc/html/s1630\_8c.js File Reference

### Variables

- var [s1630\\_8c](#)

### 30.1126.1 Variable Documentation

#### 30.1126.1.1 var s1630\_8c

**Initial value:**

```
=
[
 ["s1630", "s1630_8c.html#a0eb594073118f9401c9424c5a57000b0", null],
 ["s1630", "s1630_8c.html#ace8f3395f7bfcaae53a0e0150779f4d3", null]
]
```

Definition at line 1 of file s1630\_8c.js.

## 30.1127 doc/html/s1631\_8c.js File Reference

### Variables

- var [s1631\\_8c](#)

### 30.1127.1 Variable Documentation

#### 30.1127.1.1 var s1631\_8c

**Initial value:**

```
=
[
 ["s1631", "s1631_8c.html#a3514f00152c2ed1d05513babbdb952b", null],
 ["s1631", "s1631_8c.html#a53aaba901df5442ab7393dec45b7cc", null]
]
```

Definition at line 1 of file s1631\_8c.js.

## 30.1128 doc/html/s1700\_8c.js File Reference

### Variables

- var [s1700\\_8c](#)

### 30.1128.1 Variable Documentation

#### 30.1128.1.1 var s1700\_8c

##### Initial value:

```
=
[
 ["s1700", "s1700_8c.html#adb6f62dd936e1ebe0534f03c5041022e", null],
 ["s1700", "s1700_8c.html#ac925c030622ffad61682e63a3d800239", null]
]
```

Definition at line 1 of file s1700\_8c.js.

## 30.1129 doc/html/s1701\_8c.js File Reference

### Variables

- var [s1701\\_8c](#)

### 30.1129.1 Variable Documentation

#### 30.1129.1.1 var s1701\_8c

##### Initial value:

```
=
[
 ["s1701", "s1701_8c.html#a12a65d58ace3e42c788e36591961484a", null],
 ["s1701", "s1701_8c.html#a512329a3b4177630dec261a42898e6fb", null]
]
```

Definition at line 1 of file s1701\_8c.js.

## 30.1130 doc/html/s1705\_8c.js File Reference

### Variables

- var [s1705\\_8c](#)

### 30.1130.1 Variable Documentation

#### 30.1130.1.1 var s1705\_8c

**Initial value:**

```
=
[
 ["s1705", "s1705_8c.html#a35d011b6419bb14ffdeec84fe4795a68", null],
 ["s1705", "s1705_8c.html#a22ad4f2315e47abc067d97c5094d4058", null]
]
```

Definition at line 1 of file s1705\_8c.js.

## 30.1131 doc/html/s1706\_8c.js File Reference

### Variables

- var [s1706\\_8c](#)

### 30.1131.1 Variable Documentation

#### 30.1131.1.1 var s1706\_8c

**Initial value:**

```
=
[
 ["s1706", "s1706_8c.html#af4a0debde381347653d010f25636c0af", null],
 ["s1706", "s1706_8c.html#adf958135b6137debf09658673adfa20b", null]
]
```

Definition at line 1 of file s1706\_8c.js.

## 30.1132 doc/html/s1707\_8c.js File Reference

### Variables

- var [s1707\\_8c](#)

### 30.1132.1 Variable Documentation

#### 30.1132.1.1 var s1707\_8c

**Initial value:**

```
=
[
 ["s1707", "s1707_8c.html#af9229b6e83db6072d144db8376230188", null],
 ["s1707", "s1707_8c.html#a13cd0a521a04d0b4b47107e048167d", null]
]
```

Definition at line 1 of file s1707\_8c.js.

## 30.1133 doc/html/s1708\_8c.js File Reference

### Variables

- var [s1708\\_8c](#)

#### 30.1133.1 Variable Documentation

##### 30.1133.1.1 var s1708\_8c

###### Initial value:

```
=
[
 ["s1708", "s1708_8c.html#a1e0e8dc5b64ef91c7ce22dda5712670d", null],
 ["s1708", "s1708_8c.html#a7ee444215733b2481127af6519e2e284", null]
]
```

Definition at line 1 of file s1708\_8c.js.

## 30.1134 doc/html/s1710\_8c.js File Reference

### Variables

- var [s1710\\_8c](#)

#### 30.1134.1 Variable Documentation

##### 30.1134.1.1 var s1710\_8c

###### Initial value:

```
=
[
 ["s1710", "s1710_8c.html#a44f63887e573a4453b97871f4c3d4392", null],
 ["s1710", "s1710_8c.html#abee883b856eb89bfd5594198077d3b9a", null]
]
```

Definition at line 1 of file s1710\_8c.js.

## 30.1135 doc/html/s1711\_8c.js File Reference

### Variables

- var [s1711\\_8c](#)

### 30.1135.1 Variable Documentation

#### 30.1135.1.1 var s1711\_8c

**Initial value:**

```
=
[
 ["s1711", "s1711_8c.html#ad3e7ecff87a8515cd38820fbdaeb63e8", null],
 ["s1711", "s1711_8c.html#a3c1ed6980d680a0617a7a328cea57f7b", null]
]
```

Definition at line 1 of file s1711\_8c.js.

## 30.1136 doc/html/s1712\_8c.js File Reference

### Variables

- var [s1712\\_8c](#)

### 30.1136.1 Variable Documentation

#### 30.1136.1.1 var s1712\_8c

**Initial value:**

```
=
[
 ["s1712", "s1712_8c.html#a20c8e656dd00cde713676ba8b137b674", null],
 ["s1712", "s1712_8c.html#a0d46472ada115e5265f31981fd336571", null]
]
```

Definition at line 1 of file s1712\_8c.js.

## 30.1137 doc/html/s1713\_8c.js File Reference

### Variables

- var [s1713\\_8c](#)

### 30.1137.1 Variable Documentation

#### 30.1137.1.1 var s1713\_8c

**Initial value:**

```
=
[
 ["s1713", "s1713_8c.html#af00af2c2d81b3c49e995b5ef59f6ef9a", null],
 ["s1713", "s1713_8c.html#a84417fad7a01a33dd6a869b4dc62cd87", null]
]
```

Definition at line 1 of file s1713\_8c.js.

## 30.1138 doc/html/s1714\_8c.js File Reference

### Variables

- var [s1714\\_8c](#)

### 30.1138.1 Variable Documentation

#### 30.1138.1.1 var s1714\_8c

##### Initial value:

```
=
[
 ["s1714", "s1714_8c.html#a04f12d45afabb0e1ea23e63db4e267bb", null],
 ["SISL_CRV_CLOSED", "s1714_8c.html#a275970986604600d92d99b5d05aac901", null],
 ["SISL_CRV_OPEN", "s1714_8c.html#a82c6dce6b0c6cf85e58ca798cd55e945", null],
 ["SISL_CRV_PERIODIC", "s1714_8c.html#a6fdb4242729596e8ce8f8ac54d58a30a", null],
 ["s1714", "s1714_8c.html#a9ea5b8b11f691c34fe7e75262fb0500c", null]
]
```

Definition at line 1 of file s1714\_8c.js.

## 30.1139 doc/html/s1715\_8c.js File Reference

### Variables

- var [s1715\\_8c](#)

### 30.1139.1 Variable Documentation

#### 30.1139.1.1 var s1715\_8c

##### Initial value:

```
=
[
 ["s1715", "s1715_8c.html#a4ffe20781f8d19aeb1d2301e7735afc7", null],
 ["s1715", "s1715_8c.html#a4de5f010f709e9d6b8ea5b0ab512d71e", null]
]
```

Definition at line 1 of file s1715\_8c.js.

## 30.1140 doc/html/s1716\_8c.js File Reference

### Variables

- var [s1716\\_8c](#)



### 30.1140.1 Variable Documentation

#### 30.1140.1.1 var s1716\_8c

**Initial value:**

```
=
[
 ["s1716", "s1716_8c.html#a0d5008ce8ec41295fa8fc2b8c59f055e", null],
 ["s1716", "s1716_8c.html#a9830f5bc1f3331268a3c1c4ea07167e0", null]
]
```

Definition at line 1 of file s1716\_8c.js.

## 30.1141 doc/html/s1720\_8c.js File Reference

### Variables

- var [s1720\\_8c](#)

### 30.1141.1 Variable Documentation

#### 30.1141.1.1 var s1720\_8c

**Initial value:**

```
=
[
 ["s1720", "s1720_8c.html#a7db7d2bb37b9e44b3e8a9293fc4a3bb2", null],
 ["s1720", "s1720_8c.html#a2f48fd3fd118ce3b11361a96b1664e80", null]
]
```

Definition at line 1 of file s1720\_8c.js.

## 30.1142 doc/html/s1730\_8c.js File Reference

### Variables

- var [s1730\\_8c](#)

### 30.1142.1 Variable Documentation

#### 30.1142.1.1 var s1730\_8c

**Initial value:**

```
=
[
 ["s1730", "s1730_8c.html#a1c4a1f8500bb6da352d1d94e89c5be32", null],
 ["s1730", "s1730_8c.html#a57a12822cfb425659496e90a02678a14", null]
]
```

Definition at line 1 of file s1730\_8c.js.

## 30.1143 doc/html/s1731\_8c.js File Reference

### Variables

- var [s1731\\_8c](#)

### 30.1143.1 Variable Documentation

#### 30.1143.1.1 var s1731\_8c

##### Initial value:

```
=
[
 ["s1731", "s1731_8c.html#af1723678e50f5b3be5f77a4aa9b6da08", null],
 ["s1731", "s1731_8c.html#ac14462b868b33b0bf0c84652f079b9b6", null]
]
```

Definition at line 1 of file s1731\_8c.js.

## 30.1144 doc/html/s1732\_8c.js File Reference

### Variables

- var [s1732\\_8c](#)

### 30.1144.1 Variable Documentation

#### 30.1144.1.1 var s1732\_8c

##### Initial value:

```
=
[
 ["s1732", "s1732_8c.html#ab395d9f90c202ae754e17375fde12fal", null],
 ["s1732", "s1732_8c.html#a0cbdbaa76f144dba2d19c27560ac325a", null]
]
```

Definition at line 1 of file s1732\_8c.js.

## 30.1145 doc/html/s1733\_8c.js File Reference

### Variables

- var [s1733\\_8c](#)

### 30.1145.1 Variable Documentation

#### 30.1145.1.1 var s1733\_8c

**Initial value:**

```
=
[
 ["s1733", "s1733_8c.html#aede761c84aef8681e579480be27ea7dd", null],
 ["s1733", "s1733_8c.html#a95a165f231a7427d7854f778eb8a240f", null]
]
```

Definition at line 1 of file s1733\_8c.js.

## 30.1146 doc/html/s1741\_8c.js File Reference

### Variables

- var [s1741\\_8c](#)

### 30.1146.1 Variable Documentation

#### 30.1146.1.1 var s1741\_8c

**Initial value:**

```
=
[
 ["s1741", "s1741_8c.html#ad0cb579f137b0860f8cbe144acf2a431", null],
 ["s1741", "s1741_8c.html#a1a4188de8c523f3a94965ce788a6f71d", null]
]
```

Definition at line 1 of file s1741\_8c.js.

## 30.1147 doc/html/s1750\_8c.js File Reference

### Variables

- var [s1750\\_8c](#)

### 30.1147.1 Variable Documentation

#### 30.1147.1.1 var s1750\_8c

**Initial value:**

```
=
[
 ["s1750", "s1750_8c.html#aa0cde0a2239d2196dc686d2366afc64c", null],
 ["s1750", "s1750_8c.html#ad7417ba9f75fd5a9aa2cde8af5e8eed6", null]
]
```

Definition at line 1 of file s1750\_8c.js.

## 30.1148 doc/html/s1753\_8c.js File Reference

### Variables

- var [s1753\\_8c](#)

### 30.1148.1 Variable Documentation

#### 30.1148.1.1 var s1753\_8c

##### Initial value:

```
=
[
 ["s1753", "s1753_8c.html#a4e04bf0ef6b21605a2585fb4c063c79d", null],
 ["s1753", "s1753_8c.html#a9c1895747dd3ad8222e58b622a1fb462", null]
]
```

Definition at line 1 of file s1753\_8c.js.

## 30.1149 doc/html/s1754\_8c.js File Reference

### Variables

- var [s1754\\_8c](#)

### 30.1149.1 Variable Documentation

#### 30.1149.1.1 var s1754\_8c

##### Initial value:

```
=
[
 ["s1754", "s1754_8c.html#a8678f43cd292ef59e829d0635d21f9c5", null],
 ["s1754", "s1754_8c.html#a7b2cce6b758dc22c3d5120f84f4d762b", null]
]
```

Definition at line 1 of file s1754\_8c.js.

## 30.1150 doc/html/s1755\_8c.js File Reference

### Variables

- var [s1755\\_8c](#)

### 30.1150.1 Variable Documentation

#### 30.1150.1.1 var s1755\_8c

**Initial value:**

```
=
[
 ["s1755", "s1755_8c.html#a8734373c7548dad90ef20f937da922ef", null],
 ["s1755", "s1755_8c.html#a07ae936702dedadda59d8d570c20f586", null]
]
```

Definition at line 1 of file s1755\_8c.js.

## 30.1151 doc/html/s17702d\_8c.js File Reference

### Variables

- var [s17702d\\_8c](#)

### 30.1151.1 Variable Documentation

#### 30.1151.1.1 var s17702d\_8c

**Initial value:**

```
=
[
 ["copy2", "s17702d_8c.html#a94d298c6c82c75789d11ea2fad958ff1", null],
 ["copy3", "s17702d_8c.html#abf5f71af6eec81e8ff8b99c9dae9f0b1", null],
 ["decr2", "s17702d_8c.html#aa8ea14f873c1bff5a2e825ff20759e33", null],
 ["incr2", "s17702d_8c.html#a66086fa9df80019f1e7929b6368f30dc", null],
 ["S1770_2D", "s17702d_8c.html#ae7fb2e682dc4304d9ef57e060624fa55", null],
 ["set_order", "s17702d_8c.html#af1da7d699e660aee38aff2d1d660da8", null],
 ["SINGULAR", "s17702d_8c.html#a87d5f56e2e026ba605bc9303c4fceaafa", null],
 ["s1770_2D", "s17702d_8c.html#a8ee503d46024d4c317e6390b57fd9c52", null]
]
```

Definition at line 1 of file s17702d\_8c.js.

## 30.1152 doc/html/s1770\_8c.js File Reference

### Variables

- var [s1770\\_8c](#)

### 30.1152.1 Variable Documentation

#### 30.1152.1.1 var s1770\_8c

**Initial value:**

```
=
[
 ["s1770", "s1770_8c.html#ae23588e589367d602793e45c7f85a03a", null],
 ["s1770", "s1770_8c.html#a780f75fe9d360fbfelca156df74c8750", null]
]
```

Definition at line 1 of file s1770\_8c.js.

### 30.1153 doc/html/s1771\_8c.js File Reference

#### Variables

- var [s1771\\_8c](#)

#### 30.1153.1 Variable Documentation

##### 30.1153.1.1 var s1771\_8c

**Initial value:**

```
=
[
 ["s1771", "s1771_8c.html#a7609a453b844a2b3a12f80755e7a6294", null],
 ["s1771", "s1771_8c.html#a648e035471a0aacb650f9d206ee3bf7c", null]
]
```

Definition at line 1 of file s1771\_8c.js.

### 30.1154 doc/html/s1772\_8c.js File Reference

#### Variables

- var [s1772\\_8c](#)

### 30.1154.1 Variable Documentation

#### 30.1154.1.1 var s1772\_8c

**Initial value:**

```
=
[
 ["copy2", "s1772_8c.html#a94d298c6c82c75789d11ea2fad958ff1", null],
 ["copy3", "s1772_8c.html#abf5f71af6eec81e8ff8b99c9dae9f0b1", null],
 ["decr2", "s1772_8c.html#aa8ea14f873c1bff5a2e825ff20759e33", null],
 ["incr2", "s1772_8c.html#a66086fa9df80019f1e7929b6368f30dc", null],
 ["s1772", "s1772_8c.html#a928afcd76735443482324cc52e8cb3a4", null],
 ["set_order", "s1772_8c.html#af1da7d699e660aee38aff2d1d660da8", null],
 ["SINGULAR", "s1772_8c.html#a87d5f56e2e026ba605bc9303c4fceaafa", null],
 ["s1772", "s1772_8c.html#ac375d1a4f8cc68ef9e9975c5eb97bbd4", null]
]
```

Definition at line 1 of file s1772\_8c.js.

## 30.1155 doc/html/s1773\_8c.js File Reference

### Variables

- var [s1773\\_8c](#)

### 30.1155.1 Variable Documentation

#### 30.1155.1.1 var s1773\_8c

**Initial value:**

```
=
[
 ["s1773", "s1773_8c.html#ab9402a7ae24a7a51760a038f873ae123", null],
 ["s1773", "s1773_8c.html#a1b8c84f7a560d6aaf077181cabe7f52", null]
]
```

Definition at line 1 of file s1773\_8c.js.

## 30.1156 doc/html/s1774\_8c.js File Reference

### Variables

- var [s1774\\_8c](#)

### 30.1156.1 Variable Documentation

30.1156.1.1 `var s1774_8c`

#### Initial value:

```
=
[
 ["s1774", "s1774_8c.html#a21800a6dba1144464a9e5c0448d0648d", null],
 ["s1774", "s1774_8c.html#aa14e06e74f9ce43d0973338cdc0a0d3d", null]
]
```

Definition at line 1 of file s1774\_8c.js.

### 30.1157 doc/html/s1775\_8c.js File Reference

#### Variables

- `var s1775_8c`

### 30.1157.1 Variable Documentation

30.1157.1.1 `var s1775_8c`

#### Initial value:

```
=
[
 ["s1775", "s1775_8c.html#aa115a33cce76971296a745a3dfc258ff", null],
 ["s1775", "s1775_8c.html#a1f9662c192d2a9eca80a55819654832d", null]
]
```

Definition at line 1 of file s1775\_8c.js.

### 30.1158 doc/html/s1780\_8c.js File Reference

#### Variables

- `var s1780_8c`

### 30.1158.1 Variable Documentation

30.1158.1.1 `var s1780_8c`

#### Initial value:

```
=
[
 ["s1780", "s1780_8c.html#ae3b7e58fbd6f44203cc526682fde382", null],
 ["s1780", "s1780_8c.html#a75790ec10957b2266827c5c4bd179941", null]
]
```

Definition at line 1 of file s1780\_8c.js.



## 30.1159 doc/html/s1785\_8c.js File Reference

### Variables

- var [s1785\\_8c](#)

#### 30.1159.1 Variable Documentation

##### 30.1159.1.1 var s1785\_8c

###### Initial value:

```
=
[
 ["s1785", "s1785_8c.html#a7e312a93012661af40279e2322058247", null],
 ["s1785", "s1785_8c.html#a474c8d138f62f1dc89f14ac8f5616565", null]
]
```

Definition at line 1 of file s1785\_8c.js.

## 30.1160 doc/html/s1786\_8c.js File Reference

### Variables

- var [s1786\\_8c](#)

#### 30.1160.1 Variable Documentation

##### 30.1160.1.1 var s1786\_8c

###### Initial value:

```
=
[
 ["s1786", "s1786_8c.html#a92cbf525ee29c8519a05fc9cd9a4a69e", null],
 ["fevalcProc", "s1786_8c.html#a6f5d68d6da132de46dd79668ba91bb37", null],
 ["s1786", "s1786_8c.html#ac58d8ac51e7cbac85f86849ef604ea65", null]
]
```

Definition at line 1 of file s1786\_8c.js.

## 30.1161 doc/html/s1787\_8c.js File Reference

### Variables

- var [s1787\\_8c](#)

### 30.1161.1 Variable Documentation

30.1161.1.1 `var s1787_8c`

**Initial value:**

```
=
[
 ["s1787", "s1787_8c.html#a11c2763709988e29bfc25952b086b357", null],
 ["s1787", "s1787_8c.html#a0541ea96e7e8fcd333c6fd3cd0b13c4", null]
]
```

Definition at line 1 of file s1787\_8c.js.

### 30.1162 doc/html/s1788\_8c.js File Reference

#### Variables

- `var s1788_8c`

### 30.1162.1 Variable Documentation

30.1162.1.1 `var s1788_8c`

**Initial value:**

```
=
[
 ["s1788", "s1788_8c.html#a79ced9cdb5ef648c5b0ca7ed585713aa", null],
 ["s1788", "s1788_8c.html#aa76e1a7505d819ce6319f7c1a6e7ea3e", null]
]
```

Definition at line 1 of file s1788\_8c.js.

### 30.1163 doc/html/s1789\_8c.js File Reference

#### Variables

- `var s1789_8c`

### 30.1163.1 Variable Documentation

30.1163.1.1 `var s1789_8c`

**Initial value:**

```
=
[
 ["s1789", "s1789_8c.html#a5befd0675445f19b13c4f7288f9f932c", null],
 ["s1789", "s1789_8c.html#a1ccc6ecfefada63d7bcad50f72d5e119", null]
]
```

Definition at line 1 of file s1789\_8c.js.

## 30.1164 doc/html/s1790\_8c.js File Reference

### Variables

- var [s1790\\_8c](#)

#### 30.1164.1 Variable Documentation

##### 30.1164.1.1 var s1790\_8c

###### Initial value:

```
=
[
 ["s1790", "s1790_8c.html#a3ce6598076009ea8061ed79bf9584f91", null],
 ["s1790", "s1790_8c.html#a40f7a78962da4f28a2ece3ccb04ce666", null]
]
```

Definition at line 1 of file s1790\_8c.js.

## 30.1165 doc/html/s1791\_8c.js File Reference

### Variables

- var [s1791\\_8c](#)

#### 30.1165.1 Variable Documentation

##### 30.1165.1.1 var s1791\_8c

###### Initial value:

```
=
[
 ["s1791", "s1791_8c.html#a42a61237e2617155dc0dadf35776f5e3", null],
 ["s1791", "s1791_8c.html#a5721c0c9b593c252bc4d033d5ac6f642", null]
]
```

Definition at line 1 of file s1791\_8c.js.

## 30.1166 doc/html/s1792\_8c.js File Reference

### Variables

- var [s1792\\_8c](#)

### 30.1166.1 Variable Documentation

30.1166.1.1 `var s1792_8c`

#### Initial value:

```
=
[
 ["s1792", "s1792_8c.html#aca1a540f503102e0556b1d2a42378f93", null],
 ["s1792", "s1792_8c.html#af4b3c2d321fc86f2a24c721cce053cee", null]
]
```

Definition at line 1 of file s1792\_8c.js.

### 30.1167 doc/html/s1795\_8c.js File Reference

#### Variables

- `var s1795_8c`

### 30.1167.1 Variable Documentation

30.1167.1.1 `var s1795_8c`

#### Initial value:

```
=
[
 ["s1795", "s1795_8c.html#a672d7f40168a03a4148f456867da18b7", null],
 ["s1795", "s1795_8c.html#ad4be6942d3725e0dd63bec5791b6a8cc", null]
]
```

Definition at line 1 of file s1795\_8c.js.

### 30.1168 doc/html/s1796\_8c.js File Reference

#### Variables

- `var s1796_8c`

### 30.1168.1 Variable Documentation

30.1168.1.1 `var s1796_8c`

#### Initial value:

```
=
[
 ["s1796", "s1796_8c.html#ae0903d18fe3755edbd81601347a8e067", null],
 ["s1796", "s1796_8c.html#a0ed4b30590356c9f4be85c9155b67d5a", null]
]
```

Definition at line 1 of file s1796\_8c.js.

## 30.1169 doc/html/s1797\_8c.js File Reference

### Variables

- var [s1797\\_8c](#)

#### 30.1169.1 Variable Documentation

##### 30.1169.1.1 var s1797\_8c

###### Initial value:

```
=
[
 ["s1797", "s1797_8c.html#a1c1acc3a62f4f516c6af659dade52a37", null],
 ["s1797", "s1797_8c.html#ae4e54adf6d1c8302bca8dc913d242af8", null]
]
```

Definition at line 1 of file s1797\_8c.js.

## 30.1170 doc/html/s1830\_8c.js File Reference

### Variables

- var [s1830\\_8c](#)

#### 30.1170.1 Variable Documentation

##### 30.1170.1.1 var s1830\_8c

###### Initial value:

```
=
[
 ["s1830", "s1830_8c.html#ac9b478dc0b37c2e4f219f1bfa2c92c68", null],
 ["s1830", "s1830_8c.html#a5e67c0ec6af8c243236a930dadb1f9d1", null]
]
```

Definition at line 1 of file s1830\_8c.js.

## 30.1171 doc/html/s1834\_8c.js File Reference

### Variables

- var [s1834\\_8c](#)

### 30.1171.1 Variable Documentation

#### 30.1171.1.1 var s1834\_8c

**Initial value:**

```
=
[
 ["s1834", "s1834_8c.html#a64cbc5c796dd6c75fad57ffa3d161b34", null],
 ["s1834", "s1834_8c.html#a45e46152948c711003013e019d3b3782", null]
]
```

Definition at line 1 of file s1834\_8c.js.

### 30.1172 doc/html/s1839\_8c.js File Reference

**Variables**

- var [s1839\\_8c](#)

#### 30.1172.1 Variable Documentation

##### 30.1172.1.1 var s1839\_8c

**Initial value:**

```
=
[
 ["s1839", "s1839_8c.html#a81dd74ed7d43b444407685292e1de4d4", null],
 ["s1839", "s1839_8c.html#a382799b6bfb725b4ff45e65d87965b43", null]
]
```

Definition at line 1 of file s1839\_8c.js.

### 30.1173 doc/html/s1840\_8c.js File Reference

**Variables**

- var [s1840\\_8c](#)

#### 30.1173.1 Variable Documentation

##### 30.1173.1.1 var s1840\_8c

**Initial value:**

```
=
[
 ["s1840", "s1840_8c.html#a7b7be0678ffad39f96259df0c1c2645a", null],
 ["s1840", "s1840_8c.html#ae83296350dce7a5195bbe10dc9cf705f", null]
]
```

Definition at line 1 of file s1840\_8c.js.

## 30.1174 doc/html/s1850\_8c.js File Reference

### Variables

- var [s1850\\_8c](#)

#### 30.1174.1 Variable Documentation

##### 30.1174.1.1 var s1850\_8c

###### Initial value:

```
=
[
 ["s1850", "s1850_8c.html#a665bd743c4b047fa4a20980c996aacda", null],
 ["s1850", "s1850_8c.html#a6a2e79aa2fdf3c1985fdaef153f542de", null]
]
```

Definition at line 1 of file s1850\_8c.js.

## 30.1175 doc/html/s1851\_8c.js File Reference

### Variables

- var [s1851\\_8c](#)

#### 30.1175.1 Variable Documentation

##### 30.1175.1.1 var s1851\_8c

###### Initial value:

```
=
[
 ["s1851", "s1851_8c.html#a4cb6dffc8672b711c2b7052610cdbfd48", null],
 ["s1851", "s1851_8c.html#aa67212524fc6e8ea0d35c9f367f9dfd9", null]
]
```

Definition at line 1 of file s1851\_8c.js.

## 30.1176 doc/html/s1852\_8c.js File Reference

### Variables

- var [s1852\\_8c](#)

### 30.1176.1 Variable Documentation

30.1176.1.1 `var s1852_8c`

**Initial value:**

```
=
[
 ["s1852", "s1852_8c.html#aa2a042fc13e197b22d57cab6b169e676", null],
 ["s1852", "s1852_8c.html#a3d0c0523066186753109c4347f85b22a", null]
]
```

Definition at line 1 of file s1852\_8c.js.

### 30.1177 doc/html/s1853\_8c.js File Reference

**Variables**

- `var s1853_8c`

#### 30.1177.1 Variable Documentation

30.1177.1.1 `var s1853_8c`

**Initial value:**

```
=
[
 ["s1853", "s1853_8c.html#a4648d8be668765760517ff0d212efe36", null],
 ["s1853", "s1853_8c.html#abd36c946780d421748b983ab2ec8bc8d", null]
]
```

Definition at line 1 of file s1853\_8c.js.

### 30.1178 doc/html/s1854\_8c.js File Reference

**Variables**

- `var s1854_8c`

#### 30.1178.1 Variable Documentation

30.1178.1.1 `var s1854_8c`

**Initial value:**

```
=
[
 ["s1854", "s1854_8c.html#a4b65fd23ceeb92c325eec4e77b7bd819", null],
 ["s1854", "s1854_8c.html#af4e7417292fb7b6615fd39d5f28fe281", null]
]
```

Definition at line 1 of file s1854\_8c.js.



## 30.1179 doc/html/s1855\_8c.js File Reference

### Variables

- var [s1855\\_8c](#)

#### 30.1179.1 Variable Documentation

##### 30.1179.1.1 var s1855\_8c

###### Initial value:

```
=
[
 ["s1855", "s1855_8c.html#ad2b751f0ea135580dd36f40db5bf348c", null],
 ["s1855", "s1855_8c.html#a48f345c05911304f9534c00187768db0", null]
]
```

Definition at line 1 of file s1855\_8c.js.

## 30.1180 doc/html/s1856\_8c.js File Reference

### Variables

- var [s1856\\_8c](#)

#### 30.1180.1 Variable Documentation

##### 30.1180.1.1 var s1856\_8c

###### Initial value:

```
=
[
 ["s1856", "s1856_8c.html#a8d17bb9cdd5246b69158f9c01f871305", null],
 ["s1856", "s1856_8c.html#af3d52e3b042e946679a41bfff5dfe223", null]
]
```

Definition at line 1 of file s1856\_8c.js.

## 30.1181 doc/html/s1857\_8c.js File Reference

### Variables

- var [s1857\\_8c](#)

### 30.1181.1 Variable Documentation

30.1181.1.1 `var s1857_8c`

#### Initial value:

```
=
[
 ["s1857", "s1857_8c.html#a6be0ffcd196ba7a9f9dcaba5b8d00395", null],
 ["s1857", "s1857_8c.html#a21269a7593217f4aa2733eed442175bc", null]
]
```

Definition at line 1 of file s1857\_8c.js.

### 30.1182 doc/html/s1858\_8c.js File Reference

#### Variables

- `var s1858_8c`

### 30.1182.1 Variable Documentation

30.1182.1.1 `var s1858_8c`

#### Initial value:

```
=
[
 ["s1858", "s1858_8c.html#af9bfe323d2d60cdc76444103ee3481fd", null],
 ["s1858", "s1858_8c.html#a554fb7aa59bddf5e20462f0d7dd9641a", null]
]
```

Definition at line 1 of file s1858\_8c.js.

### 30.1183 doc/html/s1859\_8c.js File Reference

#### Variables

- `var s1859_8c`

### 30.1183.1 Variable Documentation

30.1183.1.1 `var s1859_8c`

#### Initial value:

```
=
[
 ["s1859", "s1859_8c.html#a93c373f9577c114c6db83980702326cf", null],
 ["s1859", "s1859_8c.html#a474fdbe04906e04d8d8351d4a72d3f48", null]
]
```

Definition at line 1 of file s1859\_8c.js.

## 30.1184 doc/html/s1860\_8c.js File Reference

### Variables

- var [s1860\\_8c](#)

#### 30.1184.1 Variable Documentation

##### 30.1184.1.1 var s1860\_8c

###### Initial value:

```
=
[
 ["s1860", "s1860_8c.html#a1e1a8be89d3800e4e70bcba7b4ac08ee", null],
 ["s1860", "s1860_8c.html#ac86d120adc27d2ecb653339a9336d022", null]
]
```

Definition at line 1 of file s1860\_8c.js.

## 30.1185 doc/html/s1870\_8c.js File Reference

### Variables

- var [s1870\\_8c](#)

#### 30.1185.1 Variable Documentation

##### 30.1185.1.1 var s1870\_8c

###### Initial value:

```
=
[
 ["s1870", "s1870_8c.html#aa8f22c39cf06fa92c501241955b9959c", null],
 ["s1870", "s1870_8c.html#a9acd1de6c8ceecfe1e290f014350a047", null]
]
```

Definition at line 1 of file s1870\_8c.js.

## 30.1186 doc/html/s1871\_8c.js File Reference

### Variables

- var [s1871\\_8c](#)

### 30.1186.1 Variable Documentation

30.1186.1.1 `var s1871_8c`

#### Initial value:

```
=
[
 ["s1871", "s1871_8c.html#a555fcb5ab210814792c3d515e4407750", null],
 ["s1871", "s1871_8c.html#a973722c65e9fa3155b1433bd01da2332", null]
]
```

Definition at line 1 of file s1871\_8c.js.

### 30.1187 doc/html/s1880\_8c.js File Reference

#### Variables

- `var s1880_8c`

### 30.1187.1 Variable Documentation

30.1187.1.1 `var s1880_8c`

#### Initial value:

```
=
[
 ["s1880", "s1880_8c.html#adf62faf690bf998018baac89112755da", null],
 ["s1880", "s1880_8c.html#a0be05ce148374b803d8f5f1eb3485bdb", null]
]
```

Definition at line 1 of file s1880\_8c.js.

### 30.1188 doc/html/s1890\_8c.js File Reference

#### Variables

- `var s1890_8c`

### 30.1188.1 Variable Documentation

30.1188.1.1 `var s1890_8c`

#### Initial value:

```
=
[
 ["s1890", "s1890_8c.html#ad6b06233018038a9040b6654bad41e80", null],
 ["s1890", "s1890_8c.html#af504cc7a4dc4d2d23e5dc06ae87e3075", null]
]
```

Definition at line 1 of file s1890\_8c.js.

## 30.1189 doc/html/s1891\_8c.js File Reference

### Variables

- var [s1891\\_8c](#)

### 30.1189.1 Variable Documentation

#### 30.1189.1.1 var s1891\_8c

##### Initial value:

```
=
[
 ["MAX_SIZE", "s1891_8c.html#a0592dba56693fad79136250c11e5a7fe", null],
 ["S1891", "s1891_8c.html#af067c27d78997910d1d8aea06c02d225", null],
 ["s1891", "s1891_8c.html#acd7628686af47b2af41e63f84d7ffc8d", null]
]
```

Definition at line 1 of file s1891\_8c.js.

## 30.1190 doc/html/s1893\_8c.js File Reference

### Variables

- var [s1893\\_8c](#)

### 30.1190.1 Variable Documentation

#### 30.1190.1.1 var s1893\_8c

##### Initial value:

```
=
[
 ["S1893", "s1893_8c.html#a60c88aa238f5edbe12d910164b02b3a7", null],
 ["s1893", "s1893_8c.html#a82ce4a46d6e6d9202b495a1960af10f1", null]
]
```

Definition at line 1 of file s1893\_8c.js.

## 30.1191 doc/html/s1894\_8c.js File Reference

### Variables

- var [s1894\\_8c](#)

### 30.1191.1 Variable Documentation

#### 30.1191.1.1 var s1894\_8c

**Initial value:**

```
=
[
 ["s1894", "s1894_8c.html#af700ddc4bf0f472e5668b4369ae57a87", null],
 ["s1894", "s1894_8c.html#a99bf48bacd3b9ea98574ba8393294434", null]
]
```

Definition at line 1 of file s1894\_8c.js.

### 30.1192 doc/html/s1896\_8c.js File Reference

**Variables**

- var [s1896\\_8c](#)

#### 30.1192.1 Variable Documentation

##### 30.1192.1.1 var s1896\_8c

**Initial value:**

```
=
[
 ["s1896", "s1896_8c.html#a066848aff8c99dce0b143df72eab003f", null],
 ["s1896", "s1896_8c.html#ab404f2c2e6a05b1c0b3f19782b1ede67", null]
]
```

Definition at line 1 of file s1896\_8c.js.

### 30.1193 doc/html/s1897\_8c.js File Reference

**Variables**

- var [s1897\\_8c](#)

#### 30.1193.1 Variable Documentation

##### 30.1193.1.1 var s1897\_8c

**Initial value:**

```
=
[
 ["MAX_IK", "s1897_8c.html#ac79ba5168f3ea53c51d1f9aa515cf7b3", null],
 ["s1897", "s1897_8c.html#a19c36dcc4b203e29e4d9a49a06c64ca0", null],
 ["s1897", "s1897_8c.html#a1fbbf1920937a33f66202cb96c39a5cb", null]
]
```

Definition at line 1 of file s1897\_8c.js.

## 30.1194 doc/html/s1900\_8c.js File Reference

### Variables

- var [s1900\\_8c](#)

#### 30.1194.1 Variable Documentation

##### 30.1194.1.1 var s1900\_8c

###### Initial value:

```
=
[
 ["s1900", "s1900_8c.html#a9448930545d6384f02aed5041a0968ce", null],
 ["s1900", "s1900_8c.html#a9cd8432125d6a0c688d40bfe567663be", null]
]
```

Definition at line 1 of file s1900\_8c.js.

## 30.1195 doc/html/s1901\_8c.js File Reference

### Variables

- var [s1901\\_8c](#)

#### 30.1195.1 Variable Documentation

##### 30.1195.1.1 var s1901\_8c

###### Initial value:

```
=
[
 ["s1901", "s1901_8c.html#a8bbe2100595e58d4531194358d162e01", null],
 ["fknotsProc", "s1901_8c.html#a693628c216123daeabc374eaf6826103", null],
 ["fparamProc", "s1901_8c.html#a340939cdef71c179f28742c34ace9588", null],
 ["s1901", "s1901_8c.html#a02b22ba4aef3e76d04f1c64efef75ce6", null]
]
```

Definition at line 1 of file s1901\_8c.js.

## 30.1196 doc/html/s1902\_8c.js File Reference

### Variables

- var [s1902\\_8c](#)

### 30.1196.1 Variable Documentation

30.1196.1.1 `var s1902_8c`

#### Initial value:

```
=
[
 ["s1902", "s1902_8c.html#ae2b3cb8648e7042b8936b60e213b9b35", null],
 ["s1902", "s1902_8c.html#a674ca8a9ea7509f21e5ba08face1b4ac", null]
]
```

Definition at line 1 of file s1902\_8c.js.

### 30.1197 doc/html/s1903\_8c.js File Reference

#### Variables

- `var s1903_8c`

### 30.1197.1 Variable Documentation

30.1197.1.1 `var s1903_8c`

#### Initial value:

```
=
[
 ["s1903", "s1903_8c.html#a4e7145d035b2703c74a2c8a5277c444c", null],
 ["s1903", "s1903_8c.html#ae0bad153a5779b416d2488c60ba79216", null]
]
```

Definition at line 1 of file s1903\_8c.js.

### 30.1198 doc/html/s1904\_8c.js File Reference

#### Variables

- `var s1904_8c`

### 30.1198.1 Variable Documentation

30.1198.1.1 `var s1904_8c`

#### Initial value:

```
=
[
 ["s1904", "s1904_8c.html#a22edc87c279177ab6960b5d0c6e5c177", null],
 ["s1904", "s1904_8c.html#ae79dfd4451e99871b71bcc5dfc55b662", null]
]
```

Definition at line 1 of file s1904\_8c.js.



## 30.1199 doc/html/s1905\_8c.js File Reference

### Variables

- var [s1905\\_8c](#)

#### 30.1199.1 Variable Documentation

##### 30.1199.1.1 var s1905\_8c

###### Initial value:

```
=
[
 ["s1905", "s1905_8c.html#a2e876c8cc0b9f63c7b608f1918a91c7a", null],
 ["s1905", "s1905_8c.html#a4512aade7d64f0d49829f1b83acd38f", null]
]
```

Definition at line 1 of file s1905\_8c.js.

## 30.1200 doc/html/s1906\_8c.js File Reference

### Variables

- var [s1906\\_8c](#)

#### 30.1200.1 Variable Documentation

##### 30.1200.1.1 var s1906\_8c

###### Initial value:

```
=
[
 ["s1906", "s1906_8c.html#ab730d8827ee4522e1d927074c697dace", null],
 ["s1906", "s1906_8c.html#a4273aca5c105ed71d3f44203edd5ceda", null]
]
```

Definition at line 1 of file s1906\_8c.js.

## 30.1201 doc/html/s1907\_8c.js File Reference

### Variables

- var [s1907\\_8c](#)

### 30.1201.1 Variable Documentation

#### 30.1201.1.1 var s1907\_8c

**Initial value:**

```
=
[
 ["s1907", "s1907_8c.html#ad5de6531944574d48d9e515c4c3fca54", null],
 ["s1907", "s1907_8c.html#a367084d3c28c7becfd81091a328ce3fa", null]
]
```

Definition at line 1 of file s1907\_8c.js.

### 30.1202 doc/html/s1908\_8c.js File Reference

**Variables**

- var [s1908\\_8c](#)

#### 30.1202.1 Variable Documentation

##### 30.1202.1.1 var s1908\_8c

**Initial value:**

```
=
[
 ["MAX_SIZE", "s1908_8c.html#a0592dba56693fad79136250c11e5a7fe", null],
 ["s1908", "s1908_8c.html#af541a900f8e68873a8dfbcd7a6f9fd5f", null],
 ["s1908", "s1908_8c.html#a5dce781e9972f6cba17926ba985e4d43", null]
]
```

Definition at line 1 of file s1908\_8c.js.

### 30.1203 doc/html/s1909\_8c.js File Reference

**Variables**

- var [s1909\\_8c](#)

#### 30.1203.1 Variable Documentation

##### 30.1203.1.1 var s1909\_8c

**Initial value:**

```
=
[
 ["s1909", "s1909_8c.html#a2d322e6ef8dbf202db80442168dab7bc", null],
 ["s1909", "s1909_8c.html#a8b2c33cc07b3f5cb64618417ff26e849", null]
]
```

Definition at line 1 of file s1909\_8c.js.

## 30.1204 doc/html/s1910\_8c.js File Reference

### Variables

- var [s1910\\_8c](#)

#### 30.1204.1 Variable Documentation

##### 30.1204.1.1 var s1910\_8c

###### Initial value:

```
=
[
 ["s1910", "s1910_8c.html#a1fc65e678b865c0896c2312f436319e3", null],
 ["s1910", "s1910_8c.html#afda52b1548a546f017c5d80da9f7cf4f", null]
]
```

Definition at line 1 of file s1910\_8c.js.

## 30.1205 doc/html/s1911\_8c.js File Reference

### Variables

- var [s1911\\_8c](#)

#### 30.1205.1 Variable Documentation

##### 30.1205.1.1 var s1911\_8c

###### Initial value:

```
=
[
 ["s1911", "s1911_8c.html#a1a6fce5b8d91f01dd4ca17c9753b0583", null],
 ["s1911", "s1911_8c.html#ab57c25f9facb47928a2a4d446c5f57a", null]
]
```

Definition at line 1 of file s1911\_8c.js.

## 30.1206 doc/html/s1912\_8c.js File Reference

### Variables

- var [s1912\\_8c](#)

### 30.1206.1 Variable Documentation

#### 30.1206.1.1 var s1912\_8c

**Initial value:**

```
=
[
 ["s1912", "s1912_8c.html#ac5f70f28673ca19ef1420a422b7879db", null],
 ["fknotsProc", "s1912_8c.html#a693628c216123daeabc374eaf6826103", null],
 ["fparamProc", "s1912_8c.html#a340939cdef71c179f28742c34ace9588", null],
 ["s1912", "s1912_8c.html#a915b77d52378968aacedaf18d2de35c9", null]
]
```

Definition at line 1 of file s1912\_8c.js.

### 30.1207 doc/html/s1916\_8c.js File Reference

**Variables**

- var [s1916\\_8c](#)

#### 30.1207.1 Variable Documentation

##### 30.1207.1.1 var s1916\_8c

**Initial value:**

```
=
[
 ["s1916", "s1916_8c.html#a7a35efb8c803a604237dc80b9c95f302", null],
 ["s1916", "s1916_8c.html#a915b77d52378968aacedaf18d2de35c9", null]
]
```

Definition at line 1 of file s1916\_8c.js.

### 30.1208 doc/html/s1917\_8c.js File Reference

**Variables**

- var [s1917\\_8c](#)

#### 30.1208.1 Variable Documentation

##### 30.1208.1.1 var s1917\_8c

**Initial value:**

```
=
[
 ["s1917", "s1917_8c.html#a7e6afc77cb472bd70e59fd3d159c13bc", null],
 ["s1917", "s1917_8c.html#a13a76bbf8eb432fbef82413bd29f3eb8", null]
]
```

Definition at line 1 of file s1917\_8c.js.

## 30.1209 doc/html/s1918\_8c.js File Reference

### Variables

- var [s1918\\_8c](#)

### 30.1209.1 Variable Documentation

#### 30.1209.1.1 var s1918\_8c

##### Initial value:

```
=
[
 ["s1918", "s1918_8c.html#a79424fb2554c0554d6030d0f1ac2d3c7", null],
 ["s1918", "s1918_8c.html#a5dbb084e4a89ca7a791dbc35b06af2b9", null]
]
```

Definition at line 1 of file s1918\_8c.js.

## 30.1210 doc/html/s1919\_8c.js File Reference

### Variables

- var [s1919\\_8c](#)

### 30.1210.1 Variable Documentation

#### 30.1210.1.1 var s1919\_8c

##### Initial value:

```
=
[
 ["s1919", "s1919_8c.html#a844d9e6129aae459a1ae2a9821e8ec88", null],
 ["s1919", "s1919_8c.html#a677e7cfa6f36691e4b4e8d90ca2d55d9", null]
]
```

Definition at line 1 of file s1919\_8c.js.

## 30.1211 doc/html/s1920\_8c.js File Reference

### Variables

- var [s1920\\_8c](#)

### 30.1211.1 Variable Documentation

30.1211.1.1 `var s1920_8c`

**Initial value:**

```
=
[
 ["s1920", "s1920_8c.html#a5d5cf9b50e3af38bdb813c1feb031a28", null],
 ["s1920", "s1920_8c.html#a63ff26ca818677270088aec48e1a9b47", null]
]
```

Definition at line 1 of file s1920\_8c.js.

### 30.1212 doc/html/s1921\_8c.js File Reference

#### Variables

- `var s1921_8c`

### 30.1212.1 Variable Documentation

30.1212.1.1 `var s1921_8c`

**Initial value:**

```
=
[
 ["s1921", "s1921_8c.html#a54327164da028c304eb6cb8a35cd8d74", null],
 ["s1921", "s1921_8c.html#a9647f729d243840aa37af614dabe36dc", null]
]
```

Definition at line 1 of file s1921\_8c.js.

### 30.1213 doc/html/s1924\_8c.js File Reference

#### Variables

- `var s1924_8c`

### 30.1213.1 Variable Documentation

30.1213.1.1 `var s1924_8c`

**Initial value:**

```
=
[
 ["s1924", "s1924_8c.html#aef77ed666cdf4fee79b028b3835fa8cf", null],
 ["s1924", "s1924_8c.html#a99220dc4bd23bb1f9048c757b967a161", null]
]
```

Definition at line 1 of file s1924\_8c.js.

## 30.1214 doc/html/s1925\_8c.js File Reference

### Variables

- var [s1925\\_8c](#)

#### 30.1214.1 Variable Documentation

##### 30.1214.1.1 var s1925\_8c

###### Initial value:

```
=
[
 ["s1925", "s1925_8c.html#a845d6c868ae305a35bbad07a696574de", null],
 ["s1925_MAX_ARRAY_SIZE", "s1925_8c.html#acfd9b0888b2fa7c92930501e996ab1fe", null],
 ["s1925", "s1925_8c.html#ada46ce038fc3e28ea98dc7b6bc9fe479", null]
]
```

Definition at line 1 of file s1925\_8c.js.

## 30.1215 doc/html/s1926\_8c.js File Reference

### Variables

- var [s1926\\_8c](#)

#### 30.1215.1 Variable Documentation

##### 30.1215.1.1 var s1926\_8c

###### Initial value:

```
=
[
 ["s1926", "s1926_8c.html#a98c09e45d22650de337f5ae577626964", null],
 ["s1926", "s1926_8c.html#a90330bd1c15720281c38b4dcf1d492d8", null]
]
```

Definition at line 1 of file s1926\_8c.js.

## 30.1216 doc/html/s1927\_8c.js File Reference

### Variables

- var [s1927\\_8c](#)

### 30.1216.1 Variable Documentation

#### 30.1216.1.1 var s1927\_8c

**Initial value:**

```
=
[
 ["s1927", "s1927_8c.html#abcd7955ad43cc6de94ad1cfe8192d65e", null],
 ["s1927", "s1927_8c.html#a31f68f728089445762ddf6ef07f93eid", null]
]
```

Definition at line 1 of file s1927\_8c.js.

### 30.1217 doc/html/s1930\_8c.js File Reference

**Variables**

- var [s1930\\_8c](#)

#### 30.1217.1 Variable Documentation

##### 30.1217.1.1 var s1930\_8c

**Initial value:**

```
=
[
 ["s1930", "s1930_8c.html#acb112f1405be2c5eb8daa3ad33bbe29f", null],
 ["s1930", "s1930_8c.html#a41903fa4d83fe83e3766f084c9294ae1", null]
]
```

Definition at line 1 of file s1930\_8c.js.

### 30.1218 doc/html/s1931\_8c.js File Reference

**Variables**

- var [s1931\\_8c](#)

#### 30.1218.1 Variable Documentation

##### 30.1218.1.1 var s1931\_8c

**Initial value:**

```
=
[
 ["s1931", "s1931_8c.html#ab1ed3c793fe8c28a35302f79f926aa06", null],
 ["s1931", "s1931_8c.html#ac2e9c821c6cce5a7d35b50c38b246405", null]
]
```

Definition at line 1 of file s1931\_8c.js.



## 30.1219 doc/html/s1931unit\_8c.js File Reference

### Variables

- var [s1931unit\\_8c](#)

#### 30.1219.1 Variable Documentation

##### 30.1219.1.1 var s1931unit\_8c

###### Initial value:

```
=
[
 ["S1931UNIT", "s1931unit_8c.html#a16d3bae4164afd017621b41abdc7c9a8", null],
 ["s1931unit", "s1931unit_8c.html#aeb9b9d99b0371a1c6d325f2d68d25b94", null]
]
```

Definition at line 1 of file s1931unit\_8c.js.

## 30.1220 doc/html/s1932\_8c.js File Reference

### Variables

- var [s1932\\_8c](#)

#### 30.1220.1 Variable Documentation

##### 30.1220.1.1 var s1932\_8c

###### Initial value:

```
=
[
 ["S1932", "s1932_8c.html#aa78b655bbefa3057f0d5b7c5a400d4c3", null],
 ["s1932", "s1932_8c.html#a71254496de0f20b31d3036851515987f", null]
]
```

Definition at line 1 of file s1932\_8c.js.

## 30.1221 doc/html/s1933\_8c.js File Reference

### Variables

- var [s1933\\_8c](#)

### 30.1221.1 Variable Documentation

#### 30.1221.1.1 var s1933\_8c

**Initial value:**

```
=
[
 ["s1933", "s1933_8c.html#ae8767619c8972beb0325749b5bdc723f", null],
 ["s1933", "s1933_8c.html#a312c97e6e061b3dd80f246c9c4b607cf", null]
]
```

Definition at line 1 of file s1933\_8c.js.

### 30.1222 doc/html/s1934\_8c.js File Reference

**Variables**

- var [s1934\\_8c](#)

#### 30.1222.1 Variable Documentation

##### 30.1222.1.1 var s1934\_8c

**Initial value:**

```
=
[
 ["s1934", "s1934_8c.html#a07b4d6fd48c5be386af7a9d2c7de66ea", null],
 ["s1934", "s1934_8c.html#aae2cf2dd30e9de48400971d30610584f", null]
]
```

Definition at line 1 of file s1934\_8c.js.

### 30.1223 doc/html/s1935\_8c.js File Reference

**Variables**

- var [s1935\\_8c](#)

#### 30.1223.1 Variable Documentation

##### 30.1223.1.1 var s1935\_8c

**Initial value:**

```
=
[
 ["s1935", "s1935_8c.html#aa854c23ad8bcc3d4111eebb581a2f70e", null],
 ["s1935", "s1935_8c.html#a3f1fa57d175d00c2b405cf015f7a3279", null]
]
```

Definition at line 1 of file s1935\_8c.js.

## 30.1224 doc/html/s1936\_8c.js File Reference

### Variables

- var [s1936\\_8c](#)

### 30.1224.1 Variable Documentation

#### 30.1224.1.1 var s1936\_8c

##### Initial value:

```
=
[
 ["MAX_SIZE", "s1936_8c.html#a0592dba56693fad79136250c11e5a7fe", null],
 ["S1936", "s1936_8c.html#a80aa31fc9052aefed38fb9691e503851", null],
 ["s1936", "s1936_8c.html#ab4b5ae59dcf3e58f195599783badf046", null]
]
```

Definition at line 1 of file s1936\_8c.js.

## 30.1225 doc/html/s1937\_8c.js File Reference

### Variables

- var [s1937\\_8c](#)

### 30.1225.1 Variable Documentation

#### 30.1225.1.1 var s1937\_8c

##### Initial value:

```
=
[
 ["S1937", "s1937_8c.html#a7759b2cab0af26b3ec412eaf72dd8d1d", null],
 ["s1937", "s1937_8c.html#a7368f3e7a1856c40a3db4bc3dcef8a99", null]
]
```

Definition at line 1 of file s1937\_8c.js.

## 30.1226 doc/html/s1938\_8c.js File Reference

### Variables

- var [s1938\\_8c](#)

### 30.1226.1 Variable Documentation

#### 30.1226.1.1 var s1938\_8c

**Initial value:**

```
=
[
 ["MAX_SIZE", "s1938_8c.html#a0592dba56693fad79136250c11e5a7fe", null],
 ["S1938", "s1938_8c.html#a49fe8ad56fb9551be254a3048ab6ff1b", null],
 ["s1938", "s1938_8c.html#aad8935f0cbf156bd064e2c7185d7947e", null]
]
```

Definition at line 1 of file s1938\_8c.js.

### 30.1227 doc/html/s1940\_8c.js File Reference

**Variables**

- var [s1940\\_8c](#)

#### 30.1227.1 Variable Documentation

##### 30.1227.1.1 var s1940\_8c

**Initial value:**

```
=
[
 ["S1940", "s1940_8c.html#a236b66f678433f488b2d94f298d4b908", null],
 ["s1940", "s1940_8c.html#a43246d0dcf0fdd4096230f5970f3c113", null]
]
```

Definition at line 1 of file s1940\_8c.js.

### 30.1228 doc/html/s1941\_8c.js File Reference

**Variables**

- var [s1941\\_8c](#)

#### 30.1228.1 Variable Documentation

##### 30.1228.1.1 var s1941\_8c

**Initial value:**

```
=
[
 ["S1941", "s1941_8c.html#ab41dfe3b05343e734f70442f8706be3f", null],
 ["s1941", "s1941_8c.html#a53b9c6b2f9ff6f52ed3116bb6306bb5a", null]
]
```

Definition at line 1 of file s1941\_8c.js.

## 30.1229 doc/html/s1942\_8c.js File Reference

### Variables

- var [s1942\\_8c](#)

### 30.1229.1 Variable Documentation

#### 30.1229.1.1 var s1942\_8c

##### Initial value:

```
=
[
 ["s1942", "s1942_8c.html#ac0fba87501fcelccd921c91414abad36", null],
 ["s1942", "s1942_8c.html#a0028158729ab71a3da6df69951854a8e", null]
]
```

Definition at line 1 of file s1942\_8c.js.

## 30.1230 doc/html/s1943\_8c.js File Reference

### Variables

- var [s1943\\_8c](#)

### 30.1230.1 Variable Documentation

#### 30.1230.1.1 var s1943\_8c

##### Initial value:

```
=
[
 ["s1943", "s1943_8c.html#a8c817b948930cbc3c60d3a4593008e74", null],
 ["s1943", "s1943_8c.html#a4117475641c9e4fb6a48d8bc619598e9", null]
]
```

Definition at line 1 of file s1943\_8c.js.

## 30.1231 doc/html/s1944\_8c.js File Reference

### Variables

- var [s1944\\_8c](#)

### 30.1231.1 Variable Documentation

30.1231.1.1 var s1944\_8c

**Initial value:**

```
=
[
 ["s1944", "s1944_8c.html#a8926b14f3f594eb5e7bb4efedac1b2ba", null],
 ["s1944", "s1944_8c.html#a156c6bd691d787b34eee04c95530c914", null]
]
```

Definition at line 1 of file s1944\_8c.js.

### 30.1232 doc/html/s1945\_8c.js File Reference

**Variables**

- var [s1945\\_8c](#)

#### 30.1232.1 Variable Documentation

30.1232.1.1 var s1945\_8c

**Initial value:**

```
=
[
 ["s1945", "s1945_8c.html#a2dc881db3c56d7373bbff803454d55d0", null],
 ["s1945", "s1945_8c.html#a038c2a1b715688f6f69607b526de1e0c", null]
]
```

Definition at line 1 of file s1945\_8c.js.

### 30.1233 doc/html/s1946\_8c.js File Reference

**Variables**

- var [s1946\\_8c](#)

#### 30.1233.1 Variable Documentation

30.1233.1.1 var s1946\_8c

**Initial value:**

```
=
[
 ["s1946", "s1946_8c.html#ac7e1c8f9e383a3df8b0b0e21308718b2", null],
 ["s1946", "s1946_8c.html#aa93838095013b9516075ef52aaa5f7ee", null]
]
```

Definition at line 1 of file s1946\_8c.js.

## 30.1234 doc/html/s1947\_8c.js File Reference

### Variables

- var [s1947\\_8c](#)

### 30.1234.1 Variable Documentation

#### 30.1234.1.1 var s1947\_8c

##### Initial value:

```
=
[
 ["s1947", "s1947_8c.html#a9f488e8e5a988c9fd1023bbae1bf3c06", null],
 ["s1947", "s1947_8c.html#ad3699f35b3badf3776bb073f9ce151b7", null]
]
```

Definition at line 1 of file s1947\_8c.js.

## 30.1235 doc/html/s1948\_8c.js File Reference

### Variables

- var [s1948\\_8c](#)

### 30.1235.1 Variable Documentation

#### 30.1235.1.1 var s1948\_8c

##### Initial value:

```
=
[
 ["s1948", "s1948_8c.html#a41df2968dc07d9f2c214f40b8db583d6", null],
 ["s1948", "s1948_8c.html#a975a2cf7881b34c991b79a8975594f71", null]
]
```

Definition at line 1 of file s1948\_8c.js.

## 30.1236 doc/html/s1949\_8c.js File Reference

### Variables

- var [s1949\\_8c](#)

### 30.1236.1 Variable Documentation

#### 30.1236.1.1 var s1949\_8c

**Initial value:**

```
=
[
 ["s1949", "s1949_8c.html#a542401b20028844ccab4298e4a27c305", null],
 ["s1949", "s1949_8c.html#a0ee6676434ce6d8853d2b9dc5f4b4f86", null]
]
```

Definition at line 1 of file s1949\_8c.js.

### 30.1237 doc/html/s1950\_8c.js File Reference

**Variables**

- var [s1950\\_8c](#)

#### 30.1237.1 Variable Documentation

##### 30.1237.1.1 var s1950\_8c

**Initial value:**

```
=
[
 ["s1950", "s1950_8c.html#a01fdb1e92896634d2667c0331c4c0ba4", null],
 ["s1950", "s1950_8c.html#acbdd8aa7fb5ae03dd9159990be8088ea", null]
]
```

Definition at line 1 of file s1950\_8c.js.

### 30.1238 doc/html/s1951\_8c.js File Reference

**Variables**

- var [s1951\\_8c](#)

#### 30.1238.1 Variable Documentation

##### 30.1238.1.1 var s1951\_8c

**Initial value:**

```
=
[
 ["s1951", "s1951_8c.html#af2293574eb448e576e1ea6e07f140dfa", null],
 ["s1951", "s1951_8c.html#ab2e8aa2d4fb9c13806fcfc38af75f7d6", null]
]
```

Definition at line 1 of file s1951\_8c.js.



## 30.1239 doc/html/s1953\_8c.js File Reference

### Variables

- var [s1953\\_8c](#)

### 30.1239.1 Variable Documentation

#### 30.1239.1.1 var s1953\_8c

##### Initial value:

```
=
[
 ["s1953", "s1953_8c.html#ac2b895632b9997427ec3aa1b6b86df57", null],
 ["s1953", "s1953_8c.html#a412e8c2338296109e645a10e6f995cea", null]
]
```

Definition at line 1 of file s1953\_8c.js.

## 30.1240 doc/html/s1954\_8c.js File Reference

### Variables

- var [s1954\\_8c](#)

### 30.1240.1 Variable Documentation

#### 30.1240.1.1 var s1954\_8c

##### Initial value:

```
=
[
 ["s1954", "s1954_8c.html#a31623ec0e969382fad7ce4c8a1598ba8", null],
 ["s1954", "s1954_8c.html#a19f2837c42e8d3a244d9795ab574a0dd", null]
]
```

Definition at line 1 of file s1954\_8c.js.

## 30.1241 doc/html/s1955\_8c.js File Reference

### Variables

- var [s1955\\_8c](#)

### 30.1241.1 Variable Documentation

#### 30.1241.1.1 var s1955\_8c

**Initial value:**

```
=
[
 ["s1955", "s1955_8c.html#aa0d94dd7e714cbd474a6f6afd1c8bd04", null],
 ["s1955", "s1955_8c.html#a7ce774d20fcb38d980c57b2c58fc553a", null]
]
```

Definition at line 1 of file s1955\_8c.js.

### 30.1242 doc/html/s1956\_8c.js File Reference

**Variables**

- var [s1956\\_8c](#)

#### 30.1242.1 Variable Documentation

##### 30.1242.1.1 var s1956\_8c

**Initial value:**

```
=
[
 ["s1956", "s1956_8c.html#a5b26eae30f31f67adc0c11c860a36a35", null],
 ["s1956", "s1956_8c.html#a126fd1ed9a2e8d51cef04caed9efc463", null]
]
```

Definition at line 1 of file s1956\_8c.js.

### 30.1243 doc/html/s1957\_8c.js File Reference

**Variables**

- var [s1957\\_8c](#)

#### 30.1243.1 Variable Documentation

##### 30.1243.1.1 var s1957\_8c

**Initial value:**

```
=
[
 ["s1957", "s1957_8c.html#a1aeead56c9dad3efff4d95e5270a770f2", null],
 ["s1957", "s1957_8c.html#a0d9c97be58e612bf36a7e5a8770ca4b3", null]
]
```

Definition at line 1 of file s1957\_8c.js.

## 30.1244 doc/html/s1958\_8c.js File Reference

### Variables

- var [s1958\\_8c](#)

#### 30.1244.1 Variable Documentation

##### 30.1244.1.1 var s1958\_8c

###### Initial value:

```
=
[
 ["s1958", "s1958_8c.html#aba55281b345a84317f767942097c9e88", null],
 ["s1958", "s1958_8c.html#aFb09d9e55f0637d6475cb34184159610", null]
]
```

Definition at line 1 of file s1958\_8c.js.

## 30.1245 doc/html/s1959\_8c.js File Reference

### Variables

- var [s1959\\_8c](#)

#### 30.1245.1 Variable Documentation

##### 30.1245.1.1 var s1959\_8c

###### Initial value:

```
=
[
 ["s1959", "s1959_8c.html#ae79aad26cb69ae0daf58161997bd8d39", null],
 ["s1959", "s1959_8c.html#ad6f564a465cfcb1753ec48623e467ad1", null]
]
```

Definition at line 1 of file s1959\_8c.js.

## 30.1246 doc/html/s1960\_8c.js File Reference

### Variables

- var [s1960\\_8c](#)

### 30.1246.1 Variable Documentation

#### 30.1246.1.1 var s1960\_8c

**Initial value:**

```
=
[
 ["s1960", "s1960_8c.html#a93650ba21515281b5786de1d2db835c3", null],
 ["s1960", "s1960_8c.html#a12ec960b2fa694469af65052b3e78f37", null]
]
```

Definition at line 1 of file s1960\_8c.js.

### 30.1247 doc/html/s1961\_8c.js File Reference

**Variables**

- var [s1961\\_8c](#)

#### 30.1247.1 Variable Documentation

##### 30.1247.1.1 var s1961\_8c

**Initial value:**

```
=
[
 ["s1961", "s1961_8c.html#ab74fbad2da6426ff6091c94a1cbdfb93", null],
 ["s1961", "s1961_8c.html#abdad5f166043bdce3e93bd0cb8a573c1", null]
]
```

Definition at line 1 of file s1961\_8c.js.

### 30.1248 doc/html/s1962\_8c.js File Reference

**Variables**

- var [s1962\\_8c](#)

#### 30.1248.1 Variable Documentation

##### 30.1248.1.1 var s1962\_8c

**Initial value:**

```
=
[
 ["s1962", "s1962_8c.html#abc343df8b5785df0d1cd2be611eee74e", null],
 ["s1962", "s1962_8c.html#aed83f404d2099e90e9028bc8ce45eee8", null]
]
```

Definition at line 1 of file s1962\_8c.js.

## 30.1249 doc/html/s1963\_8c.js File Reference

### Variables

- var [s1963\\_8c](#)

### 30.1249.1 Variable Documentation

#### 30.1249.1.1 var s1963\_8c

##### Initial value:

```
=
[
 ["s1963", "s1963_8c.html#a8488e865634f22cbf9cd2b50e8675117", null],
 ["s1963", "s1963_8c.html#abd9cfe765a45d6afff26cda2bf06c118", null]
]
```

Definition at line 1 of file s1963\_8c.js.

## 30.1250 doc/html/s1965\_8c.js File Reference

### Variables

- var [s1965\\_8c](#)

### 30.1250.1 Variable Documentation

#### 30.1250.1.1 var s1965\_8c

##### Initial value:

```
=
[
 ["s1965", "s1965_8c.html#a74ae6d370384bbd305c566cf538665ea", null],
 ["s1965", "s1965_8c.html#a945dc87c87a97fbcc4b4b59b057955", null]
]
```

Definition at line 1 of file s1965\_8c.js.

## 30.1251 doc/html/s1966\_8c.js File Reference

### Variables

- var [s1966\\_8c](#)

### 30.1251.1 Variable Documentation

30.1251.1.1 `var s1966_8c`

#### Initial value:

```
=
[
 ["s1966", "s1966_8c.html#ae95daceceblce20d36dbccf90016e9d2", null],
 ["s1966", "s1966_8c.html#aeeeca6edae7ecfc3c1b8d3f1e0c9592a9", null]
]
```

Definition at line 1 of file s1966\_8c.js.

### 30.1252 doc/html/s1967\_8c.js File Reference

#### Variables

- `var s1967_8c`

### 30.1252.1 Variable Documentation

30.1252.1.1 `var s1967_8c`

#### Initial value:

```
=
[
 ["s1967", "s1967_8c.html#a390235277c65264616727dac31294c4e", null],
 ["s1967", "s1967_8c.html#ac6330023975d6d90f1a171d82fd8c41a", null]
]
```

Definition at line 1 of file s1967\_8c.js.

### 30.1253 doc/html/s1968\_8c.js File Reference

#### Variables

- `var s1968_8c`

### 30.1253.1 Variable Documentation

30.1253.1.1 `var s1968_8c`

#### Initial value:

```
=
[
 ["s1968", "s1968_8c.html#acb61af634a88be2affdfbe109b0dce40", null],
 ["s1968", "s1968_8c.html#adb78baf33c957adbfe38271e17389d25", null]
]
```

Definition at line 1 of file s1968\_8c.js.

## 30.1254 doc/html/s1986\_8c.js File Reference

### Variables

- var [s1986\\_8c](#)

#### 30.1254.1 Variable Documentation

##### 30.1254.1.1 var s1986\_8c

###### Initial value:

```
=
[
 ["s1986", "s1986_8c.html#ac86ce654b6e6d4c3a2dec233b1581262", null],
 ["s1986", "s1986_8c.html#a3bb2556dc60f0a820fc2ba72f0b3d2a2", null]
]
```

Definition at line 1 of file s1986\_8c.js.

## 30.1255 doc/html/s1987\_8c.js File Reference

### Variables

- var [s1987\\_8c](#)

#### 30.1255.1 Variable Documentation

##### 30.1255.1.1 var s1987\_8c

###### Initial value:

```
=
[
 ["s1987", "s1987_8c.html#af422f292633f9ab231ead78ed736da6c", null],
 ["s1987", "s1987_8c.html#a5b15e768d153a1bc5366a729f573d935", null]
]
```

Definition at line 1 of file s1987\_8c.js.

## 30.1256 doc/html/s1988\_8c.js File Reference

### Variables

- var [s1988\\_8c](#)

### 30.1256.1 Variable Documentation

#### 30.1256.1.1 var s1988\_8c

**Initial value:**

```
=
[
 ["s1988", "s1988_8c.html#a6273281b0ee9321b6ce8ceb4f597bf58", null],
 ["s1988", "s1988_8c.html#ac90891f06d1f165bf4b60d7344d8493b", null]
]
```

Definition at line 1 of file s1988\_8c.js.

### 30.1257 doc/html/s1989\_8c.js File Reference

**Variables**

- var [s1989\\_8c](#)

#### 30.1257.1 Variable Documentation

##### 30.1257.1.1 var s1989\_8c

**Initial value:**

```
=
[
 ["s1989", "s1989_8c.html#a02b33a06f6c825233be52bfc43ea5318", null],
 ["s1989", "s1989_8c.html#a7a8021f5e5762b0ad6a24720df9e1567", null]
]
```

Definition at line 1 of file s1989\_8c.js.

### 30.1258 doc/html/s1990\_8c.js File Reference

**Variables**

- var [s1990\\_8c](#)

#### 30.1258.1 Variable Documentation

##### 30.1258.1.1 var s1990\_8c

**Initial value:**

```
=
[
 ["s1990", "s1990_8c.html#adde63d686829c5900f0879df204db2c7", null],
 ["s1990", "s1990_8c.html#ab28572ba520d26e749bf12b76ea325b8", null]
]
```

Definition at line 1 of file s1990\_8c.js.



## 30.1259 doc/html/s1991\_8c.js File Reference

### Variables

- var [s1991\\_8c](#)

#### 30.1259.1 Variable Documentation

##### 30.1259.1.1 var s1991\_8c

###### Initial value:

```
=
[
 ["s1991", "s1991_8c.html#acb6f458aec338edb12620ab8853cb781", null],
 ["s1991", "s1991_8c.html#a0923d7923d3369bba81c2340176319cf", null]
]
```

Definition at line 1 of file s1991\_8c.js.

## 30.1260 doc/html/s1992\_8c.js File Reference

### Variables

- var [s1992\\_8c](#)

#### 30.1260.1 Variable Documentation

##### 30.1260.1.1 var s1992\_8c

###### Initial value:

```
=
[
 ["s1992", "s1992_8c.html#a834f6a97dee6c2ac119a9dc0f274a2d3", null],
 ["s1992", "s1992_8c.html#a7a5bd51c6f1a033735d40e78c88f139b", null],
 ["s1992cu", "s1992_8c.html#a100175ae477544966a4cc3ce2f0488d8", null],
 ["s1992su", "s1992_8c.html#af1a99e9bc10d1a6873c738964e8aa633", null]
]
```

Definition at line 1 of file s1992\_8c.js.

## 30.1261 doc/html/s1993\_8c.js File Reference

### Variables

- var [s1993\\_8c](#)

### 30.1261.1 Variable Documentation

#### 30.1261.1.1 var s1993\_8c

**Initial value:**

```
=
[
 ["s1993", "s1993_8c.html#a8bc6b3835dfc5dbf39924c9949e07e49", null],
 ["s1993", "s1993_8c.html#a6beb0050a071d1a8397003193e7f5b36", null]
]
```

Definition at line 1 of file s1993\_8c.js.

### 30.1262 doc/html/s1994\_8c.js File Reference

**Variables**

- var [s1994\\_8c](#)

#### 30.1262.1 Variable Documentation

##### 30.1262.1.1 var s1994\_8c

**Initial value:**

```
=
[
 ["s1994", "s1994_8c.html#a80674b1ab4412cf5aa6ed04c39fca7d9", null],
 ["s1994", "s1994_8c.html#a2c4581b9571930c0d02a1e694629bde9", null]
]
```

Definition at line 1 of file s1994\_8c.js.

### 30.1263 doc/html/s2500\_8c.js File Reference

**Variables**

- var [s2500\\_8c](#)

#### 30.1263.1 Variable Documentation

##### 30.1263.1.1 var s2500\_8c

**Initial value:**

```
=
[
 ["s2500", "s2500_8c.html#a4d7d2275501950766e9e645ed24fdec3", null],
 ["s2500", "s2500_8c.html#a99b29fb11524f2398e40dcadea1b3ac7", null]
]
```

Definition at line 1 of file s2500\_8c.js.

## 30.1264 doc/html/s2501\_8c.js File Reference

### Variables

- var [s2501\\_8c](#)

#### 30.1264.1 Variable Documentation

##### 30.1264.1.1 var s2501\_8c

###### Initial value:

```
=
[
 ["s2501", "s2501_8c.html#a142dfb5c25ae4e9f2bd468bc52f766d9", null],
 ["s2501", "s2501_8c.html#a85ed6a2755f92d22249b8b7c621d21a5", null]
]
```

Definition at line 1 of file s2501\_8c.js.

## 30.1265 doc/html/s2502\_8c.js File Reference

### Variables

- var [s2502\\_8c](#)

#### 30.1265.1 Variable Documentation

##### 30.1265.1.1 var s2502\_8c

###### Initial value:

```
=
[
 ["s2502", "s2502_8c.html#a9b31bb52ea63cc1e0e28756a22a0e5b1", null],
 ["s2502", "s2502_8c.html#a1803d49d8babc9fdf28517055664b5d5", null]
]
```

Definition at line 1 of file s2502\_8c.js.

## 30.1266 doc/html/s2503\_8c.js File Reference

### Variables

- var [s2503\\_8c](#)

### 30.1266.1 Variable Documentation

30.1266.1.1 `var s2503_8c`

**Initial value:**

```
=
[
 ["s2503", "s2503_8c.html#afd21d17ff576ad35a27497e21a883c9c", null],
 ["s2503", "s2503_8c.html#a15df620ee6a3afaa9abbe2793a9406ec", null]
]
```

Definition at line 1 of file s2503\_8c.js.

### 30.1267 doc/html/s2504\_8c.js File Reference

**Variables**

- `var s2504_8c`

### 30.1267.1 Variable Documentation

30.1267.1.1 `var s2504_8c`

**Initial value:**

```
=
[
 ["s2504", "s2504_8c.html#a473a72ee34c168beee444b1d5e8c3f62", null],
 ["s2504", "s2504_8c.html#a3d35c27348369cdf129a6302780084e7", null]
]
```

Definition at line 1 of file s2504\_8c.js.

### 30.1268 doc/html/s2505\_8c.js File Reference

**Variables**

- `var s2505_8c`

### 30.1268.1 Variable Documentation

30.1268.1.1 `var s2505_8c`

**Initial value:**

```
=
[
 ["s2505", "s2505_8c.html#a71b422f405b7bedd0f58fe67b301dc46", null],
 ["s2505", "s2505_8c.html#af304c78340c4fd2a090ba19abedd626d", null]
]
```

Definition at line 1 of file s2505\_8c.js.

## 30.1269 doc/html/s2506\_8c.js File Reference

### Variables

- var [s2506\\_8c](#)

#### 30.1269.1 Variable Documentation

##### 30.1269.1.1 var s2506\_8c

###### Initial value:

```
=
[
 ["s2506", "s2506_8c.html#a9e75a7847ba99a656ef30f53cda9fb5a", null],
 ["s2506", "s2506_8c.html#abf9d9e5affbdf7cc8a300916842b63b1", null]
]
```

Definition at line 1 of file s2506\_8c.js.

## 30.1270 doc/html/s2507\_8c.js File Reference

### Variables

- var [s2507\\_8c](#)

#### 30.1270.1 Variable Documentation

##### 30.1270.1.1 var s2507\_8c

###### Initial value:

```
=
[
 ["s2507", "s2507_8c.html#a70cac93ddedc634a39cd987a69142928", null],
 ["s2507", "s2507_8c.html#a0057655f02b71f4f3a13adbf2d76332e", null]
]
```

Definition at line 1 of file s2507\_8c.js.

## 30.1271 doc/html/s2508\_8c.js File Reference

### Variables

- var [s2508\\_8c](#)

### 30.1271.1 Variable Documentation

30.1271.1.1 `var s2508_8c`

**Initial value:**

```
=
[
 ["s2508", "s2508_8c.html#a2a670524c161e63723338d6bc73e4eef", null],
 ["s2508", "s2508_8c.html#ab1798cd3dae4aec076fd9717a1d7cb7f", null]
]
```

Definition at line 1 of file s2508\_8c.js.

### 30.1272 doc/html/s2509\_8c.js File Reference

#### Variables

- `var s2509_8c`

### 30.1272.1 Variable Documentation

30.1272.1.1 `var s2509_8c`

**Initial value:**

```
=
[
 ["s2509", "s2509_8c.html#a421067631ac948901d41bc58127f8c6a", null],
 ["s2509", "s2509_8c.html#a380286a79bef03fddd98baf33e6845f2", null]
]
```

Definition at line 1 of file s2509\_8c.js.

### 30.1273 doc/html/s2510\_8c.js File Reference

#### Variables

- `var s2510_8c`

### 30.1273.1 Variable Documentation

30.1273.1.1 `var s2510_8c`

**Initial value:**

```
=
[
 ["s2510", "s2510_8c.html#a3ffee703742b34b2fe2e11f035132467", null],
 ["s2510", "s2510_8c.html#aa3ec3764187c063a66d4832e2b2f38b5", null]
]
```

Definition at line 1 of file s2510\_8c.js.

## 30.1274 doc/html/s2511\_8c.js File Reference

### Variables

- var [s2511\\_8c](#)

#### 30.1274.1 Variable Documentation

##### 30.1274.1.1 var s2511\_8c

###### Initial value:

```
=
[
 ["s2511", "s2511_8c.html#a8f9afde3c256398ee36e446598a63c", null],
 ["s2511", "s2511_8c.html#a8f1c6305a53616371d111f8529697ad5", null]
]
```

Definition at line 1 of file s2511\_8c.js.

## 30.1275 doc/html/s2512\_8c.js File Reference

### Variables

- var [s2512\\_8c](#)

#### 30.1275.1 Variable Documentation

##### 30.1275.1.1 var s2512\_8c

###### Initial value:

```
=
[
 ["s2512", "s2512_8c.html#a72441239d3de4ee8ba3fb2a03d9cc7e5", null],
 ["s2512", "s2512_8c.html#a4dc2649dd1c1e4fa081f6bcfcefd50e0", null]
]
```

Definition at line 1 of file s2512\_8c.js.

## 30.1276 doc/html/s2513\_8c.js File Reference

### Variables

- var [s2513\\_8c](#)

### 30.1276.1 Variable Documentation

#### 30.1276.1.1 var s2513\_8c

**Initial value:**

```
=
[
 ["s2513", "s2513_8c.html#a487ad0e3da62b489f2dceca111ab677c", null],
 ["s2513", "s2513_8c.html#aabfa0c2c0afa6b10d54bfc5977d3f23", null]
]
```

Definition at line 1 of file s2513\_8c.js.

### 30.1277 doc/html/s2514\_8c.js File Reference

**Variables**

- var [s2514\\_8c](#)

#### 30.1277.1 Variable Documentation

##### 30.1277.1.1 var s2514\_8c

**Initial value:**

```
=
[
 ["s2514", "s2514_8c.html#ac95c36c7fc0ccabc4c747c8bc8c528ad", null],
 ["s2514", "s2514_8c.html#a08c58aadf1cefe9787d5b7f8ea20453c", null]
]
```

Definition at line 1 of file s2514\_8c.js.

### 30.1278 doc/html/s2515\_8c.js File Reference

**Variables**

- var [s2515\\_8c](#)

#### 30.1278.1 Variable Documentation

##### 30.1278.1.1 var s2515\_8c

**Initial value:**

```
=
[
 ["s2515", "s2515_8c.html#a5cda4f8382a8baf0b61b508a8dcb0e92", null],
 ["s2515", "s2515_8c.html#a605dd31bd838d07c2f4ce9cbdb69344f", null]
]
```

Definition at line 1 of file s2515\_8c.js.



## 30.1279 doc/html/s2516\_8c.js File Reference

### Variables

- var [s2516\\_8c](#)

### 30.1279.1 Variable Documentation

#### 30.1279.1.1 var s2516\_8c

##### Initial value:

```
=
[
 ["s2516", "s2516_8c.html#aad40157aeef970f1afa238a2e142bd60", null],
 ["s2516", "s2516_8c.html#a5a69ef752de55c44379c250ae30e2b8f", null]
]
```

Definition at line 1 of file s2516\_8c.js.

## 30.1280 doc/html/s2532\_8c.js File Reference

### Variables

- var [s2532\\_8c](#)

### 30.1280.1 Variable Documentation

#### 30.1280.1.1 var s2532\_8c

##### Initial value:

```
=
[
 ["s2532", "s2532_8c.html#a0222a12dbb6061ef233bcfacfc94810c", null],
 ["s2532", "s2532_8c.html#a6757deb4f32f78a8499f7d0d92ccd32e", null]
]
```

Definition at line 1 of file s2532\_8c.js.

## 30.1281 doc/html/s2533\_8c.js File Reference

### Variables

- var [s2533\\_8c](#)

### 30.1281.1 Variable Documentation

30.1281.1.1 `var s2533_8c`

**Initial value:**

```
=
[
 ["s2533", "s2533_8c.html#ae9cf4d238e9b4273f048c481fea7889e", null],
 ["s2533", "s2533_8c.html#aa0240d0e738aa20a2cc06b7b2fa70654", null]
]
```

Definition at line 1 of file s2533\_8c.js.

### 30.1282 doc/html/s2534\_8c.js File Reference

**Variables**

- `var s2534_8c`

#### 30.1282.1 Variable Documentation

30.1282.1.1 `var s2534_8c`

**Initial value:**

```
=
[
 ["s2534", "s2534_8c.html#a8d03810260fea89852580810b106ff59", null],
 ["s2534", "s2534_8c.html#a5bdf9fa86ed953814edfd028bbe437ec", null]
]
```

Definition at line 1 of file s2534\_8c.js.

### 30.1283 doc/html/s2535\_8c.js File Reference

**Variables**

- `var s2535_8c`

#### 30.1283.1 Variable Documentation

30.1283.1.1 `var s2535_8c`

**Initial value:**

```
=
[
 ["s2535", "s2535_8c.html#ae27db3010a36ca4111d3bd8b24e35357", null],
 ["s2535", "s2535_8c.html#a76680a535ea63087af69ab022cad9639", null]
]
```

Definition at line 1 of file s2535\_8c.js.

## 30.1284 doc/html/s2536\_8c.js File Reference

### Variables

- var [s2536\\_8c](#)

#### 30.1284.1 Variable Documentation

##### 30.1284.1.1 var s2536\_8c

###### Initial value:

```
=
[
 ["s2536", "s2536_8c.html#a5ed8c67af3816fa5e69e171e049c8795", null],
 ["s2536", "s2536_8c.html#a3373ec59ea4c62106d0cb248d8d0fac8", null]
]
```

Definition at line 1 of file s2536\_8c.js.

## 30.1285 doc/html/s2540\_8c.js File Reference

### Variables

- var [s2540\\_8c](#)

#### 30.1285.1 Variable Documentation

##### 30.1285.1.1 var s2540\_8c

###### Initial value:

```
=
[
 ["s2540", "s2540_8c.html#aeb5a6af0cd7b651179d7028ca12738ac", null],
 ["s2540", "s2540_8c.html#a13fa46720c3ff6fb9836404d099e7d07", null]
]
```

Definition at line 1 of file s2540\_8c.js.

## 30.1286 doc/html/s2541\_8c.js File Reference

### Variables

- var [s2541\\_8c](#)

### 30.1286.1 Variable Documentation

#### 30.1286.1.1 var s2541\_8c

**Initial value:**

```
=
[
 ["s2541", "s2541_8c.html#aac8bfde742d1b94e7ba7fae5d6788dab", null],
 ["s2541", "s2541_8c.html#a237e411ceb881aac87b94fbde03a1d95", null]
]
```

Definition at line 1 of file s2541\_8c.js.

### 30.1287 doc/html/s2542\_8c.js File Reference

**Variables**

- var [s2542\\_8c](#)

#### 30.1287.1 Variable Documentation

##### 30.1287.1.1 var s2542\_8c

**Initial value:**

```
=
[
 ["s2542", "s2542_8c.html#a26ba475c9a0f6a789eb101d850af22ae", null],
 ["s2542", "s2542_8c.html#a4b23a44f7ff69a9190fcd55008c4d148", null]
]
```

Definition at line 1 of file s2542\_8c.js.

### 30.1288 doc/html/s2543\_8c.js File Reference

**Variables**

- var [s2543\\_8c](#)

#### 30.1288.1 Variable Documentation

##### 30.1288.1.1 var s2543\_8c

**Initial value:**

```
=
[
 ["s2543", "s2543_8c.html#aa683afaa39e2aef9ec2780d26079f1e9", null],
 ["s2543", "s2543_8c.html#a617bcd5068cda0f10dc4e9acd3e2b484", null]
]
```

Definition at line 1 of file s2543\_8c.js.

## 30.1289 doc/html/s2544\_8c.js File Reference

### Variables

- var [s2544\\_8c](#)

### 30.1289.1 Variable Documentation

#### 30.1289.1.1 var s2544\_8c

##### Initial value:

```
=
[
 ["s2544", "s2544_8c.html#a84cdee037024bb4ad69a5c3576a8cf94", null],
 ["s2544", "s2544_8c.html#a5f5d710746400640c7bc3924679b081c", null]
]
```

Definition at line 1 of file s2544\_8c.js.

## 30.1290 doc/html/s2545\_8c.js File Reference

### Variables

- var [s2545\\_8c](#)

### 30.1290.1 Variable Documentation

#### 30.1290.1.1 var s2545\_8c

##### Initial value:

```
=
[
 ["s2545", "s2545_8c.html#aa3625e0c61b75e10a40edbbc6f633f69", null],
 ["s2545", "s2545_8c.html#a41de0ed9938c634f47a9cdb2153376b6", null]
]
```

Definition at line 1 of file s2545\_8c.js.

## 30.1291 doc/html/s2550\_8c.js File Reference

### Variables

- var [s2550\\_8c](#)

### 30.1291.1 Variable Documentation

30.1291.1.1 `var s2550_8c`

**Initial value:**

```
=
[
 ["s2550", "s2550_8c.html#a7467fb874321237e877e4e6281aabafb", null],
 ["s2550", "s2550_8c.html#ab4d3acef06d32976596b8e891eb08069", null]
]
```

Definition at line 1 of file s2550\_8c.js.

### 30.1292 doc/html/s2551\_8c.js File Reference

**Variables**

- `var s2551_8c`

### 30.1292.1 Variable Documentation

30.1292.1.1 `var s2551_8c`

**Initial value:**

```
=
[
 ["s2551", "s2551_8c.html#a442aba2fe0e9d1f49fd5a98d6f895462", null],
 ["s2551", "s2551_8c.html#a9a9e439ea38dbe0d3fda019d8bafb612", null]
]
```

Definition at line 1 of file s2551\_8c.js.

### 30.1293 doc/html/s2553\_8c.js File Reference

**Variables**

- `var s2553_8c`

### 30.1293.1 Variable Documentation

30.1293.1.1 `var s2553_8c`

**Initial value:**

```
=
[
 ["s2553", "s2553_8c.html#a172f512637d72486554910f16d92242b", null],
 ["s2553", "s2553_8c.html#ac60405e3e343a34c992c6c4bfe845172", null]
]
```

Definition at line 1 of file s2553\_8c.js.

## 30.1294 doc/html/s2554\_8c.js File Reference

### Variables

- var [s2554\\_8c](#)

#### 30.1294.1 Variable Documentation

##### 30.1294.1.1 var s2554\_8c

###### Initial value:

```
=
[
 ["s2554", "s2554_8c.html#a4250cbc624fbc7721f91ae94228853a8", null],
 ["s2554", "s2554_8c.html#ab4aeb0e94291ed4c9081364c995337", null]
]
```

Definition at line 1 of file s2554\_8c.js.

## 30.1295 doc/html/s2555\_8c.js File Reference

### Variables

- var [s2555\\_8c](#)

#### 30.1295.1 Variable Documentation

##### 30.1295.1.1 var s2555\_8c

###### Initial value:

```
=
[
 ["s2555", "s2555_8c.html#ab05c8828c32b4f12c0aa0cb5c0a40509", null],
 ["s2555", "s2555_8c.html#a35950511e590abf071ec7e691e8018c1", null]
]
```

Definition at line 1 of file s2555\_8c.js.

## 30.1296 doc/html/s2556\_8c.js File Reference

### Variables

- var [s2556\\_8c](#)

### 30.1296.1 Variable Documentation

30.1296.1.1 `var s2556_8c`

**Initial value:**

```
=
[
 ["s2556", "s2556_8c.html#ae80e009e69a1e440913e7ea77b695919", null],
 ["s2556", "s2556_8c.html#a639b34165f9d7bfcef5d4b00e7026234", null]
]
```

Definition at line 1 of file s2556\_8c.js.

### 30.1297 doc/html/s2557\_8c.js File Reference

**Variables**

- `var s2557_8c`

### 30.1297.1 Variable Documentation

30.1297.1.1 `var s2557_8c`

**Initial value:**

```
=
[
 ["s2557", "s2557_8c.html#a164c2bd7adc605b0ca9c3ace703aa7d8", null],
 ["s2557", "s2557_8c.html#a94cf3c7be106e38adc8871a4656bfef7", null]
]
```

Definition at line 1 of file s2557\_8c.js.

### 30.1298 doc/html/s2558\_8c.js File Reference

**Variables**

- `var s2558_8c`

### 30.1298.1 Variable Documentation

30.1298.1.1 `var s2558_8c`

**Initial value:**

```
=
[
 ["s2558", "s2558_8c.html#a288eb78f482bc364cb5f710268a6a985", null],
 ["s2558", "s2558_8c.html#a53845d56d990a847b1ccbbd691aa4ee2", null]
]
```

Definition at line 1 of file s2558\_8c.js.



## 30.1299 doc/html/s2559\_8c.js File Reference

### Variables

- var [s2559\\_8c](#)

#### 30.1299.1 Variable Documentation

##### 30.1299.1.1 var s2559\_8c

###### Initial value:

```
=
[
 ["s2559", "s2559_8c.html#af7f92acdd065d9dc7b72846180db005a", null],
 ["s2559", "s2559_8c.html#aa214528f28ad5f7415e8490ff57243ff", null]
]
```

Definition at line 1 of file s2559\_8c.js.

## 30.1300 doc/html/s2560\_8c.js File Reference

### Variables

- var [s2560\\_8c](#)

#### 30.1300.1 Variable Documentation

##### 30.1300.1.1 var s2560\_8c

###### Initial value:

```
=
[
 ["s2560", "s2560_8c.html#a1f9c67b4ca3321bf99d4d76f50832175", null],
 ["s2560", "s2560_8c.html#ab0f341f619a4a04c811d7c9c546d5fac", null]
]
```

Definition at line 1 of file s2560\_8c.js.

## 30.1301 doc/html/s2561\_8c.js File Reference

### Variables

- var [s2561\\_8c](#)

### 30.1301.1 Variable Documentation

30.1301.1.1 var s2561\_8c

**Initial value:**

```
=
[
 ["s2561", "s2561_8c.html#a994c13158ea772cbacfae7fdb087ea79", null],
 ["s2561", "s2561_8c.html#a3170c8b6c7134c9588b7d2739dd5551e", null]
]
```

Definition at line 1 of file s2561\_8c.js.

### 30.1302 doc/html/s2562\_8c.js File Reference

#### Variables

- var [s2562\\_8c](#)

### 30.1302.1 Variable Documentation

30.1302.1.1 var s2562\_8c

**Initial value:**

```
=
[
 ["s2562", "s2562_8c.html#a4d8239c7bf886b530acefc9568b9c148", null],
 ["s2562", "s2562_8c.html#a45964a15239cba527d6639ac8d037644", null]
]
```

Definition at line 1 of file s2562\_8c.js.

### 30.1303 doc/html/s6addcurve\_8c.js File Reference

#### Variables

- var [s6addcurve\\_8c](#)

### 30.1303.1 Variable Documentation

30.1303.1.1 var s6addcurve\_8c

**Initial value:**

```
=
[
 ["S6ADDCURVE", "s6addcurve_8c.html#ac68a3efe3fd76c850d49a7e60d256db5", null],
 ["s6addcurve", "s6addcurve_8c.html#a2ff4b8d15c68b87f85b4dcb6f28b9a9d", null]
]
```

Definition at line 1 of file s6addcurve\_8c.js.

## 30.1304 doc/html/s6affdist\_8c.js File Reference

### Variables

- var [s6affdist\\_8c](#)

#### 30.1304.1 Variable Documentation

##### 30.1304.1.1 var s6affdist\_8c

###### Initial value:

```
=
[
 ["S6AFFDIST", "s6affdist_8c.html#a9618d268d1513e3bf866ce8ae5a1479b", null],
 ["s6affdist", "s6affdist_8c.html#a8c4b95d6f1acd4ae2eb0ea5269a62a11", null]
]
```

Definition at line 1 of file s6affdist\_8c.js.

## 30.1305 doc/html/s6ang\_8c.js File Reference

### Variables

- var [s6ang\\_8c](#)

#### 30.1305.1 Variable Documentation

##### 30.1305.1.1 var s6ang\_8c

###### Initial value:

```
=
[
 ["S6ANG", "s6ang_8c.html#a3c5bc78a99382a60efb86386ac88dc16", null],
 ["s6ang", "s6ang_8c.html#a51df2fd4b947cdced9cf60effaa99b73", null]
]
```

Definition at line 1 of file s6ang\_8c.js.

## 30.1306 doc/html/s6angle\_8c.js File Reference

### Variables

- var [s6angle\\_8c](#)

### 30.1306.1 Variable Documentation

#### 30.1306.1.1 var s6angle\_8c

**Initial value:**

```
=
[
 ["S6ANGLE", "s6angle_8c.html#aaadf9755caa77f601eb0bdd9fdd9a4e5", null],
 ["s6angle", "s6angle_8c.html#a576f12c38ad86a65726f971477290cfe", null]
]
```

Definition at line 1 of file s6angle\_8c.js.

## 30.1307 doc/html/s6castelja\_8c.js File Reference

### Variables

- var [s6castelja\\_8c](#)

### 30.1307.1 Variable Documentation

#### 30.1307.1.1 var s6castelja\_8c

**Initial value:**

```
=
[
 ["S6DECASTELJAU", "s6castelja_8c.html#a6fc444779a1c5bc6722cc3203ef90898", null],
 ["s6deCasteljau", "s6castelja_8c.html#a33e813e1fc6e2dfbf6e69df2a26e6cac", null]
]
```

Definition at line 1 of file s6castelja\_8c.js.

## 30.1308 doc/html/s6chpar\_8c.js File Reference

### Variables

- var [s6chpar\\_8c](#)

### 30.1308.1 Variable Documentation

#### 30.1308.1.1 var s6chpar\_8c

**Initial value:**

```
=
[
 ["S6CHPAR", "s6chpar_8c.html#a6bc188257f3209e82c48420abab5aaa6", null],
 ["s6chpar", "s6chpar_8c.html#abf77d8075b532741cd8cfbbcb35e8b", null]
]
```

Definition at line 1 of file s6chpar\_8c.js.

## 30.1309 doc/html/s6crss\_8c.js File Reference

### Variables

- var [s6crss\\_8c](#)

### 30.1309.1 Variable Documentation

#### 30.1309.1.1 var s6crss\_8c

##### Initial value:

```
=
[
 ["S6CRSS", "s6crss_8c.html#acf0ca9217120a824a0bbc19229bb862d", null],
 ["s6crss", "s6crss_8c.html#a336bccca08dedd525a044d6fbbafda3d2", null]
]
```

Definition at line 1 of file s6crss\_8c.js.

## 30.1310 doc/html/s6crvature\_8c.js File Reference

### Variables

- var [s6crvature\\_8c](#)

### 30.1310.1 Variable Documentation

#### 30.1310.1.1 var s6crvature\_8c

##### Initial value:

```
=
[
 ["S6CURVATURE", "s6crvature_8c.html#a26fad67421bf84e53104cd1747942bac", null],
 ["s6crvature", "s6crvature_8c.html#a2d1ed0197ce0a554f0905097832e5667", null]
]
```

Definition at line 1 of file s6crvature\_8c.js.

## 30.1311 doc/html/s6crvcheck\_8c.js File Reference

### Variables

- var [s6crvcheck\\_8c](#)

### 30.1311.1 Variable Documentation

#### 30.1311.1.1 var s6crvcheck\_8c

**Initial value:**

```
=
[
 ["S6CRVCHECK", "s6crvcheck_8c.html#a044eb3a4b59cd57dfcaa93508c01a9b2", null],
 ["s6crvcheck", "s6crvcheck_8c.html#ae702ec538a65924ee0a6b5310fa1b075", null]
]
```

Definition at line 1 of file s6crvcheck\_8c.js.

## 30.1312 doc/html/s6curvrad\_8c.js File Reference

**Variables**

- var [s6curvrad\\_8c](#)

### 30.1312.1 Variable Documentation

#### 30.1312.1.1 var s6curvrad\_8c

**Initial value:**

```
=
[
 ["S6CURVRAD", "s6curvrad_8c.html#a0a1cbfd7f19e573f1534678f4af8bcd7", null],
 ["s6curvrad", "s6curvrad_8c.html#a0868b595e8286dc1e7494afeaf148fdc", null]
]
```

Definition at line 1 of file s6curvrad\_8c.js.

## 30.1313 doc/html/s6decomp\_8c.js File Reference

**Variables**

- var [s6decomp\\_8c](#)

### 30.1313.1 Variable Documentation

#### 30.1313.1.1 var s6decomp\_8c

**Initial value:**

```
=
[
 ["S6DECOMP", "s6decomp_8c.html#a688e17aa7e50581a93f2125cd045f4de", null],
 ["s6decomp", "s6decomp_8c.html#a3ec124c98793420ece87b04c1fd9718c", null]
]
```

Definition at line 1 of file s6decomp\_8c.js.

## 30.1314 doc/html/s6degnorm\_8c.js File Reference

### Variables

- var [s6degnorm\\_8c](#)

#### 30.1314.1 Variable Documentation

##### 30.1314.1.1 var s6degnorm\_8c

###### Initial value:

```
=
[
 ["S6DEGNORM", "s6degnorm_8c.html#a9f043d8f2a39cd4493a14ff660524eab", null],
 ["s6degnorm", "s6degnorm_8c.html#a18263570cde4087ccacf7291e5b86fa7", null]
]
```

Definition at line 1 of file s6degnorm\_8c.js.

## 30.1315 doc/html/s6dertopt\_8c.js File Reference

### Variables

- var [s6dertopt\\_8c](#)

#### 30.1315.1 Variable Documentation

##### 30.1315.1.1 var s6dertopt\_8c

###### Initial value:

```
=
[
 ["S6DERTOPT", "s6dertopt_8c.html#adb1c9d25e1cd66a4c17f880ed0ca9209", null],
 ["s6dertopt", "s6dertopt_8c.html#a795f8e351a96dfeb4677cfffdc44cfce2", null]
]
```

Definition at line 1 of file s6dertopt\_8c.js.

## 30.1316 doc/html/s6diff\_8c.js File Reference

### Variables

- var [s6diff\\_8c](#)

### 30.1316.1 Variable Documentation

#### 30.1316.1.1 var s6diff\_8c

**Initial value:**

```
=
[
 ["S6DIFF", "s6diff_8c.html#a52f2ec9621cd96b5615373339612f483", null],
 ["s6diff", "s6diff_8c.html#afcb16ff486c070d8ab17984792f8d019", null]
]
```

Definition at line 1 of file s6diff\_8c.js.

### 30.1317 doc/html/s6dist\_8c.js File Reference

**Variables**

- var [s6dist\\_8c](#)

#### 30.1317.1 Variable Documentation

##### 30.1317.1.1 var s6dist\_8c

**Initial value:**

```
=
[
 ["S6DIST", "s6dist_8c.html#aeedd4ab2682ab3fcadd5dc74b34d72e5", null],
 ["s6dist", "s6dist_8c.html#abf2d8ba2b1b09b76a3995451c8a3871a", null]
]
```

Definition at line 1 of file s6dist\_8c.js.

### 30.1318 doc/html/s6dline\_8c.js File Reference

**Variables**

- var [s6dline\\_8c](#)

#### 30.1318.1 Variable Documentation

##### 30.1318.1.1 var s6dline\_8c

**Initial value:**

```
=
[
 ["S6DLINE", "s6dline_8c.html#a7d9aba73a62e98d7700f5cc71feb2c45", null],
 ["s6dline", "s6dline_8c.html#a4eb6806bf84772599b5ad8c3184cdf03", null]
]
```

Definition at line 1 of file s6dline\_8c.js.



## 30.1319 doc/html/s6dplane\_8c.js File Reference

### Variables

- var [s6dplane\\_8c](#)

#### 30.1319.1 Variable Documentation

##### 30.1319.1.1 var s6dplane\_8c

###### Initial value:

```
=
[
 ["S6DPLANE", "s6dplane_8c.html#a133f94fd304dae3d48b685f3ce27952d", null],
 ["s6dplane", "s6dplane_8c.html#a87003f3f940d7e9f43ee852d02bd0c06", null]
]
```

Definition at line 1 of file s6dplane\_8c.js.

## 30.1320 doc/html/s6drawseq\_8c.js File Reference

### Variables

- var [s6drawseq\\_8c](#)

#### 30.1320.1 Variable Documentation

##### 30.1320.1.1 var s6drawseq\_8c

###### Initial value:

```
=
[
 ["S6DRAWSEQ", "s6drawseq_8c.html#abb0d778c2b13e4b9143096ad525b2e6", null],
 ["s6drawseq", "s6drawseq_8c.html#a556a3d8485d6ef340453c058d8098dc2", null],
 ["s6line", "s6drawseq_8c.html#aa297adc147bdb171116578256e9ab29a", null],
 ["s6move", "s6drawseq_8c.html#a52f54508d7c8698b4484ae09c8fe0f7f", null]
]
```

Definition at line 1 of file s6drawseq\_8c.js.

## 30.1321 doc/html/s6equal\_8c.js File Reference

### Variables

- var [s6equal\\_8c](#)

### 30.1321.1 Variable Documentation

#### 30.1321.1.1 var s6equal\_8c

**Initial value:**

```
=
[
 ["S6EQUAL", "s6equal_8c.html#a05da7fbced1106716585bc6441cbad13", null],
 ["s6equal", "s6equal_8c.html#aaf4f250c73dbac97e39ca7378d2bcbff", null]
]
```

Definition at line 1 of file s6equal\_8c.js.

### 30.1322 doc/html/s6err\_8c.js File Reference

**Variables**

- var [s6err\\_8c](#)

#### 30.1322.1 Variable Documentation

##### 30.1322.1.1 var s6err\_8c

**Initial value:**

```
=
[
 ["S6ERR", "s6err_8c.html#a7869604fc2bc0a0e0dc94fae2c82c817", null],
 ["s6err", "s6err_8c.html#a0167287f293e82ee78fe9d9650a57ca9", null]
]
```

Definition at line 1 of file s6err\_8c.js.

### 30.1323 doc/html/s6existbox\_8c.js File Reference

**Variables**

- var [s6existbox\\_8c](#)

#### 30.1323.1 Variable Documentation

##### 30.1323.1.1 var s6existbox\_8c

**Initial value:**

```
=
[
 ["S6EXISTBOX", "s6existbox_8c.html#a645c371a463bd4597717a1696257c688", null],
 ["s6existbox", "s6existbox_8c.html#a6c06fd32283f71f44906a065af629599", null]
]
```

Definition at line 1 of file s6existbox\_8c.js.

## 30.1324 doc/html/s6findfac\_8c.js File Reference

### Variables

- var [s6findfac\\_8c](#)

#### 30.1324.1 Variable Documentation

##### 30.1324.1.1 var s6findfac\_8c

###### Initial value:

```
=
[
 ["S6FINDFAC", "s6findfac_8c.html#ac841a22706ed1d66248ede6e5048c8ca", null],
 ["s6findfac", "s6findfac_8c.html#a61a90f39e6f3e087394aaee188805e", null]
]
```

Definition at line 1 of file s6findfac\_8c.js.

## 30.1325 doc/html/s6fndintv\_8c.js File Reference

### Variables

- var [s6fndintv\\_8c](#)

#### 30.1325.1 Variable Documentation

##### 30.1325.1.1 var s6fndintv\_8c

###### Initial value:

```
=
[
 ["SFNDINTVL", "s6fndintv_8c.html#af649c083c01b5e823c602582961b646f", null],
 ["s6fndintv1", "s6fndintv_8c.html#aa399b9d02e274b4c18c6cb0c34cb116c", null]
]
```

Definition at line 1 of file s6fndintv\_8c.js.

## 30.1326 doc/html/s6herm\_8c.js File Reference

### Variables

- var [s6herm\\_8c](#)

### 30.1326.1 Variable Documentation

#### 30.1326.1.1 var s6herm\_8c

**Initial value:**

```
=
[
 ["S6HERM", "s6herm_8c.html#a96e45525e882ad80bf597c3db2ba0797", null],
 ["s6herm", "s6herm_8c.html#af8b2797bde0e5dd1a671752b8bd7b3cf", null]
]
```

Definition at line 1 of file s6herm\_8c.js.

### 30.1327 doc/html/s6herm\_\_bez\_8c.js File Reference

**Variables**

- var [s6herm\\_\\_bez\\_8c](#)

#### 30.1327.1 Variable Documentation

##### 30.1327.1.1 var s6herm\_\_bez\_8c

**Initial value:**

```
=
[
 ["S6HERMITE_BEZIER", "s6herm__bez_8c.html#a36fc301409b7a5b958924911a927a71a", null],
 ["s6hermite_bezier", "s6herm__bez_8c.html#aee846e0428939cf56de479b09058add2", null]
]
```

Definition at line 1 of file s6herm\_\_bez\_8c.js.

### 30.1328 doc/html/s6idcon\_8c.js File Reference

**Variables**

- var [s6idcon\\_8c](#)

#### 30.1328.1 Variable Documentation

##### 30.1328.1.1 var s6idcon\_8c

**Initial value:**

```
=
[
 ["S6IDCON", "s6idcon_8c.html#a2af77f50ad3fc89be205eba38f844c4c", null],
 ["s6idcon", "s6idcon_8c.html#a5167ce75e2b84f14bceaaa49f920c48f", null]
]
```

Definition at line 1 of file s6idcon\_8c.js.

## 30.1329 doc/html/s6idcpt\_8c.js File Reference

### Variables

- var [s6idcpt\\_8c](#)

#### 30.1329.1 Variable Documentation

##### 30.1329.1.1 var s6idcpt\_8c

###### Initial value:

```
=
[
 ["S6IDCPT", "s6idcpt_8c.html#a8572104e61cd1b7b1c6a1789a5e59e8b", null],
 ["s6idcpt", "s6idcpt_8c.html#aca82e0758dc21ec7dbe1b0b37251f010", null]
]
```

Definition at line 1 of file s6idcpt\_8c.js.

## 30.1330 doc/html/s6idedg\_8c.js File Reference

### Variables

- var [s6idedg\\_8c](#)

#### 30.1330.1 Variable Documentation

##### 30.1330.1.1 var s6idedg\_8c

###### Initial value:

```
=
[
 ["S6IDEDG", "s6idedg_8c.html#aba0d67648f173100ac45016261aadd82", null],
 ["s6idedg", "s6idedg_8c.html#a1198cc3b46fdb9d3504abed53dcb4c9d", null]
]
```

Definition at line 1 of file s6idedg\_8c.js.

## 30.1331 doc/html/s6identify\_8c.js File Reference

### Variables

- var [s6identify\\_8c](#)

### 30.1331.1 Variable Documentation

#### 30.1331.1.1 var s6identify\_8c

**Initial value:**

```
=
[
 ["S6IDENTIFY", "s6identify_8c.html#ab631a87fe6bfd230408f39af40ba4b56", null],
 ["s6identify", "s6identify_8c.html#a0788f16c57aefb9d79a2c00a7ddad9af", null]
]
```

Definition at line 1 of file s6identify\_8c.js.

### 30.1332 doc/html/s6idget\_8c.js File Reference

**Variables**

- var [s6idget\\_8c](#)

#### 30.1332.1 Variable Documentation

##### 30.1332.1.1 var s6idget\_8c

**Initial value:**

```
=
[
 ["S6IDGET", "s6idget_8c.html#a78f556bc28612bf71d52cf5f4736fb27", null],
 ["s6idget", "s6idget_8c.html#a228b26dba5eb138af035e636006a0587", null]
]
```

Definition at line 1 of file s6idget\_8c.js.

### 30.1333 doc/html/s6idint\_8c.js File Reference

**Variables**

- var [s6idint\\_8c](#)

#### 30.1333.1 Variable Documentation

##### 30.1333.1.1 var s6idint\_8c

**Initial value:**

```
=
[
 ["S6IDINT", "s6idint_8c.html#acddd2b11e0a5d0833df9a3b1d698094a", null],
 ["s6idint", "s6idint_8c.html#af55007f23c080999d4323349fa432c2a", null]
]
```

Definition at line 1 of file s6idint\_8c.js.

## 30.1334 doc/html/s6idklist\_8c.js File Reference

### Variables

- var [s6idklist\\_8c](#)

#### 30.1334.1 Variable Documentation

##### 30.1334.1.1 var s6idklist\_8c

###### Initial value:

```
=
[
 ["S6IDKLIST", "s6idklist_8c.html#ae30e25e1923a505fca4d8d717d7b8d64", null],
 ["s6idklist", "s6idklist_8c.html#a2dce0fc5acab84bf248773b41f294bbe", null]
]
```

Definition at line 1 of file s6idklist\_8c.js.

## 30.1335 doc/html/s6idkpt\_8c.js File Reference

### Variables

- var [s6idkpt\\_8c](#)

#### 30.1335.1 Variable Documentation

##### 30.1335.1.1 var s6idkpt\_8c

###### Initial value:

```
=
[
 ["S6IDKPT", "s6idkpt_8c.html#a9cdb66d9ecc5c082ad0c17786c154aa6", null],
 ["s6idkpt", "s6idkpt_8c.html#a07728432bcac58961e45bef761ca0c15", null]
]
```

Definition at line 1 of file s6idkpt\_8c.js.

## 30.1336 doc/html/s6idlis\_8c.js File Reference

### Variables

- var [s6idlis\\_8c](#)

### 30.1336.1 Variable Documentation

#### 30.1336.1.1 var s6idlis\_8c

**Initial value:**

```
=
[
 ["S6IDLIS", "s6idlis_8c.html#ab229e11e340f375f26908e4d1281c6a9", null],
 ["s6idlis", "s6idlis_8c.html#a0843f5040f7f11f931a76d85fe465021", null]
]
```

Definition at line 1 of file s6idlis\_8c.js.

### 30.1337 doc/html/s6idnpt\_8c.js File Reference

**Variables**

- var [s6idnpt\\_8c](#)

#### 30.1337.1 Variable Documentation

##### 30.1337.1.1 var s6idnpt\_8c

**Initial value:**

```
=
[
 ["S6IDNPT", "s6idnpt_8c.html#a129f7e227953427208ef8ccf87a2b67a", null],
 ["s6idnpt", "s6idnpt_8c.html#a138f2f281e16c38b65adcc9a7c6c9b16", null]
]
```

Definition at line 1 of file s6idnpt\_8c.js.

### 30.1338 doc/html/s6idput\_8c.js File Reference

**Variables**

- var [s6idput\\_8c](#)

#### 30.1338.1 Variable Documentation

##### 30.1338.1.1 var s6idput\_8c

**Initial value:**

```
=
[
 ["S6IDPUT", "s6idput_8c.html#afd37cbabd0ff2565b68f594940d21854", null],
 ["s6idput", "s6idput_8c.html#a3c8cb17e9dc024fb0e541bca04186ff0", null]
]
```

Definition at line 1 of file s6idput\_8c.js.



## 30.1339 doc/html/s6inv4\_8c.js File Reference

### Variables

- var [s6inv4\\_8c](#)

#### 30.1339.1 Variable Documentation

##### 30.1339.1.1 var s6inv4\_8c

###### Initial value:

```
=
[
 ["S6INV4", "s6inv4_8c.html#a857a7d701c40da166947982927a5dd6b", null],
 ["s6inv4", "s6inv4_8c.html#a4db5f9686061d17476e2aecad965252e", null]
]
```

Definition at line 1 of file s6inv4\_8c.js.

## 30.1340 doc/html/s6invert\_8c.js File Reference

### Variables

- var [s6invert\\_8c](#)

#### 30.1340.1 Variable Documentation

##### 30.1340.1.1 var s6invert\_8c

###### Initial value:

```
=
[
 ["S6INVERT", "s6invert_8c.html#a997cf2309b2ab801f18b3c832a14290b", null],
 ["s6invert", "s6invert_8c.html#a82f814ee44540bb24d2c04bc5add30fa", null]
]
```

Definition at line 1 of file s6invert\_8c.js.

## 30.1341 doc/html/s6knotmult\_8c.js File Reference

### Variables

- var [s6knotmult\\_8c](#)

### 30.1341.1 Variable Documentation

#### 30.1341.1.1 var s6knotmult\_8c

**Initial value:**

```
=
[
 ["S6KNOTMULT", "s6knotmult_8c.html#a019e18124f54b338304b5a43f6c61084", null],
 ["s6knotmult", "s6knotmult_8c.html#aa0ca077eb0bac9c9d1ca0ca013d6acde", null]
]
```

Definition at line 1 of file s6knotmult\_8c.js.

### 30.1342 doc/html/s6length\_8c.js File Reference

**Variables**

- var [s6length\\_8c](#)

#### 30.1342.1 Variable Documentation

##### 30.1342.1.1 var s6length\_8c

**Initial value:**

```
=
[
 ["S6LENGTH", "s6length_8c.html#a767c68f7534983e95f61411a7dbe8b18", null],
 ["s6length", "s6length_8c.html#a4cbe26a79c8c65d3de133bc410284362", null]
]
```

Definition at line 1 of file s6length\_8c.js.

### 30.1343 doc/html/s6line\_8c.js File Reference

**Variables**

- var [s6line\\_8c](#)

#### 30.1343.1 Variable Documentation

##### 30.1343.1.1 var s6line\_8c

**Initial value:**

```
=
[
 ["S6LINE", "s6line_8c.html#a464a9074552d5db5f33f86836f2b063d", null],
 ["s6line", "s6line_8c.html#a506e5de24c3b032d09ca07ab943bea5c", null]
]
```

Definition at line 1 of file s6line\_8c.js.

## 30.1344 doc/html/s6lprj\_8c.js File Reference

### Variables

- var [s6lprj\\_8c](#)

#### 30.1344.1 Variable Documentation

##### 30.1344.1.1 var s6lprj\_8c

###### Initial value:

```
=
[
 ["S6LPRJ", "s6lprj_8c.html#a004ef1e03c0aceba2c4193d0e719da43", null],
 ["s6lprj", "s6lprj_8c.html#ad18893baf6ae8c078f0c3a988343baa7", null]
]
```

Definition at line 1 of file s6lprj\_8c.js.

## 30.1345 doc/html/s6lufacp\_8c.js File Reference

### Variables

- var [s6lufacp\\_8c](#)

#### 30.1345.1 Variable Documentation

##### 30.1345.1.1 var s6lufacp\_8c

###### Initial value:

```
=
[
 ["S6LUFACP", "s6lufacp_8c.html#a22b6042f1b88759f9f9bc8450a357cc8", null],
 ["s6lufacp", "s6lufacp_8c.html#a7928ab7cd261b6b9a5deec9823e18b74", null]
]
```

Definition at line 1 of file s6lufacp\_8c.js.

## 30.1346 doc/html/s6lusolp\_8c.js File Reference

### Variables

- var [s6lusolp\\_8c](#)

### 30.1346.1 Variable Documentation

#### 30.1346.1.1 var s6lusolp\_8c

**Initial value:**

```
=
[
 ["S6LUSOLP", "s6lusolp_8c.html#a18d5b0dbcf7cbb509be34cffa88c30c1", null],
 ["s6lusolp", "s6lusolp_8c.html#a4c0b16a24b1478981eafaa76e18de353", null]
]
```

Definition at line 1 of file s6lusolp\_8c.js.

### 30.1347 doc/html/s6metric\_8c.js File Reference

**Variables**

- var [s6metric\\_8c](#)

#### 30.1347.1 Variable Documentation

##### 30.1347.1.1 var s6metric\_8c

**Initial value:**

```
=
[
 ["S6METRIC", "s6metric_8c.html#ac0aeaf33961c1cf1d9a38ce2a81a6f07", null],
 ["s6metric", "s6metric_8c.html#ad2f21b9fd696bcfffebf4ff599c24ff5", null]
]
```

Definition at line 1 of file s6metric\_8c.js.

### 30.1348 doc/html/s6move\_8c.js File Reference

**Variables**

- var [s6move\\_8c](#)

#### 30.1348.1 Variable Documentation

##### 30.1348.1.1 var s6move\_8c

**Initial value:**

```
=
[
 ["S6MOVE", "s6move_8c.html#ab3e4f32fe1c029cfe3f0e2dd5b6bf011", null],
 ["s6move", "s6move_8c.html#aa0b696eeefa27818ca8d4ad70cb3e99ed", null]
]
```

Definition at line 1 of file s6move\_8c.js.

## 30.1349 doc/html/s6mulvec\_8c.js File Reference

### Variables

- var [s6mulvec\\_8c](#)

#### 30.1349.1 Variable Documentation

##### 30.1349.1.1 var s6mulvec\_8c

###### Initial value:

```
=
[
 ["S6MULVEC", "s6mulvec_8c.html#a197128589eb422ba5b5380884f2cb32d", null],
 ["s6mulvec", "s6mulvec_8c.html#a302d23e7e5af682d7ecf15b59d7c6e4d", null]
]
```

Definition at line 1 of file s6mulvec\_8c.js.

## 30.1350 doc/html/s6mvec\_8c.js File Reference

### Variables

- var [s6mvec\\_8c](#)

#### 30.1350.1 Variable Documentation

##### 30.1350.1.1 var s6mvec\_8c

###### Initial value:

```
=
[
 ["S6MVEC", "s6mvec_8c.html#a0c5eac9eac55c41c90a1b5d4c6e7a0af", null],
 ["s6mvec", "s6mvec_8c.html#ae68867523bc91432656b4032b5f94112", null]
]
```

Definition at line 1 of file s6mvec\_8c.js.

## 30.1351 doc/html/s6newbox\_8c.js File Reference

### Variables

- var [s6newbox\\_8c](#)

### 30.1351.1 Variable Documentation

#### 30.1351.1.1 var s6newbox\_8c

**Initial value:**

```
=
[
 ["S6NEWBOX", "s6newbox_8c.html#a80bdb02860ab488616fdcc3b7fea3a31", null],
 ["s6newbox", "s6newbox_8c.html#a1cf9e9f8ea8d1311324eace684cc377d", null]
]
```

Definition at line 1 of file s6newbox\_8c.js.

### 30.1352 doc/html/s6norm\_8c.js File Reference

**Variables**

- var [s6norm\\_8c](#)

#### 30.1352.1 Variable Documentation

##### 30.1352.1.1 var s6norm\_8c

**Initial value:**

```
=
[
 ["S6NORM", "s6norm_8c.html#abf455c4d8e356823d97ecc511604ca5f", null],
 ["s6norm", "s6norm_8c.html#a19390630ed1341ab54a0eb2776276a0c", null]
]
```

Definition at line 1 of file s6norm\_8c.js.

### 30.1353 doc/html/s6ratder\_8c.js File Reference

**Variables**

- var [s6ratder\\_8c](#)

#### 30.1353.1 Variable Documentation

##### 30.1353.1.1 var s6ratder\_8c

**Initial value:**

```
=
[
 ["S6RATDER", "s6ratder_8c.html#a063cf39116101938f54c80e79fe26c4b", null],
 ["s6ratder", "s6ratder_8c.html#a0fefc63294377a0650c07bcff86c6717", null]
]
```

Definition at line 1 of file s6ratder\_8c.js.

## 30.1354 doc/html/s6rotax\_8c.js File Reference

### Variables

- var [s6rotax\\_8c](#)

#### 30.1354.1 Variable Documentation

##### 30.1354.1.1 var s6rotax\_8c

###### Initial value:

```
=
[
 ["S6ROTAX", "s6rotax_8c.html#a635d38c53fd72e0eb2be74b1cd2eab31", null],
 ["s6rotax", "s6rotax_8c.html#a69df2fc294e5ff516f7055742c19f2ee", null]
]
```

Definition at line 1 of file s6rotax\_8c.js.

## 30.1355 doc/html/s6rotmat\_8c.js File Reference

### Variables

- var [s6rotmat\\_8c](#)

#### 30.1355.1 Variable Documentation

##### 30.1355.1.1 var s6rotmat\_8c

###### Initial value:

```
=
[
 ["S6ROTMAT", "s6rotmat_8c.html#a94540cb2345f671e2fb12e1e1018ac95", null],
 ["s6rotmat", "s6rotmat_8c.html#a002798ab6371ee28c5a7cd51c229ad90", null]
]
```

Definition at line 1 of file s6rotmat\_8c.js.

## 30.1356 doc/html/s6schoen\_8c.js File Reference

### Variables

- var [s6schoen\\_8c](#)

### 30.1356.1 Variable Documentation

#### 30.1356.1.1 var s6schoen\_8c

**Initial value:**

```
=
[
 ["S6SCHOEN", "s6schoen_8c.html#add2d4df20ecfd9af20478df0cf108415", null],
 ["s6schoen", "s6schoen_8c.html#acbd883798bfb4524c7551b22e0d42877", null]
]
```

Definition at line 1 of file s6schoen\_8c.js.

### 30.1357 doc/html/s6scpr\_8c.js File Reference

**Variables**

- var [s6scpr\\_8c](#)

#### 30.1357.1 Variable Documentation

##### 30.1357.1.1 var s6scpr\_8c

**Initial value:**

```
=
[
 ["S6SCPR", "s6scpr_8c.html#ae0303e3d897d74693bfeda2685a671ee", null],
 ["s6scpr", "s6scpr_8c.html#a40208ad1a68943542bb9aa05fc1d0d06", null]
]
```

Definition at line 1 of file s6scpr\_8c.js.

### 30.1358 doc/html/s6sortpar\_8c.js File Reference

**Variables**

- var [s6sortpar\\_8c](#)

#### 30.1358.1 Variable Documentation

##### 30.1358.1.1 var s6sortpar\_8c

**Initial value:**

```
=
[
 ["S6SORTPAR", "s6sortpar_8c.html#ada7d7b2269768599979515f92d93fc45", null],
 ["s6sortpar", "s6sortpar_8c.html#a6e0679bdaf720915f7b62c846c649305", null]
]
```

Definition at line 1 of file s6sortpar\_8c.js.



## 30.1359 doc/html/s6sratder\_8c.js File Reference

### Variables

- var [s6sratder\\_8c](#)

#### 30.1359.1 Variable Documentation

##### 30.1359.1.1 var s6sratder\_8c

###### Initial value:

```
=
[
 ["S6SRATDER", "s6sratder_8c.html#a1cec956106edf5e66a511ad1fc401eeb", null],
 ["s6sratder", "s6sratder_8c.html#a8ff72acc1bf5029d23c2413dd60efce9", null]
]
```

Definition at line 1 of file s6sratder\_8c.js.

## 30.1360 doc/html/s6strider\_8c.js File Reference

### Variables

- var [s6strider\\_8c](#)

#### 30.1360.1 Variable Documentation

##### 30.1360.1.1 var s6strider\_8c

###### Initial value:

```
=
[
 ["S6STRIDER", "s6strider_8c.html#a9420da05d5e965c859088e28b6d76e88", null],
 ["s6strider", "s6strider_8c.html#a5e5ce51ae18855baeaa64e091eab12c6", null]
]
```

Definition at line 1 of file s6strider\_8c.js.

## 30.1361 doc/html/s6takunion\_8c.js File Reference

### Variables

- var [s6takunion\\_8c](#)

### 30.1361.1 Variable Documentation

#### 30.1361.1.1 var s6takunion\_8c

**Initial value:**

```
=
[
 ["S6TAKEUNION", "s6takunion_8c.html#a5208d1a25bb96397654bf171929c4dfc", null],
 ["s6takeunion", "s6takunion_8c.html#ac950a0db8775ec752295efe918b1437d", null]
]
```

Definition at line 1 of file s6takunion\_8c.js.

### 30.1362 doc/html/s6twonorm\_8c.js File Reference

**Variables**

- var [s6twonorm\\_8c](#)

#### 30.1362.1 Variable Documentation

##### 30.1362.1.1 var s6twonorm\_8c

**Initial value:**

```
=
[
 ["S6TWONORM", "s6twonorm_8c.html#aaccffb62e19263aebaaf015b9a0e53df", null],
 ["s6twonorm", "s6twonorm_8c.html#a8dae3bd50bf7607282d493eebe504351", null]
]
```

Definition at line 1 of file s6twonorm\_8c.js.

### 30.1363 doc/html/s9adsimp\_8c.js File Reference

**Variables**

- var [s9adsimp\\_8c](#)

#### 30.1363.1 Variable Documentation

##### 30.1363.1.1 var s9adsimp\_8c

**Initial value:**

```
=
[
 ["S9ADSIMP", "s9adsimp_8c.html#a38cd209f5ec01ffe03bc9f6af9fddd91", null],
 ["s9adsimp", "s9adsimp_8c.html#a8fd1c9dfa80574a33aacd6c89375f9be", null]
]
```

Definition at line 1 of file s9adsimp\_8c.js.

## 30.1364 doc/html/s9adstep\_8c.js File Reference

### Variables

- var [s9adstep\\_8c](#)

#### 30.1364.1 Variable Documentation

##### 30.1364.1.1 var s9adstep\_8c

###### Initial value:

```
=
[
 ["S9ADSTEP", "s9adstep_8c.html#a3fc6930442b1a55563f686043787d9a6", null],
 ["s9adstep", "s9adstep_8c.html#ae4b9864fe962edbc6b905ce9a3663fd0", null]
]
```

Definition at line 1 of file s9adstep\_8c.js.

## 30.1365 doc/html/s9boundimp\_8c.js File Reference

### Variables

- var [s9boundimp\\_8c](#)

#### 30.1365.1 Variable Documentation

##### 30.1365.1.1 var s9boundimp\_8c

###### Initial value:

```
=
[
 ["S9BOUNDIMP", "s9boundimp_8c.html#a8db74aad6f21980b8bb7a22e43ffddb6", null],
 ["s9boundimp", "s9boundimp_8c.html#af88a19b4eb19f86a784c0e5fecc94a9c", null]
]
```

Definition at line 1 of file s9boundimp\_8c.js.

## 30.1366 doc/html/s9boundit\_8c.js File Reference

### Variables

- var [s9boundit\\_8c](#)

### 30.1366.1 Variable Documentation

#### 30.1366.1.1 var s9boundit\_8c

**Initial value:**

```
=
[
 ["S9BOUNDIT", "s9boundit_8c.html#a4d5c7c5503a88b39fac7dc5df92d8de3", null],
 ["s9boundit", "s9boundit_8c.html#a4fc3d38497c6dee699ccb978f42afded", null]
]
```

Definition at line 1 of file s9boundit\_8c.js.

### 30.1367 doc/html/s9clipimp\_8c.js File Reference

**Variables**

- var [s9clipimp\\_8c](#)

#### 30.1367.1 Variable Documentation

##### 30.1367.1.1 var s9clipimp\_8c

**Initial value:**

```
=
[
 ["S9CLIPIMP", "s9clipimp_8c.html#a239aaee9a37f54bb2cef68a06e87a5ba", null],
 ["s9clipimp", "s9clipimp_8c.html#a4e242d424e250ad264c3a95b5ef2faae", null]
]
```

Definition at line 1 of file s9clipimp\_8c.js.

### 30.1368 doc/html/s9clipit\_8c.js File Reference

**Variables**

- var [s9clipit\\_8c](#)

#### 30.1368.1 Variable Documentation

##### 30.1368.1.1 var s9clipit\_8c

**Initial value:**

```
=
[
 ["S9CLIPIT", "s9clipit_8c.html#a71376b9c4eb67357ea151d6a770688b7", null],
 ["s9clipit", "s9clipit_8c.html#acce291c32ed4dfca1c5b48b3cdceb1f2", null]
]
```

Definition at line 1 of file s9clipit\_8c.js.

## 30.1369 doc/html/s9conmarch\_8c.js File Reference

### Variables

- var [s9conmarch\\_8c](#)

#### 30.1369.1 Variable Documentation

##### 30.1369.1.1 var s9conmarch\_8c

###### Initial value:

```
=
[
 ["S9CONMARCH", "s9conmarch_8c.html#a8bef0e5e6c53d8bf855c17a6d2d46807", null],
 ["s9conmarch", "s9conmarch_8c.html#ade2fcd614208fa9de6755fb38b89cfaf", null]
]
```

Definition at line 1 of file s9conmarch\_8c.js.

## 30.1370 doc/html/s9iterate\_8c.js File Reference

### Variables

- var [s9iterate\\_8c](#)

#### 30.1370.1 Variable Documentation

##### 30.1370.1.1 var s9iterate\_8c

###### Initial value:

```
=
[
 ["S9ITERATE", "s9iterate_8c.html#a7bf9741589513d5161cd93fa4be8119b", null],
 ["s9iterate", "s9iterate_8c.html#a145dfbbdce306bc13048f283d3e4d302", null]
]
```

Definition at line 1 of file s9iterate\_8c.js.

## 30.1371 doc/html/s9iterimp\_8c.js File Reference

### Variables

- var [s9iterimp\\_8c](#)

### 30.1371.1 Variable Documentation

#### 30.1371.1.1 var s9iterimp\_8c

**Initial value:**

```
=
[
 ["S9ITERIMP", "s9iterimp_8c.html#a74c9fe8b5b9e1b31b4a0672efad86a26", null],
 ["s9iterimp", "s9iterimp_8c.html#ac1ad95062a5b32009141c9a21f9a8d87", null]
]
```

Definition at line 1 of file s9iterimp\_8c.js.

### 30.1372 doc/html/s9smplknot\_8c.js File Reference

**Variables**

- var [s9smplknot\\_8c](#)

#### 30.1372.1 Variable Documentation

##### 30.1372.1.1 var s9smplknot\_8c

**Initial value:**

```
=
[
 ["S9SIMPLE_KNOT", "s9smplknot_8c.html#a02a6310cafd59b07701624ab7b22b903", null],
 ["s9simple_knot", "s9smplknot_8c.html#a6353f6acd9d39814e7ec5c93ef25eb6d", null]
]
```

Definition at line 1 of file s9smplknot\_8c.js.

### 30.1373 doc/html/s9surmarch\_8c.js File Reference

**Variables**

- var [s9surmarch\\_8c](#)

#### 30.1373.1 Variable Documentation

##### 30.1373.1.1 var s9surmarch\_8c

**Initial value:**

```
=
[
 ["S9SURMARCH", "s9surmarch_8c.html#ae3cedf3fd5412862b698e60d4efa0b25", null],
 ["s9surmarch", "s9surmarch_8c.html#a3f0fdc6f5fc9bfcfb5c0f6ed8aldac", null]
]
```

Definition at line 1 of file s9surmarch\_8c.js.

## 30.1374 doc/html/search/all\_0.js File Reference

### Variables

- var [searchData](#)

### 30.1374.1 Variable Documentation

#### 30.1374.1.1 var searchData

##### Initial value:

```
=
[
 ['2dpoly_5ffor_5fs2m_2eh', ['2dpoly_for_s2m.h', ['../2dpoly__for__s2m_8h.html', 1, '']]]
]
```

Definition at line 1 of file all\_0.js.

## 30.1375 doc/html/search/all\_1.js File Reference

### Variables

- var [searchData](#)

### 30.1375.1 Variable Documentation

#### 30.1375.1.1 var searchData

##### Initial value:

```
=
[
 ['_5f_5fpower', ['__power', ['../namespacestd.html#a97b81ceb78c6f342eb91ad62ba86260f', 1, 'std: __power(_Tp __x, _Integer __n, _MonoidOperation __oper)'], ['../namespacestd.html#ac866b28c409b5c4b6b13e17fd3b5c85', 1, 'std: __power(_Tp __x, _Integer __n)']],
 ['_5fcopyright_5fh', ['_COPYRIGHT_H', ['../gotools-core_2include_2go_tools_2geometry_2copyright_8h.html#abdfb6b142e6341a6d5e35b87ccbfab07', 1, 'copyright.h']],
 ['_5ferrormacros_5fh', ['_ERRORMACROS_H', ['../aux2_8h.html#a3f97e52087e510f4d7c49c7eb28624c0', 1, 'aux2.h']],
],
 ['_5figeslib', ['_IGESLIB', ['../igeslib__doxymain_8h.html#aa60f7acaa57d814a1403f7bb16981c1d', 1, 'igeslib_doxymain.h']],
 ['_5flrsplineevavgrid_5fh', ['_LRSPLINEEVAVGRID_H', ['../_l_r_spline_eval_grid_8h.html#a861377e9079b564d42abe23579c6992b', 1, 'LRSplineEvalGrid.h']],
 ['_5foffsetutils_5fh', ['_OFFSETUTILS_H', ['../_creators_offset_utils_8h.html#ad9fc2da1082c23bb539309710e76122f', 1, 'CreatorsOffsetUtils.h']],
 ['_5fvolumehandler_5fh', ['_VOLUMEFILEHANDLER_H', ['../_volume_model_file_handler_8h.html#a1d5505728388942e4ea90d4be08daaea', 1, 'VolumeModelFileHandler.h']],
],
]
```

Definition at line 1 of file all\_1.js.

## 30.1376 doc/html/search/all\_10.js File Reference

### Variables

- var [searchData](#)
- [StreamUtils](#) [h](#)
- [StreamUtils](#) int [idx](#)
- [StreamUtils](#) int int [idx](#) [const](#)
- [StreamUtils](#) int int [idx](#) int i int [i](#)
- [StreamUtils](#) int int [idx](#) int i int int i int int i int int num int [num](#)

### 30.1376.1 Variable Documentation

#### 30.1376.1.1 [StreamUtils](#) int int [idx](#) int i int int i int int i int int num int int num int int num int int num int int i int int i [const](#) [bound]

#### Initial value:

```
=0'], ['./class_go_1_1_composite_model.html#a1546764e88cba9c13fa2bea819e9fe2c',
,1,'Go::CompositeModel::tessellate(double density, std::vector&
lt; shared_ptr< GeneralMesh > > & amp; meshes) const =0'], ['./
class_go_1_1_curve_model.html#a5aa6f99880fd5126485b39a59932a30f',1,'
Go::CurveModel::tessellate(std::vector< shared_ptr< GeneralMesh > &
amp; meshes) const ', ['./class_go_1_1_curve_model.html#
a2cc71ae48a6d1024c8779c1fde08eb80',1,'Go::CurveModel::tessellate(int resolution[], std::vector<
shared_ptr< GeneralMesh > > & amp; meshes) const ', ['./
class_go_1_1_curve_model.html#a13361c5bf5d010aa4da56ae5569070e0',1,'
Go::CurveModel::tessellate(double density, std::vector< shared_ptr<
GeneralMesh > > & amp; meshes) const ', ['./class_go_1_1_lft_curve_segment.html#
ad3a6be9011d0466686c7095dd6e54f85',1,'Go::ftCurveSegment::tessellate()', ['./
class_go_1_1_lft_curve.html#a3fa46cd0cd1b6301996463d9fc95589c',1,'
Go::ftCurve::tessellate(std::vector< shared_ptr< LineStrip &
gt; > & amp; meshes) const ', ['./class_go_1_1_lft_curve.html#
a2305a44a658989280b9877ac04600a19',1,'Go::ftCurve::tessellate(int resolution, std::vector<
shared_ptr< LineStrip > > & amp; meshes) const ', ['./class_go_1_1_lft_curve.html#
af4679709df9d47b8f07fb7dbf82737b5',1,'Go::ftCurve::tessellate(double density, std::vector<
shared_ptr< LineStrip > > & amp; meshes) const ', ['./
class_go_1_1_int_results_comp_cv.html#a0a2c6122f9a871bf7b9666d2e0ad2a2f',1,
'Go::IntResultsCompCv::tessellate(std::vector< shared_ptr< LineStrip >
> & amp; meshes, PointCloud3D & amp; points) const ', ['./
class_go_1_1_int_results_comp_cv.html#aaa85f9ea710d6c6700b49456212c73ba',1,
'Go::IntResultsCompCv::tessellate(int resolution, std::vector< shared_ptr<
LineStrip > > & amp; meshes, PointCloud3D & amp; points) const ', ['./
class_go_1_1_int_results_comp_cv.html#a679437896e0247ae64a4375234file270',1,
'Go::IntResultsCompCv::tessellate(double density, std::vector< shared_ptr<
LineStrip > > & amp; meshes, PointCloud3D & amp; points) const ', ['./
class_go_1_1_int_results_model.html#af758fa08a1d9a2e291102486e4029128',1,
'Go::IntResultsModel::tessellate(std::vector< shared_ptr<
LineStrip > > & amp; meshes, PointCloud3D & amp; points) const =0'], ['./
class_go_1_1_int_results_model.html#a5cba20336f32b65cfd840edbf8f46bbd',1,
'Go::IntResultsModel::tessellate(int resolution, std::vector< shared_ptr<
LineStrip > > & amp; meshes, PointCloud3D & amp; points) const =0'], ['./
class_go_1_1_int_results_model.html#aa65af6225021c30ea430fc9287191cd9',1,
'Go::IntResultsModel::tessellate(double density, std::vector< shared_ptr<
LineStrip > > & amp; meshes, PointCloud3D & amp; points) const =0'], ['./
class_go_1_1_int_results_sf_model.html#ab7833b7cdf1ab93b3635e2d68cea77e4',
1,'Go::IntResultsSfModel::tessellate(std::vector< shared_ptr< LineStrip
> > & amp; meshes, PointCloud3D & amp; points) const ', ['./
class_go_1_1_int_results_sf_model.html#a0482f11c47cbd15aa53dd87f7fa59f5',
1,'Go::IntResultsSfModel::tessellate(int resolution, std::vector<
shared_ptr< LineStrip > > & amp; meshes, PointCloud3D & amp; points) const ', ['./
class_go_1_1_int_results_sf_model.html#a9f75cafb391badc9153f52c01393e207',
1,'Go::IntResultsSfModel::tessellate(double density, std::vector<
shared_ptr< LineStrip > > & amp; meshes, PointCloud3D & amp; points) const ', ['./
class_go_1_1_surface_model.html#a499dcd53fc489e79193ec08a40000321',1,
'Go::SurfaceModel::tessellate(std::vector< shared_ptr<
GeneralMesh > > & amp; meshes) const ', ['./
class_go_1_1_surface_model.html#a0cfa68b53d7d7ba43270ac0370434da2',1,
'Go::SurfaceModel::tessellate(int uv_res, std::vector< shared_ptr<
GeneralMesh > > & amp; meshes) const ', ['./class_go_1_1_surface_model.html#
a388420dafc3080e6681f70b8eb0d4f33',1,'Go::SurfaceModel::tessellate(int resolution[],
std::vector< shared_ptr< GeneralMesh > > & amp; meshes) const ', ['./
```



```

class_go_1_1_surface_model.html#ac0c7f814bd5bbba9a376c490f06a85',1,'
Go::SurfaceModel::tessellate(double density, std::vector<T>& shared_ptr<T>
GeneralMesh &T; &T; &T; &T; meshes) const', ['../class_go_1_1_surface_model.html#
ad71862c93633533cf9b92fd3999c512c',1,'Go::SurfaceModel::tessellate(const std::vector<T>&
shared_ptr<T> &T; ftFaceBase &T; &T; &T; &T; faces, int uv_res, std::vector<
T; shared_ptr<T> &T; GeneralMesh &T; &T; &T; &T; meshes) const', ['../
class_go_1_1_surface_model.html#ab8a38d3b31d12e3db759bc29328dbcb9',1,'
Go::SurfaceModel::tessellate(const std::vector<T> &T; shared_ptr<T> &T; ftFaceBase &T;
&T; &T; faces, int resolution[], std::vector<T> &T; shared_ptr<T> &T; &T; &T; meshes) const
', ['../class_go_1_1_surface_model.html#a2ffe542d114f0ae72313b5f99fd2b3d9',1,'
Go::SurfaceModel::tessellate(const std::vector<T> &T; shared_ptr<
T; ftFaceBase &T; &T; &T; faces, double density, std::vector<T> &T; shared_ptr<
T; GeneralMesh &T; &T; &T; meshes) const', ['../
class_go_1_1_curve_tessellator.html#a2903e0b478f9c9bc52de4b78a2e58554',1,'
Go::CurveTessellator::tessellate()', ['../
class_go_1_1_line_cloud_tessellator.html#a741142d2fb183073f3999ebd0987490
',1,'Go::LineCloudTessellator::tessellate()', ['../
class_go_1_1_noop_tessellator.html#a098bad44d4975404a9ecbf097a895423',1,'
Go::NoopTessellator::tessellate()', ['../
class_go_1_1_parametric_surface_tessellator.html#
a0a157fa8039d15ed49d25eccf1d6d8b',1,'Go::ParametricSurfaceTessellator::tessellate()
', ['../class_go_1_1_rectangular_surface_tessellator.html#
a98bc65dc65f847c17602033717c41843',1,'Go::RectangularSurfaceTessellator::tessellate
()', ['../class_go_1_1_rect_grid_tessellator.html#
a37c7dac1717250aae0fe930161e4406b',1,'Go::RectGridTessellator::tessellate()', ['../
class_go_1_1_tessellator.html#ac990c28ea9a36e37b26edd30cc523ac3',1,'
Go::Tessellator::tessellate()', ['../
class_go_1_1_rectangular_volume_tessellator.html#
a81d00516682f244ebac9097b6de6da78',1,'Go::RectangularVolumeTessellator::tessellate()
', ['../class_go_1_1_volume_model.html#aafc3da8a531cbdl1a8e65853e2933abe00',1,'
Go::VolumeModel::tessellate(std::vector<T> &T; shared_ptr<
T; GeneralMesh &T; &T; &T; meshes) const', ['../
class_go_1_1_volume_model.html#aef65bfbbaa2e8f5a66fc0b23630861b3',1,'
Go::VolumeModel::tessellate(int resolution[], std::vector<T> &T; shared_ptr<T>
GeneralMesh &T; &T; &T; meshes) const', ['../class_go_1_1_volume_model.html#
a3e034d7fa26dd97bb9cb8733e50c82c8',1,'Go::VolumeModel::tessellate(double density,
std::vector<T> &T; shared_ptr<T> &T; GeneralMesh &T; &T; &T; meshes) const']]],
['tessellatedctrpolygon', ['tessellatedCtrPolygon', ['../
class_go_1_1_composite_curve.html#accf52036d9edefe396c90df496e8ea2e',1,'
Go::CompositeCurve::tessellatedCtrPolygon(std::vector<T> &T; shared_ptr<
T; LineCloud &T; &T; &T; ctr_pol) const', ['../class_go_1_1_composite_curve
.html#a98d7786acc00fe2d2b642beaff493324',1,'
Go::CompositeCurve::tessellatedCtrPolygon(const std::vector<T>
&T; shared_ptr<T> &T; ParamCurve &T; &T; &T; curves, std::vector<T> &T; shared_ptr<
T; LineCloud &T; &T; &T; ctr_pol) const', ['../
class_go_1_1_composite_model.html#a60e568eb9b8531f2bc5de60ac9d8fd7',1,'
Go::CompositeModel::tessellatedCtrPolygon()', ['../
class_go_1_1_curve_model.html#a0bc9f19a78ec7aaac8b655ea7e25bc7b',1,'
Go::CurveModel::tessellatedCtrPolygon()', ['../
class_go_1_1_surface_model.html#a42d3923e99fa793ca34de335767c53f1',1,'
Go::SurfaceModel::tessellatedCtrPolygon(std::vector<T> &T; shared_ptr<
T; LineCloud &T; &T; &T; ctr_pol) const', ['../
class_go_1_1_surface_model.html#a81fc1beb1ddd467c6fd34f44a4351938',1,'
Go::SurfaceModel::tessellatedCtrPolygon(const std::vector<T>
&T; shared_ptr<T> &T; ftFaceBase &T; &T; &T; faces, std::vector<T> &T; shared_ptr<T> &T; LineCloud &T; &T; &T; ctr_pol)
const', ['../class_go_1_1_volume_model.html#ad7cfef7a9a77a4f4d1d40e7aedef0de3',1,'
Go::VolumeModel::tessellatedCtrPolygon()']]],
['tessellator', ['Tessellator', ['../class_go_1_1_tessellator.html',1,'
Go']]],
['tessellator', ['tessellator', ['../class_data_handler.html#
ab7caedcbf1e13f7456ce14b84c3b6789',1,'DataHandler::tessellator()', ['../classgv_data.html#
a56fb6a11bc06f0998bd20a67a9274836',1,'gvData::tessellator()']]],
['tessellator_2eh', ['Tessellator.h', ['../_tessellator_8h.html',1,'']],
['tessellator_5f', ['tessellator_', ['../class_data_handler.html#
ac6931f251770a4e9b125d18bb2711092',1,'DataHandler']]],
['tessellatorutils_2eh', ['TessellatorUtils.h', ['../_tessellator_utils_8h.html',1,'']],
['test1', ['test1', ['../example_8cpp.html#ac0697ad85fb379446222fcc2df02c778',1,'
example.cpp']]],
['test2', ['test2', ['../example_8cpp.html#ac7865afb3b9f6d710b1df18e501c94d6',1,'
example.cpp']]],
['test3', ['test3', ['../example_8cpp.html#a727a32bdd7e2491824ca4acd62be9b1a',1,'
example.cpp']]],
['test4', ['test4', ['../example_8cpp.html#acd897830273f9673b3894e831ceb3423',1,'
example.cpp']]],
['test5', ['test5', ['../example_8cpp.html#aad4ba83ae697b892cf4711aa7170c1f3',1,'
example.cpp']]],
['test_5fcyclic_5fknots', ['TEST_CYCLIC_KNOTS', ['../
tstcyclknt_8c.html#a77b5bb7a941d583a9aee41d2be40d163',1,'
TEST_CYCLIC_KNOTS():

```

Definition at line 8 of file all\_10.js.

**30.1376.1.2** `sisIP h`

Definition at line 8 of file `all_10.js`.

**30.1376.1.3** `StreamUtils int int idx int i int int i int int i int int num int int num int int num int int num int int int i int i [bound]`

Definition at line 8 of file `all_10.js`.

**30.1376.1.4** `StreamUtils int idx [bound]`

Definition at line 8 of file `all_10.js`.

**30.1376.1.5** `StreamUtils int int idx int i int int i int int i int int num int int num int int num int int num int num [bound]`

Definition at line 8 of file `all_10.js`.

**30.1376.1.6** `var searchData`**Initial value:**

```
=
[
 ['ol', ['ol', ['../struct_s_i_s_l_object.html#aa40c25309fff2ffd852103f2af7acd1f', 1, 'SISLObject']]],
 ['obj_5fint_5f', ['obj_int_', ['../class_go_l_l_intersector2_obj.html#ae87a91f8f4f95a31dc6a07678eabb132', 1,
 'Go::Intersector2Obj']]],
 ['objcontainer', ['ObjContainer', ['../gv_application_8h.html#a930ed0f609554d54ba43b401b20d1f37', 1, '
 gvApplication.h']]],
 ['object', ['object', ['../classgv_data.html#a13c47f9abfb21d7d5ef58957b229f70e', 1, 'gvData::object(int i)'],
 ['../classgv_data.html#a28402d21f6e677b96197b32feb40d8e4', 1, 'gvData::object(int i) const ']]],
 ['object_5fcolors_5f', ['object_colors_', ['../classgv_data.html#a4effc6f51ba7d6d6fbad2afe862d7aac', 1, '
 gvData']]],
 ['object_5ffrom_5fstream', ['object_from_stream', ['
 ../_stream_utils_8h.html#a4894fce509ab613c659adfb8c6e4f3a4', 1, 'object_from_stream(std::istream &is, T &obj):
```

Definition at line 1 of file `all_10.js`.

**30.1377** `doc/html/search/all_11.js` File Reference**Variables**

- `var searchData`
- `jquery js`

**30.1377.1** Variable Documentation**30.1377.1.1** `jquery js`

Definition at line 3 of file `all_11.js`.

## 30.1377.1.2 var searchData

**Initial value:**

```
=
[
 ['p', ['p', ['../jquery_8js.html#a2335e57f79b6acfb6de59c235dc8a83e', 1, 'p() :
```

Definition at line 1 of file all\_11.js.

## 30.1378 doc/html/search/all\_12.js File Reference

**Variables**

- var [searchData](#)
- hholder [cpp](#)

## 30.1378.1 Variable Documentation

## 30.1378.1.1 example sisl\_view\_demo cpp

Definition at line 3 of file all\_12.js.

## 30.1378.1.2 var searchData

**Initial value:**

```
=
[
 ['qrz', ['QRZ', ['../namespace_n_e_w_m_a_t.html#ae1eac58bd06f44ae7f02b39163b2fd80', 1, 'NEWMAT:QRZ(Matrix
&, UpperTriangularMatrix &);)], ['../namespace_n_e_w_m_a_t.html#a3786173220a9e54601def50dc380b32f', 1, '
NEWMAT:QRZ(const Matrix &, Matrix &, Matrix &);)], ['
../hholder_8cpp.html#abf43bc890b60a01fda0c7db9631bcf64', 1, 'QRZ(Matrix &X, UpperTriangularMatrix &U):
```

Definition at line 1 of file all\_12.js.

## 30.1379 doc/html/search/all\_13.js File Reference

**Variables**

- var [searchData](#)
- sisIP [h](#)

## 30.1379.1 Variable Documentation

### 30.1379.1.1 sisIP h

Definition at line 18 of file all\_13.js.

### 30.1379.1.2 var searchData

#### Initial value:

```
=
[
 ['r1_5f', ['r1_', ['../class_go_1_1_ellipse.html#a773aaeb5188d2e6d4e9034fbd4a30123', 1, 'Go::Ellipse::r1_()'],
 ['../class_go_1_1_hyperbola.html#ac624bd16d4a32d9b74361c141421dc81', 1, 'Go::Hyperbola::r1_()']],
 ['r1_5fcol_5fi_5fd', ['R1_Col_I_D', ['../class_n_e_w_m_a_t_1_1_r1_col_i_d.html', 1, 'NEWMAT']],
 ['r1_5f', ['R1_R1', ['../class_r_b_d_c_o_m_m_o_n_1_1_r1.html', 1, 'RBD_COMMON']],
 ['r1_5f', ['R1_R1', ['../class_r_b_d_c_o_m_m_o_n_1_1_r1.html#a278e42bea46cab3e7579f8357975e34', 1,
 'RBD_COMMON::R1_R1']],
 ['r2_5f', ['r2_', ['../class_go_1_1_ellipse.html#a2c90fff391d88f4349a96482c1e4d0fc', 1, 'Go::Ellipse::r2_()'],
 ['../class_go_1_1_hyperbola.html#aa0d052580ed57b97c4974580cdd4733c', 1, 'Go::Hyperbola::r2_()']],
 ['r_5f', ['r_', ['../struct_go_1_1_param_surface_1_1_degenerate__info.html#af2004242e813ed444c5fdb9b265071',
 1, 'Go::ParamSurface::degenerate_info']],
 ['r_5fpath_5f', ['r_path_', ['../class_path_type.html#a6698f5a904d409bc6779015e5ef36aca', 1, 'PathType']],
 ['radialedges', ['radialEdges', ['../class_go_1_1_ft_volume.html#aef559a6497b829cf31faae6942263d97', 1,
 'Go::ftVolume']],
 ['radius', ['radius', ['../class_go_1_1_cylinder_int.html#a4493b91edd175e4166b7ab07a9b35c2c', 1,
 'Go::CylinderInt::radius()'], ['../class_go_1_1_sphere_int.html#abb8450f4ca8990149d73b1eea2f3015d', 1,
 'Go::SphereInt::radius()']],
 ['radius_5f', ['radius_', ['../class_go_1_1_circle.html#a9c349501a00ac666fb501c5a3360c0f9', 1,
 'Go::Circle::radius_()'], ['../class_go_1_1_cone.html#a5bd969d6fb325d1dcc994007031ef23f', 1, 'Go::Cone::radius_()'],
 ['../class_go_1_1_cylinder.html#a1fcl04ecd7893897a5fbbdb94e3f5f3', 1, 'Go::Cylinder::radius_()'],
 ['../class_go_1_1_sphere.html#a4e42c3da8d108c5ea687dcc4cf4652cc', 1, 'Go::Sphere::radius_()']],
 ['radix', ['Radix', ['../class_n_e_w_m_a_t_1_1_floating_point_precision.html#afb7383880d8bbd99bab8e25e3758aa7f', 1, 'NEWMAT::FloatingPointPrecision']],
 ['raiseorder', ['raiseOrder', ['../class_go_1_1_spline_curve.html#a53bcb9a51debc1441a4008cadb904dc2', 1,
 'Go::SplineCurve::raiseOrder()'], ['../class_go_1_1_spline_surface.html#a58f9393ed2771ce6167c4504a9df80e8', 1,
 'Go::SplineSurface::raiseOrder()'], ['../class_go_1_1_spline_volume.html#a306ad4a1f5ab97ab1087360331a51a26', 1,
 'Go::SplineVolume::raiseOrder()']],
 ['randomnoise_2eh', ['randomnoise.h', ['../randomnoise.h', 1, ''']],
 ['range_5ferror', ['Range_error', ['../class_r_b_d_c_o_m_m_o_n_1_1_range__error.html', 1, 'RBD_COMMON']],
 ['range_5ferror', ['Range_error', ['../class_r_b_d_c_o_m_m_o_n_1_1_range__error.html#a8bc12b7074abb363df50ddb82c1c9db', 1, 'RBD_COMMON::Range_error']],
 ['rank_5finfo', ['rank_info', ['../structrank__info.html', 1, 'rank_info'], ['../sis1_p_8h.html#ad075872fe80a1bd569afbd76ea3f2ae5', 1, 'rank_info():

```

Definition at line 1 of file all\_13.js.

## 30.1380 doc/html/search/all\_14.js File Reference

### Variables

- var [searchData](#)
- [s1001 c](#)

## 30.1380.1 Variable Documentation

### 30.1380.1.1 s1001 c

Definition at line 5 of file all\_14.js.

## 30.1380.1.2 var searchData

**Initial value:**

```
=
[
 ['s', ['S', ['../namespace_go.html#a95d383314d9ef7d277424ee672473063a9175c6d2bfb458461de6f564a532dda5', 1, 'Go']]],
 ['s1', ['s1', ['../struct_s_i_s_l_object.html#a932bc80aef806de501a097405e6c5422', 1, 'SISLObject']]],
 ['s1001', ['S1001', ['../s1001_8c.html#a56ccf110d6c3ffd6ec1bc77ba9bf017f', 1, 'S1001():
```

Definition at line 1 of file all\_14.js.

## 30.1381 doc/html/search/all\_15.js File Reference

### Variables

- var [searchData](#)
- [PrParamUtil](#) [h](#)
- [PrParamUtil](#) [std::vector](#) & [lt](#)
- [GeneralMesh](#) & [gt](#)
- & [amp](#)
- [meshes](#) [const](#)
- [tstcyclknt](#) [c](#)

### 30.1381.1 Variable Documentation

#### 30.1381.1.1 [int](#) & [amp](#) [[property](#)], [[bound](#)]

Definition at line 35 of file all\_15.js.

#### 30.1381.1.2 [tstcyclknt](#) [c](#)

Definition at line 47 of file all\_15.js.

#### 30.1381.1.3 [meshes](#) [const](#)

**Initial value:**

```

=0'], ['../class_go_1_1_composite_model.html#a1546764e88cba9c13fa2bea819e9fe2c'
,1,'Go::CompositeModel::tessellate(double density, std::vector&
lt; shared_ptr<lt; GeneralMesh > > & meshes) const =0'], ['../
class_go_1_1_curve_model.html#a5aa6f99880fd5126485b39a59932a30f',1,'
Go::CurveModel::tessellate(std::vector<lt; shared_ptr<lt; GeneralMesh > > &
& meshes) const ', ['../class_go_1_1_curve_model.html#
a2cc71ae48a6d1024c8779c1fde08eb80',1,'Go::CurveModel::tessellate(int resolution[], std::vector<
shared_ptr<lt; GeneralMesh > > & meshes) const ', ['../
class_go_1_1_curve_model.html#a13361c5bf5d010aa4da56ae5569070e0',1,'
Go::CurveModel::tessellate(double density, std::vector< shared_ptr<lt;
GeneralMesh > > & meshes) const ', ['../class_go_1_1ft_curve_segment.html#
ad3a6be9011d0466686c7095dd6e54f85',1,'Go::ftCurveSegment::tessellate()', ['../
class_go_1_1ft_curve.html#a3fa46cd0cd1b6301996463d9fc95589c',1,'
Go::ftCurve::tessellate(std::vector< shared_ptr<lt; LineStrip &
gt; > & meshes) const ', ['../class_go_1_1ft_curve.html#
a2305a44a658989280b9877ac04600a19',1,'Go::ftCurve::tessellate(int resolution, std::vector< shared_ptr&
lt; LineStrip > > & meshes) const ', ['../class_go_1_1ft_curve.html#
af4679709df9d47b8f07fb7dbf82737b5',1,'Go::ftCurve::tessellate(double density, std::vector<
shared_ptr<lt; LineStrip > > & meshes) const ', ['../
class_go_1_1_int_results_comp_cv.html#a0a2c6122f9a871bf7b9666d2e0ad2a2f',1,
'Go::IntResultsCompCv::tessellate(std::vector< shared_ptr<lt; LineStrip >
> & meshes, PointCloud3D & meshes; points) const ', ['../
class_go_1_1_int_results_comp_cv.html#aaa85f9ea710d6c6700b49456212c73ba',1,
'Go::IntResultsCompCv::tessellate(int resolution, std::vector< shared_ptr&
lt; LineStrip > > & meshes, PointCloud3D & meshes; points) const ', ['../
class_go_1_1_int_results_comp_cv.html#a679437896e0247ae64a4375234f1e270',1,
'Go::IntResultsCompCv::tessellate(double density, std::vector< shared_ptr&
lt; LineStrip > > & meshes, PointCloud3D & meshes; points) const ', ['../
class_go_1_1_int_results_model.html#af758fa08ald9a2e291102486e4029128',1,'
Go::IntResultsModel::tessellate(std::vector< shared_ptr&
lt; LineStrip > > & meshes, PointCloud3D & meshes; points) const =0'], ['../
class_go_1_1_int_results_model.html#a5c5ba20336f32b65cfd840edbf8f46bbd',1,'
Go::IntResultsModel::tessellate(int resolution, std::vector< shared_ptr<
LineStrip > > & meshes, PointCloud3D & meshes; points) const =0'], ['../
class_go_1_1_int_results_model.html#aa65af6225021c30ea430fc9287191cd9',1,'
Go::IntResultsModel::tessellate(double density, std::vector< shared_ptr&
lt; LineStrip > > & meshes, PointCloud3D & meshes; points) const =0'], ['../
class_go_1_1_int_results_sf_model.html#ab7833b7cdf1ab93b3635e2d68cea77e4'
,1,'Go::IntResultsSfModel::tessellate(std::vector< shared_ptr< LineStrip
> > & meshes, PointCloud3D & meshes; points) const ', ['../
class_go_1_1_int_results_sf_model.html#a0482f111c47cb1d5aa53dd87f7fa59f5'
,1,'Go::IntResultsSfModel::tessellate(int resolution, std::vector<
shared_ptr< LineStrip > > & meshes, PointCloud3D & meshes; points) const ', ['../
class_go_1_1_int_results_sf_model.html#a9f75cafb391badc9153f52c01393e207'
,1,'Go::IntResultsSfModel::tessellate(double density, std::vector<
shared_ptr< LineStrip > > & meshes, PointCloud3D & meshes; points) const ', ['../
class_go_1_1_surface_model.html#a499dcd53fc489e79193ec08a40000321',1,'
Go::SurfaceModel::tessellate(std::vector< shared_ptr&
lt; GeneralMesh > > & meshes) const ', ['../
class_go_1_1_surface_model.html#a0cfa68b53d7d7ba43270ac0370434da2',1,'
Go::SurfaceModel::tessellate(int uv_res, std::vector< shared_ptr<
GeneralMesh > > & meshes) const ', ['../class_go_1_1_surface_model.html#
a388420dafc3080e6681f70b8eb0d4f33',1,'Go::SurfaceModel::tessellate(int resolution[],
std::vector< shared_ptr< GeneralMesh > > & meshes) const ', ['../
class_go_1_1_surface_model.html#ac0c7f814bd5bbbba9a376c490f0f06a85',1,'
Go::SurfaceModel::tessellate(double density, std::vector< shared_ptr<
GeneralMesh > > & meshes) const ', ['../class_go_1_1_surface_model.html#
ad71862c93633533cf9b92fd3999c512c',1,'Go::SurfaceModel::tessellate(const std::vector<
shared_ptr< ftFaceBase > > & faces, int uv_res, std::vector&
lt; shared_ptr< GeneralMesh > > & meshes) const ', ['../
class_go_1_1_surface_model.html#ab8a38d3b31d12e3db759bc29328dbcb9',1,'
Go::SurfaceModel::tessellate(const std::vector< shared_ptr< ftFaceBase >
> & faces, int resolution[], std::vector< shared_ptr< GeneralMesh > > & meshes) const
', ['../class_go_1_1_surface_model.html#a2ffe542d114f0ae72313b5f99fd2b3d9',1,'
Go::SurfaceModel::tessellate(const std::vector< shared_ptr&
lt; ftFaceBase > > & faces, double density, std::vector< shared_ptr&
lt; GeneralMesh > > & meshes) const ', ['../
class_go_1_1_curve_tessellator.html#a2903e0b478f9c9bc52de4b78a2e58554',1,'
Go::CurveTessellator::tessellate()', ['../
class_go_1_1_line_cloud_tessellator.html#a741142d2fb183073f3999edb0987490
',1,'Go::LineCloudTessellator::tessellate()', ['../
class_go_1_1_noop_tessellator.html#a098bad44d4975404a9ecbf097a895423',1,'
Go::NoopTessellator::tessellate()', ['../
class_go_1_1_parametric_surface_tessellator.html#
af0a157fa8039d15ed49d25eccf1d6d8b',1,'Go::ParametricSurfaceTessellator::tessellate()'
], ['../class_go_1_1_rectangular_surface_tessellator.html#
a98bc65dc65f847c17602033717c41843',1,'Go::RectangularSurfaceTessellator::tessellate
()', ['../class_go_1_1_rect_grid_tessellator.html#
a37c7dac1717250aae0fe930161e4406b',1,'Go::RectGridTessellator::tessellate()', ['../
class_go_1_1_tessellator.html#ac990c28ea9a36e37b26edd30cc523ac3',1,'
Go::Tessellator::tessellate()', ['../
class_go_1_1_rectangular_volume_tessellator.html#
a81d00516682f244ebac9097b6de6da78',1,'Go::RectangularVolumeTessellator::tessellate()'
], ['../class_go_1_1_volume_model.html#afc3da8a531cbdl1a8e65853e2933abe00',1,'
Go::VolumeModel::tessellate(std::vector< shared_ptr&
lt; GeneralMesh > > & meshes) const ', ['../
class_go_1_1_volume_model.html#aef65bfbbaa2e8f5a66fc0b23630861b3',1,'

```

```

Go::VolumeModel::tessellate(int resolution[], std::vector<< shared_ptr<<
GeneralMesh > > & amp; meshes) const ', ['../class_go_1_1_volume_model.html#
a3e034d7fa26dd97bb9cb8733e50c82c8', 1, 'Go::VolumeModel::tessellate(double density,
std::vector<< shared_ptr<< GeneralMesh > > & amp; meshes) const ']]],
['tessellatedctrpolygon', ['tessellatedCtrPolygon', ['../
class_go_1_1_composite_curve.html#accf52036d9edefe396c90df496e8ea2e', 1, '
Go::CompositeCurve::tessellatedCtrPolygon(std::vector<< shared_ptr
<< LineCloud > > & amp; ctr_pol) const '], ['../class_go_1_1_composite_curve
.html#a98d7786acc00fe2d2b642beaff493324', 1, '
Go::CompositeCurve::tessellatedCtrPolygon(const std::vector<<
shared_ptr<< ParamCurve > > & amp; curves, std::vector<< shared_ptr<&
lt; LineCloud > > & amp; ctr_pol) const '], ['../
class_go_1_1_composite_model.html#a60e568eb9b8531f2bc5de60acb9d8fd7', 1, '
Go::CompositeModel::tessellatedCtrPolygon()'], ['../
class_go_1_1_curve_model.html#a0bc9f19a78ec7aaac8b655ea7e25bc7b', 1, '
Go::CurveModel::tessellatedCtrPolygon()'], ['../
class_go_1_1_surface_model.html#a42d3923e99fa793ca34de60ac335767c53f1', 1, '
Go::SurfaceModel::tessellatedCtrPolygon(std::vector<< shared_ptr<&
lt; LineCloud > > & amp; ctr_pol) const '], ['../
class_go_1_1_surface_model.html#a81fcb1beb1ddd467c6fd34f44a4351938', 1, '
Go::SurfaceModel::tessellatedCtrPolygon(const std::vector<<
shared_ptr<< ftFaceBase > > & amp; faces, std::vector<< shared_ptr<< LineCloud > > & amp; ctr_pol)
const '], ['../class_go_1_1_volume_model.html#ad7cfeb7a9a77a4f4d1d40e7aedef0de3', 1, '
Go::VolumeModel::tessellatedCtrPolygon()']]]],
['tessellator', ['Tessellator', ['../class_go_1_1_tessellator.html', 1, '
Go']]],
['tessellator', ['tessellator', ['../class_data_handler.html#
ab7caedcbf1e13f7456ce14b84c3b6789', 1, 'DataHandler::tessellator()'], ['../classgv_data.html#
a56fb6a11bc06f0998bd20a67a9274836', 1, 'gvData::tessellator()']]],
['tessellator_2eh', ['Tessellator.h', ['../_tessellator_8h.html', 1, '']]],
['tessellator_5f', ['tessellator_', ['../class_data_handler.html#
ac6931f251770a4e9b125d18bb2711092', 1, 'DataHandler']]],
['tessellatorutils_2eh', ['TessellatorUtils.h', ['../_tessellator_utils_8h.html', 1, '']]],
['test1', ['test1', ['../example_8cpp.html#ac0697ad85fb379446222fcc2df02c778', 1, '
example.cpp']]],
['test2', ['test2', ['../example_8cpp.html#ac7865afb3b9f6d710b1df18e501c94d6', 1, '
example.cpp']]],
['test3', ['test3', ['../example_8cpp.html#a727a32bdd7e2491824ca4acd62be9b1a', 1, '
example.cpp']]],
['test4', ['test4', ['../example_8cpp.html#acd897830273f9673b3894e831ceb3423', 1, '
example.cpp']]],
['test5', ['test5', ['../example_8cpp.html#aad4ba83ae697b892cf4711aa7170c1f3', 1, '
example.cpp']]],
['test_5fcyclic_5fknots', ['TEST_CYCLIC_KNOTS', ['../
tstcyclknt_8c.html#a77b5bb7a941d583a9aee41d2be40d163', 1, '
TEST_CYCLIC_KNOTS():#160

```

Definition at line 35 of file all\_15.js.

### 30.1381.1.4 double &gt;

Definition at line 35 of file all\_15.js.

### 30.1381.1.5 PrParamUtil h

Definition at line 27 of file all\_15.js.

### 30.1381.1.6 std::vector &lt;

Definition at line 27 of file all\_15.js.

## 30.1381.1.7 var searchData

## Initial value:

```

=
[
 ['the_20filter_20routines_20main_20page', ['The filter routines main page', ['./filters_mainpage.html', 1, '']],
 ['the_20half_20dedge_20data_20structure_20and_20adaption_20to_20ttl', ['The Half-Edge Data Structure and Adaption to TTL', ['./halfedge.html', 1, '']],
 ['the_20g2_20dformat_20c_20gotools_20file_20format_20for_20geometry_20entities', ['The g2-format, GoTools file format for geometry entities', ['./streamable_doc.html', 1, '']],
 ['t', ['t', ['./class_n_e_w_m_a_t_l_l_matrix_type.html#a87734af76e2bd752f88e7b4976b52bbb', 1, 'NEWMAT::MatrixType::t()'], ['./class_n_e_w_m_a_t_l_l_matrix_band_width.html#a9282c4cccc63ddeadc55483508c5c2794', 1, 'NEWMAT::MatrixBandWidth::t()'], ['./class_n_e_w_m_a_t_l_l_base_matrix.html#a9b2e4b5ac85906a6c0007c40df340b33', 1, 'NEWMAT::BaseMatrix::t()'], ['./namespace_go.html#a95d383314d9ef7d277424ee672473063ad80deac00a5f9e9bea01b087b3bc6474', 1, 'Go::T()']],
 ['t1', ['t1', ['./class_pr_triangle.html#a3c5e90139c8ad3fe42bf807f5a68b7ba', 1, 'PrTriangle::t1() const'], ['./class_pr_triangle.html#a3b7f193f51f1ba9aceeeae8d414fld3a', 1, 'PrTriangle::t1()']],
 ['t2', ['t2', ['./class_pr_triangle.html#a3b4afbf990eb31ec311069ee83b5b01b', 1, 'PrTriangle::t2() const'], ['./class_pr_triangle.html#a41d3422b3e374cba8ed081021eb43fa7', 1, 'PrTriangle::t2()']],
 ['t3', ['t3', ['./class_pr_triangle.html#a9aa253de3e236af87f94b8dc5c5cfl9f', 1, 'PrTriangle::t3() const'], ['./class_pr_triangle.html#ac1063043a62da70ab3a33dfad452e1f2', 1, 'PrTriangle::t3()']],
 ['t_5f', ['t_', ['./struct_go_1_l_param_surface_1_degenerate__info.html#aa4a1b5a04e6f9581c46bf8c50ec6e408', 1, 'Go::ParamSurface::degenerate_info::t_()'], ['./class_pr_filterbank.html#a8ba3a2c1b525d44832be09fc86afe136', 1, 'PrFilterbank::t_()'], ['./class_pr_threshold.html#a561PrThreshold::t_()']],
 ['tab_5fstring', ['TAB_STRING', ['./sisl__view__demo_8cpp.html#a0d3b5deaf6bcb60d1dfb8c6dbc234446', 1, 'sisl_view_demo.cpp']],
 ['table_5f', ['table_', ['./class_go_1_ltp_topology_table.html#a144bdf017ad7970a1ald85386652fbd7', 1, 'Go::tpTopologyTable']],
 ['tag', ['Tag', ['./class_n_e_w_m_a_t_l_l_general_matrix.html#ac86f41d9beeda0ba938e8f777167bf47', 1, 'NEWMAT::GeneralMatrix::Tag() const'], ['./class_n_e_w_m_a_t_l_l_general_matrix.html#acb8f3d5391d96487f1094e4914f88585', 1, 'NEWMAT::GeneralMatrix::tag()']],
 ['tangent', ['tangent', ['./class_go_1_lft_curve_segment.html#ad3cf5a26861e7ed344ccf146e59d0457', 1, 'Go::ftCurveSegment::tangent()'], ['./class_go_1_lft_curve.html#af97541ce68c9b3a6769d16d38b09267c', 1, 'Go::ftCurve::tangent()'], ['./class_go_1_lft_edge.html#a62badabelbae46b1ba238ead31889f51', 1, 'Go::ftEdge::tangent()'], ['./class_go_1_lft_edge_base.html#aed5b154957fdd7a97df9726053c32eef', 1, 'Go::ftEdgeBase::tangent()'], ['./class_go_1_lft_sf_edge.html#aec5af9042c2a159c2b8dce97f88eb4ad', 1, 'Go::ftSSfEdge::tangent()'], ['./class_go_1_ltp_edge.html#ad854b7324cfe2d5ef2edc4342d97c267', 1, 'Go::tpEdge::tangent()']],
 ['tangent_5f2d_5f1', ['tangent_2d_1', ['./struct_go_1_l_second_order_properties.html#a2a04d22fd412a880614e77120022ed82', 1, 'Go::SecondOrderProperties']],
 ['tangent_5f2d_5f2', ['tangent_2d_2', ['./struct_go_1_l_second_order_properties.html#ae150a2eb73072238b54e2c3f5e130dff', 1, 'Go::SecondOrderProperties']],
 ['tangent_5f2d_5fis_5fcached', ['tangent_2d_is_cached', ['./struct_go_1_l_second_order_properties.html#a43338efc4bd449d02635099c6ae9e082', 1, 'Go::SecondOrderProperties']],
 ['tangent_5f3d', ['tangent_3d', ['./struct_go_1_l_second_order_properties.html#a186f4ccbf51d36ce342dlab3cala7eda', 1, 'Go::SecondOrderProperties']],
 ['tangent_5fis_5foriented', ['tangent_2d_is_oriented', ['./struct_go_1_l_second_order_properties.html#a77419f5e3c6b905374de860cf8713de6', 1, 'Go::SecondOrderProperties']],
 ['tangent_5ftol', ['tangent_tol', ['./class_go_1_l_intersection_point.html#a44981526bad57ccb080ad4735311d2d2', 1, 'Go::IntersectionPoint']],
 ['tangentcone', ['tangentCone', ['./class_go_1_l_bounded_surface.html#ac311070548b96d789d21758e5ee7ec88', 1, 'Go::BoundedSurface::tangentCone()'], ['./class_go_1_l_composite_surface.html#a9bfe05b31ca0bab1be6adb707586a18f', 1, 'Go::CompositeSurface::tangentCone()'], ['./class_go_1_l_cylinder.html#a583676f3507e5b9e5cb8d3cd9f2772f0', 1, 'Go::Cylinder::tangentCone()'], ['./class_go_1_l_disc.html#a7a7b3943b053c0a274af05b8d5284cc9', 1, 'Go::Disc::tangentCone()'], ['./class_go_1_l_param_surface.html#a0892308334265519c050b32f050e065e', 1, 'Go::ParamSurface::tangentCone()'], ['./class_go_1_l_plane.html#a08c1d6807c5866963922e8265393fa62', 1, 'Go::Plane::tangentCone()'], ['./class_go_1_l_sphere.html#a3f02936582bc72857d2af3b1b9962c63', 1, 'Go::Sphere::tangentCone()'], ['./class_go_1_l_spline_surface.html#ac4f32c873766da700a3f78ebf5d95c37', 1, 'Go::SplineSurface::tangentCone()'], ['./class_go_1_l_surface_of_linear_extrusion.html#ac9cf10e65169359291347f3569eeadb2', 1, 'Go::SurfaceOfLinearExtrusion::tangentCone()'], ['./class_go_1_l_surface_of_revolution.html#a07bc005c7579121b29e4ef1e0ed', 1, 'Go::SurfaceOfRevolution::tangentCone()'], ['./class_go_1_l_torus.html#ac8849f0b6a5a7083c8286ebf83d8346f', 1, 'Go::Torus::tangentCone()'], ['./class_go_1_l_r_spline_surface.html#acde98e9907ecaa6b5ba085540f50db94', 1, 'Go::LRSplineSurface::tangentCone()'], ['./class_go_1_l_cone_volume.html#alf7d1a81eae6e98a3422a2320aa932bd', 1, 'Go::ConeVolume::tangentCone()'], ['./class_go_1_l_cylinder_volume.html#a4d5146512b6324e7616b54c3c7fb3a51', 1, 'Go::CylinderVolume::tangentCone()'], ['./class_go_1_l_parallelepiped.html#a65b0a65033c8c65f6a2884fe2e6a11ca', 1, 'Go::Parallelepiped::tangentCone()'], ['./class_go_1_l_parallel_volume.html#af5b51bc450da49f76b73f038618a57b6', 1, 'Go::ParallelVolume::tangentCone()'], ['./class_go_1_l_sphere_volume.html#a6f5a787cfalabfd98de924e4dc36f262', 1, 'Go::SphereVolume::tangentCone()'], ['./class_go_1_l_spline_volume.html#ac01dec30b44329e7cc9d97a02f9c85d7', 1, 'Go::SplineVolume::tangentCone()'], ['./class_go_1_l_surface_on_volume.html#a346db16aaa51c7e3cf525d4f4918bc2d', 1, 'Go::SurfaceOnVolume::tangentCone()'], ['./class_go_1_l_torus_volume.html#a4e5d384ca79038f368e5f447959407c2', 1, 'Go::TorusVolume::tangentCone()']],
 ['tangentdomain', ['TangentDomain', ['./namespace_go.html#a683c7cd0653c849d16af423f40a6daf6', 1, 'Go']],
 ['tangential_5fpoin', ['TANGENTIAL_POINT', ['./namespace_go.html#a3f21d9ac1d33fe9bf8364573e0126af2a7a751867640d143dba6d355e3ad044ce', 1, 'Go']],
 ['tangentisoriented', ['tangentIsOriented', ['./class_go_1_l_intersection_point.html#a6de8bbbe86de15714906fb08e8469a27', 1, 'Go::IntersectionPoint']],
 ['tangentpointinginwards', ['tangentPointingInwards', ['./class_go_1_l_intersection_point.html#a7d2d0f72efc426b982994f0355860fda', 1, 'Go::IntersectionPoint']],
 ['tanlenfromradius', ['tanLenFromRadius', ['./namespace_go.html#a87059cc6cddba0b64ae502b9c4d52a52', 1, 'Go']],
],
 ['tanthetaovertwo', ['tanThetaOverTwo', ['

```



```
../class_pr_prm_sym_mean_value.html#a9f07830f6d01f9430204d6b090da8823',1,'PrPrmSymMeanValue::tanThetaOverTwo()'],['
../class_pr_prm_wachspres.html#a3b09c67987bea5e0a6f2ac95755ca258',1,'PrPrmWachspres::tanThetaOverTwo()'],['
../_pr_param_util_8h.html#acf689ef17a246312b527f1c2b8fa1266',1,'tanThetaOverTwo():
```

Definition at line 1 of file all\_15.js.

## 30.1382 doc/html/search/all\_16.js File Reference

### Variables

- var [searchData](#)

#### 30.1382.1 Variable Documentation

##### 30.1382.1.1 var searchData

Definition at line 1 of file all\_16.js.

## 30.1383 doc/html/search/all\_17.js File Reference

### Variables

- var [searchData](#)
- & [gt](#)

#### 30.1383.1 Variable Documentation

##### 30.1383.1.1 & gt

Definition at line 14 of file all\_17.js.

## 30.1383.1.2 var searchData

## Initial value:

```
=
[
 ['v', ['v', ['../class_go_1_1ft_point.html#a5f721b935eeefed8e81d0cf416cf0f2a', 1, 'Go::ftPoint::v()'], ['
 ../class_pr_nested_node.html#a4fb696c20bedfb1aa45377050dfc8509', 1, 'PrNestedNode::v() const'], ['
 ../class_pr_nested_node.html#aa0688bdd1fcd5d49dd45e3cbb56e0cec', 1, 'PrNestedNode::v()'], ['
 ../class_pr_node.html#a7001ff1832d48c579c1b69deddb00db9', 1, 'PrNode::v() const'], ['../class_pr_node.html#a8b59204989f4b
 PrNode::v()'], ['../class_pr_p_g_node.html#adaalf1049de852bdad29fdd715d39419', 1, 'PrPGNode::v() const'], ['
 ../class_pr_p_g_node.html#a8476c3c6b52627b627a31ec7f0b0d046', 1, 'PrPGNode::v()']],
 ['v_5f', ['v_', ['../class_go_1_1ft_point.html#ae66d28de47384fb0fd2f113281a83848', 1, 'Go::ftPoint::v_()'], ['
 ../class_pr_prm_shp_pres.html#a1b91934fd02891b562a62a7def09b645', 1, 'PrPrmShpPres::v_()']],
 ['v_5fmax', ['v_max', ['
 ../struct_go_1_1_l_r_spline_surface_1_1_b_s_key.html#a6751c5eed5caa610e8dac5f3558b4702', 1, 'Go::LRSplineSurface::BSKey'
]],
 ['v_5fmin', ['v_min', ['
 ../struct_go_1_1_l_r_spline_surface_1_1_b_s_key.html#a917055d1f2ba2783786ba6345d73cbd8', 1, 'Go::LRSplineSurface::BSKey:
]],
 ['v_5fmult1', ['v_mult1', ['
 ../struct_go_1_1_l_r_spline_surface_1_1_b_s_key.html#ac8f082b1e3b79de0d2531198fd3648af', 1, 'Go::LRSplineSurface::BSKey'
]],
 ['v_5fmult2', ['v_mult2', ['
 ../struct_go_1_1_l_r_spline_surface_1_1_b_s_key.html#a04b64c2a027326729a138282d8015159', 1, 'Go::LRSplineSurface::BSKey'
]],
 ['val', ['val', ['../struct_go_1_1_inverse_factorial.html#ab0c85f28bd408da23e82368e710e1fa2', 1, '
 Go::InverseFactorial']],
 ['valid', ['valid', ['../class_go_1_1_bounding_box.html#a3082546a7b34a3dd3e51b7ae822f74c6', 1, '
 Go::BoundingBox::valid()'], ['
 ../class_n_e_w_m_a_t_l_1_matrix_type.html#af1adc0086f5d4c3328ad4a698fa57c90a0dad346d65e5540202aecf9f55dc0d', 1, 'NEWMAT
]],
 ['validate', ['validate', ['../class_go_1_1_intersection_pool.html#ac09a6a264bdf2d5e8136774287a5293b', 1, '
 Go::IntersectionPool::validate()'], ['../class_go_1_1tp_topology_table.html#a82cf7ad03299185589e0ac28bf6c30a6',
 1, 'Go::tpTopologyTable::Validate()']],
 ['validatesiblingpools', ['validateSiblingPools', ['
 ../class_go_1_1_intersector.html#adc08c6934cb9c8289adf1a3762bdc59', 1, 'Go::Intersector']],
 ['validentity', ['validEntity', ['../struct_go_1_1_entity_list.html#a4579a25ef0627fd75736eec3082a5fc3', 1, '
 Go::EntityList']],
 ['value', ['value', ['
 ../struct_go_1_1_factorial.html#a579daed3b4275abe013c8f88077f888fa4acbf42abf2d3a50f62969f536eacb98', 1, 'Go::Factorial::v
]],
 ['value', ['value', ['
 ../struct_go_1_1_factorial_3_011_01_4.html#a654cb8f1088a190314946d0cb460482caf04561aa2fcbf1fb66fbc964a355d268', 1, 'Go::F
]],

```

Definition at line 1 of file all\_17.js.

## 30.1384 doc/html/search/all\_18.js File Reference

## Variables

- var [searchData](#)
- garch [cpp](#)

## 30.1384.1 Variable Documentation

30.1384.1.1 garch [cpp](#)

Definition at line 3 of file all\_18.js.

## 30.1384.1.2 var searchData

## Initial value:

```
=
[
 ['want_5ffstream', ['WANT_FSTREAM', ['../garch_8cpp.html#a37d625862c54954e237886ad4bb5e75f', 1, '
 WANT_FSTREAM():
]],

```

Definition at line 1 of file all\_18.js.

## 30.1385 doc/html/search/all\_19.js File Reference

### Variables

- var [searchData](#)
- transfutils [cpp](#)

### 30.1385.1 Variable Documentation

#### 30.1385.1.1 transfutils cpp

Definition at line 6 of file all\_19.js.

#### 30.1385.1.2 var searchData

### Initial value:

```
=
[
 ['x', ['x', ['../class_r_b_d___c_o_m_m_o_n_l_l_r_l___r_l.html#a4812b378711ea5f78a5441b96134f569', 1,
 RBD_COMMON::Rl_Rl::x()'], ['../classhetriang_l_l_dart.html#a54e0bbd87448ac2b3e9bddc8db856c59', 1, 'hetriang::Dart::x()'],
 ['../class_go_l_lttl_point.html#a2edfd5b3a2a64c8c3af3e02c06ae2357', 1, 'Go::ttlPoint::x()'], ['
 ../classhetriang_l_l_node.html#a6d6619cfedeaef9e468da7670da5f', 1, 'hetriang::Node::x()'], ['
 ../class_go_l_l_array.html#a3c8aa548649a570e01713ce17c1d784c', 1, 'Go::Array::x() const'], ['
 ../class_go_l_l_array.html#ac87b81b08bef59f57cc13c5531115110', 1, 'Go::Array::x()'], ['../class_pr_nested_node.html#a69747
 PrNestedNode::x() const'], ['../class_pr_nested_node.html#a613b6c4885ec0097d356410119e8a4b9', 1, '
 PrNestedNode::x()'], ['../class_pr_node.html#a5a5865db2f78a4ccfd762634b4899e39', 1, 'PrNode::x() const'], ['
 ../class_pr_node.html#af1089d11e7ee670efb2007351752c1b6', 1, 'PrNode::x()'], ['
 ../class_pr_p_g_node.html#af56272e39fbc9e4b0273b83d9d9aa970', 1, 'PrPGNode::x() const'], ['../class_pr_p_g_node.html#a373
 PrPGNode::x()'], ['../classvector3t.html#a85c1f558e7e9d6800afa25b4d95b4e1f', 1, 'vector3t::x(void) const'], ['
 ../classvector3t.html#a4e3bd29a46eclbcl9dc4a06313829e6a', 1, 'vector3t::x(void)'], ['
 ../classshed_l_l_dart.html#a39d20327f3eeafbc53f06f79f53d840', 1, 'hed::Dart::x()'], ['
 ../classshed_l_l_node.html#ae5240e90452911b4635f8668f2d4129c', 1, 'hed::Node::x()']],
 ['x_5faxis_5f', ['x_axis_', ['../class_go_l_l_cone.html#a97719276e5e89fc874cee2df6327c46d', 1,
 Go::Cone::x_axis_()'], ['../class_go_l_l_cylinder.html#aa070cf5deafb3498f2601d9a2847ffc5', 1, 'Go::Cylinder::x_axis_()'],
 ../class_go_l_l_sphere.html#a6865f0ab0d027141c49dceb412a613b4', 1, 'Go::Sphere::x_axis_()'], ['
 ../class_go_l_l_torus.html#ae07499b7a7a24f9c29bcf451ef48b14b', 1, 'Go::Torus::x_axis_()']],
 ['xfixed', ['XFIXED',
 ../namespace_go.html#a821a750d4a6740898072cf759246eb87a1e4145becb15ce54486dd57c554a8716', 1, 'Go']],
 ['xrot', ['xrot', ['../transfutils_8h.html#a5175ae8b353bb511616aec812502269d', 1, 'xrot() :
```

Definition at line 1 of file all\_19.js.

## 30.1386 doc/html/search/all\_1a.js File Reference

### Variables

- var [searchData](#)
- transfutils [cpp](#)

### 30.1386.1 Variable Documentation

#### 30.1386.1.1 transfutils cpp

Definition at line 6 of file all\_1a.js.

## 30.1386.1.2 var searchData

## Initial value:

```
=
[
 ['y', ['y', ['../classhetriang_l1_l_dart.html#af210f3f33a71b195c745e54cd7165070', 1, 'hetriang::Dart::y()', ['
 ../class_go_l1_lttl_point.html#aa68070e89bc646943dcfff514e3d194f', 1, 'Go::ttlPoint::y()', ['
 ../classhetriang_l1_l_node.html#a9e14e02fa7d2ec93a21f26108a6d1b36', 1, 'hetriang::Node::y()', ['
 ../class_go_l1_l_array.html#a7503e9570b4c3e5df9d7c6a2f2882df4', 1, 'Go::Array::y() const ', ['
 ../class_go_l1_l_array.html#ad343c6549d1fd8aae5abfb88ae509cf6', 1, 'Go::Array::y()', ['../class_pr_nested_node.html#ac2aa9
 PrNestedNode::y() const ', ['../class_pr_nested_node.html#a437a7f299f4aeb739d3c86fd434a5673', 1, 'PrNestedNode::y()
 ', ['../class_pr_node.html#ae837126218bc3668de1896a232be37bf', 1, 'PrNode::y() const ', ['
 ../class_pr_node.html#a5aa29384b673ee37acbd1cf7f95059e5', 1, 'PrNode::y()', ['
 ../class_pr_p_g_node.html#a73c1340c393a2d29be3ab2aca97dfb88', 1, 'PrPGNode::y() const ', ['../class_pr_p_g_node.html#acf6
 PrPGNode::y()'], ['../classvector3t.html#ab0f2c22060b6068c3104d8c6992a8b89', 1, 'vector3t::y(void) const ', ['
 ../classvector3t.html#af88d3df5d1b648785874485afc10a876', 1, 'vector3t::y(void)', ['
 ../classhed_l1_l_dart.html#a20849660f038888f05ba65daf0a99ae5', 1, 'hed::Dart::y()'], ['../classhed_l1_l_node.html#ab71715188
 1, 'hed::Node::y()']],
 ['y_5faxis_5f', ['y_axis_', ['../class_go_l1_l_cone.html#a04725fd933cb2a6d53a58ae1ebd24f11', 1,
 Go::Cone::y_axis_()], ['../class_go_l1_l_cylinder.html#aeaf20ee6ba23475fb4926e3c31274742', 1, 'Go::Cylinder::y_axis_()'], ['
 ../class_go_l1_l_sphere.html#a3b495bbd4eaa7a34ea93ad464b7c6c5a', 1, 'Go::Sphere::y_axis_()'], ['
 ../class_go_l1_l_torus.html#a045f285b84522d667f44f32cfc428810', 1, 'Go::Torus::y_axis_()']],
 ['yfixed', ['YFIXED', ['
 ../namespace_go.html#a821a750d4a6740898072cf759246eb87a344e219d46e24ae18ec2ff7fe11afddd', 1, 'Go']],
 ['yrot', ['yrot', ['../transfutils_8h.html#aa796e5a13fbe734d62adecbe1c549960', 1, 'yrot():

```

Definition at line 1 of file all\_1a.js.

## 30.1387 doc/html/search/all\_1b.js File Reference

## Variables

- var [searchData](#)
- [jquery.js](#)

## 30.1387.1 Variable Documentation

## 30.1387.1.1 jquery.js

Definition at line 3 of file all\_1b.js.

## 30.1387.1.2 var searchData

## Initial value:

```
=
[
 ['z', ['z', ['../class_go_l1_lttl_point.html#ac3852f5d65cea6fc03ded418c52b1224', 1, 'Go::ttlPoint::z()'], ['
 ../classhetriang_l1_l_node.html#a3a3e75bffc96f2230699b7ec641afa70', 1, 'hetriang::Node::z()'], ['
 ../class_go_l1_l_array.html#aae59550f97c7b789f9fedfef6214fca4', 1, 'Go::Array::z() const ', ['
 ../class_go_l1_l_array.html#ab13fabbb125a68879f994e0c0a492755b', 1, 'Go::Array::z()'], ['
 ../class_pr_nested_node.html#ae5e6fab6058c33f1c08c5e7b335502ba', 1, 'PrNestedNode::z() const ', ['../class_pr_nested_node
 PrNestedNode::z()'], ['../class_pr_node.html#ad6eec21be11c8ba6ee3a0e4fe9d4693a', 1, 'PrNode::z() const ', ['
 ../class_pr_node.html#a323e95d93b0be8200fbfb5cf0b98afcl', 1, 'PrNode::z()'], ['
 ../class_pr_p_g_node.html#a302fb7445d4022c990244d61768aaa22', 1, 'PrPGNode::z() const ', ['
 ../class_pr_p_g_node.html#aab8d9580b6112f5342804c503aa5aa78', 1, 'PrPGNode::z()'], ['../classvector3t.html#ac756398d08e462
 '], ['../classvector3t.html#a40e10484bca97074094c63c9ea3f458b', 1, 'vector3t::z(void)'], ['
 ../classhed_l1_l_node.html#a812130c3e029cdec9bc9a47a1e53a7', 1, 'hed::Node::z()'], ['
 ../jquery_8js.html#adc18d83abfd9f87d396e8fd6b6ac0fe1', 1, 'Z():

```

Definition at line 1 of file all\_1b.js.

## 30.1388 doc/html/search/all\_1c.js File Reference

### Variables

- var [searchData](#)

### 30.1388.1 Variable Documentation

#### 30.1388.1.1 var searchData

Definition at line 1 of file all\_1c.js.

## 30.1389 doc/html/search/all\_2.js File Reference

### Variables

- var [searchData](#)
- mouse [cpp](#)
- mouse int [dim](#)
- mouse int [double aepsge](#)
- mouse int [double](#) int [constdir =0](#)
- mouse int [double](#) int [bool repara](#)
- PrParamUtil [h](#)

### 30.1389.1 Variable Documentation

#### 30.1389.1.1 mouse int double aepsge [property], [bound]

Definition at line 120 of file all\_2.js.

#### 30.1389.1.2 mouse int double int constdir =0 [property], [bound]

Definition at line 169 of file all\_2.js.

#### 30.1389.1.3 mouse cpp

Definition at line 120 of file all\_2.js.

#### 30.1389.1.4 mouse int dim [property], [bound]

Definition at line 120 of file all\_2.js.

## 30.1389.1.5 PrParamUtil h

Definition at line 178 of file all\_2.js.

## 30.1389.1.6 mouse int double int bool repar [property], [bound]

## Initial value:

```
=true)'], ['../class_go_1_l_approx_surf.html#ae046f4d99fff00a02c1707e92b5053dd', 1, '
 Go::ApproxSurf::ApproxSurf(shared_ptr<t; SplineSurface &
gt; &srf, const std::vector<t; double > &points, const std::vector<
lt; double > &parvals, int dim, double aepsge, int
constdir=0, bool approx_orig=false, bool close_belt=false, int nmb_stabil=0, bool
repar=true)'], ['../class_go_1_l_approx_surf.html#
a7781c1001f19a97499b193907d693b15', 1, 'Go::ApproxSurf::ApproxSurf()']],
['approxsurf', ['ApproxSurf', ['../class_go_1_l_approx_surf.html', 1, '
Go']],
['approxsurf_2eh', ['ApproxSurf.h', ['../_approx_surf_8h.html', 1, '']],
['approx_tol_5f', ['approx_tol_', ['../class_go_1_lft_smooth_surf.html#
aec9cdafc2633142a81e2fd4a8388ba45', 1, 'Go::ftSmoothSurf::approx_tol()'], ['../
class_go_1_lft_surface_set.html#a6af113d25a69c498a856f7c1e5e7a2b3', 1, '
Go::ftSurfaceSet::approx_tol()'], ['../
class_go_1_l_surface_model.html#a6472ae026aa52cebc79d2f43f0917e64', 1, '
Go::SurfaceModel::approx_tol()']],
['approxvolparamcurve', ['approxVolParamCurve', ['../namespace_go_1_l_volume_tools.html#
aa500ad08b2e111caedb2c15ba3c6999f', 1, 'Go::VolumeTools']],
['approxweight_5f', ['approxweight_', ['../class_go_1_lft_surface_set.html#
a0a15b621cc8fadbb91909c748c5d4f1', 1, 'Go::ftSurfaceSet']],
['aq', ['aQ', ['../jquery_8js.html#a79eb58dc6cdf0aef563d5dcded27df5', 1, 'jquery.js']],
['ar_5fld_5fft', ['ar_ld_ft', ['../class_n_e_w_m_a_t_1_l_f_t___controller.html#
aa2d9e1babelle9df86986a62dd93ebac', 1, 'NEWMAT::FFT_Controller']],
['arclength', ['arcLength', ['../class_go_1_lft_curve_segment.html#
a953153d17ea3bfff870bf270c716d487d', 1, 'Go::ftCurveSegment::arcLength()'], ['../
class_go_1_lft_curve.html#a054c7514fc2e82eafcbfc8e1714061ea', 1, '
Go::ftCurve::arcLength()']],
['area', ['area', ['../class_go_1_lft_surface.html#
aad67b36ef653ff7e4c09ea06456a0d0', 1, 'Go::ftSurface::area()'], ['../
class_go_1_l_bounded_surface.html#a351925baa7e536730663a87ea0f0fce006', 1, '
Go::BoundedSurface::area()'], ['../
class_go_1_l_composite_surface.html#a6a68591f99fe0ce92a8442843a6e403b', 1, '
Go::CompositeSurface::area()'], ['../
class_go_1_l_elementary_surface.html#ab20ceab1a6924addf76e0d04850de566', 1, '
Go::ElementarySurface::area()'], ['../
class_go_1_l_param_surface.html#a0417c918e78108fb3a22f969fcc0498a', 1, '
Go::ParamSurface::area()'], ['../
class_go_1_l_spline_surface.html#ab0e8c17c1d43ddab2fd56ee045c2c183', 1, '
Go::SplineSurface::area()'], ['../
class_go_1_l_surface_of_linear_extrusion.html#
af7839cc0ddb64b5033ae98b733e0ea14', 1, 'Go::SurfaceOfLinearExtrusion::area()'], ['../
class_go_1_l_surface_of_revolution.html#a6b877c9c54faa2f1fec69be7d303118c
', 1, 'Go::SurfaceOfRevolution::area()'], ['../
class_go_1_l_element2_d.html#a2321c59da3c5c143cfb2ccca7b791dbf', 1, '
Go::Element2D::area()'], ['../class_go_1_l_l_r_spline_surface
.html#aab242113dae866aa4e4abf04dfc8dc85', 1, 'Go::LRSplineSurface::area()'], ['../
class_go_1_l_surface_on_volume.html#ad16e1c297d24c01995f1c407a4f6d418', 1, '
Go::SurfaceOnVolume::area()'], ['../namespace_go.html#
a8cc8f744af2fc5541e493ffdd3d2e1f5', 1, 'Go::area(const Array<t; T, 2 &t; *c)'], ['../
namespace_go.html#a04f0f159a17bcc67285bb7349debd945', 1, 'Go::area(const Array<
lt; T, 3 &t; *c)'], ['../_pr_param_util_8h.html#a88d3831b85d9d3912354527e5fe6d73c', 1, '
area(const double &x0, const double &y0, const double &x1, const double &
&y1, const double &x2, const double &y2):
```

Definition at line 169 of file all\_2.js.

## 30.1389.1.7 var searchData

Definition at line 1 of file all\_2.js.

## 30.1390 doc/html/search/all\_3.js File Reference

### Variables

- var [searchData](#)
- jquery [js](#)

### 30.1390.1 Variable Documentation

#### 30.1390.1.1 jquery js

Definition at line 3 of file all\_3.js.

#### 30.1390.1.2 var searchData

##### Initial value:

```
=
[
 ['b', ['b', ['../class_go_1_1_line2_d_int.html#a1bf14760b8c49e8b71cd24fa7a38cd35', 1, 'Go::Line2DInt::b()'], [
 '../class_go_1_1_plane_int.html#aa61eda35ce195ebbb59321069d0b20f1', 1, 'Go::PlaneInt::b()'], [
 '../jquery_8js.html#aa4026ad5544b958e54ce5e106fa1c805', 1, 'b():
```

Definition at line 1 of file all\_3.js.

## 30.1391 doc/html/search/all\_4.js File Reference

### Variables

- var [searchData](#)
- jquery [js](#)
- jquery [shared\\_ptr](#) & [It](#)
- [IntersectionPoint](#) & [gt](#)
- & [amp](#)
- [pnt](#) [const](#)
- [dist](#)
- [cpos1](#)
- [double](#) [gpos2](#) []
- [pt\\_cv](#)
- [pt\\_su](#)
- [bool](#) [second\\_order](#)
- [newmat4](#) [cpp](#)
- [basisValues](#)
- [basisDerivs\\_u](#)
- [basisDerivs\\_v](#)
- [bool](#) [evaluate\\_from\\_right](#)
- [s17702d](#) [c](#)
- [s17702d](#) [int](#) [nmb\\_cvs](#)
- [init\\_basis](#)
- [double](#) [tol](#)
- [double](#) [int](#) [double](#) [double](#) [degree](#)

### 30.1391.1 Variable Documentation

#### 30.1391.1.1 `s17702d int const BsplineBasis& amp [property], [bound]`

Definition at line 176 of file `all_4.js`.

#### 30.1391.1.2 `basisDerivs_u`

Definition at line 289 of file `all_4.js`.

#### 30.1391.1.3 `basisDerivs_v`

Definition at line 289 of file `all_4.js`.

#### 30.1391.1.4 `basisValues`

Definition at line 289 of file `all_4.js`.

#### 30.1391.1.5 `s17702d c`

Definition at line 384 of file `all_4.js`.

#### 30.1391.1.6 `pnt const`

Definition at line 176 of file `all_4.js`.

#### 30.1391.1.7 `cpos1`

Definition at line 193 of file `all_4.js`.

#### 30.1391.1.8 `newmat4 cpp`

Definition at line 244 of file `all_4.js`.

#### 30.1391.1.9 `double int double double degree`

Definition at line 516 of file `all_4.js`.

#### 30.1391.1.10 `dist`

Definition at line 181 of file `all_4.js`.



**30.1391.1.11** `bool evaluate_from_right`

Definition at line 289 of file all\_4.js.

**30.1391.1.12** `double gpos2[]`

Definition at line 193 of file all\_4.js.

**30.1391.1.13** `double& gt`

Definition at line 176 of file all\_4.js.

**30.1391.1.14** `init_basis`

Definition at line 516 of file all\_4.js.

**30.1391.1.15** `jquery js`

Definition at line 3 of file all\_4.js.

**30.1391.1.16** `std::vector& lt [bound]`

Definition at line 3 of file all\_4.js.

**30.1391.1.17** `double int nmb_cvs [property], [bound]`

Definition at line 384 of file all\_4.js.

**30.1391.1.18** `pt_cv`

Definition at line 193 of file all\_4.js.

**30.1391.1.19** `pt_su`

Definition at line 193 of file all\_4.js.

**30.1391.1.20** `var searchData`**Initial value:**

```
=
[
 ['c', ['c', ['../class_go_1_1_line2_d_int.html#a267566d852799214325a5702eb5cef12', 1, 'Go::Line2DInt::c()'], [
 '../class_go_1_1_plane_int.html#a4f9d4c6288fb0ee151a466648816fca1', 1, 'Go::PlaneInt::c()'], [
 '../jquery_8js.html#abce695e0af988ece0826d9ad59b8160d', 1, 'c():
```

Definition at line 1 of file all\_4.js.

**30.1391.1.21 bool second\_order**

Definition at line 193 of file all\_4.js.

**30.1391.1.22 double int double tol**

Definition at line 516 of file all\_4.js.

**30.1392 doc/html/search/all\_5.js File Reference****Variables**

- var [searchData](#)
- [fft](#) `cpp`

**30.1392.1 Variable Documentation****30.1392.1.1 fft `cpp`**

Definition at line 17 of file all\_5.js.

**30.1392.1.2 var searchData****Initial value:**

```
=
[
 ['d', ['d', ['../struct_go_1_1_l_r_spline_surface_1_1_refinement2_d.html#a8a68737e378b8daee92599fcb690ef3c',
 ,1,'Go::LRSplineSurface::Refinement2D::d()'], ['
 ../class_go_1_1_plane_int.html#adbef9642db16186c9411dce8c3fd7c92',1,'Go::PlaneInt::d()'], ['
 ../namespace_go.html#a95d383314d9ef7d277424ee672473063ada8cdab2225fa4f764ba2ff6d553f3d4',1,'Go::D()']],
 ['dart', ['Dart', ['../classhetriang_1_1_dart.html',1,'hetriang']],
 ['dart', ['Dart', ['../classshed_1_1_dart.html',1,'hed']],
 ['dart', ['Dart', ['../classhetriang_1_1_dart.html#ab39198f2792ee004668b8fc7f6a2d54f',1,'
 hetriang::Dart::Dart()'], ['../classhetriang_1_1_dart.html#ad19d89b2aed2b3a8d928be1625c41522',1,'hetriang::Dart::Dart(Edge
 *edge, bool dir=true)'], ['../classhetriang_1_1_dart.html#a38272a3789a247be3b18bd270c7f7e0e',1,'
 hetriang::Dart::Dart(const Dart &dart)'], ['../classshed_1_1_dart.html#a141548d2908ef5847fd81370d57c07a0',1,'
 hed::Dart::Dart()'], ['../classshed_1_1_dart.html#a40d2832ae9a9d61alcc16c9b171a2d14',1,'hed::Dart::Dart(Edge *edge, bool
 dir=true)'], ['../classshed_1_1_dart.html#a75fe6236394e8e039alb3ad44582bf4f',1,'hed::Dart::Dart(const Dart
 &dart)']],
 ['data', ['Data', ['../class_n_e_w_m_a_t_1_1_simple_int_array.html#a224921d28dcde7d9131ae08d30adba79',1,'
 NEWMAT::SimpleIntArray::Data()'], ['
 ../class_n_e_w_m_a_t_1_1_simple_int_array.html#aa2caf98cf436c591d84b0c4ffa1f95d2',1,'NEWMAT::SimpleIntArray::Data() cons
 ../class_matrix_row_col.html#a9900af353a47350b3346718cd4a6007f',1,'MatrixRowCol::Data()'], ['../class_matrix_row_col.htm
 MatrixRowCol::data()']],
 ['data_5f', ['data_', ['../classgv_application.html#a15780eaf17a8bf73fece3f89300fd5b4',1,'
 gvApplication::data_()'], ['../classgv_view.html#ae6f37f89a9bf67440090589d576924db',1,'gvView::data_()']],
 ['data_5fpoints_5f', ['data_points_', ['
 ../struct_go_1_1_l_s_smooth_data.html#a48391b2e0bfef8d763a55015de021dee',1,'Go::LSSmoothData']],
 ['datahandler', ['DataHandler', ['../class_data_handler.html',1,'DataHandler'], ['
 ../class_data_handler.html#a51d845c1d9bbeef44607ec7592ba3c1d',1,'DataHandler::DataHandler()']],
 ['datahandler_2eh', ['DataHandler.h', ['../_data_handler_8h.html',1,'']],
 ['datahandlervolandlr', ['DataHandlerVolAndLR', ['../class_data_handler_vol_and_l_r.html',1,'
 DataHandlerVolAndLR'], ['../class_data_handler_vol_and_l_r.html#aa9165829b2bcbfd4bbb5c2afbb4ea6c0',1,'
 DataHandlerVolAndLR::DataHandlerVolAndLR()']],
 ['datahandlervolandlr_2eh', ['DataHandlerVolAndLR.h', ['../_data_handler_vol_and_l_r_8h.html',1,'']],
 ['datalossok', ['DataLossOK', ['../class_n_e_w_m_a_t_1_1_matrix_type.html#a0e6e05568d71ac2843e99e7b4959b6e0
 ',1,'NEWMAT::MatrixType']],
 ['datapointsiz', ['dataPointSize', ['
 ../struct_go_1_1_l_s_smooth_data.html#a535cfb16e7a24d9389902429bbe5b64b',1,'Go::LSSmoothData']],
 ['dbgqqq', ['DBGqqq', ['../sisl__aux_8cpp.html#a22e3c68f7ee44d24da231675c4aceaafa',1,'sisl__aux.cpp']],
 ['dct', ['DCT', ['../namespace_n_e_w_m_a_t.html#a8d6c3e8e2abaccf76b86956d403842cb',1,'NEWMAT::DCT()'], ['
 ../fft_8cpp.html#a924dff49d847f7fc827760a9adad7f75',1,'DCT()::
```

Definition at line 1 of file all\_5.js.

## 30.1393 doc/html/search/all\_6.js File Reference

### Variables

- var [searchData](#)
- PrPathTriangleSeq [h](#)
- PrPathTriangleSeq [const](#)
- [eval\\_2\\_crv c](#)
- [eval\\_2\\_crv Point & amp](#)
- [pnt int nder](#)
- [pnt int std::vector & lt](#)
- [Point & gt](#)

### 30.1393.1 Variable Documentation

#### 30.1393.1.1 [der Point& amp](#) [bound]

Definition at line 153 of file [all\\_6.js](#).

#### 30.1393.1.2 [eval\\_2\\_crv c](#)

Definition at line 153 of file [all\\_6.js](#).

#### 30.1393.1.3 [pnt const](#) [bound]

### Initial value:

```
=0'], ['./class_go_1_1_eval_curve_set.html#a6cad51d0922a7dedac3a31b2026ab0a4', 1,
'Go::EvalCurveSet::eval(double t)=0'], ['./
class_go_1_1_eval_curve_set.html#af94dfe476d9a31a9283263e1c7eb8bb3', 1,
Go::EvalCurveSet::eval(double t, int n, std::vector<lt; std::vector<
lt; Point > > & amp;der)=0'], ['./
class_go_1_1_eval_param_curve.html#a765082ce27f3a3ac77f9dbda11001cf1', 1,
Go::EvalParamCurve::eval(double t) const ', ['./
class_go_1_1_eval_param_curve.html#ae053eebea18f9dbe4a3289a7aeb2563d', 1,
Go::EvalParamCurve::eval(double t, int n, Point der[]) const ', ['./
class_go_1_1_eval_surface.html#a2048831767bc076026fb999eb2f1036c', 1,
Go::EvalSurface::eval(double u, double v) const =0'], ['./
class_go_1_1_eval_surface.html#ac30c4ea3682ad3433ed510bebf64c968', 1,
Go::EvalSurface::eval(double u, double v, int n, Point der[]) const =0'], ['./
class_go_1_1_int_crv_evaluator.html#ab9f4d31119d434a7399b4f57a99c7efb', 1,
Go::IntCrvEvaluator::eval(double t)'], ['./
class_go_1_1_int_crv_evaluator.html#a212ac4e6d0896e12fef2d5a0bd895e58', 1,
Go::IntCrvEvaluator::eval(double t, int n, std::vector<lt; std::vector<
lt; Point > > & amp;der)'], ['./class_go_1_1_lift_curve.html#
ae512f443f85b1704a433e11bb8782af5', 1, 'Go::LiftCurve::eval(double t) const ', ['./
class_go_1_1_lift_curve.html#a2cb2e79f40e4d61f22303b5bcf2aeba3', 1,
Go::LiftCurve::eval(double t, int n, Point der[]) const ', ['./
class_go_1_1_project_curve.html#af98b1ced4a661a5876e60db1d659f4fa', 1,
Go::ProjectCurve::eval(double t) const ', ['./
class_go_1_1_project_curve.html#a96186d4cf98720c6777bfd7686f27f49', 1,
Go::ProjectCurve::eval(double t, Go::Point seed) const ', ['./
class_go_1_1_project_curve.html#aa04bfd8be7761c82b715e59ab2537bc3', 1,
Go::ProjectCurve::eval(double t, int n, Go::Point der[]) const ', ['./
class_go_1_1_project_curve_and_cross_tan.html#
a2887b1adafc03d983eb8405ea079f828', 1, 'Go::ProjectCurveAndCrossTan::eval(double t)'], ['./
class_go_1_1_project_curve_and_cross_tan.html#
afe33ce9e2590e0bfe7808194acce9ebb', 1, 'Go::ProjectCurveAndCrossTan::eval(double t, int n,
std::vector<lt; std::vector<lt; Go::Point > > & amp;ders)'], ['./
class_go_1_1_project_intersection_curve.html#
a6c51a345c6bc7eef9ad51c6a6af1a6a7', 1, 'Go::ProjectIntersectionCurve::eval(double t) const ', ['./
class_go_1_1_project_intersection_curve.html#
```

```

ae83c55aba4c48f4f9e67ee10a469eebd',1,'Go::ProjectIntersectionCurve::eval(double t, int n,
Point der[]) const ', ['./class_go_1_1_smooth_transition.html#
a9dec775aeaa43960e2f31389536c020d',1,'Go::SmoothTransition::eval(double t)', ['./
class_go_1_1_smooth_transition.html#a3014fd12933c038db7eb267fbed90b50',1,'
Go::SmoothTransition::eval(double t, int n, std::vector<t; std::vector&
lt; Point > > & &der)', ['./class_go_1_1_space_int_crv.
html#ab9a922c9e907889610ba88f507340ed3',1,'Go::SpaceIntCrv::eval(double t) const ', ['./
class_go_1_1_space_int_crv.html#a52581fc9d01325d625eff7e90fd5ad23',1,'
Go::SpaceIntCrv::eval(double t, int n, Point der[]) const ', ['./
class_go_1_1_trim_curve.html#a383077ca958c414fc6c39ecde49df68f',1,'
Go::TrimCurve::eval(double t)', ['./class_go_1_1_trim_curve.html
#a94b8b25ba0369b8bf01a5e18caafa7da',1,'Go::TrimCurve::eval(double t, int n, std::vector&
lt; std::vector<t; Point > > & &der)', ['./
class_go_1_1_bezier_triangle.html#a04e7a7850a4292cd300aac604e17efec',1,'
Go::BezierTriangle::eval()', ['./
class_go_1_1_eval_functor_curve.html#adc7d79d2ad002096776641323d421fc0',1,'
Go::EvalFunctorCurve::eval(double t) const ', ['./
class_go_1_1_eval_functor_curve.html#ad765b1fdca8d47430a90b05d08358c42',1,'
Go::EvalFunctorCurve::eval(double t, int n, Point der[]) const ', ['./
class_go_1_1_eval_functor_surface.html#a4bba1fd389a8a21408c76279077d8c98'
,1,'Go::EvalFunctorSurface::eval(double u, double v) const ', ['./
class_go_1_1_eval_functor_surface.html#a52a9888c724819231d867b5f8ecc9edb'
,1,'Go::EvalFunctorSurface::eval(double u, double v, int n,
Point der[]) const ', ['./class_go_1_1_l_r_b_spline2_d.html#
af0a9b4ba330f30b6a7cceeaa8bd8e40b',1,'Go::LRBSpline2D::eval()', ['./
class_go_1_1_volume_parameter_curve.html#
a9fe5deedf6c920a026949df25a804eeb',1,'Go::VolumeParameterCurve::eval(double t) const ', ['./
class_go_1_1_volume_parameter_curve.html#
a7a1967e05c656a34b29c9e455c6985d3',1,'Go::VolumeParameterCurve::eval(double t, int n,
Point der[]) const ', ['./class_go_1_1_volume_space_curve.html#
a9c680f8db2892c4000e199d3591c4ec6',1,'Go::VolumeSpaceCurve::eval(double t) const ', ['./
class_go_1_1_volume_space_curve.html#adf40f504a4ffcd83d09f88dd023deb69',1,'
Go::VolumeSpaceCurve::eval(double t, int n, Point der[]) const ']]],
['eval_5f2_5fcrv', ['EVAL_2_CRV', ['./eval__2__crv_8c.html#
ae21399dc6a70f8168d0e3926598430cc',1,'EVAL_2_CRV():

```

Definition at line 26 of file all\_6.js.

#### 30.1393.1.4 Point&t

Definition at line 185 of file all\_6.js.

#### 30.1393.1.5 PrPathTriangleSeq h

Definition at line 26 of file all\_6.js.

#### 30.1393.1.6 pnt int std::vector& lt

Definition at line 185 of file all\_6.js.

#### 30.1393.1.7 pnt int nder

Definition at line 185 of file all\_6.js.

## 30.1393.1.8 var searchData

## Initial value:

```
=
[
 ['e', ['E', ['../namespace_go.html#a95d383314d9ef7d277424ee672473063a67661f5f4694b7e01c81e2e32d5841d0', 1, 'Go']],
 ['e1_5f', ['e1_', ['../struct_go_1_l_face_connectivity.html#a42876bc3d54e64697555761d3667382e', 1, 'Go::FaceConnectivity::e1_()'], ['../class_go_1_ltp_table_entry.html#ab2e9e30e315ec2fd05d5d37114296884', 1, 'Go::tpTableEntry::e1_()']],
 ['e2_5f', ['e2_', ['../struct_go_1_l_face_connectivity.html#a56d1d7a434f3fd07682e85488b369fb8', 1, 'Go::FaceConnectivity::e2_()'], ['../class_go_1_ltp_table_entry.html#a2e23e6a84effa6180933fe8e6fa6e61c', 1, 'Go::tpTableEntry::e2_()']],
 ['e2max', ['e2max', ['../struct_s_i_s_lbox.html#a669492a1c5428ae5d107e54a63052903', 1, 'SISLbox']],
 ['e2min', ['e2min', ['../struct_s_i_s_lbox.html#ae83586cc85d8c03f69f07976bdd48e20', 1, 'SISLbox']],
 ['each', ['each', ['../jquery_8js.html#a871ff39db627c54c710a3e9909b8234c', 1, 'jquery.js']],
 ['eatwhite', ['eatwhite', ['../namespace_go_1_l_utils.html#a43780996b9bc44a2d902858e953f8050', 1, 'Go::Utils']],
 ['ec', ['ec', ['../struct_s_i_s_l_point.html#aaab08c937d85029f5d86c469a21bf45d', 1, 'SISLPoint']],
 ['ecoeff', ['ecoeff', ['../struct_s_i_s_l_dir.html#a9cae019213555ec9a755671f7c12ba98', 1, 'SISLdir::ecoeff()'], ['../struct_s_i_s_l_curve.html#ad8783e20499cc2468b34e22d322354ad', 1, 'SISLCurve::ecoeff()'], ['../struct_s_i_s_l_surf.html#af123d50549798fca464196d4bc6321e', 1, 'SISLSurf::ecoeff()'], ['../struct_s_i_s_l_point.html#aa674e75fad6c907c12ade842728d6eb4', 1, 'SISLPoint::ecoeff()']],
 ['edg', ['edg', ['../struct_s_i_s_l_object.html#a3b5ff84d39e82a228481c7806a424ee7', 1, 'SISLObject']],
 ['edg_5fidx_5f1_5f', ['edg_idx_1', ['../struct_go_1_l_volume_adjacency_info.html#ab19327268c61359eb382df49d03f9095', 1, 'Go::VolumeAdjacencyInfo']],
 ['edg_5fidx_5f2_5f', ['edg_idx_2', ['../struct_go_1_l_volume_adjacency_info.html#a41ce7795661cc2ff8be7e16bcd260678', 1, 'Go::VolumeAdjacencyInfo']],
 ['edge', ['Edge', ['../classhetriang_1_l_edge.html', 1, 'hetriang']],
 ['edge', ['edge', ['../class_go_1_l_point_on_edge.html#aaaae9baee52e5453776627caa2fa9f37c', 1, 'Go::PointOnEdge::edge()'], ['../class_go_1_l_composite_box.html#a5f582420b4561b6bdb20f3c6d64d474e', 1, 'Go::CompositeBox::edge()'], ['../classhetriang_1_l_edge.html#ac66aa8601c31b0f664fb020e02d5e0bc', 1, 'hetriang::Edge::Edge()'], ['../classhed_1_l_edge.html#afcf345e2e15a83e18dbf14dd2eaeacd7', 1, 'hed::Edge::Edge()']],
 ['edge', ['Edge', ['../classhed_1_l_edge.html', 1, 'hed']],
 ['edge_5f', ['edge_', ['../struct_go_1_l_sample_point_data.html#abba15d6076f86102c17f7c4bb857ac13', 1, 'Go::SamplePointData']],
 ['edge_5f1', ['edge_1', ['../struct_s_i_s_l_intpt.html#a46dc1e8c213299854359e559636234b9', 1, 'SISLIntpt']],
 ['edge_5f2', ['edge_2', ['../struct_s_i_s_l_intpt.html#a6a017813f6d6c57e694fdd3332a8a233', 1, 'SISLIntpt']],
 ['edge_5facute_5fangle', ['EDGE_ACUTE_ANGLE', ['../namespace_go.html#a2353a3d581dde8e8be2ad76914f0f584ae81304fe056a752a83af4310b525b2e8', 1, 'Go']],
 ['edge_5fcurves_5fequal', ['edge_curves_equal', ['../namespace_go_1_l_coons_patch_volume_gen.html#af3266d664434db11a0a73a5ba524e652', 1, 'Go::CoonsPatchVolumeGen']],
 ['edge_5fpar_5f', ['edge_par_', ['../struct_go_1_l_sample_point_data.html#ae50653d346158f4c60b5d7d1a4438422', 1, 'Go::SamplePointData']],
 ['edge_5fposition_5fdiscont', ['EDGE_POSITION_DISCONT', ['../namespace_go.html#a2353a3d581dde8e8be2ad76914f0f584a8ae1a2b3ebb821b186678cda5a0459c6', 1, 'Go']],
 ['edge_5fscales_5f', ['edge_scales_', ['../class_go_1_lft_chart_surface.html#a9212705a80c34a260750dc8c9126a15e', 1, 'Go::ftChartSurface']],
 ['edge_5fsequence', ['edge_sequence', ['../_pr_path_triangle_seq_8h.html#ab5a2fa93378e83175ca952ed72782fa5', 1, 'edge_sequence(const std::vector<int > &tr_seq, const PrTriangulation_OP &t):#160

```

Definition at line 1 of file all\_6.js.

## 30.1394 doc/html/search/all\_7.js File Reference

## Variables

- var [searchData](#)
- & [gt](#)

## 30.1394.1 Variable Documentation

## 30.1394.1.1 &amp; gt

Definition at line 47 of file all\_7.js.

30.1394.1.2 `var searchData`

Definition at line 1 of file `all_7.js`.

30.1395 `doc/html/search/all_8.js` File Reference

## Variables

- `var searchData`
- `generic_graph_algorithms_implementation` `h`
- `generic_graph_algorithms_implementation` `const int xs`
- `generic_graph_algorithms_implementation` `const int const int ys`
- `generic_graph_algorithms_implementation` `const int const int const int x0`
- `generic_graph_algorithms_implementation` `const int const int const int const int y0`
- `generic_graph_algorithms_implementation` `const int const int const int const int const int doubleBuffer`
- `generic_graph_algorithms_implementation` `const int const int const int const int const int const int two_sided`
- `generic_graph_algorithms_implementation` `const int const int const int const int const int const int const int lighting`
- `generic_graph_algorithms_implementation` `const int const int const int const int const int const int const int const int normalize`
- `generic_graph_algorithms_implementation` `const int const int const int const int const int const int const int const int const int smooth`
- `generic_graph_algorithms_implementation` `const int const int const int const int const int const int const int const int const int const double xtrans`
- `generic_graph_algorithms_implementation` `const int const int const int const int const int const int const int const int const int const double const double ytrans`
- `generic_graph_algorithms_implementation` `const int const int const int const int const int const int const int const int const int const double const double const double ztrans`
- `generic_graph_algorithms_implementation` `const int const int const int const int const int const int const int const int const int const double const double const double const double xscale`
- `generic_graph_algorithms_implementation` `const int const int const int const int const int const int const int const int const int const double const double const double const double const double yscale`
- `generic_graph_algorithms_implementation` `const int const int const int const int const int const int const int const int const int const double const double const double const double const double const double zscale`
- `generic_graph_algorithms_implementation` `const int const int const int const int const int const int const int const int const int const double const double const double const double const double const double const double int & amp`
- `tx`
- `ty`
- `const int texture_mode`
- `const int const char *const texfile =NULL):&#160`
- `gl_aux` `cpp`

## 30.1395.1 Variable Documentation

30.1395.1.1 `int& amp` `[property]`, `[bound]`, `[constrained]`

Definition at line 92 of file `all_8.js`.

30.1395.1.2 `gl_aux` `cpp`

Definition at line 620 of file `all_8.js`.

30.1395.1.3 `gl_aux const int const int const int const int const int doubleBuffer [property], [bound], [constrained]`

Definition at line 92 of file all\_8.js.

30.1395.1.4 `generic_graph_algorithms_implementation h`

Definition at line 92 of file all\_8.js.

30.1395.1.5 `gl_aux const int const int const int const int const int const int const int lighting [property], [bound], [constrained]`

Definition at line 92 of file all\_8.js.

30.1395.1.6 `gl_aux const int const int const int const int const int const int const int const int const int normalize [property], [bound], [constrained]`

Definition at line 92 of file all\_8.js.

30.1395.1.7 `var searchData`

Definition at line 1 of file all\_8.js.

30.1395.1.8 `gl_aux const int const int const int const int const int const int const int const int const int const int smooth [property], [bound], [constrained]`

Definition at line 92 of file all\_8.js.

30.1395.1.9 `const int const char *const texfile =NULL):&#160`

Definition at line 620 of file all\_8.js.

30.1395.1.10 `const int texture_mode`

Definition at line 620 of file all\_8.js.

30.1395.1.11 `gl_aux const int const int const int const int const int const int const int two_sided [property], [bound], [constrained]`

Definition at line 92 of file all\_8.js.

30.1395.1.12 tx

Definition at line 620 of file all\_8.js.

30.1395.1.13 ty

Definition at line 620 of file all\_8.js.

30.1395.1.14 **gl\_aux const int const int const int x0** [property], [bound], [constrained]

Definition at line 92 of file all\_8.js.

30.1395.1.15 **gl\_aux const int xs** [property], [bound], [constrained]

Definition at line 92 of file all\_8.js.

30.1395.1.16 **gl\_aux const int const int const int const int const int const int const int const int const int const double const double const double const double xscale** [property], [bound], [constrained]

Definition at line 92 of file all\_8.js.

30.1395.1.17 **gl\_aux const int const int const int const int const int const int const int const int const int const int const double xtrans** [property], [bound], [constrained]

Definition at line 92 of file all\_8.js.

30.1395.1.18 **gl\_aux const int const int const int const int y0** [property], [bound], [constrained]

Definition at line 92 of file all\_8.js.

30.1395.1.19 **gl\_aux const int const int ys** [property], [bound], [constrained]

Definition at line 92 of file all\_8.js.

30.1395.1.20 **gl\_aux const int const int const int const int const int const int const int const int const int const int const double const double const double const double const double const double yscale** [property], [bound], [constrained]

Definition at line 92 of file all\_8.js.



30.1395.1.21 **gl\_aux const int const int const int const int const int const int const int const int const int const int const double const double ytrans** [property], [bound], [constrained]

Definition at line 92 of file all\_8.js.

30.1395.1.22 **gl\_aux const int const int const int const int const int const int const int const int const int const int const int const double const double const double const double const double const double const double zscale** [property], [bound], [constrained]

Definition at line 92 of file all\_8.js.

30.1395.1.23 **gl\_aux const int const int const int const int const int const int const int const int const int const int const int const double const double const double ztrans** [property], [bound], [constrained]

Definition at line 92 of file all\_8.js.

## 30.1396 doc/html/search/all\_9.js File Reference

### Variables

- var [searchData](#)
- T & [gt](#)
- int [n](#)
- int [double tolerance1](#)
- int [double double tolerance2](#)
- int [double double int mm](#)
- int [double double int double initpars\\_v \[\]](#)
- int [double double int double int nn](#)
- int [double double int double int double double int double double std::vector & lt](#)
- [dims](#)
- [double param\\_v \[\]](#)

### 30.1396.1 Variable Documentation

#### 30.1396.1.1 [dims](#)

Definition at line 68 of file all\_9.js.

#### 30.1396.1.2 [int& gt](#)

Definition at line 5 of file all\_9.js.

#### 30.1396.1.3 [int double double int double initpars\\_v\[\]](#) [bound]

Definition at line 5 of file all\_9.js.

**30.1396.1.4** `int int std::vector& lt` [bound]

Definition at line 5 of file all\_9.js.

**30.1396.1.5** `double int mm` [bound]

Definition at line 5 of file all\_9.js.

**30.1396.1.6** `int int n` [bound]

Definition at line 5 of file all\_9.js.

**30.1396.1.7** `double int int nn` [bound]

Definition at line 5 of file all\_9.js.

**30.1396.1.8** `double param_v[]`

Definition at line 75 of file all\_9.js.

**30.1396.1.9** `var searchData`

**Initial value:**

```
=
[
 ['hahnssurfacegen_2eh', ['HahnsSurfaceGen.h', ['../_hahns_surface_gen_8h.html', 1, '']],
 ['halfedgecollapse', ['halfEdgeCollapse', ['../class_pr_thin.html#a8562782f8ebd8645b6ef5dee869f913e', 1, 'PrThin']],
 ['handle', ['Handle', ['../class_handle.html', 1, 'Handle<
```

Definition at line 1 of file all\_9.js.

**30.1396.1.10** `int double double int double int double double int double tolerance1` [bound]

Definition at line 5 of file all\_9.js.

**30.1396.1.11** `int double double int double int double double int double double tolerance2` [bound]

Definition at line 5 of file all\_9.js.

## 30.1397 doc/html/search/all\_a.js File Reference

### Variables

- var [searchData](#)

### 30.1397.1 Variable Documentation

#### 30.1397.1.1 var searchData

##### Initial value:

```
=
[
 ['i', ['i', ['../class_n_e_w_m_a_t_l_l_matrix_type.html#acb7b50521ae4e68a7cf332d180a19bc8', 1, '
 NEWMAT::MatrixType::i()'], ['../class_n_e_w_m_a_t_l_l_base_matrix.html#a4227de5c79ebbe72cb883dc44a4adc1f', 1, '
 NEWMAT::BaseMatrix::i()']],
 ['icopy', ['icopy', ['../struct_s_i_s_l_curve.html#a20f717dfda7626d2381b6d852aad3b0', 1, 'SISLCurve::icopy()
 '], ['../struct_s_i_s_l_surf.html#aa626e2db7658d0247f79c931ac118e56', 1, 'SISLSurf::icopy()'], ['
 ../struct_s_i_s_l_point.html#abb277b543c1cfce82467725ead2544c2', 1, 'SISLPoint::icopy()']],
 ['id'
```

Definition at line 1 of file all\_a.js.

## 30.1398 doc/html/search/all\_b.js File Reference

### Variables

- var [searchData](#)
- [jacobi.cpp](#)

### 30.1398.1 Variable Documentation

#### 30.1398.1.1 jacobi.cpp

Definition at line 3 of file all\_b.js.

#### 30.1398.1.2 var searchData

##### Initial value:

```
=
[
 ['jacobi', ['Jacobi', ['../namespace_n_e_w_m_a_t.html#a2836cab2994eacfc0e7a3b5197d0e9a', 1, '
 NEWMAT::Jacobi(const SymmetricMatrix & , DiagonalMatrix &)'], ['
 ../namespace_n_e_w_m_a_t.html#a556f131206374d64bb896c8f815e8fd6', 1, 'NEWMAT::Jacobi(const SymmetricMatrix & , Diagona
 & , DiagonalMatrix & , Matrix &)'], ['../namespace_n_e_w_m_a_t.html#ab7f1b7f360924503ff4e0a681602cbc0'
 , 1, 'NEWMAT::Jacobi(const SymmetricMatrix & , DiagonalMatrix & , SymmetricMatrix & , Matrix & ,
 bool=true)'], ['../jacobi_8cpp.html#ad370e3456b419cfe2eae25f2300fd5b1', 1, 'Jacobi(const SymmetricMatrix & X,
 DiagonalMatrix & D, SymmetricMatrix & A, Matrix & V, bool eivec):
```

Definition at line 1 of file all\_b.js.

## 30.1399 doc/html/search/all\_c.js File Reference

### Variables

- var [searchData](#)
- [newmat6](#) `cpp`

### 30.1399.1 Variable Documentation

#### 30.1399.1.1 [newmat6](#) `cpp`

Definition at line 26 of file `all_c.js`.

#### 30.1399.1.2 [var searchData](#)

#### Initial value:

```
=
[
 ['k', ['k', ['../jquery_8js.html#ab26645c014aa005ecedef329ecf58c99', 1, 'jquery.js']],
 ['kdim_5f', ['kdim_', ['../class_go_1_1_smooth_surf.html#aecbfa2bdb4142f2207c88249237cf358', 1,
 Go::SmoothSurf::kdim_()'], ['../class_go_1_1_smooth_surf_set.html#a16650b6975a4aaf03bf49ccd70a973e1', 1,
 Go::SmoothSurfSet::kdim_()']],
 ['keep_5fsing', ['KEEP_SING', ['
 ../namespace_go.html#a2d40b57e4fc6eed03a9fbeebe181e7c5a5950a0471c4dedf23ebab3d521693e9e', 1, 'Go']]],
 ['key', ['Key', ['../classmaterial__appearance.html#a74882356eb34bffa5a8683cccbee97bc', 1,
 material_appearance']]],
 ['key_5f', ['key_', ['../class_heap_node.html#ae07076e227f6a24b9a70d2dff7031110', 1, 'HeapNode::key_()'], ['
 ../class_heap_node2.html#a5834899269c09b811d1f2cf807e41e22', 1, 'HeapNode2::key_()']],
 ['kink', ['kink', ['../struct_go_1_ltp_tolerances.html#a587c6924aaf46d129e70defebf9cdf73', 1,
 Go::tpTolerances']]],
 ['kkl_5f', ['kkl_', ['../class_go_1_1_smooth_surf.html#ad616047930638a6351bdd0b2e82dc9d1', 1, 'Go::SmoothSurf
 ']]],
 ['kk2_5f', ['kk2_', ['../class_go_1_1_smooth_surf.html#a1e00323a454408b86c724dd46fbc1b9c', 1, 'Go::SmoothSurf
 ']]],
 ['knl_5f', ['knl_', ['../class_go_1_1_smooth_surf.html#a548d3b669c60ffdf971feac58a2053', 1, 'Go::SmoothSurf
 ']]],
 ['kn2_5f', ['kn2_', ['../class_go_1_1_smooth_surf.html#a1ab34f0fee55320b22914774b1e943c6', 1, 'Go::SmoothSurf
 ']]],
 ['kncond_5f', ['kncond_', ['../class_go_1_1_smooth_surf.html#a63899626d5ffddfc45f9d6aa22f44dd4', 1,
 Go::SmoothSurf::kncond_()'], ['../class_go_1_1_smooth_surf_set.html#aa530cc21c18294d89efb3e986b5f1ad3', 1,
 Go::SmoothSurfSet::kncond_()']],
 ['knconstraint_5f', ['knconstraint_', ['../class_go_1_1_smooth_surf.html#a9a80424597864d0adf36ef300ca2a41a'
 ../class_go_1_1_smooth_surf_set.html#ab4b15fa0722035377b4651a00d147e5f', 1, 'Go::SmoothSurfSet::knconstraint_()']],
 ['knotepsilon', ['knotEpsilon', ['../class_go_1_1_go_tools.html#af44ef84796819b793f2b196a7b296b9c', 1,
 Go::GoTools']]],
 ['knotinterval', ['knotInterval', ['../class_go_1_1_bspline_basis.html#a305a7e9c5ded401fbb4de72c0e9d2693', 1,
 'Go::BsplineBasis']]],
 ['knotintervalfuzzy', ['knotIntervalFuzzy', ['
 ../class_go_1_1_bspline_basis.html#a0ae41ca8567fa19eaccdce8b228f1580', 1, 'Go::BsplineBasis::knotIntervalFuzzy()'], ['
 ../class_go_1_1_param0_function_int.html#adf18f61ba3429e8337f680bc6ed9eb4a', 1, 'Go::Param0FunctionInt::knotIntervalFuzzy
 ../class_go_1_1_param1_function_int.html#a4cfe2132fa258b820edaf776f5383396', 1, 'Go::Param1FunctionInt::knotIntervalFuzzy
 ../class_go_1_1_param2_function_int.html#a751cf94d3a521e66fde43b34d920662a', 1, 'Go::Param2FunctionInt::knotIntervalFuzzy
 ../class_go_1_1_param_curve_int.html#ab7260dc6baf8fa87b13c773360cced24', 1,
 Go::ParamCurveInt::knotIntervalFuzzy()'], ['../class_go_1_1_param_surface_int.html#af16dd54625073ebccc4cac921c9d6ac2', 1,
 Go::ParamSurfaceInt::knotIntervalFuzzy()'], ['../class_go_1_1_spline1_function_int.html#ae54cf5b82d26a4171300d8fbb3de2d
 Go::Spline1FunctionInt::knotIntervalFuzzy()'], ['
 ../class_go_1_1_spline2_function_int.html#a7229f63cd797e93f4aed7b4f3f8650ba', 1, 'Go::Spline2FunctionInt::knotIntervalFuz
 ../class_go_1_1_spline_curve_int.html#aaccc36c235ff72d3989818aa3d99b024', 1, 'Go::SplineCurveInt::knotIntervalFuzzy()'], [
 ../class_go_1_1_spline_surface_int.html#ab48552a70547a940f282bf4f9eec5adf', 1, 'Go::SplineSurfaceInt::knotIntervalFuzzy()
 ../class_go_1_1_mesh2_d.html#a7038b8a5c1eal06fac4c1b5b0bbddc7e', 1, 'Go::Mesh2D::knotIntervalFuzzy()']],
 ['knotmultiplicities', ['knotMultiplicities', ['
 ../class_go_1_1_bspline_basis.html#a91f3e774d6f812df432c2f38fcf44f8e', 1, 'Go::BsplineBasis']]],
 ['knotmultiplicity', ['knotMultiplicity', ['
 ../class_go_1_1_bspline_basis.html#aca0cff53a80af7a8cfd0cc64acea7c95', 1, 'Go::BsplineBasis']]],
 ['knots', ['knots', ['../class_go_1_1_block_solution.html#af0bdaab56a91706e89bcfafe49ad2ae8f', 1,
 Go::BlockSolution::knots()'], ['../class_go_1_1_isogeometric_block.html#ad515166c60c7e702dc9f27d48f7122d3', 1,
 Go::IsogeometricBlock::knots()'], ['../class_go_1_1_sf_solution.html#a74a9a0c6cc3dc7e3ddfcab6ac8031cdc', 1,
 Go::SfSolution::knots()'], ['../class_go_1_1_vol_solution.html#a9d6a56296f7300e21f6f371dcc5d08a0', 1,

```

```

 Go::VolSolution::knots()]]],
['knots_5fto_5finsert', ['knots_to_insert', [
 ../namespace_go_1_1_l_r_spline_utils.html#a4f98f5746feb419caf049335e2bleda7', 1, 'Go::LRSplineUtils']]],
['knotsbegin', ['knotsBegin', ['../class_go_1_1_spline_curve.html#a9617b6d88db3ed4e11ed498a40d89581', 1, '
Go::SplineCurve::knotsBegin()'], ['../class_go_1_1_spline_curve.html#ab7cfb2dc285d0378062d55d18c0acac2', 1, '
Go::SplineCurve::knotsBegin() const '], ['../class_go_1_1_mesh2_d.html#a752b5850413c524460efed453321a688', 1, '
Go::Mesh2D::knotsBegin()']]],
['knotsend', ['knotsEnd', ['../class_go_1_1_spline_curve.html#acf20e5283b690c8f22aae9fb147fc210', 1, '
Go::SplineCurve::knotsEnd()'], ['../class_go_1_1_spline_curve.html#a9fe54c39a94b2c2575f6d0511f721145', 1, '
Go::SplineCurve::knotsEnd() const '], ['../class_go_1_1_mesh2_d.html#aicedb03145e626d0bd834ca339f091db', 1, '
Go::Mesh2D::knotsEnd()']]],
['knotspan', ['knotSpan', ['../class_go_1_1_spline_surface.html#a4ae188d833ba7e018a60361975ca1754', 1, '
Go::SplineSurface::knotSpan()'], ['../class_go_1_1_spline_volume.html#aaffe28b54eaalc97dele2a3e2229b608f', 1, '
Go::SplineVolume::knotSpan()']]],
['knotssimple', ['knotsSimple', ['../class_go_1_1_bspline_basis.html#aa2e75ab7996534ff328f96d5b2cfadde', 1, '
Go::BsplineBasis']]],
['kp', ['KP', ['../class_n_e_w_m_a_t_1_1_matrix_type.html#a6a5c67f2185e7691d2d8e85e7f209130', 1, '
NEWMAT::MatrixType::KP()'], ['../class_n_e_w_m_a_t_1_1_k_p_matrix.html#a7b84f8e43bac4608bf225867306c9450', 1, '
NEWMAT::KPMatrix::KP()'], ['../class_matrix_row_col.html#a8a788cf7179884e2100b77f845cea7dd', 1, 'MatrixRowCol::KP()'], [
../namespace_n_e_w_m_a_t.html#adac429a80804603f506371d42304b686', 1, 'NEWMAT::KP()'], [
../newmat_6_cpp.html#ae3166f1201bb79865ac1945a9ebc64d6', 1, 'KP():

```

Definition at line 1 of file all\_c.js.

## 30.1400 doc/html/search/all\_d.js File Reference

### Variables

- var [searchData](#)
- mouse [cpp](#)
- mouse [double epsge](#)
- mouse [double bool init\\_mba =false](#)
- mouse [double bool double mba\\_level =0.0](#)
- mouse [double bool double bool closest\\_dist =true](#)
- mouse [double bool double bool bool repar](#)

### 30.1400.1 Variable Documentation

30.1400.1.1 mouse [double bool double bool closest\\_dist =true](#) [bound]

Definition at line 184 of file all\_d.js.

30.1400.1.2 mouse [cpp](#)

Definition at line 20 of file all\_d.js.

30.1400.1.3 mouse [double epsge](#) [bound]

Definition at line 20 of file all\_d.js.

30.1400.1.4 mouse [double bool init\\_mba =false](#) [bound]

Definition at line 184 of file all\_d.js.

## 30.1400.1.5 mouse double bool double mba\_level =0.0 [bound]

Definition at line 184 of file all\_d.js.

## 30.1400.1.6 mouse double bool double bool bool repar [bound]

Initial value:

```
=false)']]],
 ['lrsurfapprox', ['LRSurfApprox', ['../class_go_1_1_l_r_surf_approx.html', 1, 'Go']],
 ['lrsurfapprox_2eh', ['LRSurfApprox.h', ['../_l_r_surf_approx_8h.html', 1, ''']],
 ['lrsurfsmoothls', ['LRSurfSmoothLS', ['../class_go_1_1_l_r_surf_smooth_ls.html#a8acba4937645e7d7c619ce67ae71c5b8', 1, 'Go::LRSurfSmoothLS::LRSurfSmoothLS(shared_ptr<LRSplineSurface > surf, std::vector<int > &coef_known)', ['../class_go_1_1_l_r_surf_smooth_ls.html#ad3d1d6e980d0818742352b8719702ac8', 1, 'Go::LRSurfSmoothLS::LRSurfSmoothLS()']]],
 ['lrsurfsmoothls', ['LRSurfSmoothLS', ['../class_go_1_1_l_r_surf_smooth_ls.html', 1, 'Go']],
 ['lrsurfsmoothls_2eh', ['LRSurfSmoothLS.h', ['../_l_r_surf_smooth_ls_8h.html', 1, ''']],
 ['lrsurfstitch_2eh', ['LRSurfStitch.h', ['../_l_r_surf_stitch_8h.html', 1, ''']],
 ['lsf', ['LSF', ['../newmatrc_8h.html#ae75c691eb2643cc7fe952b12f074040c', 1, 'newmatrc.h']],
 ['lsmat_5f', ['LSmat_', ['../struct_go_1_1_l_s_smooth_data.html#ab548ff8514223b089b87c15a6c83199e', 1, 'Go::LSSmoothData']],
 ['lsright_5f', ['LSright_', ['../struct_go_1_1_l_s_smooth_data.html#a58d1fcd7509e9686f9c8c7a7932c1646', 1, 'Go::LSSmoothData']],
 ['lssmoothdata', ['LSSmoothData', ['../struct_go_1_1_l_s_smooth_data.html#adbe08d7ac0e51e41aad91021aaef8014', 1, 'Go::LSSmoothData']],
 ['lssmoothdata', ['LSSmoothData', ['../struct_go_1_1_l_s_smooth_data.html', 1, 'Go']],
 ['lt', ['LT', ['../class_n_e_w_m_a_t_l_1_matrix_type.html#a6a88aac41b54a529b61a9c9433f4b583a0edcd4d5f3d58272ad0e0a36f5da704a', 1, 'NEWMAT::MatrixType']],
 ['ltlexpoint', ['ltLexPoint', ['../main_8cpp.html#a3ad6217edc1c1d915556be09a7a6e512', 1, 'main.cpp']],
 ['lubksb', ['lubksb', ['../class_n_e_w_m_a_t_l_1_crout_matrix.html#afcfb1bcd9b105d4be4f13b9dc85a0f98', 1, 'NEWMAT::CroutMatrix::lubksb()'], ['../class_n_e_w_m_a_t_l_1_band_l_u_matrix.html#aa6b6344001665e9241bd7a3e6564c7da', 1, 'NEWMAT::BandLUMatrix::lubksb()']],
 ['ludeco', ['LUDeco', ['../class_n_e_w_m_a_t_l_1_matrix_type.html#afladc0086f5d4c3328ad4a698fa57c90ae624050aed576ddc4119f40c426cc61b', 1, 'NEWMAT::MatrixType']],
 ['ludecomp', ['LUDecomp', ['../namespace_go.html#a77c8de2bdc12a47b62ef5c306ac0e46d', 1, 'Go']],
 ['ludecomp_2eh', ['LUDecomp.h', ['../_l_u_decomp_8h.html', 1, ''']],
 ['ludecomp_5fimplementation_2eh', ['LUDecomp_implementation.h', ['../_l_u_decomp_implementation_8h.html', 1, ''']],
 ['lusolvesystem', ['LUSolveSystem', ['../namespace_go.html#afd9ea9a56159c9a5841819ebf13280b8', 1, 'Go']],
 ['lw_5fset_5f', ['lw_set_', ['../class_go_1_1_param_curve_int.html#af38741b5ba619e02139aa5fa2b17d302', 1, 'Go::ParamCurveInt::lw_set()'], ['../class_go_1_1_param_surface_int.html#a13bd0a26899cdabc17854eb1b4c230aa', 1, 'Go::ParamSurfaceInt::lw_set()']]
]
```

Definition at line 184 of file all\_d.js.

## 30.1400.1.7 var searchData

Initial value:

```
=
[
 ['l', ['L', ['../jquery_8js.html#a38ee4c0b5f4fe2a18d0c783af540d253', 1, 'jquery.js']],
 ['l_5f', ['l_', ['../struct_go_1_1_param_surface_1_1_degenerate__info.html#a9e9e4dd3a9bad4fc495e1ca428d1c4e9', 1, 'Go::ParamSurface::degenerate_info']],
 ['l_5fpath_5f', ['l_path_', ['../class_path_type.html#a3849d634c4440974a3e8664f0adf2c88', 1, 'PathType']],
 ['label_5f', ['label_', ['../class_multi_dijkstra.html#a7e2736e3c6d24b2a5b451c5d6f4b02ff', 1, 'MultiDijkstra']],
 ['label_5fdisplay', ['label_display', ['../struct_go_1_1_i_g_e_sdirentary.html#a2b8d7c0041720a2cc10d1eab59d69171', 1, 'Go::IGESdirentary']],
 ['labelbdynode', ['labelBdyNode', ['../class_pr_organized_points.html#acab5b170fefc327f7db0aac6dac3347b', 1,
```

```

 'PrOrganizedPoints']]],
['labelNode', ['labelNode', ['../class_pr_organized_points.html#a02406e295c174e57548497a85e987988', 1, '
PrOrganizedPoints::labelNode(int i, int ic, std::vector<&int > &component) const '], ['
../class_pr_organized_points.html#ae7d58624e3dabfac3af4a976ecd46ca2', 1, 'PrOrganizedPoints::labelNode(int i, int ic, int
&index, std::vector<&int > &component, std::vector<&int > &newIndex) const ']]],
['lackingparameter', ['lackingParameter', ['
../class_go_1_l_intersection_pool.html#a204c700b87c2881793dc64850c98ca03', 1, 'Go::IntersectionPool']]],
['lackingparametervalue', ['lackingParameterValue', ['
../class_go_1_l_intersection_pool.html#afd672f1d2375a2c42b178c9aa139363a', 1, 'Go::IntersectionPool']]],
['large_5fdistance_5f', ['large_distance_', ['../class_dijkstra.html#af7d4f8f644cdf1330bca5ce819fa58a2', 1, '
Dijkstra::large_distance_()'], ['../class_multi_dijkstra.html#a4cc650703fd03dae2d075afb06563c2f', 1, '
MultiDijkstra::large_distance_()']]]],
['largestmultinline', ['largestMultInLine', ['
../class_go_1_l_mesh2_d.html#a6f5984896001959aacf751265b29208c', 1, 'Go::Mesh2D']]],
['last', ['last', ['../class_r_b_d_c_o_m_m_o_n_l_l_tracer.html#a93a99ea96d8fc44d27b51629c2071cd1', 1, '
RBD_COMMON::Tracer']]],
['last_5fchange_5f', ['last_change_', ['
../struct_go_1_l_registration_result.html#a7df6e84c504a75d83f47aa2de92d747d', 1, 'Go::RegistrationResult']]],
['last_5ffile_5fname_5f', ['last_file_name_', ['
../classgv_application.html#a69d06231d8c542521ae2e638c7033a23', 1, 'gvApplication']]],
['last_5fmouse_5fpos_5f', ['last_mouse_pos_', ['../classgv_view.html#a08505548abel1069e5600ae0b17271a7', 1, '
gvView']]],
['last_5fnewton_5fiteration_5f', ['last_newton_iteration_', ['
../struct_go_1_l_registration_result.html#a0955123270a8c94b76919e794f6e8e67', 1, 'Go::RegistrationResult']]],
['last_5fnonlarger_5fknotvalue_5fix', ['last_nonlarger_knotvalue_ix', ['
../namespace_go_1_l_mesh2_d_utils.html#a8a567523d54d76c3a532ce9a9749f58e', 1, 'Go::Mesh2DUtils']]],
['last_5fx', ['last_x', ['../mouse_8h.html#abbf254766520cad4d7fe189eb184194c', 1, 'last_x():

```

Definition at line 1 of file all\_d.js.

## 30.1401 doc/html/search/all\_e.js File Reference

### Variables

- var [searchData](#)
- [glutils.cpp](#)

### 30.1401.1 Variable Documentation

#### 30.1401.1.1 glutils sisl\_view\_demo.cpp

Definition at line 10 of file all\_e.js.

#### 30.1401.1.2 var searchData

#### Initial value:

```

=
[
['m3_5fdet', ['m3_det', ['../gv_utilities_8h.html#a15f8dd61e74e6e8637e610a774df9607', 1, 'gvUtilities.h']]],
['m3_5fidentity', ['m3_identity', ['../gv_utilities_8h.html#a6091a5b9ab587c9a4b692565e6e1d6c6', 1, '
gvUtilities.h']]],
['m3_5finverse', ['m3_inverse', ['../gv_utilities_8h.html#a3347b853ddaa80d8f92360e764b433e8', 1, '
gvUtilities.h']]],
['m4_5fdet', ['m4_det', ['../gv_utilities_8h.html#acdaa244d22eb4f9c9c0bda4b5d94e5b3', 1, 'gvUtilities.h']]],
['m4_5finverse', ['m4_inverse', ['../gv_utilities_8h.html#a708f7916c5b2c579c1096435f8f481f4', 1, '
gvUtilities.h']]],
['m4_5fsubmat', ['m4_submat', ['../gv_utilities_8h.html#a94b4064202801e77aa5bd9b975b37d0c', 1, 'gvUtilities.h
']]],
['m_5f', ['m_', ['../class_go_1_lft_chart_surface.html#a5a87666799aa8b8369f97049e534b943', 1, '
Go::ftChartSurface::m_()'], ['../class_pr_mat.html#a4808d661baa0b26acc1c87a5647e18ab', 1, 'PrMat::m_()'], ['
../class_go_1_l_solve_c_g.html#ac8ef49ab38d13d40f19874a80534d645', 1, 'Go::SolveCG::M_()']]]],
['m_5fpi', ['M_PI', ['../glutils_8cpp.html#ae71449b1cc6e6250b91f539153a7a0d3', 1, 'M_PI():

```

Definition at line 1 of file all\_e.js.

## 30.1402 doc/html/search/all\_f.js File Reference

### Variables

- var [searchData](#)
- [sisl](#) h

### 30.1402.1 Variable Documentation

#### 30.1402.1.1 [sisl](#) h

Definition at line 38 of file all\_f.js.

#### 30.1402.1.2 var searchData

Definition at line 1 of file all\_f.js.

## 30.1403 doc/html/search/classes\_0.js File Reference

### Variables

- var [searchData](#)
- [double](#)
- [&gt;](#)

### 30.1403.1 Variable Documentation

#### 30.1403.1.1 [double](#)

Definition at line 15 of file classes\_0.js.

#### 30.1403.1.2 [&gt;](#)

Definition at line 15 of file classes\_0.js.

#### 30.1403.1.3 var searchData

### Initial value:

```
=
[
 ['adapctcurve', ['AdaptCurve', ['../class_go_1_1_adapt_curve.html', 1, 'Go']],
 ['addedmatrix', ['AddedMatrix', ['../class_n_e_w_m_a_t_1_1_added_matrix.html', 1, 'NEWMAT']],
 ['adjacencyinfo', ['AdjacencyInfo', ['../struct_go_1_1_adjacency_info.html', 1, 'Go']],
 ['alg2delem', ['Alg2DElem', ['../struct_go_1_1_alg2_d_elem.html', 1, 'Go']],
 ['alg3delem', ['Alg3DElem', ['../struct_go_1_1_alg3_d_elem.html', 1, 'Go']],
 ['algobj2dint', ['AlgObj2DInt', ['../class_go_1_1_alg_obj2_d_int.html', 1, 'Go']],
 ['algobj3dint', ['AlgObj3DInt', ['../class_go_1_1_alg_obj3_d_int.html', 1, 'Go']],
 ['algobjectint', ['AlgObjectInt', ['../class_go_1_1_alg_object_int.html', 1, 'Go']],
 ['approxcrvtoseqs', ['ApproxCrvToSeqs', ['../class_go_1_1_approx_crv_to_seqs.html', 1, 'Go']],
 ['approxcurve', ['ApproxCurve', ['../class_go_1_1_approx_curve.html', 1, 'Go']],
 ['approxsurf', ['ApproxSurf', ['../class_go_1_1_approx_surf.html', 1, 'Go']],
 ['array', ['Array', ['../class_go_1_1_array.html', 1, 'Go']],
 ['array_3c_20double_2c_202_20_3e', ['Array<double>']]]]
```

Definition at line 1 of file classes\_0.js.



## 30.1404 doc/html/search/classes\_1.js File Reference

### Variables

- var [searchData](#)
- & gt

#### 30.1404.1 Variable Documentation

##### 30.1404.1.1 & gt

Definition at line 7 of file classes\_1.js.

##### 30.1404.1.2 var searchData

#### Initial value:

```
=
[
 ['bad_5falloc', ['Bad_alloc', ['../class_r_b_d___c_o_m_m_o_n_1_1_bad_alloc.html', 1, 'RBD_COMMON']],
 ['bandlumatrix', ['BandLUMatrix', ['../class_n_e_w_m_a_t_1_1_band_l_u_matrix.html', 1, 'NEWMAT']],
 ['bandmatrix', ['BandMatrix', ['../class_n_e_w_m_a_t_1_1_band_matrix.html', 1, 'NEWMAT']],
 ['barycoordsystem', ['BaryCoordSystem', ['../class_go_1_1_bary_coord_system.html', 1, 'Go']],
 ['barycoordsystem_3c_202_20_3e', ['BaryCoordSystem<
```

Definition at line 1 of file classes\_1.js.

## 30.1405 doc/html/search/classes\_10.js File Reference

### Variables

- var [searchData](#)

#### 30.1405.1 Variable Documentation

##### 30.1405.1.1 var searchData

#### Initial value:

```
=
[
 ['quadmesh', ['QuadMesh', ['../class_go_1_1_quad_mesh.html', 1, 'Go']],
 ['qualityresults', ['QualityResults', ['../class_go_1_1_quality_results.html', 1, 'Go']]
]
```

Definition at line 1 of file classes\_10.js.

## 30.1406 doc/html/search/classes\_11.js File Reference

### Variables

- var [searchData](#)

### 30.1406.1 Variable Documentation

#### 30.1406.1.1 var searchData

Definition at line 1 of file classes\_11.js.

## 30.1407 doc/html/search/classes\_12.js File Reference

### Variables

- var [searchData](#)

### 30.1407.1 Variable Documentation

#### 30.1407.1.1 var searchData

Definition at line 1 of file classes\_12.js.

## 30.1408 doc/html/search/classes\_13.js File Reference

### Variables

- var [searchData](#)

### 30.1408.1 Variable Documentation

#### 30.1408.1.1 var searchData

#### Initial value:

```
=
[
 ['tessellator', ['Tessellator', ['./class_go_1_1_tessellator.html', 1, 'Go']],
 ['testclass', ['TestClass', ['./class_test_class.html', 1, '']]],
 ['testindomain', ['TestInDomain', ['./class_go_1_1_test_in_domain.html', 1, 'Go']],
 ['time_5flapse', ['time_lapse', ['./classtime__lapse.html', 1, '']]],
 ['torus', ['Torus', ['./class_go_1_1_torus.html', 1, 'Go']],
 ['torusvolume', ['TorusVolume', ['./class_go_1_1_torus_volume.html', 1, 'Go']],
 ['tpedge', ['tpEdge', ['./class_go_1_1tp_edge.html', 1, 'Go']],
 ['tpface', ['tpFace', ['./class_go_1_1tp_face.html', 1, 'Go']],
 ['tpmarchpoint', ['tpMarchPoint', ['./class_go_1_1tp_march_point.html', 1, 'Go']],
 ['tptableentry', ['tpTableEntry', ['./class_go_1_1tp_table_entry.html', 1, 'Go']],
 ['tptolerances', ['tpTolerances', ['./struct_go_1_1tp_tolerances.html', 1, 'Go']],
 ['tptopologicalinfo', ['tpTopologicalInfo', ['./struct_go_1_1tp_topological_info.html', 1, 'Go']],
 ['tptopologytable', ['tpTopologyTable', ['./class_go_1_1tp_topology_table.html', 1, 'Go']],
 ['tracer', ['Tracer', ['./class_r_b_d___c_o_m_m_o_n_1_1_tracer.html', 1, 'RBD_COMMON']],
 ['transposedmatrix', ['TransposedMatrix', ['./class_n_e_w_m_a_t_1_1_transposed_matrix.html', 1, 'NEWMAT']],
 ['triangle', ['Triangle', ['./class_go_1_1_triangle.html', 1, 'Go']],
 ['triangulation', ['Triangulation', ['./classhed_1_1_triangulation.html', 1, 'hed']],
 ['triangulation', ['Triangulation', ['./classhetriang_1_1_triangulation.html', 1, 'hetriang']],
 ['trimcurve', ['TrimCurve', ['./class_go_1_1_trim_curve.html', 1, 'Go']],
 ['ttlpoint', ['ttlPoint', ['./class_go_1_1ttl_point.html', 1, 'Go']],
 ['ttltraits', ['TTLtraits', ['./structhed_1_1_t_t_t_traits.html', 1, 'hed']],
 ['ttltraits', ['TTLtraits', ['./structhetriang_1_1_t_t_t_traits.html', 1, 'hetriang']]
]
```

Definition at line 1 of file classes\_13.js.

## 30.1409 doc/html/search/classes\_14.js File Reference

### Variables

- var [searchData](#)

### 30.1409.1 Variable Documentation

#### 30.1409.1.1 var searchData

##### Initial value:

```
=
[
 ['unfnodetype', ['UnfNodeType', ['../class_unf_node_type.html', 1, '']]],
 ['unknownerror', ['UnknownError', ['../class_go_1_1_coons_patch_gen_1_1_un_known_error.html', 1, 'Go::CoonsPatchGen']]],
 ['upperbandmatrix', ['UpperBandMatrix', ['../class_n_e_w_m_a_t_1_1_upper_band_matrix.html', 1, 'NEWMAT']]],
 ['uppertriangularmatrix', ['UpperTriangularMatrix', ['../class_n_e_w_m_a_t_1_1_upper_triangular_matrix.html', 1, 'NEWMAT']]]
]
```

Definition at line 1 of file classes\_14.js.

## 30.1410 doc/html/search/classes\_15.js File Reference

### Variables

- var [searchData](#)

### 30.1410.1 Variable Documentation

#### 30.1410.1.1 var searchData

##### Initial value:

```
=
[
 ['vector3t', ['vector3t', ['../classvector3t.html', 1, '']]],
 ['vectorexception', ['VectorException', ['../class_n_e_w_m_a_t_1_1_vector_exception.html', 1, 'NEWMAT']]],
 ['vertex', ['Vertex', ['../class_go_1_1_vertex.html', 1, 'Go']]],
 ['volboundarycondition', ['VolBoundaryCondition', ['../class_go_1_1_vol_boundary_condition.html', 1, 'Go']]],
 ['volpointbdcond', ['VolPointBdCond', ['../class_go_1_1_vol_point_bd_cond.html', 1, 'Go']]],
 ['volsolution', ['VolSolution', ['../class_go_1_1_vol_solution.html', 1, 'Go']]],
 ['volumeadjacency', ['VolumeAdjacency', ['../class_go_1_1_volume_adjacency.html', 1, 'Go']]],
 ['volumeadjacencyinfo', ['VolumeAdjacencyInfo', ['../struct_go_1_1_volume_adjacency_info.html', 1, 'Go']]],
 ['volumemodel', ['VolumeModel', ['../class_go_1_1_volume_model.html', 1, 'Go']]],
 ['volumemodelfilehandler', ['VolumeModelFileHandler', ['../class_go_1_1_volume_model_file_handler.html', 1, 'Go']]],
 ['volumeparametercurve', ['VolumeParameterCurve', ['../class_go_1_1_volume_parameter_curve.html', 1, 'Go']]],
 ['volumespacecurve', ['VolumeSpaceCurve', ['../class_go_1_1_volume_space_curve.html', 1, 'Go']]]
]
```

Definition at line 1 of file classes\_15.js.

## 30.1411 doc/html/search/classes\_16.js File Reference

### Variables

- var [searchData](#)

#### 30.1411.1 Variable Documentation

##### 30.1411.1.1 var searchData

#### Initial value:

```
=
[
 ['zero_5fparameter_5fspan_5ferror', ['Zero_Parameter_Span_Error', ['
 ../class_go_1_1_zero___parameter___span___error.html', 1, 'Go']]]
]
```

Definition at line 1 of file classes\_16.js.

## 30.1412 doc/html/search/classes\_2.js File Reference

### Variables

- var [searchData](#)

#### 30.1412.1 Variable Documentation

##### 30.1412.1.1 var searchData

Definition at line 1 of file classes\_2.js.

## 30.1413 doc/html/search/classes\_3.js File Reference

### Variables

- var [searchData](#)

### 30.1413.1 Variable Documentation

#### 30.1413.1.1 var searchData

##### Initial value:

```
=
[
 ['dart', ['Dart', ['../classhed_1_1_dart.html', 1, 'hed']],
 ['dart', ['Dart', ['../classhetriang_1_1_dart.html', 1, 'hetriang']],
 ['datahandler', ['DataHandler', ['../class_data_handler.html', 1, ''']],
 ['datahandlervolandlr', ['DataHandlerVolAndLR', ['../class_data_handler_vol_and_lr.html', 1, ''']],
 ['defaultdatahandler', ['DefaultDataHandler', ['../class_default_data_handler.html', 1, ''']],
 ['degenerate_5finfo', ['degenerate_info', ['../struct_go_1_1_param_surface_1_1degenerate__info.html', 1, 'Go::ParamSurface']],
 ['degeneratedintersectioncurve', ['DegeneratedIntersectionCurve', ['../class_go_1_1_degenerated_intersection_curve.html', 1, 'Go']],
 ['diagedmatrix', ['DiagedMatrix', ['../class_n_e_w_m_a_t_1_1_diaged_matrix.html', 1, 'NEWMAT']],
 ['diagonalmatrix', ['DiagonalMatrix', ['../class_n_e_w_m_a_t_1_1_diagonal_matrix.html', 1, 'NEWMAT']],
 ['dijkstra', ['Dijkstra', ['../class_dijkstra.html', 1, ''']],
 ['directioncone', ['DirectionCone', ['../class_go_1_1_direction_cone.html', 1, 'Go']],
 ['disc', ['Disc', ['../class_go_1_1_disc.html', 1, 'Go']],
 ['domain', ['Domain', ['../class_go_1_1_domain.html', 1, 'Go']],
 ['domain_5ferror', ['Domain_error', ['../class_r_b_d__c_o_m_m_o_n_1_1_domain__error.html', 1, 'RBD_COMMON']],
],
 ['double_5fpair_5fhash', ['double_pair_hash', ['../struct_go_1_1_l_r_spline_surface_1_1double_pair_hash.html', 1, 'Go::LRSplineSurface']]
]
```

Definition at line 1 of file classes\_3.js.

## 30.1414 doc/html/search/classes\_4.js File Reference

### Variables

- var [searchData](#)

### 30.1414.1 Variable Documentation

#### 30.1414.1.1 var searchData

##### Initial value:

```
=
[
 ['edge', ['Edge', ['../classhed_1_1_edge.html', 1, 'hed']],
 ['edge', ['Edge', ['../classhetriang_1_1_edge.html', 1, 'hetriang']],
 ['edgetype', ['EdgeType', ['../class_edge_type.html', 1, ''']],
 ['edgevertex', ['EdgeVertex', ['../class_go_1_1_edge_vertex.html', 1, 'Go']],
 ['element2d', ['Element2D', ['../class_go_1_1_element2_d.html', 1, 'Go']],
 ['elementarycurve', ['ElementaryCurve', ['../class_go_1_1_elementary_curve.html', 1, 'Go']],
 ['elementarysurface', ['ElementarySurface', ['../class_go_1_1_elementary_surface.html', 1, 'Go']],
 ['elementaryvolume', ['ElementaryVolume', ['../class_go_1_1_elementary_volume.html', 1, 'Go']],
 ['elemkey', ['ElemKey', ['../struct_go_1_1_l_r_spline_surface_1_1_elem_key.html', 1, 'Go::LRSplineSurface']],
],
 ['ellipse', ['Ellipse', ['../class_go_1_1_ellipse.html', 1, 'Go']],
 ['entitylist', ['EntityList', ['../struct_go_1_1_entity_list.html', 1, 'Go']],
 ['eofexception', ['EofException', ['../class_go_1_1_streamable_1_1_eof_exception.html', 1, 'Go::Streamable']],
],
 ['evalcurve', ['EvalCurve', ['../class_go_1_1_eval_curve.html', 1, 'Go']],
 ['evalcurveset', ['EvalCurveSet', ['../class_go_1_1_eval_curve_set.html', 1, 'Go']],
 ['evalfunctorcurve', ['EvalFunctorCurve', ['../class_go_1_1_eval_functor_curve.html', 1, 'Go']],
 ['evalfunctorsurface', ['EvalFunctorSurface', ['../class_go_1_1_eval_functor_surface.html', 1, 'Go']],
 ['evaloffsetsurface', ['EvalOffsetSurface', ['../class_go_1_1_eval_offset_surface.html', 1, 'Go']],
 ['evalparamcurve', ['EvalParamCurve', ['../class_go_1_1_eval_param_curve.html', 1, 'Go']],
 ['evalsurface', ['EvalSurface', ['../class_go_1_1_eval_surface.html', 1, 'Go']]
]
```

Definition at line 1 of file classes\_4.js.

## 30.1415 doc/html/search/classes\_5.js File Reference

### Variables

- var [searchData](#)
- & gt

### 30.1415.1 Variable Documentation

#### 30.1415.1.1 & gt

Definition at line 9 of file classes\_5.js.

#### 30.1415.1.2 var searchData

#### Initial value:

```
=
[
 ['faceadjacency', ['FaceAdjacency', ['../class_go_1_1_face_adjacency.html', 1, 'Go']]],
 ['faceconnectivity', ['FaceConnectivity', ['../struct_go_1_1_face_connectivity.html', 1, 'Go']]],
 ['faceconnectivityutils', ['FaceConnectivityUtils', ['../class_go_1_1_face_connectivity_utils.html', 1, 'Go']
]],
 ['facesetquality', ['FaceSetQuality', ['../class_go_1_1_face_set_quality.html', 1, 'Go']]],
 ['facesetrepair', ['FaceSetRepair', ['../class_go_1_1_face_set_repair.html', 1, 'Go']]],
 ['factorial', ['Factorial', ['../struct_go_1_1_factorial.html', 1, 'Go']]],
 ['factorial_3c_201_20_3e', ['Factorial<
```

Definition at line 1 of file classes\_5.js.

## 30.1416 doc/html/search/classes\_6.js File Reference

### Variables

- var [searchData](#)
- [const T\\*](#) & gt

### 30.1416.1 Variable Documentation

#### 30.1416.1.1 [const T\\*](#) & gt

Definition at line 13 of file classes\_6.js.

## 30.1416.1.2 var searchData

## Initial value:

```
=
[
 ['garch11_5fll', ['GARCH11_LL', ['./class_g_a_r_c_h11___l_l.html', 1, '']]],
 ['generalmatrix', ['GeneralMatrix', ['./class_n_e_w_m_a_t_l_l_general_matrix.html', 1, 'NEWMAT']]],
 ['generalmesh', ['GeneralMesh', ['./class_go_l_l_general_mesh.html', 1, 'Go']]],
 ['genericmatrix', ['GenericMatrix', ['./class_n_e_w_m_a_t_l_l_generic_matrix.html', 1, 'NEWMAT']]],
 ['generictrimesh', ['GenericTriMesh', ['./class_go_l_l_generic_tri_mesh.html', 1, 'Go']]],
 ['geomobject', ['GeomObject', ['./class_go_l_l_geom_object.html', 1, 'Go']]],
 ['geomobjectint', ['GeomObjectInt', ['./class_go_l_l_geom_object_int.html', 1, 'Go']]],
 ['geotol', ['GeoTol', ['./class_go_l_l_geo_tol.html', 1, 'Go']]],
 ['getsubmatrix', ['GetSubMatrix', ['./class_n_e_w_m_a_t_l_l_get_sub_matrix.html', 1, 'NEWMAT']]],
 ['go_5fiterator_5ftraits', ['go_iterator_traits', ['./struct_go_l_lgo__iterator__traits.html', 1, 'Go']]],
 ['go_5fiterator_5ftraits_3c_20const_20t_20_2a_20_3e', ['go_iterator_traits<
```

Definition at line 1 of file classes\_6.js.

## 30.1417 doc/html/search/classes\_7.js File Reference

## Variables

- var [searchData](#)
- [hed::Node](#) & [gt](#)

## 30.1417.1 Variable Documentation

## 30.1417.1.1 hed::Node&amp; gt

Definition at line 4 of file classes\_7.js.

## 30.1417.1.2 var searchData

## Initial value:

```
=
[
 ['handle', ['Handle', ['./class_handle.html', 1, '']]],
 ['handle_3c_20hed_3a_3anode_20_3e', ['Handle<
```

Definition at line 1 of file classes\_7.js.

## 30.1418 doc/html/search/classes\_8.js File Reference

## Variables

- var [searchData](#)

### 30.1418.1 Variable Documentation

#### 30.1418.1.1 var searchData

Definition at line 1 of file classes\_8.js.

## 30.1419 doc/html/search/classes\_9.js File Reference

### Variables

- var [searchData](#)

### 30.1419.1 Variable Documentation

#### 30.1419.1.1 var searchData

#### Initial value:

```
=
[
 ['janitor', ['Janitor', ['./class_r_b_d___c_o_m_m_o_n_1_1_janitor.html', 1, 'RBD_COMMON']]]
]
```

Definition at line 1 of file classes\_9.js.

## 30.1420 doc/html/search/classes\_a.js File Reference

### Variables

- var [searchData](#)

### 30.1420.1 Variable Documentation

#### 30.1420.1.1 var searchData

#### Initial value:

```
=
[
 ['kpmatrix', ['KPMatrix', ['./class_n_e_w_m_a_t_1_1_k_p_matrix.html', 1, 'NEWMAT']]]
]
```

Definition at line 1 of file classes\_a.js.



## 30.1421 doc/html/search/classes\_b.js File Reference

### Variables

- var [searchData](#)

### 30.1421.1 Variable Documentation

#### 30.1421.1.1 var searchData

##### Initial value:

```
=
[
 ['length_5ferror', ['Length_error', ['./class_r_b_d_c_o_m_m_o_n_l_l_length_error.html', 1, 'RBD_COMMON']],
],
 ['liftcurve', ['LiftCurve', ['./class_go_1_l_lift_curve.html', 1, 'Go']],
 ['line', ['Line', ['./class_go_1_l_line.html', 1, 'Go']],
 ['line2dint', ['Line2DInt', ['./class_go_1_l_line2_d_int.html', 1, 'Go']],
 ['linearequationsolver', ['LinearEquationSolver', ['./class_n_e_w_m_a_t_l_l_linear_equation_solver.html', 1, 'NEWMAT']],
],
 ['linecloud', ['LineCloud', ['./class_go_1_l_line_cloud.html', 1, 'Go']],
 ['linecloudtesselator', ['LineCloudTesselator', ['./class_go_1_l_line_cloud_tesselator.html', 1, 'Go']],
 ['linestrip', ['LineStrip', ['./class_go_1_l_line_strip.html', 1, 'Go']],
 ['ll_5fd_5ffi', ['LL_D_FI', ['./class_n_e_w_m_a_t_l_l_l_l_d_f_i.html', 1, 'NEWMAT']],
 ['loadandstoreflag', ['LoadAndStoreFlag', ['./class_load_and_store_flag.html', 1, '']],
 ['lockeddirdistfunc', ['LockedDirDistFunc', ['./class_go_1_l_locked_dir_dist_func.html', 1, 'Go']],
 ['logandsign', ['LogAndSign', ['./class_n_e_w_m_a_t_l_l_log_and_sign.html', 1, 'NEWMAT']],
 ['logic_5ferror', ['Logic_error', ['./class_r_b_d_c_o_m_m_o_n_l_l_logic_error.html', 1, 'RBD_COMMON']],
 ['loop', ['Loop', ['./class_go_1_l_loop.html', 1, 'Go']],
 ['lowerbandmatrix', ['LowerBandMatrix', ['./class_n_e_w_m_a_t_l_l_lower_band_matrix.html', 1, 'NEWMAT']],
 ['lowertriangularmatrix', ['LowerTriangularMatrix', ['./class_n_e_w_m_a_t_l_l_lower_triangular_matrix.html', 1, 'NEWMAT']],
],
 ['lrbspline2d', ['LRBSpline2D', ['./class_go_1_l_l_r_b_spline2_d.html', 1, 'Go']],
 ['lrsplineevalgrid', ['LRSplineEvalGrid', ['./class_go_1_l_l_r_spline_eval_grid.html', 1, 'Go']],
 ['lrsplinesurface', ['LRSplineSurface', ['./class_go_1_l_l_r_spline_surface.html', 1, 'Go']],
 ['lrsurfapprox', ['LRSurfApprox', ['./class_go_1_l_l_r_surf_approx.html', 1, 'Go']],
 ['lrsurfsmoothls', ['LRSurfSmoothLS', ['./class_go_1_l_l_r_surf_smooth_ls.html', 1, 'Go']],
 ['lssmoothdata', ['LSSmoothData', ['./struct_go_1_l_l_s_smooth_data.html', 1, 'Go']]
]
]
```

Definition at line 1 of file classes\_b.js.

## 30.1422 doc/html/search/classes\_c.js File Reference

### Variables

- var [searchData](#)
- [double](#)
- [&gt;](#)

### 30.1422.1 Variable Documentation

#### 30.1422.1.1 double

Definition at line 15 of file classes\_c.js.

## 30.1422.1.2 &amp; gt

Definition at line 15 of file classes\_c.js.

## 30.1422.1.3 var searchData

## Initial value:

```
=
[
 ['marchpoint', ['MarchPoint', ['./class_go_1_1_march_point.html', 1, 'Go']],
 ['matedmatrix', ['MatedMatrix', ['./class_n_e_w_m_a_t_1_1_mated_matrix.html', 1, 'NEWMAT']],
 ['material_5fappearance', ['material_appearance', ['./classmaterial__appearance.html', 1, ''']],
 ['matrix', ['Matrix', ['./class_n_e_w_m_a_t_1_1_matrix.html', 1, 'NEWMAT']],
 ['matrixbandwidth', ['MatrixBandWidth', ['./class_n_e_w_m_a_t_1_1_matrix_band_width.html', 1, 'NEWMAT']],
 ['matrixcol', ['MatrixCol', ['./class_matrix_col.html', 1, ''']],
 ['matrixcolx', ['MatrixColX', ['./class_matrix_col_x.html', 1, ''']],
 ['matrixinput', ['MatrixInput', ['./class_n_e_w_m_a_t_1_1_matrix_input.html', 1, 'NEWMAT']],
 ['matrixrow', ['MatrixRow', ['./class_matrix_row.html', 1, ''']],
 ['matrixrowcol', ['MatrixRowCol', ['./class_matrix_row_col.html', 1, ''']],
 ['matrixtype', ['MatrixType', ['./class_n_e_w_m_a_t_1_1_matrix_type.html', 1, 'NEWMAT']],
 ['matrixxd', ['MatrixXD', ['./class_go_1_1_matrix_x_d.html', 1, 'Go']],
 ['matrixxd_3c_20double_2c_202_20_3e', ['MatrixXD<
```

Definition at line 1 of file classes\_c.js.

## 30.1423 doc/html/search/classes\_d.js File Reference

## Variables

- var [searchData](#)

## 30.1423.1 Variable Documentation

## 30.1423.1.1 var searchData

## Initial value:

```
=
[
 ['negatedmatrix', ['NegatedMatrix', ['./class_n_e_w_m_a_t_1_1_negated_matrix.html', 1, 'NEWMAT']],
 ['negshiftedmatrix', ['NegShiftedMatrix', ['./class_n_e_w_m_a_t_1_1_neg_shifted_matrix.html', 1, 'NEWMAT']],
 ['node', ['Node', ['./classhetriang_1_1_node.html', 1, 'hetriang']],
 ['node', ['Node', ['./classhed_1_1_node.html', 1, 'hed']],
 ['nonevaluableintersectioncurve', ['NonEvaluableIntersectionCurve', ['./class_go_1_1_non_evaluable_intersection_curve.html', 1, 'Go']],
 ['nonlinearleast_squares', ['NonLinearLeastSquares', ['./class_n_e_w_m_a_t_1_1_non_linear_least_squares.html', 1, 'NEWMAT']],
 ['nooptesselator', ['NoopTesselator', ['./class_go_1_1_noop_tesselator.html', 1, 'Go']],
 ['notdefinedexception', ['NotDefinedException', ['./class_n_e_w_m_a_t_1_1_not_defined_exception.html', 1, 'NEWMAT']],
 ['notsquareexception', ['NotSquareException', ['./class_n_e_w_m_a_t_1_1_not_square_exception.html', 1, 'NEWMAT']],
 ['npdexception', ['NPDException', ['./class_n_e_w_m_a_t_1_1_n_p_d_exception.html', 1, 'NEWMAT']],
 ['nricmatrix', ['nricMatrix', ['./class_n_e_w_m_a_t_1_1_nric_matrix.html', 1, 'NEWMAT']]
]
```

Definition at line 1 of file classes\_d.js.

## 30.1424 doc/html/search/classes\_e.js File Reference

### Variables

- var [searchData](#)

#### 30.1424.1 Variable Documentation

##### 30.1424.1.1 var searchData

#### Initial value:

```
=
[
 ['objectheader', ['ObjectHeader', ['./class_go_1_1_object_header.html', 1, 'Go']],
 ['onedimsolve', ['OneDimSolve', ['./class_r_b_d___c_o_m_m_o_n_1_1_one_dim_solve.html', 1, 'RBD_COMMON']],
 ['out_5fof_5frange', ['Out_of_range', ['./class_r_b_d___c_o_m_m_o_n_1_1_out_of__range.html', 1, 'RBD_COMMON
 ']]],
 ['overflow_5ferror', ['Overflow_error', ['./class_r_b_d___c_o_m_m_o_n_1_1_overflow__error.html', 1, '
 RBD_COMMON']],
 ['overflowexception', ['OverflowException', ['./class_n_e_w_m_a_t_1_1_overflow_exception.html', 1, 'NEWMAT']
]]
]
```

Definition at line 1 of file classes\_e.js.

## 30.1425 doc/html/search/classes\_f.js File Reference

### Variables

- var [searchData](#)
- & [gt](#)

#### 30.1425.1 Variable Documentation

##### 30.1425.1.1 & gt

Definition at line 27 of file classes\_f.js.

## 30.1425.1.2 var searchData

## Initial value:

```
=
[
 ['par0funcintersector', ['Par0FuncIntersector', ['../class_go_1_1_par0_func_intersector.html', 1, 'Go']]],
 ['par1funcintersector', ['Par1FuncIntersector', ['../class_go_1_1_par1_func_intersector.html', 1, 'Go']]],
 ['par2funcintersector', ['Par2FuncIntersector', ['../class_go_1_1_par2_func_intersector.html', 1, 'Go']]],
 ['parabola', ['Parabola', ['../class_go_1_1_parabola.html', 1, 'Go']]],
 ['parallelepiped', ['Parallelepiped', ['../class_go_1_1_parallelepiped.html', 1, 'Go']]],
 ['param0functionint', ['Param0FunctionInt', ['../class_go_1_1_param0_function_int.html', 1, 'Go']]],
 ['param1functionint', ['Param1FunctionInt', ['../class_go_1_1_param1_function_int.html', 1, 'Go']]],
 ['param2functionint', ['Param2FunctionInt', ['../class_go_1_1_param2_function_int.html', 1, 'Go']]],
 ['paramcurve', ['ParamCurve', ['../class_go_1_1_param_curve.html', 1, 'Go']]],
 ['paramcurveint', ['ParamCurveInt', ['../class_go_1_1_param_curve_int.html', 1, 'Go']]],
 ['parametricsurfacepropertysheet', ['ParametricSurfacePropertySheet', ['
 ../class_parametric_surface_property_sheet.html', 1, '']]],
 ['parametricsurfacetessellator', ['ParametricSurfaceTessellator', ['
 ../class_go_1_1_parametric_surface_tessellator.html', 1, 'Go']]],
 ['paramfunctionint', ['ParamFunctionInt', ['../class_go_1_1_param_function_int.html', 1, 'Go']]],
 ['paramgeomint', ['ParamGeomInt', ['../class_go_1_1_param_geom_int.html', 1, 'Go']]],
 ['paramobjectint', ['ParamObjectInt', ['../class_go_1_1_param_object_int.html', 1, 'Go']]],
 ['parampointint', ['ParamPointInt', ['../class_go_1_1_param_point_int.html', 1, 'Go']]],
 ['paramsurface', ['ParamSurface', ['../class_go_1_1_param_surface.html', 1, 'Go']]],
 ['paramsurfaceint', ['ParamSurfaceInt', ['../class_go_1_1_param_surface_int.html', 1, 'Go']]],
 ['paramvolume', ['ParamVolume', ['../class_go_1_1_param_volume.html', 1, 'Go']]],
 ['pathtype', ['PathType', ['../class_path_type.html', 1, '']]],
 ['plane', ['Plane', ['../class_go_1_1_plane.html', 1, 'Go']]],
 ['planeint', ['PlaneInt', ['../class_go_1_1_plane_int.html', 1, 'Go']]],
 ['point', ['Point', ['../class_go_1_1_point.html', 1, 'Go']]],
 ['pointcloud', ['PointCloud', ['../class_go_1_1_point_cloud.html', 1, 'Go']]],
 ['pointcloud_3c_203_20_3e', ['PointCloud<
```

Definition at line 1 of file classes\_f.js.

## 30.1426 doc/html/search/defines\_0.js File Reference

## Variables

- var [searchData](#)

## 30.1426.1 Variable Documentation

## 30.1426.1.1 var searchData

## Initial value:

```
=
[
 ['_5fcopyright_5fh', ['_COPYRIGHT_H', ['
 ../gotools-core_2include_2go_tools_2geometry_2copyright_8h.html#abdfb6b142e6341a6d5e35b87ccbfab07', 1, 'copyright.h']]],
 ['_5ferrormacros_5fh', ['_ERRORMACROS_H', ['../aux2_8h.html#a3f97e52087e510f4d7c49c7eb28624c0', 1, 'aux2.h']]],
 ['_5figeslib', ['_IGESLIB', ['../igeslib_doxymain_8h.html#aa60f7acaa57d814a1403f7bb16981c1d', 1, '
 igeslib_doxymain.h']]],
 ['_5flrsplineevavgrid_5fh', ['_LRSPLINEEVAVGRID_H', ['
 ../_1_r_spline_eval_grid_8h.html#a861377e9079b564d42abe23579c6992b', 1, 'LRSplineEvalGrid.h']]],
 ['_5foffsetutils_5fh', ['_OFFSETUTILS_H', ['
 ../_creators_offset_utils_8h.html#ad9fc2da1082c23bb539309710e76122f', 1, 'CreatorsOffsetUtils.h']]],
 ['_5fvolumehandler_5fh', ['_VOLUMEFILEHANDLER_H', ['
 ../_volume_model_file_handler_8h.html#a1d5505728388942e4ea90d4be08daaea', 1, 'VolumeModelFileHandler.h']]]
]
```

Definition at line 1 of file defines\_0.js.

## 30.1427 doc/html/search/defines\_1.js File Reference

### Variables

- var [searchData](#)
- [errormacros](#) h

### 30.1427.1 Variable Documentation

#### 30.1427.1.1 [errormacros](#) h

Definition at line 7 of file [defines\\_1.js](#).

#### 30.1427.1.2 var [searchData](#)

##### Initial value:

```
=
[
 ['aepsco', ['AEPSCO', ['../sisl_p_8h.html#a431756bcd46dae9cee579fc4f8d482e9', 1, 'sislP.h']]],
 ['aepsge', ['AEPSGE', ['../sisl_p_8h.html#af789738847b509ac079cbab0963130b1', 1, 'sislP.h']]],
 ['always_5ferror_5fif', ['ALWAYS_ERROR_IF', ['../errormacros_8h.html#a86611c37e530eb06c9b077fed130b2dc', 1, 'errormacros.h']]],
 ['angular_5ftolerance', ['ANGULAR_TOLERANCE', ['../sisl_p_8h.html#ac74392922bc63d29f53a31ad9fff6fe8', 1, 'sislP.h']]],
 ['assert', ['ASSERT', ['../errormacros_8h.html#a0966b817b229d48e5ffc7feab19a0be6', 1, 'ASSERT():
```

Definition at line 1 of file [defines\\_1.js](#).

## 30.1428 doc/html/search/defines\_10.js File Reference

### Variables

- var [searchData](#)

### 30.1428.1 Variable Documentation

#### 30.1428.1.1 var [searchData](#)

##### Initial value:

```
=
[
 ['voidp', ['VOIDP', ['../sisl_p_8h.html#aa261d4359cc2e4f4df5dace402c1b809', 1, 'sislP.h']]]
]
```

Definition at line 1 of file [defines\\_10.js](#).

## 30.1429 doc/html/search/defines\_11.js File Reference

### Variables

- var [searchData](#)
- garch [cpp](#)

### 30.1429.1 Variable Documentation

#### 30.1429.1.1 garch [cpp](#)

Definition at line 3 of file defines\_11.js.

#### 30.1429.1.2 var [searchData](#)

##### Initial value:

```
=
[
 ['want_5ffstream', ['WANT_FSTREAM', ['../garch_8cpp.html#a37d625862c54954e237886ad4bb5e75f', 1, '
 WANT_FSTREAM():
```

Definition at line 1 of file defines\_11.js.

## 30.1430 doc/html/search/defines\_2.js File Reference

### Variables

- var [searchData](#)

### 30.1430.1 Variable Documentation

#### 30.1430.1.1 var [searchData](#)

##### Initial value:

```
=
[
 ['bool_5flib', ['bool_LIB', ['../boolean_8h.html#ae034ee292f37bf7f1a5488c0afe12cfd', 1, 'boolean.h']]]
```

Definition at line 1 of file defines\_2.js.

## 30.1431 doc/html/search/defines\_3.js File Reference

### Variables

- var [searchData](#)
- [s17702d c](#)

#### 30.1431.1 Variable Documentation

##### 30.1431.1.1 [s17702d c](#)

Definition at line 11 of file defines\_3.js.

##### 30.1431.1.2 [var searchData](#)

#### Initial value:

```
=
[
 ['catch', ['Catch', ['../myexcept_8h.html#af19f2de9b4bccab916654c258aa6f2d4', 1, 'myexcept.h']]],
 ['catchall', ['CatchAll', ['../myexcept_8h.html#a549ceb8bbab061440f4698483d6ad179', 1, 'myexcept.h']]],
 ['catchandthrow', ['CatchAndThrow', ['../myexcept_8h.html#a3a7fd3e342806240062de1b14ff917ca', 1, 'myexcept.h'
]]],
 ['check', ['CHECK', ['../_bspline_basis_8h.html#a70246e9ac30d8d3e1fba7ca97cd5f4e8', 1, 'BsplineBasis.h']]],
 ['const', ['CONST', ['../sisl_p_8h.html#a0c33b494a68ce28497e7ce8e5e95feff', 1, 'sislP.h']]],
 ['construct', ['CONSTRUCT', ['../construct_8c.html#a158cc7b98afd3caaf2cf9c39be593b5', 1, 'construct.c']]],
 ['constvoidp', ['CONSTVOIDP', ['../sisl_p_8h.html#a3fe7ac8c946b70b8f70092b9921359a3', 1, 'sislP.h']]],
 ['control_5fword_5flib', ['CONTROL_WORD_LIB', ['../controlw_8h.html#ac3f4cc42dfc44a8de07e00956f1f192e', 1, '
 controlw.h']]],
 ['copy2', ['copy2', ['../s17702d_8c.html#a94d298c6c82c75789d11ea2fad958ff1', 1, 'copy2() :
```

Definition at line 1 of file defines\_3.js.

## 30.1432 doc/html/search/defines\_4.js File Reference

### Variables

- var [searchData](#)
- [s17702d c](#)

#### 30.1432.1 Variable Documentation

##### 30.1432.1.1 [s17702d c](#)

Definition at line 5 of file defines\_4.js.

## 30.1432.1.2 var searchData

## Initial value:

```
=
[
 ['dbgqqq', ['DBGqqq', ['./sisl__aux_8cpp.html#a22e3c68f7ee44d24da231675c4aceafa', 1, 'sisl_aux.cpp']],
 ['debug_5ferror_5fif', ['DEBUG_ERROR_IF', ['./errormacros_8h.html#ad369e2a5e24adc64ecb0e600750372f5', 1, 'errormacros.h']],
 ['decr2', ['decr2', ['./s17702d_8c.html#aa8ea14f873c1bff5a2e825fff20759e33', 1, 'decr2() :
```

Definition at line 1 of file defines\_4.js.

## 30.1433 doc/html/search/defines\_5.js File Reference

## Variables

- var [searchData](#)

## 30.1433.1 Variable Documentation

## 30.1433.1.1 var searchData

## Initial value:

```
=
[
 ['eps', ['EPS', ['./_pr_geodesics_8h.html#a6ebf6899d6c1c8b7b9d09be872c05aae', 1, 'PrGeodesics.h']],
 ['error_5fif', ['ERROR_IF', ['./aux2_8h.html#a9553d62d235151ba08f888c15714d321', 1, 'aux2.h']],
 ['esc_5fstring', ['ESC_STRING', ['./sisl__view__demo_8cpp.html#a3b6ad63436c143876aa93611c3328d8d', 1, 'sisl_view_demo.cpp']],
 ['ev_5fcv_5foff', ['EV_CV_OFF', ['./ev__cv__off_8c.html#af570ffd133e823ba65e4594bbba6e69a', 1, 'ev_cv_off.c']],
 ['eval_5f2_5fcrv', ['EVAL_2_CRV', ['./eval__2__crv_8c.html#ae21399dc6a70f8168d0e3926598430cc', 1, 'eval_2_crv.c']],
 ['eval_5fcrv_5farc', ['EVAL_CRV_ARC', ['./evalcrvarc_8c.html#a8c6fa6c95e08223f26bb7f9a6ba8eccc', 1, 'evalcrvarc.c']]
]
```

Definition at line 1 of file defines\_5.js.

## 30.1434 doc/html/search/defines\_6.js File Reference

## Variables

- var [searchData](#)



### 30.1434.1 Variable Documentation

#### 30.1434.1.1 var searchData

**Initial value:**

```
=
[
 ['free_5fcheck', ['FREE_CHECK', ['../myexcept_8h.html#a38496bbc880a5b060716ea39a9e67990', 1, 'myexcept.h']]]
]
```

Definition at line 1 of file defines\_6.js.

## 30.1435 doc/html/search/defines\_7.js File Reference

### Variables

- var [searchData](#)
- [config h](#)

### 30.1435.1 Variable Documentation

#### 30.1435.1.1 config h

Definition at line 5 of file defines\_7.js.

#### 30.1435.1.2 var searchData

**Initial value:**

```
=
[
 ['glut_5fdisable_5fatexit_5fhack', ['GLUT_DISABLE_ATEXIT_HACK', [
 ../mouse_8cpp.html#ab07b3c93a1236fc83b162fb0a8945d72', 1, 'mouse.cpp']]],
 ['glutils_5fh_5fincluded', ['GLUTILS_H_INCLUDED', ['../glutils_8h.html#acd507b0c3c98dc8199ba953051d907ff', 1,
 'glutils.h']]],
 ['go_5fapi', ['GO_API', ['../config_8h.html#aa9fe0b1a0029021ae339602cdfc4d8dd', 1, 'GO_API():
```

Definition at line 1 of file defines\_7.js.

## 30.1436 doc/html/search/defines\_8.js File Reference

### Variables

- var [searchData](#)

### 30.1436.1 Variable Documentation

#### 30.1436.1.1 var searchData

**Initial value:**

```
=
[
 ['hp_5fs1880', ['HP_S1880', ['../hp__s1880_8c.html#ae6033f654b8655d58b42ee58d81bce23', 1, 'hp_s1880.c']]]
]
```

Definition at line 1 of file defines\_8.js.

### 30.1437 doc/html/search/defines\_9.js File Reference

#### Variables

- var [searchData](#)
- [s17702d c](#)

#### 30.1437.1 Variable Documentation

##### 30.1437.1.1 [s17702d c](#)

Definition at line 3 of file defines\_9.js.

##### 30.1437.1.2 var searchData

**Initial value:**

```
=
[
 ['incr2', ['incr2', ['../s17702d_8c.html#a66086fa9df80019f1e7929b6368f30dc', 1, 'incr2() :
```

Definition at line 1 of file defines\_9.js.

### 30.1438 doc/html/search/defines\_a.js File Reference

#### Variables

- var [searchData](#)
- [s1891 c](#)

## 30.1438.1 Variable Documentation

### 30.1438.1.1 s1891 c

Definition at line 14 of file defines\_a.js.

### 30.1438.1.2 var searchData

#### Initial value:

```
=
[
 ['m_5fpi', ['M_PI', ['../glutils_8cpp.html#ae71449b1cc6e6250b91f539153a7a0d3', 1, 'glutils.cpp']],
 ['make3d', ['MAKE3D', ['../make3_d_8c.html#af73dcd2ed246a565df1162b610461e64', 1, 'make3D.c']],
 ['make_5fcv_5fcyclic', ['MAKE_CV_CYCLIC', ['../mk_cv_cycl_8c.html#afd0d637cd833f1e12447d183a72bf1ee', 1, 'mk_cv_cycl.c']],
 ['make_5fcv_5fkreg', ['MAKE_CV_KREG', ['../makecvkreg_8c.html#a3d78c8854b03c4e271f7fcdc2642aaaa', 1, 'makecvkreg.c']],
 ['make_5fsf_5fkreg', ['MAKE_SF_KREG', ['../makesfkreg_8c.html#ab2dd7beaefalb53540f56163621670cd', 1, 'makesfkreg.c']],
 ['make_5ftracks', ['MAKE_TRACKS', ['../maketracks_8c.html#af500037a7945500574264d2dda094220', 1, 'maketracks.c']],
 ['matrixtypeunsp', ['MatrixTypeUnSp', ['../newmat_8h.html#af54b3134157ba2977e1f765c2f15c01f', 1, 'newmat.h']],
],
 ['max', ['MAX', ['../aux2_8h.html#a99ec4acc4ecb2dc3c2d05da15d0e3f', 1, 'aux2.h']],
 ['max_5fcurves', ['MAX_CURVES', ['../sisl__view__demo_8cpp.html#a389f931619cd0eb55d3854c3556c3e93', 1, 'sisl_view_demo.cpp']],
 ['max_5fik', ['MAX_IK', ['../s1897_8c.html#ac79ba5168f3ea53c51d1f9aa515cf7b3', 1, 's1897.c']],
 ['max_5flines', ['MAX_LINES', ['../sisl__view__demo_8cpp.html#a07e2b531c72985b064c431c05dfbc5fc', 1, 'sisl_view_demo.cpp']],
 ['max_5fsize', ['MAX_SIZE', ['../s1891_8c.html#a0592dba56693fad79136250c11e5a7fe', 1, 'MAX_SIZE():
```

Definition at line 1 of file defines\_a.js.

## 30.1439 doc/html/search/defines\_b.js File Reference

### Variables

- var [searchData](#)

## 30.1439.1 Variable Documentation

### 30.1439.1.1 var searchData

#### Initial value:

```
=
[
 ['ndequal', ['NDEQUAL', ['../aux2_8h.html#adcab733c45d4a4268b7adda3fb6296ef', 1, 'aux2.h']],
 ['ndequal2', ['NDEQUAL2', ['../aux2_8h.html#a8511c5447adf91b377927225414d8303', 1, 'aux2.h']],
 ['new_5fdelete', ['NEW_DELETE', ['../myexcept_8h.html#a7814d199e1da1e1a6255fa85f8e8c320', 1, 'myexcept.h']],
 ['newknots', ['NEWKNOTS', ['../newknots_8c.html#a2cfe4b9fd93d72364a6776e3f87406b9', 1, 'newknots.c']],
 ['newmat_5flib', ['NEWMAT_LIB', ['../newmat_8h.html#acc34192eabbe4f4cd977a4bab78a4bdd0', 1, 'newmat.h']],
 ['newmatap_5flib', ['NEWMATAP_LIB', ['../newmatap_8h.html#a036ff4b2a9e34bb7de87024107995c9', 1, 'newmatap.h']],
],
 ['newmatio_5flib', ['NEWMATIO_LIB', ['../newmatio_8h.html#a65fb8e2c45ceffd1871256a348617cbf', 1, 'newmatio.h']],
],
 ['newmatnl_5flib', ['NEWMATNL_LIB', ['../newmatnl_8h.html#aed6d29a446241bb9799d3e281972c9c1', 1, 'newmatnl.h']],
],
 ['newmatrc_5flib', ['NEWMATRC_LIB', ['../newmatrc_8h.html#a96d76d6bee402806b06fcce48adf34e0', 1, 'newmatrc.h']],
],
 ['newmatrm_5flib', ['NEWMATRM_LIB', ['../newmatrm_8h.html#a57c1191464ba3297be3ac91c6a339c60', 1, 'newmatrm.h']],
]
]
```

Definition at line 1 of file defines\_b.js.

## 30.1440 doc/html/search/defines\_c.js File Reference

### Variables

- var [searchData](#)
- [aux2](#) h

### 30.1440.1 Variable Documentation

#### 30.1440.1.1 [aux2](#) h

Definition at line 3 of file `defines_c.js`.

#### 30.1440.1.2 var [searchData](#)

#### Initial value:

```
=
[
 ['pi', ['PI', ['./aux2_8h.html#a598a3330b3c21701223ee0ca14316eca', 1, 'PI() :
```

Definition at line 1 of file `defines_c.js`.

## 30.1441 doc/html/search/defines\_d.js File Reference

### Variables

- var [searchData](#)
- [errormacros](#) h

### 30.1441.1 Variable Documentation

#### 30.1441.1.1 [errormacros](#) h

Definition at line 7 of file `defines_d.js`.

#### 30.1441.1.2 var [searchData](#)

#### Initial value:

```
=
[
 ['rasterutils_5fh_5fincluded', ['RASTERUTILS_H_INCLUDED', ['./raster_8h.html#ae6d95c22bae73000aac0efdc2997c55d', 1, 'raster.h']],
 ['refine_5fall', ['REFINE_ALL', ['./refine__all_8c.html#a7021f2e1b451a7d88475a011da9cffe9', 1, 'refine_all.c']],
 ['rel_5fcomp_5fres', ['REL_COMP_RES', ['./sisl_p_8h.html#a82d2181b25bb9fa66c5f5b705bfff0915', 1, 'sislP.h']],
 ['rel_5fpar_5fres', ['REL_PAR_RES', ['./sisl_p_8h.html#a673bad4c1fd60a19bda31bfb9ab40fd', 1, 'sislP.h']],
 ['report', ['REPORT', ['./errormacros_8h.html#a787099914a94ed31fa544b985b55752f', 1, 'REPORT() :
```

Definition at line 1 of file `defines_d.js`.

## 30.1442 doc/html/search/defines\_e.js File Reference

### Variables

- var [searchData](#)
- [s1366 c](#)

### 30.1442.1 Variable Documentation

#### 30.1442.1.1 [s1366 c](#)

Definition at line 109 of file defines\_e.js.

#### 30.1442.1.2 var [searchData](#)

Definition at line 1 of file defines\_e.js.

## 30.1443 doc/html/search/defines\_f.js File Reference

### Variables

- var [searchData](#)
- [myexcept h](#)

### 30.1443.1 Variable Documentation

#### 30.1443.1.1 [myexcept h](#)

Definition at line 6 of file defines\_f.js.

#### 30.1443.1.2 var [searchData](#)

#### Initial value:

```
=
[
 ['tab_5fstring', ['TAB_STRING', ['../sisl__view__demo_8cpp.html#a0d3b5deaf6bcb60d1dfb8c6dbc234446', 1, '
 sisl_view_demo.cpp']],
 ['test_5fcyclic_5fknots', ['TEST_CYCLIC_KNOTS', ['../tstcyclic_8c.html#a77b5bb7a941d583a9aee41d2be40d163',
 1, 'tstcyclic.c']],
 ['test_5fsuite_5fsize', ['TEST_SUITE_SIZE', ['../test_suite_8h.html#abbc02a22a96cfac45964a8fd615181b', 1, '
 testSuite.h']],
 ['throw', ['Throw', ['../myexcept_8h.html#a7554028bfd929fe4391cb3f9ed7b9da1', 1, 'Throw() :
```

Definition at line 1 of file defines\_f.js.

## 30.1444 doc/html/search/enums\_0.js File Reference

### Variables

- var [searchData](#)

### 30.1444.1 Variable Documentation

#### 30.1444.1.1 var searchData

##### Initial value:

```
=
[
 ['algorithmchoice', ['AlgorithmChoice', ['../namespace_go.html#a9ec4373c3aa2905cb547164492f510d1', 1, 'Go']]]
 ['attribute', ['Attribute', ['../class_n_e_w_m_a_t_1_1_matrix_type.html#af1adc0086f5d4c3328ad4a698fa57c90',
 1, 'NEWMAT::MatrixType']]]
]
```

Definition at line 1 of file enums\_0.js.

## 30.1445 doc/html/search/enums\_1.js File Reference

### Variables

- var [searchData](#)

### 30.1445.1 Variable Documentation

#### 30.1445.1.1 var searchData

##### Initial value:

```
=
[
 ['bdconditiontype', ['BdConditionType', ['../namespace_go.html#a2ef4e46477cac9d1ab25de9f9be20fed7', 1, 'Go']]]
]
```

Definition at line 1 of file enums\_1.js.

## 30.1446 doc/html/search/enums\_2.js File Reference

### Variables

- var [searchData](#)

### 30.1446.1 Variable Documentation

#### 30.1446.1.1 var searchData

**Initial value:**

```
=
[
 ['classtype', ['ClassType', ['../namespace_go.html#a105371b9620b32708c1de810631c2fe2', 1, 'Go']]],
 ['closestpointlevel', ['closestPointLevel', ['../namespace_go.html#a64e7b23868d471d8f59b8e280437195e', 1, 'Go']]],
 ['coefstatus', ['CoefStatus', ['../namespace_go.html#a24654e7a71d69eef31552a072adeecc4', 1, 'Go']]],
 ['condtype', ['CondType', ['../class_go_1_1_spline_interpolator.html#aacf4f80d98ea6b554729125fafd24a78', 1, 'Go::SplineInterpolator']]]
]
```

Definition at line 1 of file enums\_2.js.

## 30.1447 doc/html/search/enums\_3.js File Reference

### Variables

- var [searchData](#)

### 30.1447.1 Variable Documentation

#### 30.1447.1.1 var searchData

**Initial value:**

```
=
[
 ['direction2d', ['Direction2D', ['../namespace_go.html#a821a750d4a6740898072cf759246eb87', 1, 'Go']]]
]
```

Definition at line 1 of file enums\_3.js.

## 30.1448 doc/html/search/enums\_4.js File Reference

### Variables

- var [searchData](#)

### 30.1448.1 Variable Documentation

#### 30.1448.1.1 var searchData

##### Initial value:

```
=
[
 ['envmodeset', ['EnvModeSet', ['./gv_texture_8h.html#abfb34fc206cb7e7d2ede80704c1fcc86', 1, 'gvTexture.h']]],
 ['estimatedirection', ['EstimateDirection', ['./namespace_go.html#a73ae93efa43d949b788530375bae4a09', 1, 'Go'
]]],
 ['evalkind', ['EvalKind', ['./namespace_go.html#a11cd92eeab3979030879414ab681ae8f', 1, 'Go']]]
]
```

Definition at line 1 of file enums\_4.js.

### 30.1449 doc/html/search/enums\_5.js File Reference

#### Variables

- var [searchData](#)

### 30.1449.1 Variable Documentation

#### 30.1449.1.1 var searchData

##### Initial value:

```
=
[
 ['fileformat', ['FileFormat', ['./namespace_go.html#a1b060f2a0c10fd0bc766c070bf3f86d4', 1, 'Go']]],
 ['ftcurvetype', ['ftCurveType', ['./namespace_go.html#ad80c2d41480925fa1892bb443097b4ae', 1, 'Go']]],
 ['ftmessages', ['ftmessages', ['./namespace_go.html#aa919fb2218b20ac19a8f859200d69ca2', 1, 'Go']]],
 ['fttangpriority', ['ftTangPriority', ['./namespace_go.html#ab182c80bd5555f4f7adee434150ede71', 1, 'Go']]]
]
```

Definition at line 1 of file enums\_5.js.

### 30.1450 doc/html/search/enums\_6.js File Reference

#### Variables

- var [searchData](#)



## 30.1450.1 Variable Documentation

### 30.1450.1.1 var searchData

#### Initial value:

```
=
[
 ['igessection', ['IGESSection', ['../namespace_go.html#a95d383314d9ef7d277424ee672473063', 1, 'Go']]],
 ['intersectiontype', ['IntersectionType', ['../namespace_go.html#a8dbaf65eacdc80d8d1714419e976ed2b', 1, 'Go']
]],
 ['intptclassification', ['IntPtClassification', ['../namespace_go.html#a96d472ee6c1174d35dc179b133c76bde', 1
 , 'Go']]],
 ['intptlocation', ['IntPtLocation', ['../namespace_go.html#a7d692aadd79b236fb05b5b4135cd0ad', 1, 'Go']]],
 ['iteratortype', ['IteratorType', ['../namespace_go.html#a241482b98287c4f940c75ef70671e05c', 1, 'Go']]]
]
```

Definition at line 1 of file enums\_6.js.

## 30.1451 doc/html/search/enums\_7.js File Reference

### Variables

- var [searchData](#)

## 30.1451.1 Variable Documentation

### 30.1451.1.1 var searchData

#### Initial value:

```
=
[
 ['linktype', ['LinkType', ['../namespace_go.html#aeb8f1bba4a2e383337bd140d61a65556', 1, 'Go']]],
 ['lsf'
]
```

Definition at line 1 of file enums\_7.js.

## 30.1452 doc/html/search/enums\_8.js File Reference

### Variables

- var [searchData](#)

### 30.1452.1 Variable Documentation

#### 30.1452.1.1 var searchData

##### Initial value:

```
=
[
 ['magfilterset', ['MagFilterSet', ['../gv_texture_8h.html#abbb19582793c79e4ec8fe451f9e236e6', 1, 'gvTexture.h
 ']]],
 ['minfilterset', ['MinFilterSet', ['../gv_texture_8h.html#a22888cb2b2c0262ed85c7ac8aa7c1c2f', 1, 'gvTexture.h
 ']]]
]
```

Definition at line 1 of file enums\_8.js.

## 30.1453 doc/html/search/enums\_9.js File Reference

### Variables

- var [searchData](#)

### 30.1453.1 Variable Documentation

#### 30.1453.1.1 var searchData

##### Initial value:

```
=
[
 ['nodestatus', ['NodeStatus', ['
 ../generic__graph__algorithms__implementation_8h.html#a8c61272fccb797a8d283bb4ac2d86cdd', 1, 'generic_graph_algorithms_imp
 'normalconemethod', ['NormalConeMethod', ['
 ../class_go_1_1_spline_surface.html#a60563fc410b940f853913f586cda19fc', 1, 'Go::SplineSurface']]
]
```

Definition at line 1 of file enums\_9.js.

## 30.1454 doc/html/search/enums\_a.js File Reference

### Variables

- var [searchData](#)

### 30.1454.1 Variable Documentation

#### 30.1454.1.1 var searchData

**Initial value:**

```
=
[
 ['offsetsurfacestatus', ['OffsetSurfaceStatus', [
 ../_offset_surface_utils_8h.html#a49c34a8a952371206ac5113543270589', 1, 'OffsetSurfaceUtils.h']]]
]
```

Definition at line 1 of file enums\_a.js.

## 30.1455 doc/html/search/enums\_b.js File Reference

### Variables

- var [searchData](#)

### 30.1455.1 Variable Documentation

#### 30.1455.1.1 var searchData

**Initial value:**

```
=
[
 ['pointsequencetype', ['PointSequenceType', [
 ..namespace_go.html#a8fd32afd3a4d4892e4020b0580de70e1', 1, 'Go
 ']]],
 ['prbdyparamkind', ['PrBdyParamKind', [
 .._pr_parametrize_bdy_8h.html#ac4f0ea96d827517d164d141b76822905', 1
 , 'PrParametrizeBdy.h']]],
 ['prparamstartvector', ['PrParamStartVector', [
 .._pr_parametrize_int_8h.html#ab93ec75afebf4127286a826175e2184e', 1, 'PrParametrizeInt.h']]]
]
```

Definition at line 1 of file enums\_b.js.

## 30.1456 doc/html/search/enums\_c.js File Reference

### Variables

- var [searchData](#)

### 30.1456.1 Variable Documentation

#### 30.1456.1.1 var searchData

**Initial value:**

```
=
[
 ['registrationreturntype', ['RegistrationReturnType', [
 './namespace_go.html#a4d9923f0997091e744526f5654084584', 1, 'Go']]]
]
```

Definition at line 1 of file enums\_c.js.

### 30.1457 doc/html/search/enums\_d.js File Reference

#### Variables

- var [searchData](#)

### 30.1457.1 Variable Documentation

#### 30.1457.1.1 var searchData

**Initial value:**

```
=
[
 ['singularityclassification', ['SingularityClassification', [
 './namespace_go.html#a2d40b57e4fc6eed03a9fbeebe181e7c5', 1, 'Go']]],
 ['singularitytype', ['SingularityType', ['./namespace_go.html#a3f21d9ac1d33fe9bf8364573e0126af2', 1, 'Go']]]
 ,
 ['subdivisionclassification', ['SubdivisionClassification', [
 './namespace_go.html#a20f9976341cc350b6e0d38e92eca1583', 1, 'Go']]]
]
```

Definition at line 1 of file enums\_d.js.

### 30.1458 doc/html/search/enums\_e.js File Reference

#### Variables

- var [searchData](#)

### 30.1458.1 Variable Documentation

#### 30.1458.1.1 var searchData

**Initial value:**

```
=
[
 ['tangentdomain', ['TangentDomain', ['../namespace_go.html#a683c7cd0653c849d16af423f40a6daf6', 1, 'Go']],
 ['testsuite', ['testSuite', ['../namespace_go.html#a2353a3d581dde8e8be2ad76914f0f584', 1, 'Go']],
 ['tpjointtype', ['tpJointType', ['../namespace_go.html#a1ff99d2b83bb317e5db42679477c31c2', 1, 'Go']]
]
```

Definition at line 1 of file enums\_e.js.

## 30.1459 doc/html/search/enums\_f.js File Reference

### Variables

- var [searchData](#)

### 30.1459.1 Variable Documentation

#### 30.1459.1.1 var searchData

**Initial value:**

```
=
[
 ['wrapmodeset', ['WrapModeSet', ['../gv_texture_8h.html#adee2e25c021585161c4cab1b6751d90d', 1, 'gvTexture.h']
]]
]
```

Definition at line 1 of file enums\_f.js.

## 30.1460 doc/html/search/enumvalues\_0.js File Reference

### Variables

- var [searchData](#)

### 30.1460.1 Variable Documentation

#### 30.1460.1.1 var searchData

**Initial value:**

```
=
[
 ['areatoosmall', ['AreaTooSmall', ['
 ../namespace_go.html#a4d9923f0997091e744526f5654084584a8c50f932f2a5486c74f9faf803baf700', 1, 'Go']]
]
```

Definition at line 1 of file enumvalues\_0.js.

## 30.1461 doc/html/search/enumvalues\_1.js File Reference

### Variables

- var [searchData](#)

### 30.1461.1 Variable Documentation

#### 30.1461.1.1 var searchData

##### Initial value:

```
=
[
 ['backwards', ['BACKWARDS', ['
 ../namespace_go.html#a73ae93efa43d949b788530375bae4a09ab28d938a8509d39f43be6186d739198c', 1, 'Go']]],
 ['band'
```

Definition at line 1 of file enumvalues\_1.js.

## 30.1462 doc/html/search/enumvalues\_10.js File Reference

### Variables

- var [searchData](#)

### 30.1462.1 Variable Documentation

#### 30.1462.1.1 var searchData

##### Initial value:

```
=
[
 ['registrationok', ['RegistrationOK', ['
 ../namespace_go.html#a4d9923f0997091e744526f5654084584ad83c7e38a4aeb95705671d8fe4e7a0d6', 1, 'Go']]],
 ['repaired_5fmissing_5flink', ['REPAIRED_MISSING_LINK', ['
 ../namespace_go.html#aeb8f1bba4a2e383337bd140d61a65556a8ccee62554d8a8b6f78d16318027737f', 1, 'Go']]],
 ['rt', ['Rt', ['
 ../class_n_e_w_m_a_t_1_1_matrix_type.html#a6a88aac41b54a529b61a9c9433f4b583a4507ac17b08c3786d5394d8530b34492', 1, 'NEWMAT
 ['rv', ['RV', ['
 ../class_n_e_w_m_a_t_1_1_matrix_type.html#a6a88aac41b54a529b61a9c9433f4b583a8d78673b896275d6485cbfabd1645749', 1, 'NEWMAT
]
```

Definition at line 1 of file enumvalues\_10.js.

## 30.1463 doc/html/search/enumvalues\_11.js File Reference

### Variables

- var [searchData](#)

### 30.1463.1 Variable Documentation

#### 30.1463.1.1 var searchData

Definition at line 1 of file enumvalues\_11.js.

## 30.1464 doc/html/search/enumvalues\_12.js File Reference

### Variables

- var [searchData](#)

### 30.1464.1 Variable Documentation

#### 30.1464.1.1 var searchData

#### Initial value:

```
=
[
 ['t', ['T', ['../namespace_go.html#a95d383314d9ef7d277424ee672473063ad80deac00a5f9e9bea01b087b3bc6474', 1, 'Go']]],
 ['tangential_5fpoint', ['TANGENTIAL_POINT', ['../namespace_go.html#a3f21d9ac1d33fe9bf8364573e0126af2a7a751867640d143dba6d355e3ad044ce', 1, 'Go']]],
 ['tolerance_5ferror', ['TOLERANCE_ERROR', ['../_offset_surface_utils_8h.html#a49c34a8a952371206ac5113543270589a89c7beb3296ffda86f726e61362f7c9c', 1, 'OffsetSurfaceUt']]],
 ['toofewpoints', ['TooFewPoints', ['../namespace_go.html#a4d9923f0997091e744526f5654084584a409700864df743fb5ed29a95821ab1ad', 1, 'Go']]]
]
```

Definition at line 1 of file enumvalues\_12.js.

## 30.1465 doc/html/search/enumvalues\_13.js File Reference

### Variables

- var [searchData](#)

### 30.1465.1 Variable Documentation

#### 30.1465.1.1 var searchData

#### Initial value:

```
=
[
 ['ub', ['UB', ['./class_n_e_w_m_a_t_1_1_matrix_type.html#a6a88aac41b54a529b61a9c9433f4b583aadb43fe9c465d39ca709c3abbbe41f2', 1, 'NEWMAT']]],
 ['unknown', ['UNKNOWN', ['../namespace_go.html#a2ef4e46477cac9d1ab25de9fbe20fed7a42f3d93048535945ea47f21d2c21dcc9', 1, 'Go']]],
 ['upper', ['Upper', ['./class_n_e_w_m_a_t_1_1_matrix_type.html#afladc0086f5d4c3328ad4a698fa57c90ab75af65529b3ea6d5d92d1b750da8e43', 1, 'NEWMAT']]],
 ['us', ['US', ['./class_n_e_w_m_a_t_1_1_matrix_type.html#a6a88aac41b54a529b61a9c9433f4b583ad0a88c0bea4b361a84511383a9b2adf1', 1, 'NEWMAT']]],
 ['ut', ['UT', ['./class_n_e_w_m_a_t_1_1_matrix_type.html#a6a88aac41b54a529b61a9c9433f4b583ac0726b808a3ace193cd89bc93fe18ea6', 1, 'NEWMAT']]]
]
```

Definition at line 1 of file enumvalues\_13.js.

## 30.1466 doc/html/search/enumvalues\_14.js File Reference

### Variables

- var [searchData](#)
- & gt

### 30.1466.1 Variable Documentation

#### 30.1466.1.1 & gt

Definition at line 4 of file enumvalues\_14.js.

#### 30.1466.1.2 var searchData

#### Initial value:

```
=
[
 ['valid', ['Valid', ['
 ../class_n_e_w_m_a_t_l_l_matrix_type.html#afladc0086f5d4c3328ad4a698fa57c90a0dad346d65e5540202aecf9f55dc0d', 1, 'NEWMAT
 'value', ['value', ['
 ../struct_go_l_l_factorial.html#a579daed3b4275abe013c8f88077f888fa4acbf42abf2d3a50f62969f536eacb98', 1, 'Go::Factorial::v
 ../struct_go_l_l_factorial_3_011_01_4.html#a654cb8f1088a190314946d0cb460482caf04561aa2fcbf1fb66fbc964a355d268', 1, 'Go::F
```

Definition at line 1 of file enumvalues\_14.js.

## 30.1467 doc/html/search/enumvalues\_15.js File Reference

### Variables

- var [searchData](#)

### 30.1467.1 Variable Documentation

#### 30.1467.1.1 var searchData

#### Initial value:

```
=
[
 ['wrapclamp', ['wrapClamp', ['
 ../gv_texture_8h.html#adee2e25c021585161c4cab1b6751d90da81991e50cc3280139282b93608fa871d', 1, 'gvTexture.h']]],
 ['wraprepeat', ['wrapRepeat', ['
 ../gv_texture_8h.html#adee2e25c021585161c4cab1b6751d90da0004245ea176bc058284f0652a8416e9', 1, 'gvTexture.h']]]
]
```

Definition at line 1 of file enumvalues\_15.js.



## 30.1468 doc/html/search/enumvalues\_16.js File Reference

### Variables

- var [searchData](#)

#### 30.1468.1 Variable Documentation

##### 30.1468.1.1 var searchData

###### Initial value:

```
=
[
 ['xfixed', ['XFIXED', [
 ../namespace_go.html#a821a750d4a6740898072cf759246eb87a1e4145becb15ce54486dd57c554a8716', 1, 'Go']]]
]
```

Definition at line 1 of file enumvalues\_16.js.

## 30.1469 doc/html/search/enumvalues\_17.js File Reference

### Variables

- var [searchData](#)

#### 30.1469.1 Variable Documentation

##### 30.1469.1.1 var searchData

###### Initial value:

```
=
[
 ['yfixed', ['YFIXED', [
 ../namespace_go.html#a821a750d4a6740898072cf759246eb87a344e219d46e24ae18ec2ff7fe11afddd', 1, 'Go']]]
]
```

Definition at line 1 of file enumvalues\_17.js.

## 30.1470 doc/html/search/enumvalues\_18.js File Reference

### Variables

- var [searchData](#)

### 30.1470.1 Variable Documentation

#### 30.1470.1.1 var searchData

##### Initial value:

```
=
[
 ['zero_5fdirichlet', ['ZERO_DIRICHLET', ['
 ../namespace_go.html#a2ef4e46477cac9d1ab25de9fbe20fed7aec9786d0435848f55a9401da630edbaef', 1, 'Go']]],
 ['zero_5fneumann', ['ZERO_NEUMANN', ['
 ../namespace_go.html#a2ef4e46477cac9d1ab25de9fbe20fed7a5a1060dcd56fe6e7058351bcf342e1c5', 1, 'Go']]]
]
```

Definition at line 1 of file enumvalues\_18.js.

## 30.1471 doc/html/search/enumvalues\_2.js File Reference

### Variables

- var [searchData](#)

### 30.1471.1 Variable Documentation

#### 30.1471.1.1 var searchData

Definition at line 1 of file enumvalues\_2.js.

## 30.1472 doc/html/search/enumvalues\_3.js File Reference

### Variables

- var [searchData](#)

## 30.1472.1 Variable Documentation

### 30.1472.1.1 var searchData

#### Initial value:

```
=
[
 ['d', ['D', ['../namespace_go.html#a95d383314d9ef7d277424ee672473063ada8cdab2225fa4f764ba2ff6d553f3d4', 1, 'Go']]],
 ['deg_5ftriangle', ['DEG_TRIANGLE', ['../namespace_go.html#aeb8f1bba4a2e383337bd140d61a65556a253a904e3add062958c2dfa385eelec2', 1, 'Go']]],
 ['degen_5fsrf_5fbd', ['DEGEN_SRF_BD', ['../namespace_go.html#a2353a3d581dde8e8be2ad76914f0f584a2af23e45ea95a9c948c0511760750af0', 1, 'Go']]],
 ['degen_5fsrf_5fcorner', ['DEGEN_SRF_CORNER', ['../namespace_go.html#a2353a3d581dde8e8be2ad76914f0f584a502f473df1998dd72f5102bf12275193', 1, 'Go']]],
 ['dg', ['Dg', ['../class_n_e_w_m_a_t_l_1_matrix_type.html#a6a88aac41b54a529b61a9c9433f4b583a12503ce36500638199a2806d1f2473d1', 1, 'NEWMAT diagonal', ['Diagonal', ['../class_n_e_w_m_a_t_l_1_matrix_type.html#af1adc0086f5d4c3328ad4a698fa57c90a88d5e722097ab68f94a3c4795a8b3970', 1, 'NEWMAT dir_5fhighly_5fsingular', ['DIR_HIGHLY_SINGULAR', ['../namespace_go.html#a96d472ee6c1174d35dc179b133c76bdea9aeed62bfaaeac96e0f966c90ef04f2c', 1, 'Go']]],
 ['dir_5fin', ['DIR_IN', ['../namespace_go.html#a96d472ee6c1174d35dc179b133c76bdea2b229b86fc7c49a888f4c85de0fd0023', 1, 'Go']]],
 ['dir_5fout', ['DIR_OUT', ['../namespace_go.html#a96d472ee6c1174d35dc179b133c76bdea867efb07d41953d6d8e3b6ee17a45772', 1, 'Go']]],
 ['dir_5fparallel', ['DIR_PARALLEL', ['../namespace_go.html#a96d472ee6c1174d35dc179b133c76bdea28b39001ab0ad9baeb5e05efd209f940', 1, 'Go']]],
 ['dir_5fperpendicular', ['DIR_PERPENDICULAR', ['../namespace_go.html#a96d472ee6c1174d35dc179b133c76bdea0f096a32edf421a2627ec6d6e20382e6', 1, 'Go']]],
 ['dir_5ftouch', ['DIR_TOUCH', ['../namespace_go.html#a96d472ee6c1174d35dc179b133c76bdea734d14ac74f992b6632803bd13715769', 1, 'Go']]],
 ['dir_5fundef', ['DIR_UNDEF', ['../namespace_go.html#a96d472ee6c1174d35dc179b133c76bdea820ac70fd33c863aebcf7a3b0bc691c9', 1, 'Go']]],
 ['directpart', ['DirectPart', ['../newmatrc_8h.html#ae75c691eb2643cc7fe952b12f074040cac1dd577ab18fd18fcb43d9654721c8f', 1, 'newmatrc.h']]],
 ['dirichlet', ['DIRICHLET', ['../namespace_go.html#a2ef4e46477cac9d1ab25de9f20fed7a5c504f4a19f84fedb7d51aa7db55592e', 1, 'Go']]],
 ['disp', ['disp', ['../namespace_go.html#alb060f2a0c10fd0bc766c070bf3f86d4afdb0b616372654f2b42dbf875f740077', 1, 'Go']]],
 ['divide_5fcritical', ['DIVIDE_CRITICAL', ['../namespace_go.html#a20f9976341cc350b6e0d38e92eca1583a5a94412a23dc475840ac49836f9acacc', 1, 'Go']]],
 ['divide_5fdeg', ['DIVIDE_DEG', ['../namespace_go.html#a20f9976341cc350b6e0d38e92eca1583a7024a78dfc8b657dfcd157ef0e0b1886', 1, 'Go']]],
 ['divide_5fhigh_5fsing', ['DIVIDE_HIGH_SING', ['../namespace_go.html#a20f9976341cc350b6e0d38e92eca1583a8c4ad2ea3ed668d02477f0b10b55d6ad', 1, 'Go']]],
 ['divide_5fint', ['DIVIDE_INT', ['../namespace_go.html#a20f9976341cc350b6e0d38e92eca1583afc505e9630be475c3842fb7b43f30d94', 1, 'Go']]],
 ['divide_5fknot', ['DIVIDE_KNOT', ['../namespace_go.html#a20f9976341cc350b6e0d38e92eca1583a261b563ba14323519361642ffd4fb8e6', 1, 'Go']]],
 ['divide_5fpar', ['DIVIDE_PAR', ['../namespace_go.html#a20f9976341cc350b6e0d38e92eca1583a4ad3fd9c982e0d60e139cf3d8fc16ea8', 1, 'Go']]],
 ['divide_5fsing', ['DIVIDE_SING', ['../namespace_go.html#a20f9976341cc350b6e0d38e92eca1583aff8eb5e454216f614a6f2debdd17b98', 1, 'Go']]],
 ['divided_5fsing', ['DIVIDED_SING', ['../namespace_go.html#a2d40b57e4fc6eed03a9fbbee181e7c5a98b637c26d37d470fcfb5260774f8f9', 1, 'Go']]]
]
```

Definition at line 1 of file enumvalues\_3.js.

## 30.1473 doc/html/search/enumvalues\_4.js File Reference

### Variables

- var [searchData](#)

### 30.1473.1 Variable Documentation

#### 30.1473.1.1 var searchData

##### Initial value:

```
=
[
 ['e', ['E', ['../namespace_go.html#a95d383314d9ef7d277424ee672473063a67661f5f4694b7e01c81e2e32d5841d0', 1, 'Go']]],
 ['edge_5facute_5fangle', ['EDGE_ACUTE_ANGLE', ['../namespace_go.html#a2353a3d581dde8e8be2ad76914f0f584ae81304fe056a752a83af4310b525b2e8', 1, 'Go']]],
 ['edge_5fposition_5fdiscont', ['EDGE_POSITION_DISCONT', ['../namespace_go.html#a2353a3d581dde8e8be2ad76914f0f584a8a1a2b3ebb821b186678cda5a0459c6', 1, 'Go']]],
 ['edge_5ftangential_5fdiscont', ['EDGE_TANGENTIAL_DISCONT', ['../namespace_go.html#a2353a3d581dde8e8be2ad76914f0f584a9305fbfa4465cd36461b3e3b18e9c9b6', 1, 'Go']]],
 ['edge_5fvertex_5fdistance', ['EDGE_VERTEX_DISTANCE', ['../namespace_go.html#a2353a3d581dde8e8be2ad76914f0f584a836f33be4d6e29ca350fd15ace55b10', 1, 'Go']]],
 ['embedded_5fedges', ['EMBEDDED_EDGES', ['../namespace_go.html#a2353a3d581dde8e8be2ad76914f0f584afceb4f2446a229605cf589c1500dd424', 1, 'Go']]],
 ['embedded_5ffaces', ['EMBEDDED_FACES', ['../namespace_go.html#a2353a3d581dde8e8be2ad76914f0f584a375ed83c0a0939ff318aec5983653710', 1, 'Go']]],
 ['envblend', ['envBlend', ['../gv_texture_8h.html#abfb34fc206cb7e7d2ede80704c1fcc86a66850f3a790c942c59dc2567c41cfebf', 1, 'gvTexture.h']]],
 ['envdecal', ['envDecal', ['../gv_texture_8h.html#abfb34fc206cb7e7d2ede80704c1fcc86ae9b27fa588bfa1b339103a8f9cd7d92d', 1, 'gvTexture.h']]],
 ['envmodulate', ['envModulate', ['../gv_texture_8h.html#abfb34fc206cb7e7d2ede80704c1fcc86a76402e7f3a9c65225ad37d3cd49ce1f3', 1, 'gvTexture.h']]],
 ['envreplace', ['envReplace', ['../gv_texture_8h.html#abfb34fc206cb7e7d2ede80704c1fcc86a74718d64656d27245ca720852575e124', 1, 'gvTexture.h']]]
]
```

Definition at line 1 of file enumvalues\_4.js.

### 30.1474 doc/html/search/enumvalues\_5.js File Reference

#### Variables

- var [searchData](#)

#### 30.1474.1 Variable Documentation

##### 30.1474.1.1 var searchData

Definition at line 1 of file enumvalues\_5.js.

### 30.1475 doc/html/search/enumvalues\_6.js File Reference

#### Variables

- var [searchData](#)

### 30.1475.1 Variable Documentation

#### 30.1475.1.1 var searchData

##### Initial value:

```
=
[
 ['g', ['G', ['./namespace_go.html#a95d383314d9ef7d277424ee672473063aa32eca3219a522f885088233d0cf8aa3', 1, 'Go']],
 ['geom', ['GEOM', ['./namespace_go.html#a683c7cd0653c849d16af423f40a6daf6a2eab3dfbfcbb69fb143144ec699e952e', 1, 'Go']],
 ['geometrical', ['GEOMETRICAL', ['./namespace_go.html#a9ec4373c3aa2905cb547164492f510d1a4877e954b6cbc33a24efca42fad0cb20', 1, 'Go']],
 ['global_5fsearch', ['GLOBAL_SEARCH', ['./namespace_go.html#a64e7b23868d471d8f59b8e280437195ea31db71b7b0faac9b18302d7d0b14a861', 1, 'Go']],
 ['go', ['go', ['./namespace_go.html#a1b060f2a0c10fd0bc766c070bf3f86d4a727b71e7d5e25b99928ed913a31437f0', 1, 'Go']]
]
```

Definition at line 1 of file enumvalues\_6.js.

## 30.1476 doc/html/search/enumvalues\_7.js File Reference

### Variables

- var [searchData](#)

### 30.1476.1 Variable Documentation

#### 30.1476.1.1 var searchData

##### Initial value:

```
=
[
 ['havestore', ['HaveStore', ['./newmatrc_8h.html#ae75c691eb2643cc7fe952b12f074040ca53133c14cb06987614a1d83f0c44700c', 1, 'newmatrc.h']],
 ['hermite', ['Hermite', ['./class_go_1_1_spline_interpolator.html#aacf4f80d98ea6b554729125fafd24a78a51b46c3d3f007d4d2ef458f51d85610c', 1, 'Go::Spline']],
 ['higher_5forder_5fpoint', ['HIGHER_ORDER_POINT', ['./namespace_go.html#a3f21d9ac1d33fe9bf8364573e0126af2a22f487388b26506dd36247ceb5747b41', 1, 'Go']]
]
```

Definition at line 1 of file enumvalues\_7.js.

## 30.1477 doc/html/search/enumvalues\_8.js File Reference

### Variables

- var [searchData](#)

### 30.1477.1 Variable Documentation

#### 30.1477.1.1 var searchData

##### Initial value:

```
=
[
 ['id', ['Id', ['
 ../class_n_e_w_m_a_t_1_1_matrix_type.html#a6a88aac41b54a529b61a9c9433f4b583a325261cf54ea1a0ca7228fdf8cd312db', 1, 'NEWMAT
 ' identical_5fedges', [' IDENTICAL_EDGES', ['
 ../namespace_go.html#a2353a3d581dde8e8be2ad76914f0f584a53806633e118165389ef85138f0e02b', 1, 'Go']]],
 [' identical_5ffaces', [' IDENTICAL_FACES', ['
 ../namespace_go.html#a2353a3d581dde8e8be2ad76914f0f584ad60ccf2fb14960d4649a164574749f8a', 1, 'Go']]],
 [' identical_5fvertices', [' IDENTICAL_VERTICES', ['
 ../namespace_go.html#a2353a3d581dde8e8be2ad76914f0f584a614b7fb3079450f43057f4e86b3e28ac', 1, 'Go']]],
 [' iges', [' IGES', [' ../namespace_go.html#a1b060f2a0c10fd0bc766c070bf3f86d4ae0747afbdd38bb230962db3f51df27d5
 ', 1, 'Go']]],
 [' indistinct_5fknots', [' INDISTINCT_KNOTS', ['
 ../namespace_go.html#a2353a3d581dde8e8be2ad76914f0f584a03ad515756b74f616d75a5120a42510d', 1, 'Go']]],
 [' init_5fsing', [' INIT_SING', ['
 ../namespace_go.html#a2d40b57e4fc6eed03a9fbbeebe181e7c5a490091ee185383778d6cd7293288d611', 1, 'Go']]],
 [' inside_5foutside_5fsingularity_5fbox', [' INSIDE_OUTSIDE_SINGULARITY_BOX', ['
 ../namespace_go.html#aeb8f1bba4a2e383337bd140d61a65556a9c821c88e8dc7ed1999ecd4746351587', 1, 'Go']]],
 [' isolated_5fpoint', [' ISOLATED_POINT', ['
 ../namespace_go.html#a3f21d9ac1d33fe9bf8364573e0126af2a1b8fe22dbe7702bdc99dd252ffc2ebc4', 1, 'Go']]],
 [' iterator_5fgeometric', [' Iterator_geometric', ['
 ../namespace_go.html#a241482b98287c4f940c75ef70671e05cacd4a1ab60b6688a79afb6336b99c936e', 1, 'Go']]],
 [' iterator_5fparametric', [' Iterator_parametric', ['
 ../namespace_go.html#a241482b98287c4f940c75ef70671e05cac62977f430391a4f0a8643ae07a5da4', 1, 'Go']]]
]
```

Definition at line 1 of file enumvalues\_8.js.

## 30.1478 doc/html/search/enumvalues\_9.js File Reference

### Variables

- var [searchData](#)

### 30.1478.1 Variable Documentation

#### 30.1478.1.1 var searchData

##### Initial value:

```
=
[
 [' joint_5fdisc', [' JOINT_DISC', ['
 ../namespace_go.html#a1ff99d2b83bb317e5db42679477c31c2a70a0ceb99fd902cf435fac73306b6e3', 1, 'Go']]],
 [' joint_5fg0', [' JOINT_G0', ['
 ../namespace_go.html#a1ff99d2b83bb317e5db42679477c31c2a7f5a6b29283bce2a3de4770df35da7d9', 1, 'Go']]],
 [' joint_5fg1', [' JOINT_G1', ['
 ../namespace_go.html#a1ff99d2b83bb317e5db42679477c31c2ada229f3349f036f74128f22b91778c5b', 1, 'Go']]],
 [' joint_5fgap', [' JOINT_GAP', ['
 ../namespace_go.html#a1ff99d2b83bb317e5db42679477c31c2a7a64be06e4c446d0ae5d82ef5dce4949', 1, 'Go']]],
 [' joint_5fkink', [' JOINT_KINK', ['
 ../namespace_go.html#a1ff99d2b83bb317e5db42679477c31c2ab918bde030f6bc020b44ec2422b39e25', 1, 'Go']]],
 [' joint_5fnone', [' JOINT_NONE', ['
 ../namespace_go.html#a1ff99d2b83bb317e5db42679477c31c2a45992da186ea931a1bf00b446280b433', 1, 'Go']]]
]
```

Definition at line 1 of file enumvalues\_9.js.

## 30.1479 doc/html/search/enumvalues\_a.js File Reference

### Variables

- var [searchData](#)

### 30.1479.1 Variable Documentation

#### 30.1479.1.1 var searchData

##### Initial value:

```
=
[
 ['keep_5fsing', ['KEEP_SING', ['
 ../namespace_go.html#a2d40b57e4fc6eed03a9fbbeebe181e7c5a5950a0471c4dedf23ebab3d521693e9e', 1, 'Go']]]
]
```

Definition at line 1 of file enumvalues\_a.js.

## 30.1480 doc/html/search/enumvalues\_b.js File Reference

### Variables

- var [searchData](#)

### 30.1480.1 Variable Documentation

#### 30.1480.1.1 var searchData

##### Initial value:

```
=
[
 ['lb', ['LB', ['
 ../class_n_e_w_m_a_t_l_l_matrix_type.html#a6a88aac41b54a529b61a9c9433f4b583a5738046339887f72d6ecf0855b8ace66', 1, 'NEWMAT
 ['linear_5fcvcv', ['LINEAR_CVCV', ['
 ../namespace_go.html#aeb8f1bba4a2e383337bd140d61a65556a7f4243e6c2c1e5c7f51df461855d187c', 1, 'Go']]],
 ['linear_5sfcv', ['LINEAR_SFVC', ['
 ../namespace_go.html#aeb8f1bba4a2e383337bd140d61a65556a5bde679ed6e9ce8890955ca90c1b1feb', 1, 'Go']]],
 ['linear_5sfsf', ['LINEAR_SFSE', ['
 ../namespace_go.html#aeb8f1bba4a2e383337bd140d61a65556a884aa3c499cdf465b88d556a3997378a', 1, 'Go']]],
 ['link_5fundefined', ['LINK_UNDEFINED', ['
 ../namespace_go.html#aeb8f1bba4a2e383337bd140d61a65556a1f11495e24d6b64700ce8e345bf87494', 1, 'Go']]],
 ['loadonentry', ['LoadOnEntry', ['
 ../newmatrc_8h.html#ae75c691eb2643cc7fe952b12f074040caf82aaa0389ad880499aa17ee995acd62', 1, 'newmatrc.h']]],
 ['loc_5fcorner_5fboth', ['LOC_CORNER_BOTH', ['
 ../namespace_go.html#a7d692aadd79b236fb05b5b4135cd0adad56be127936a25a10c7b50cfafaf85cc', 1, 'Go']]],
 ['loc_5fcorner_5fone', ['LOC_CORNER_ONE', ['
 ../namespace_go.html#a7d692aadd79b236fb05b5b4135cd0ada65dc06de763e9db4a605a22c74ab0303', 1, 'Go']]],
 ['loc_5fedge_5fboth', ['LOC_EDGE_BOTH', ['
 ../namespace_go.html#a7d692aadd79b236fb05b5b4135cd0ada3db4d0a9d71b8d50fd87fdc59664a13c', 1, 'Go']]],
 ['loc_5fedge_5fone', ['LOC_EDGE_ONE', ['
 ../namespace_go.html#a7d692aadd79b236fb05b5b4135cd0ada0cc998fe7d568329a67a5af22270fae8', 1, 'Go']]],
 ['loc_5finside_5fboth', ['LOC_INSIDE_BOTH', ['
 ../namespace_go.html#a7d692aadd79b236fb05b5b4135cd0ada7d4f28281cf301a763ffcfdf526f3ba2', 1, 'Go']]],
 ['loc_5foutside_5fboth', ['LOC_OUTSIDE_BOTH', ['
 ../namespace_go.html#a7d692aadd79b236fb05b5b4135cd0ada91b8e290e0e2555d6b1e96dfae8e4bae', 1, 'Go']]],
]
```

```
['local_5fsearch', ['LOCAL_SEARCH', ['
 ../namespace_go.html#a64e7b23868d471d8f59b8e280437195ea67da27ebc2731782484f21d1d9acb8fd', 1, 'Go']]],
['loop_5fconsistency', ['LOOP_CONSISTENCY', ['
 ../namespace_go.html#a2353a3d581dde8e8be2ad76914f0f584a5cc09671c391910934656977a9658ec0', 1, 'Go']]],
['loop_5fintersection', ['LOOP_INTERSECTION', ['
 ../namespace_go.html#a2353a3d581dde8e8be2ad76914f0f584a69345c1de52ee52df57afe2037aade9c', 1, 'Go']]],
['loop_5forientation', ['LOOP_ORIENTATION', ['
 ../namespace_go.html#a2353a3d581dde8e8be2ad76914f0f584af65e037e9f2b5131d88e15a3ed099081', 1, 'Go']]],
['loop_5fself_5fintersection', ['LOOP_SELF_INTERSECTION', ['
 ../namespace_go.html#a2353a3d581dde8e8be2ad76914f0f584ae5291135054d75cd8a290a808f2b3b67', 1, 'Go']]],
['lower', ['Lower', ['
 ../class_n_e_w_m_a_t_1_1_matrix_type.html#af1adc0086f5d4c3328ad4a698fa57c90a7179b6f363e7d4e90bd188392de924ba', 1, 'NEWMAT
['lt', ['LT', ['
 ../class_n_e_w_m_a_t_1_1_matrix_type.html#a6a88aac41b54a529b61a9c9433f4b583a0edcd4d5f3d58272ad0e0a36f5da704a', 1, 'NEWMAT
['ludeco', ['LUDECO', ['
 ../class_n_e_w_m_a_t_1_1_matrix_type.html#af1adc0086f5d4c3328ad4a698fa57c90ae624050aed576ddc4119f40c426cc61b', 1, 'NEWMAT
]
```

Definition at line 1 of file enumvalues\_b.js.

## 30.1481 doc/html/search/enumvalues\_c.js File Reference

### Variables

- var [searchData](#)

### 30.1481.1 Variable Documentation

#### 30.1481.1.1 var searchData

#### Initial value:

```
=
[
['maglinear', ['magLinear', ['
 ../gv_texture_8h.html#abbb19582793c79e4ec8fe451f9e236e6abe6738841e17db775ec39e0479af16b8', 1, 'gvTexture.h']]],
['magnearest', ['magNearest', ['
 ../gv_texture_8h.html#abbb19582793c79e4ec8fe451f9e236e6a656ed1cddd73643821e62a1fdedc48b8', 1, 'gvTexture.h']]],
['merged_5fcoincidence_5fsfcv_5fsfcv', ['MERGED_COINCIDENCE_SFCV_SFCV', ['
 ../namespace_go.html#aeb8f1bba4a2e383337bd140d61a65556a845c0591e6522cbb1bd7d42a3d644c4e', 1, 'Go']]],
['merged_5fsimple_5fcone', ['MERGED_SIMPLE_CONE', ['
 ../namespace_go.html#aeb8f1bba4a2e383337bd140d61a65556a33c6ebf38c4366b2005b53d95baf936e', 1, 'Go']]],
['merged_5fsimple_5fcone_5fmonotone', ['MERGED_SIMPLE_CONE_MONOTONE', ['
 ../namespace_go.html#aeb8f1bba4a2e383337bd140d61a65556a3112ff1b70eed68acf85bc8f64045cd8', 1, 'Go']]],
['merged_5fsimple_5fmonotone', ['MERGED_SIMPLE_MONOTONE', ['
 ../namespace_go.html#aeb8f1bba4a2e383337bd140d61a65556a880855a6ca44f4c7dcdad21a476df516', 1, 'Go']]],
['merged_5fundefined', ['MERGED_UNDEFINED', ['
 ../namespace_go.html#aeb8f1bba4a2e383337bd140d61a65556a2a28c174c7b328eeb951148142dd2ba6', 1, 'Go']]],
['micro_5fcvcv', ['MICRO_CVCV', ['
 ../namespace_go.html#aeb8f1bba4a2e383337bd140d61a65556a4dfb177c5df6a14182f3bf999f0a4130', 1, 'Go']]],
['micro_5fpar1func', ['MICRO_PAR1FUNC', ['
 ../namespace_go.html#aeb8f1bba4a2e383337bd140d61a65556a02e6b5db18a29d921fac920122d304bc', 1, 'Go']]],
['micro_5fpar2func', ['MICRO_PAR2FUNC', ['
 ../namespace_go.html#aeb8f1bba4a2e383337bd140d61a65556a5963a3db65d693ff97246e5078994779', 1, 'Go']]],
['micro_5fsfcv', ['MICRO_SFCV', ['
 ../namespace_go.html#aeb8f1bba4a2e383337bd140d61a65556ac885460e14ae384b04e728688ba62416', 1, 'Go']]],
['micro_5fsfpt', ['MICRO_SFPT', ['
 ../namespace_go.html#aeb8f1bba4a2e383337bd140d61a65556afdffc0f39697f2e1a759cf5bcea991285', 1, 'Go']]],
['micro_5fsfsf', ['MICRO_SFSF', ['
 ../namespace_go.html#aeb8f1bba4a2e383337bd140d61a65556a26b61b3d6e6b8e2a66202f7c3900e2e2', 1, 'Go']]],
['mini_5fcurve', ['MINI_CURVE', ['
 ../namespace_go.html#a2353a3d581dde8e8be2ad76914f0f584ad7c56a91bf38e4b2b1c294a9301ff68a', 1, 'Go']]],
['mini_5fedge', ['MINI_EDGE', ['
 ../namespace_go.html#a2353a3d581dde8e8be2ad76914f0f584ae2427cc37fe955e7ede968ea3ceea7c0', 1, 'Go']]],
['mini_5fface', ['MINI_FACE', ['
 ../namespace_go.html#a2353a3d581dde8e8be2ad76914f0f584ab640d2c0b4dde412a0a5cc233ed8574e', 1, 'Go']]],
['mini_5fsurface', ['MINI_SURFACE', ['
 ../namespace_go.html#a2353a3d581dde8e8be2ad76914f0f584ac0239da67f77b2a85f3e1467eb2465ca', 1, 'Go']]],
['minlinear', ['minLinear', ['
 ../gv_texture_8h.html#a22888cb2b2c0262ed85c7ac8aa7c1c2fa34564aeb9a54aec0ae6daa30d0ca5965', 1, 'gvTexture.h']]],
```



```
['minlinearmipmaplinear', ['minLinearMipmapLinear', [
 './gv_texture_8h.html#a22888cb2b2c0262ed85c7ac8aa7c1c2fa5bf26bc6347132e8118dc0eda75dd2c9', 1, 'gvTexture.h']]],
['minlinearmipmapnearest', ['minLinearMipmapNearest', [
 './gv_texture_8h.html#a22888cb2b2c0262ed85c7ac8aa7c1c2fa8579243951c8b2cd902ef07b1c94f84c', 1, 'gvTexture.h']]],
['minnearest', ['minNearest', [
 './gv_texture_8h.html#a22888cb2b2c0262ed85c7ac8aa7c1c2fa529132446a8de28d302618413fd64095', 1, 'gvTexture.h']]],
['minnearestmipmaplinear', ['minNearestMipmapLinear', [
 './gv_texture_8h.html#a22888cb2b2c0262ed85c7ac8aa7c1c2fae65a99cf93d997f52b3d3244e7e9e7a7', 1, 'gvTexture.h']]],
['minnearestmipmapnearest', ['minNearestMipmapNearest', [
 './gv_texture_8h.html#a22888cb2b2c0262ed85c7ac8aa7c1c2fa51d5f598d2c42800336631f9ca5d6b1f', 1, 'gvTexture.h']]]
]
```

Definition at line 1 of file enumvalues\_c.js.

## 30.1482 doc/html/search/enumvalues\_d.js File Reference

### Variables

- var [searchData](#)

### 30.1482.1 Variable Documentation

#### 30.1482.1.1 var searchData

#### Initial value:

```
=
[
 ['narrow_5fregion', ['NARROW_REGION', [
 './namespace_go.html#a2353a3d581dde8e8be2ad76914f0f584a7ce649f843503dab14b2319f52eaf41d', 1, 'Go']]],
 ['natural', ['Natural', [
 './class_go_1_1_spline_interpolator.html#aacf4f80d98ea6b554729125fafd24a78a97ae16f658015873c02850158cd655fd', 1, 'Go::SplineInterpolation']]],
 ['naturalatend', ['NaturalAtEnd', [
 './class_go_1_1_spline_interpolator.html#aacf4f80d98ea6b554729125fafd24a78a94a08e9e644774498ad7e091d8535415', 1, 'Go::SplineInterpolation']]],
 ['naturalatstart', ['NaturalAtStart', [
 './class_go_1_1_spline_interpolator.html#aacf4f80d98ea6b554729125fafd24a78a5b6b1f58b2d5ddefab7785b4e637e213', 1, 'Go::SplineInterpolation']]],
 ['neumann', ['NEUMANN', [
 './namespace_go.html#a2ef4e46477cac9d1ab25de9f20fed7aed6f828e7a7040280d06e6d5dc58f405', 1, 'Go']]],
 ['no_5fsing', ['NO_SING', [
 './namespace_go.html#a2d40b57e4fc6eed03a9fbeebe181e7c5ab1559d7dcf7654af2bcec501f5312732', 1, 'Go']]],
 ['no_5ftype', ['No_Type', [
 './namespace_go.html#a8dbaf65eacdc80d8d1714419e976ed2babcd670721a68d8943fefaf6f071c7ab7', 1, 'Go']]],
 ['non_5fiso_5fkink_5fcurve', ['NON_ISO_KINK_CURVE', [
 './_offset_surface_utils_8h.html#a49c34a8a952371206ac5113543270589a8f44525c3493edee9f3a3948eabe7bee', 1, 'OffsetSurfaceUtils']]],
 ['none', ['None', [
 './class_go_1_1_spline_interpolator.html#aacf4f80d98ea6b554729125fafd24a78a02bef9ce6e415b73a3cd9fe89aaace722', 1, 'Go::SplineInterpolation']]],
 ['not_5ffour_5fcorners', ['NOT_FOUR_CORNERS', [
 './_offset_surface_utils_8h.html#a49c34a8a952371206ac5113543270589a1db463f259d94e558afd3af00a006af', 1, 'OffsetSurfaceUtils']]]
]
```

Definition at line 1 of file enumvalues\_d.js.

## 30.1483 doc/html/search/enumvalues\_e.js File Reference

### Variables

- var [searchData](#)

### 30.1483.1 Variable Documentation

#### 30.1483.1.1 var searchData

##### Initial value:

```
=
[
 ['offset_5ffailed', ['OFFSET_FAILED', ['
 ../offset_surface_utils_8h.html#a49c34a8a952371206ac5113543270589adc3d327e65f0c46656bfc9e175fcc151', 1, 'OffsetSurfaceUt
 ['offset_5fok', ['OFFSET_OK', ['
 ../offset_surface_utils_8h.html#a49c34a8a952371206ac5113543270589addc758c86d9a578e0dcb248819d592af', 1, 'OffsetSurfaceUt
 ['ones', ['Ones', ['
 ../class_n_e_w_m_a_t_l_l_matrix_type.html#af1adc0086f5d4c3328ad4a698fa57c90a44547634c366280d11428cb090d33abd', 1, 'NEWMAT
 ['ordinary_5fpoint', ['ORDINARY_POINT', ['
 ../namespace_go.html#a3f21d9ac1d33fe9bf8364573e0126af2a1fdcba6fec487885e75718a98dd830b0', 1, 'Go']]]
]
```

Definition at line 1 of file enumvalues\_e.js.

## 30.1484 doc/html/search/enumvalues\_f.js File Reference

### Variables

- var [searchData](#)

### 30.1484.1 Variable Documentation

#### 30.1484.1.1 var searchData

##### Initial value:

```
=
[
 ['p', ['P', ['../namespace_go.html#a95d383314d9ef7d277424ee672473063ad1ce8b353acf910f8871322784488407', 1, '
 Go']]],
 ['param1', ['PARAM1', ['
 ../namespace_go.html#a683c7cd0653c849d16af423f40a6daf6a44e72e0b76476af77c20052f24d3b5be', 1, 'Go']]],
 ['param2', ['PARAM2', ['
 ../namespace_go.html#a683c7cd0653c849d16af423f40a6daf6a7c4a6ef82dbc05c929c42e8d776d0afe', 1, 'Go']]],
 ['paramcurve_5f1', ['PARAMCURVE_1', ['
 ../namespace_go.html#a11cd92eeab3979030879414ab681ae8fab087de473bf3a94dfe7e5517ff3fca7e', 1, 'Go']]],
 ['paramcurve_5f2', ['PARAMCURVE_2', ['
 ../namespace_go.html#a11cd92eeab3979030879414ab681ae8fa5d0237f746518b9794e553b00c211c3d', 1, 'Go']]],
 ['pointset sizediff', ['PointSetSizeDiff', ['
 ../namespace_go.html#a4d9923f0997091e744526f5654084584abcdec83ae9dd6bb083acac7e3b218dc3', 1, 'Go']]],
 ['post_5fiterate', ['POST_ITERATE', ['
 ../namespace_go.html#aeb8f1bba4a2e383337bd140d61a65556a7332a0add2807af6a1d8545ece9d7672', 1, 'Go']]],
 ['prbarycentre', ['PrBARYCENTRE', ['
 ../pr_parametrize_int_8h.html#ab93ec75afebf4127286a826175e2184eac1e230601ee7949ef1e75f1a4439f4e7', 1, 'PrParametrizeInt.
 ['prcentripetal', ['PrCENTRIPETAL', ['
 ../pr_parametrize_bdy_8h.html#ac4f0ea96d827517d164d141b76822905a9d8ef4f0fb695440e1e12f49b448ee1c', 1, 'PrParametrizeBdy.
 ['prchordlengthbdy', ['PrCHORDLENGTHBDY', ['
 ../pr_parametrize_bdy_8h.html#ac4f0ea96d827517d164d141b76822905a01024afd7e94cb195b4956f794df5ea3', 1, 'PrParametrizeBdy.
 ['pretop_5fat', ['pretop_AT', ['
 ../namespace_go.html#a0589ebbe872cac278faf8159c61b8292ab0740d5b95cb39891d4fbc9e8ae72315', 1, 'Go']]],
 ['pretop_5fin', ['pretop_IN', ['
 ../namespace_go.html#a0589ebbe872cac278faf8159c61b8292a418cdf83cdf220855b9ed6eec2a3f68d', 1, 'Go']]],
 ['pretop_5fon', ['pretop_ON', ['
 ../namespace_go.html#a0589ebbe872cac278faf8159c61b8292a57715e2bd06af960bad6c7fa862d717', 1, 'Go']]],
 ['pretop_5fout', ['pretop_OUT', ['
 ../namespace_go.html#a0589ebbe872cac278faf8159c61b8292a6656387d8b452630d832f8a6ba20c79d', 1, 'Go']]],
 ['pretop_5fundef', ['pretop_UNDEF', ['
 ../namespace_go.html#a0589ebbe872cac278faf8159c61b8292a3ead4181c54e7d4f7e0add68c20ad134', 1, 'Go']]],
]
```

```
['prfromuv', ['PrFROMUV', ['
 ../_pr_parametrize_int_8h.html#ab93ec75afebf4127286a826175e2184ea2740fa09d2d03e4c2d55733fd8ec3e36', 1, 'PrParametrizeInt.
['prunifbdy', ['PrUNIFBDY', ['
 ../_pr_parametrize_bdy_8h.html#ac4f0ea96d827517d164d141b76822905ac0af8acf5a5382cb8d17efec95c6f194', 1, 'PrParametrizeBdy.
['pstpoint', ['PSTPoint', ['
 ../_namespace_go.html#a8fd32afd3a4d4892e4020b0580de70e1a4eea7b565319ed23e1e46c7b519bcfe2', 1, 'Go']],
['pstpointtangent', ['PSTPointTangent', ['
 ../_namespace_go.html#a8fd32afd3a4d4892e4020b0580de70e1a44f2da44ddde83fedb06c67050ddc6e4', 1, 'Go']],
['pstscattered', ['PSTScattered', ['
 ../_namespace_go.html#a8fd32afd3a4d4892e4020b0580de70e1ae7ab6c2a6415887b29d8152b39a92f92', 1, 'Go']]
]
```

Definition at line 1 of file enumvalues\_f.js.

## 30.1485 doc/html/search/files\_0.js File Reference

### Variables

- var [searchData](#)

### 30.1485.1 Variable Documentation

#### 30.1485.1.1 var searchData

##### Initial value:

```
=
[
 ['2dpoly_5ffor_5fs2m_2eh', ['2dpoly_for_s2m.h', ['../2dpoly__for_s2m_8h.html', 1, '']]]
]
```

Definition at line 1 of file files\_0.js.

## 30.1486 doc/html/search/files\_1.js File Reference

### Variables

- var [searchData](#)

### 30.1486.1 Variable Documentation

#### 30.1486.1.1 var searchData

##### Initial value:

```
=
[
 ['adaptcurve_2eh', ['AdaptCurve.h', ['../_adapt_curve_8h.html', 1, '']]],
 ['adaptsurface_2eh', ['AdaptSurface.h', ['../_adapt_surface_8h.html', 1, '']]],
 ['algobj2dint_2eh', ['AlgObj2DInt.h', ['../_alg_obj2_d_int_8h.html', 1, '']]],
 ['algobj3dint_2eh', ['AlgObj3DInt.h', ['../_alg_obj3_d_int_8h.html', 1, '']]],
 ['algobjectint_2eh', ['AlgObjectInt.h', ['../_alg_object_int_8h.html', 1, '']]],
 ['api_2eh', ['api.h', ['../api_8h.html', 1, '']]],
 ['approxcrvtoseqs_2eh', ['ApproxCrvToSeqs.h', ['../_approx_crv_to_seqs_8h.html', 1, '']]],
 ['approxcurve_2eh', ['ApproxCurve.h', ['../_approx_curve_8h.html', 1, '']]],
 ['approxsurf_2eh', ['ApproxSurf.h', ['../_approx_surf_8h.html', 1, '']]],
 ['array_2eh', ['Array.h', ['../_array_8h.html', 1, '']]],
 ['aux2_2ecpp', ['aux2.cpp', ['../aux2_8cpp.html', 1, '']]],
 ['aux2_2eh', ['aux2.h', ['../aux2_8h.html', 1, '']]]
]
```

Definition at line 1 of file files\_1.js.

## 30.1487 doc/html/search/files\_10.js File Reference

### Variables

- var [searchData](#)

### 30.1487.1 Variable Documentation

#### 30.1487.1.1 var searchData

##### Initial value:

```
=
[
 ['quadmesh_2eh', ['QuadMesh.h', ['../_quad_mesh_8h.html', 1, '']]],
 ['qualitymodule_2ddoxymain_2eh', ['qualitymodule-doxymain.h', ['../qualitymodule-doxymain_8h.html', 1, '']]],
 ['qualityresults_2eh', ['QualityResults.h', ['../_quality_results_8h.html', 1, '']]],
 ['qualityutils_2eh', ['QualityUtils.h', ['../_quality_utils_8h.html', 1, '']]]
]
```

Definition at line 1 of file files\_10.js.

## 30.1488 doc/html/search/files\_11.js File Reference

### Variables

- var [searchData](#)

### 30.1488.1 Variable Documentation

#### 30.1488.1.1 var searchData

##### Initial value:

```
=
[
 ['randomnoise_2eh', ['randomnoise.h', ['../randomnoise_8h.html', 1, '']]],
 ['raster_2eh', ['raster.h', ['../raster_8h.html', 1, '']]],
 ['rational_2eh', ['Rational.h', ['../_rational_8h.html', 1, '']]],
 ['rectangularsurfacepropertysheet_2eh', ['RectangularSurfacePropertySheet.h', ['
 ../_rectangular_surface_property_sheet_8h.html', 1, '']]],
 ['rectangularsurfacetessellator_2eh', ['RectangularSurfaceTessellator.h', ['
 ../_rectangular_surface_tessellator_8h.html', 1, '']]],
 ['rectangularvolumepropertysheet_2eh', ['RectangularVolumePropertySheet.h', ['
 ../_rectangular_volume_property_sheet_8h.html', 1, '']]],
 ['rectangularvolumetessellator_2eh', ['RectangularVolumeTessellator.h', ['
 ../_rectangular_volume_tessellator_8h.html', 1, '']]],
 ['rectdomain_2eh', ['RectDomain.h', ['../_rect_domain_8h.html', 1, '']]],
 ['rectgrid_2eh', ['RectGrid.h', ['../_rect_grid_8h.html', 1, '']]],
 ['rectgridtessellator_2eh', ['RectGridTessellator.h', ['../_rect_grid_tessellator_8h.html', 1, '']]],
 ['refine_5fall_2ec', ['refine_all.c', ['../refine_all_8c.html', 1, '']]],
 ['registrationutils_2eh', ['RegistrationUtils.h', ['../_registration_utils_8h.html', 1, '']]],
 ['regularizeface_2eh', ['RegularizeFace.h', ['../_regularize_face_8h.html', 1, '']]],
 ['regularizefaceset_2eh', ['RegularizeFaceSet.h', ['../_regularize_face_set_8h.html', 1, '']]],
 ['regularizeutils_2eh', ['RegularizeUtils.h', ['../_regularize_utils_8h.html', 1, '']]],
 ['regularmesh_2eh', ['RegularMesh.h', ['../_regular_mesh_8h.html', 1, '']]],
 ['regularvolmesh_2eh', ['RegularVolMesh.h', ['../_regular_vol_mesh_8h.html', 1, '']]],
 ['rotatedbox_2eh', ['RotatedBox.h', ['../_rotated_box_8h.html', 1, '']]]
]
```

Definition at line 1 of file files\_11.js.

## 30.1489 doc/html/search/files\_12.js File Reference

### Variables

- var [searchData](#)

#### 30.1489.1 Variable Documentation

##### 30.1489.1.1 var searchData

Definition at line 1 of file files\_12.js.

## 30.1490 doc/html/search/files\_13.js File Reference

### Variables

- var [searchData](#)

#### 30.1490.1 Variable Documentation

##### 30.1490.1.1 var searchData

Definition at line 1 of file files\_13.js.

## 30.1491 doc/html/search/files\_14.js File Reference

### Variables

- var [searchData](#)

#### 30.1491.1 Variable Documentation

##### 30.1491.1.1 var searchData

#### Initial value:

```
=
[
 ['utils_2eh', ['Utils.h', ['../utils_8h.html', 1, '']]]
]
```

Definition at line 1 of file files\_14.js.

## 30.1492 doc/html/search/files\_15.js File Reference

### Variables

- var [searchData](#)

### 30.1492.1 Variable Documentation

#### 30.1492.1.1 var searchData

#### Initial value:

```
=
[
 ['values_2eh', ['Values.h', ['./_values_8h.html', 1, '']]],
 ['vertex_2eh', ['Vertex.h', ['./_vertex_8h.html', 1, '']]],
 ['viewlib_5fdoxymain_2eh', ['viewlib_doxymain.h', ['./viewlib__doxymain_8h.html', 1, '']]],
 ['volboundarycondition_2eh', ['VolBoundaryCondition.h', ['./_vol_boundary_condition_8h.html', 1, '']]],
 ['volpointbdcond_2eh', ['VolPointBdCond.h', ['./_vol_point_bd_cond_8h.html', 1, '']]],
 ['volsolution_2eh', ['VolSolution.h', ['./_vol_solution_8h.html', 1, '']]],
 ['volumeadjacency_2eh', ['VolumeAdjacency.h', ['./_volume_adjacency_8h.html', 1, '']]],
 ['volumeinterpolator_2eh', ['VolumeInterpolator.h', ['./_volume_interpolator_8h.html', 1, '']]],
 ['volumemodel_2eh', ['VolumeModel.h', ['./_volume_model_8h.html', 1, '']]],
 ['volumemodelcreator_2eh', ['VolumeModelCreator.h', ['./_volume_model_creator_8h.html', 1, '']]],
 ['volumemodelfilehandler_2eh', ['VolumeModelFileHandler.h', ['./_volume_model_file_handler_8h.html', 1, '']]],
],
 ['volumeparametercurve_2eh', ['VolumeParameterCurve.h', ['./_volume_parameter_curve_8h.html', 1, '']]],
 ['volumes_2eh', ['Volumes.h', ['./_volumes_8h.html', 1, '']]],
 ['volumespacecurve_2eh', ['VolumeSpaceCurve.h', ['./_volume_space_curve_8h.html', 1, '']]],
 ['volumetools_2eh', ['VolumeTools.h', ['./_volume_tools_8h.html', 1, '']]]
]
```

Definition at line 1 of file files\_15.js.

## 30.1493 doc/html/search/files\_2.js File Reference

### Variables

- var [searchData](#)

### 30.1493.1 Variable Documentation

#### 30.1493.1.1 var searchData

#### Initial value:

```

=
[
 ['bandmat_2ecpp', ['bandmat.cpp', ['../bandmat_8cpp.html', 1, '']],
 ['barycoordsystem_2eh', ['BaryCoordSystem.h', ['../_bary_coord_system_8h.html', 1, '']],
 ['barycoordsystemtriangle3d_2eh', ['BaryCoordSystemTriangle3D.h', ['
 ../_bary_coord_system_triangle3_d_8h.html', 1, '']],
 ['bdcondfunctor_2eh', ['BdCondFunctor.h', ['../_bd_cond_functor_8h.html', 1, '']],
 ['bdconditiontype_2eh', ['BdConditionType.h', ['../_bd_condition_type_8h.html', 1, '']],
 ['bernsteinmulti_2eh', ['BernsteinMulti.h', ['../_bernstein_multi_8h.html', 1, '']],
 ['bernsteinpoly_2eh', ['BernsteinPoly.h', ['../_bernstein_poly_8h.html', 1, '']],
 ['bernsteintetrahedralpoly_2eh', ['BernsteinTetrahedralPoly.h', ['../_bernstein_tetrahedral_poly_8h.html', 1, '']],
 ['bernsteintriangularpoly_2eh', ['BernsteinTriangularPoly.h', ['../_bernstein_triangular_poly_8h.html', 1, '']],
 ['bernsteinutils_2eh', ['BernsteinUtils.h', ['../_bernstein_utils_8h.html', 1, '']],
 ['beziertriangle_2eh', ['BezierTriangle.h', ['../_bezier_triangle_8h.html', 1, '']],
 ['binom_2eh', ['binom.h', ['../binom_8h.html', 1, '']],
 ['binomial_2eh', ['Binomial.h', ['../_binomial_8h.html', 1, '']],
 ['blockboundarycondition_2eh', ['BlockBoundaryCondition.h', ['../_block_boundary_condition_8h.html', 1, '']],
 ['blockpointbdcond_2eh', ['BlockPointBdCond.h', ['../_block_point_bd_cond_8h.html', 1, '']],
 ['blocksolution_2eh', ['BlockSolution.h', ['../_block_solution_8h.html', 1, '']],
 ['body_2eh', ['Body.h', ['../_body_8h.html', 1, '']],
 ['boolean_2eh', ['boolean.h', ['../boolean_8h.html', 1, '']],
 ['boundaryfunctionint_2eh', ['BoundaryFunctionInt.h', ['../_boundary_function_int_8h.html', 1, '']],
 ['boundarygeomint_2eh', ['BoundaryGeomInt.h', ['../_boundary_geom_int_8h.html', 1, '']],
 ['boundedcurve_2eh', ['BoundedCurve.h', ['../_bounded_curve_8h.html', 1, '']],
 ['boundedsurface_2eh', ['BoundedSurface.h', ['../_bounded_surface_8h.html', 1, '']],
 ['boundedutils_2eh', ['BoundedUtils.h', ['../_bounded_utils_8h.html', 1, '']],
 ['boundingbox_2eh', ['BoundingBox.h', ['../_bounding_box_8h.html', 1, '']],
 ['brent_5fminimize_2eh', ['brent_minimize.h', ['../_brent__minimize_8h.html', 1, '']],
 ['bsplinebasis_2eh', ['BsplineBasis.h', ['../_bspline_basis_8h.html', 1, '']]]
]

```

Definition at line 1 of file files\_2.js.

## 30.1494 doc/html/search/files\_3.js File Reference

### Variables

- var [searchData](#)

### 30.1494.1 Variable Documentation

#### 30.1494.1.1 var searchData

Definition at line 1 of file files\_3.js.

## 30.1495 doc/html/search/files\_4.js File Reference

### Variables

- var [searchData](#)

### 30.1495.1 Variable Documentation

#### 30.1495.1.1 var searchData

##### Initial value:

```
=
[
 ['datahandler_2eh', ['DataHandler.h', ['../_data_handler_8h.html', 1, '']]],
 ['datahandlervolandlr_2eh', ['DataHandlerVolAndLR.h', ['../_data_handler_vol_and_l_r_8h.html', 1, '']]],
 ['defaultdatahandler_2eh', ['DefaultDataHandler.h', ['../_default_data_handler_8h.html', 1, '']]],
 ['destruct_2ec', ['destruct.c', ['../destruct_8c.html', 1, '']]],
 ['direction2d_2eh', ['Direction2D.h', ['../_direction2_d_8h.html', 1, '']]],
 ['directioncone_2eh', ['DirectionCone.h', ['../_direction_cone_8h.html', 1, '']]],
 ['disc_2eh', ['Disc.h', ['../_disc_8h.html', 1, '']]],
 ['domain_2eh', ['Domain.h', ['../_domain_8h.html', 1, '']]]
]
```

Definition at line 1 of file files\_4.js.

### 30.1496 doc/html/search/files\_5.js File Reference

#### Variables

- var [searchData](#)

### 30.1496.1 Variable Documentation

#### 30.1496.1.1 var searchData

Definition at line 1 of file files\_5.js.

### 30.1497 doc/html/search/files\_6.js File Reference

#### Variables

- var [searchData](#)

### 30.1497.1 Variable Documentation

#### 30.1497.1.1 var searchData

Definition at line 1 of file files\_6.js.



## 30.1498 doc/html/search/files\_7.js File Reference

### Variables

- var [searchData](#)

### 30.1498.1 Variable Documentation

#### 30.1498.1.1 var searchData

Definition at line 1 of file files\_7.js.

## 30.1499 doc/html/search/files\_8.js File Reference

### Variables

- var [searchData](#)

### 30.1499.1 Variable Documentation

#### 30.1499.1.1 var searchData

#### Initial value:

```
=
[
 ['hahnssurfacegen_2eh', ['HahnsSurfaceGen.h', ['../_hahns_surface_gen_8h.html', 1, '']]],
 ['handle_2eh', ['Handle.h', ['../_handle_8h.html', 1, '']]],
 ['handleid_2ecpp', ['HandleId.cpp', ['../_handle_id_8cpp.html', 1, '']]],
 ['handleid_2eh', ['HandleId.h', ['../_handle_id_8h.html', 1, '']]],
 ['hedart_2eh', ['HeDart.h', ['../_he_dart_8h.html', 1, '']]],
 ['hermiteappc_2eh', ['HermiteAppC.h', ['../_hermite_app_c_8h.html', 1, '']]],
 ['hermiteapprvalsurf_2eh', ['HermiteApprEvalSurf.h', ['../_hermite_appr_eval_surf_8h.html', 1, '']]],
 ['hermiteapps_2eh', ['HermiteAppS.h', ['../_hermite_app_s_8h.html', 1, '']]],
 ['hermitegrid1d_2eh', ['HermiteGrid1D.h', ['../_hermite_grid1_d_8h.html', 1, '']]],
 ['hermitegrid1dmulti_2eh', ['HermiteGrid1DMulti.h', ['../_hermite_grid1_d_multi_8h.html', 1, '']]],
 ['hermitegrid2d_2eh', ['HermiteGrid2D.h', ['../_hermite_grid2_d_8h.html', 1, '']]],
 ['hermiteinterpolator_2eh', ['HermiteInterpolator.h', ['../_hermite_interpolator_8h.html', 1, '']]],
 ['hetraits_2eh', ['HeTraits.h', ['../_he_traits_8h.html', 1, '']]],
 ['hetriang_2ecpp', ['HeTriang.cpp', ['../_he_triang_8cpp.html', 1, '']]],
 ['hetriang_2eh', ['HeTriang.h', ['../_he_triang_8h.html', 1, '']]],
 ['hholder_2ecpp', ['hholder.cpp', ['../_hholder_8cpp.html', 1, '']]],
 ['hp_5fs1880_2ec', ['hp_s1880.c', ['../_hp_s1880_8c.html', 1, '']]],
 ['hyperbola_2eh', ['Hyperbola.h', ['../_hyperbola_8h.html', 1, '']]]
]
```

Definition at line 1 of file files\_8.js.

## 30.1500 doc/html/search/files\_9.js File Reference

### Variables

- var [searchData](#)

### 30.1500.1 Variable Documentation

#### 30.1500.1.1 var searchData

Definition at line 1 of file files\_9.js.

## 30.1501 doc/html/search/files\_a.js File Reference

### Variables

- var [searchData](#)

### 30.1501.1 Variable Documentation

#### 30.1501.1.1 var searchData

#### Initial value:

```
=
[
 ['jacobi_2ecpp', ['jacobi.cpp', ['./jacobi_8cpp.html', 1, '']]],
 ['jonvec_2eh', ['jonvec.h', ['./jonvec_8h.html', 1, '']]],
 ['jquery_2ejs', ['jquery.js', ['./jquery_8js.html', 1, '']]]
]
```

Definition at line 1 of file files\_a.js.

## 30.1502 doc/html/search/files\_b.js File Reference

### Variables

- var [searchData](#)

### 30.1502.1 Variable Documentation

#### 30.1502.1.1 var searchData

Definition at line 1 of file files\_b.js.

## 30.1503 doc/html/search/files\_c.js File Reference

### Variables

- var [searchData](#)

### 30.1503.1 Variable Documentation

#### 30.1503.1.1 var searchData

##### Initial value:

```
=
[
 ['main_2ecpp', ['main.cpp', ['./main_8cpp.html', 1, '']],
 ['main_5fhe_5fref_2eh', ['main_he_ref.h', ['./main__he__ref_8h.html', 1, '']],
 ['mainpage_5fhed_2eh', ['mainpage_hed.h', ['./mainpage__hed_8h.html', 1, '']],
 ['mainpage_5fttl_2eh', ['mainpage_ttl.h', ['./mainpage__ttl_8h.html', 1, '']],
 ['make3d_2ec', ['make3D.c', ['./make3_d_8c.html', 1, '']],
 ['makecvkreg_2ec', ['makecvkreg.c', ['./makecvkreg_8c.html', 1, '']],
 ['makesfkreg_2ec', ['makesfkreg.c', ['./makesfkreg_8c.html', 1, '']],
 ['maketracks_2ec', ['maketracks.c', ['./maketracks_8c.html', 1, '']],
 ['matrixxd_2eh', ['MatrixXD.h', ['./_matrix_x_d_8h.html', 1, '']],
 ['mesh2d_2eh', ['Mesh2D.h', ['./_mesh2_d_8h.html', 1, '']],
 ['mesh2diterator_2eh', ['Mesh2DIterator.h', ['./_mesh2_d_iterator_8h.html', 1, '']],
 ['mesh2dutils_2eh', ['Mesh2DUtils.h', ['./_mesh2_d_utils_8h.html', 1, '']],
 ['mk_5fcv_5fcycl_2ec', ['mk_cv_cycl.c', ['./mk_cv_cycl_8c.html', 1, '']],
 ['modelquality_2eh', ['ModelQuality.h', ['./_model_quality_8h.html', 1, '']],
 ['modelrepair_2eh', ['ModelRepair.h', ['./_model_repair_8h.html', 1, '']],
 ['modifysurf_2eh', ['ModifySurf.h', ['./_modify_surf_8h.html', 1, '']],
 ['mouse_2ecpp', ['mouse.cpp', ['./mouse_8cpp.html', 1, '']],
 ['mouse_2eh', ['mouse.h', ['./mouse_8h.html', 1, '']],
 ['myexcept_2ecpp', ['myexcept.cpp', ['./myexcept_8cpp.html', 1, '']],
 ['myexcept_2eh', ['myexcept.h', ['./myexcept_8h.html', 1, '']]]
]
```

Definition at line 1 of file files\_c.js.

## 30.1504 doc/html/search/files\_d.js File Reference

### Variables

- var [searchData](#)

### 30.1504.1 Variable Documentation

#### 30.1504.1.1 var searchData

##### Initial value:

```
=
[
 ['newfft_2ecpp', ['newfft.cpp', ['./newfft_8cpp.html', 1, '']],
 ['newknots_2ec', ['newknots.c', ['./newknots_8c.html', 1, '']],
 ['newmat_2eh', ['newmat.h', ['./newmat_8h.html', 1, '']],
 ['newmat1_2ecpp', ['newmat1.cpp', ['./newmat1_8cpp.html', 1, '']],
 ['newmat2_2ecpp', ['newmat2.cpp', ['./newmat2_8cpp.html', 1, '']],
 ['newmat3_2ecpp', ['newmat3.cpp', ['./newmat3_8cpp.html', 1, '']],
 ['newmat4_2ecpp', ['newmat4.cpp', ['./newmat4_8cpp.html', 1, '']],
 ['newmat5_2ecpp', ['newmat5.cpp', ['./newmat5_8cpp.html', 1, '']],
 ['newmat6_2ecpp', ['newmat6.cpp', ['./newmat6_8cpp.html', 1, '']],
 ['newmat7_2ecpp', ['newmat7.cpp', ['./newmat7_8cpp.html', 1, '']],
 ['newmat8_2ecpp', ['newmat8.cpp', ['./newmat8_8cpp.html', 1, '']],
 ['newmat9_2ecpp', ['newmat9.cpp', ['./newmat9_8cpp.html', 1, '']],
 ['newmatap_2eh', ['newmatap.h', ['./newmatap_8h.html', 1, '']],
 ['newmatex_2ecpp', ['newmatex.cpp', ['./newmatex_8cpp.html', 1, '']],
 ['newmatio_2eh', ['newmatio.h', ['./newmatio_8h.html', 1, '']],
 ['newmatn1_2ecpp', ['newmatn1.cpp', ['./newmatn1_8cpp.html', 1, '']],
 ['newmatn1_2eh', ['newmatn1.h', ['./newmatn1_8h.html', 1, '']],
 ['newmatrc_2eh', ['newmatrc.h', ['./newmatrc_8h.html', 1, '']],
 ['newmatrm_2ecpp', ['newmatrm.cpp', ['./newmatrm_8cpp.html', 1, '']],
 ['newmatrm_2eh', ['newmatrm.h', ['./newmatrm_8h.html', 1, '']],
 ['nl_5fex_2ecpp', ['nl_ex.cpp', ['./nl__ex_8cpp.html', 1, '']],
 ['nooptesselator_2eh', ['NoopTesselator.h', ['./_noop_tesselator_8h.html', 1, '']]]
]
```

Definition at line 1 of file files\_d.js.

## 30.1505 doc/html/search/files\_e.js File Reference

### Variables

- var [searchData](#)

### 30.1505.1 Variable Documentation

#### 30.1505.1.1 var searchData

#### Initial value:

```
=
[
 ['objectheader_2eh', ['ObjectHeader.h', ['../_object_header_8h.html', 1, '']]],
 ['offsetsurfaceutils_2eh', ['OffsetSurfaceUtils.h', ['../_offset_surface_utils_8h.html', 1, '']]],
 ['orientcurves_2eh', ['orientCurves.h', ['../orient_curves_8h.html', 1, '']]]
]
```

Definition at line 1 of file files\_e.js.

## 30.1506 doc/html/search/files\_f.js File Reference

### Variables

- var [searchData](#)

### 30.1506.1 Variable Documentation

#### 30.1506.1.1 var searchData

Definition at line 1 of file files\_f.js.

## 30.1507 doc/html/search/functions\_0.js File Reference

### Variables

- var [searchData](#)

### 30.1507.1 Variable Documentation

#### 30.1507.1.1 var searchData

##### Initial value:

```
=
[
 ['_5f_5fpower', ['__power', ['../namespacestd.html#a97b81ceb78c6f342eb91ad62ba86260f', 1, 'std::__power(_Tp
 __x, _Integer __n, _MonoidOperation __oper)'], ['../namespacestd.html#ac866b28c409b5c4b6b613e17fd3b5c85', 1, '
 std::__power(_Tp __x, _Integer __n)']]]
]
```

Definition at line 1 of file functions\_0.js.

## 30.1508 doc/html/search/functions\_1.js File Reference

### Variables

- var [searchData](#)
- PrPathTriangleSeq [h](#)
- PrPathTriangleSeq int [dim](#)
- PrPathTriangleSeq int [double aepsge](#)
- PrPathTriangleSeq int [double](#) int [constdir =0](#)
- PrPathTriangleSeq int [double](#) int [bool repar](#)

### 30.1508.1 Variable Documentation

#### 30.1508.1.1 PrPathTriangleSeq int double aepsge [property], [bound]

Definition at line 103 of file functions\_1.js.

#### 30.1508.1.2 PrPathTriangleSeq int double int constdir =0 [property], [bound]

Definition at line 123 of file functions\_1.js.

#### 30.1508.1.3 PrPathTriangleSeq int dim [property], [bound]

Definition at line 103 of file functions\_1.js.

#### 30.1508.1.4 PrParamUtil h

Definition at line 103 of file functions\_1.js.

### 30.1508.1.5 PrPathTriangleSeq int double int bool repar [property],[bound]

#### Initial value:

```
=true)'], ['../class_go_1_l_approx_surf.html#ae046f4d99fff00a02c1707e92b5053dd', 1, '
 Go::ApproxSurf::ApproxSurf(shared_ptr<T> SplineSurface &
 gt; &srft, const std::vector<T> &points, const std::vector<
 T> &parvals, int dim, double aepsge, int
 constmdir=0, bool approx_orig=false, bool close_belt=false, int nmb_stabil=0, bool
 repar=true)'], ['../class_go_1_l_approx_surf.html#
 a7781c1001f19a97499b193907d693b15', 1, 'Go::ApproxSurf::ApproxSurf()']],
['approxvolparamcurve', ['approxVolParamCurve', ['../namespace_go_1_l_volume_tools.html#
 aa500ad08b2e111caedb2c15ba3c6999f', 1, 'Go::VolumeTools']],
['ar_5fld_5fft', ['ar_ld_ft', ['../class_n_e_w_m_a_t_1_l_f_f_t___controller.html#
 aa2d9e1babelle9df86986a62dd93ebac', 1, 'NEWMAT::FFT_Controller']],
['arclength', ['arclength', ['../class_go_1_l_ft_curve_segment.html#
 a953153d17ea3bfff870bf270c716d487d', 1, 'Go::ftCurveSegment::arclength()'], ['../
 class_go_1_l_ft_curve.html#a054c7514fc2e82eafcbfc8e1714061ea', 1, '
 Go::ftCurve::arclength()']],
['area', ['area', ['../class_go_1_l_ft_surface.html#
 aadc67b36ef653ff7e4c09ea06456a0d0', 1, 'Go::ftSurface::area()'], ['../
 class_go_1_l_bounded_surface.html#a351925baa7e536730663a87ea0fce006', 1, '
 Go::BoundedSurface::area()'], ['../
 class_go_1_l_composite_surface.html#a6a68591f99fe0ce92a8442843a6e403b', 1, '
 Go::CompositeSurface::area()'], ['../
 class_go_1_l_elementary_surface.html#ab20ceab1a6924addf76e0d04850de566', 1, '
 Go::ElementarySurface::area()'], ['../
 class_go_1_l_param_surface.html#a0417c918e78108fb3a22f969fcc0498a', 1, '
 Go::ParamSurface::area()'], ['../
 class_go_1_l_spline_surface.html#ab0e8c17c1d43ddab2fd56ee045c2c183', 1, '
 Go::SplineSurface::area()'], ['../
 class_go_1_l_surface_of_linear_extrusion.html#
 af7839cc0ddb64b5033ae98b733e0ea14', 1, 'Go::SurfaceOfLinearExtrusion::area()'], ['../
 class_go_1_l_surface_of_revolution.html#a6b877c9c54faa2f1fec69be7d303118c
 ', 1, 'Go::SurfaceOfRevolution::area()'], ['../
 class_go_1_l_element2_d.html#a2321c59da3c5c143cfb2ccca7b791dbf', 1, '
 Go::Element2D::area()'], ['../class_go_1_l_l_r_spline_surface
 .html#aab242113dae866aa4e4abf04dfc8dc85', 1, 'Go::LRSplineSurface::area()'], ['../
 class_go_1_l_surface_on_volume.html#ad16e1c297d24c01995f1c407a4f6d418', 1, '
 Go::SurfaceOnVolume::area()'], ['../namespace_go.html#
 a8cc8f744af2fc5541e493ffdd3d2e1f5', 1, 'Go::area(const Array<T, 2> > *c)'], ['../
 namespace_go.html#a04f0f159a17bcc67285bb7349debd945', 1, 'Go::area(const Array<
 T, 3> > *c)'], ['../pr_param_util_8h.html#a88d3831b85d9d3912354527e5fe6d73c', 1, '
 area(const double &x0, const double &y0, const double &x1, const double &
 amp;y1, const double &x2, const double &y2):
```

Definition at line 123 of file functions\_1.js.

### 30.1508.1.6 var searchData

Definition at line 1 of file functions\_1.js.

## 30.1509 doc/html/search/functions\_10.js File Reference

### Variables

- var [searchData](#)
- [sisIP](#) h

### 30.1509.1 Variable Documentation

#### 30.1509.1.1 sisIP h

Definition at line 67 of file functions\_10.js.

30.1509.1.2 var searchData

Definition at line 1 of file functions\_10.js.

## 30.1510 doc/html/search/functions\_11.js File Reference

### Variables

- var [searchData](#)
- hholder [cpp](#)

### 30.1510.1 Variable Documentation

30.1510.1.1 hholder [cpp](#)

Definition at line 3 of file functions\_11.js.

30.1510.1.2 var searchData

#### Initial value:

```
=
[
 ['qrz', ['QRZ', ['../namespace_n_e_w_m_a_t.html#ae1eac58bd06f44ae7f02b39163b2fd80', 1, 'NEWMAT:QRZ (Matrix
 &, UpperTriangularMatrix &);']], ['../namespace_n_e_w_m_a_t.html#a3786173220a9e54601def50dc380b32f', 1, '
 NEWMAT:QRZ (const Matrix &, Matrix &, Matrix &);']], ['
 ../hholder_8cpp.html#abf43bc890b60a01fda0c7db9631bcf64', 1, 'QRZ (Matrix &, X, UpperTriangularMatrix &);U:##160
```

Definition at line 1 of file functions\_11.js.

## 30.1511 doc/html/search/functions\_12.js File Reference

### Variables

- var [searchData](#)
- gl\_aux [cpp](#)

### 30.1511.1 Variable Documentation

30.1511.1.1 gl\_aux [cpp](#)

Definition at line 17 of file functions\_12.js.

## 30.1511.1.2 var searchData

## Initial value:

```

=
[
 ['r1_5fr1', ['R1_R1', ['./class_r_b_d_c_o_m_m_o_n_l_l_r1_r1.html#a278e42bea46cadb3e7579f8357975e34', 1,
 'RBD_COMMON::R1_R1']],
 ['radialedges', ['radialEdges', ['./class_go_1_1ft_volume.html#aef559a6497b829cf31faae6942263d97', 1,
 'Go::ftVolume']],
 ['radius', ['radius', ['./class_go_1_1cylinder_int.html#a4493b91edd175e4166b7ab07a9b35c2c', 1,
 'Go::CylinderInt::radius()'], ['./class_go_1_1sphere_int.html#abb8450f4ca8990149d73b1eea2f3015d', 1,
 'Go::SphereInt::radius()']],
 ['radix', ['Radix', [
 ['./class_n_e_w_m_a_t_l_1_floating_point_precision.html#afb7383880d8bbd99bab8e25e3758aa7f', 1, 'NEWMAT::FloatingPointPrecision'],
 ['./class_go_1_1_spline_curve.html#a53bcb9a51debc1441a4008cadb904dc2', 1,
 'Go::SplineCurve::raiseOrder()'], ['./class_go_1_1_spline_surface.html#a58f9393ed2771ce6167c4504a9df80e8', 1,
 'Go::SplineSurface::raiseOrder()'], ['./class_go_1_1_spline_volume.html#a306ad4a1f5ab97ab1087360331a51a26', 1,
 'Go::SplineVolume::raiseOrder()']],
 ['range_5ferror', ['Range_error', [
 ['./class_r_b_d_c_o_m_m_o_n_l_l_range_error.html#a8bc12b7074abb363df50d8bd82c1c9db', 1, 'RBD_COMMON::Range_error']],
 ['rational', ['rational', ['./class_go_1_1_spline_curve.html#a243012e5d6b527a0e87cc07c6eac9378', 1,
 'Go::SplineCurve::rational()'], ['./class_go_1_1_spline_surface.html#a1cccecbf7d23b89a446af7698aebcad', 1,
 'Go::SplineSurface::rational()'], ['./class_go_1_1_r_b_spline_2_d.html#aae5e8bd36f5897c2dff2b334d342a54f', 1,
 'Go::LRBSpline2D::rational()'], ['./class_go_1_1_r_spline_surface.html#a03a776aca6be1428771bf0f88ca8d2f38', 1,
 'Go::LRSplineSurface::rational()'], ['./class_go_1_1_spline_volume.html#a094ddc31cbdcddede328579dc81de6ef6', 1,
 'Go::SplineVolume::rational()'], ['./class_go_1_1_rational.html#a986b1722c95bc708600cfc0114e0451a', 1,
 'Go::Rational::Rational()'], ['./class_go_1_1_rational.html#acadf8909244c77f7bb16036cd69d0700', 1,
 'Go::Rational::Rational(int p)'], ['./class_go_1_1_rational.html#a2bd8648917321cbfa7bb8b04279b582a', 1, 'Go::Rational::Rational(int q)']],
 ['raw', ['raw', ['./classvector3t.html#a9e486673106c5e00b4a50b7a9699082c', 1, 'vector3t']],
 ['rawdata', ['rawData', ['./class_go_1_1_line_cloud.html#a3430f9f2b895c270e16190bdcc361a60', 1,
 'Go::LineCloud::rawData()'], ['./class_go_1_1_point_cloud.html#aeb184284d03948126f32911f9bc18300', 1,
 'Go::PointCloud::rawData()'], ['./class_go_1_1_point_cloud.html#a8d46240635fff22b7c3317c8a220a8eee', 1, 'Go::PointCloud::rawData(const ')'], ['./class_go_1_1_rect_grid.html#a2228c05541484aeec6f376769db48a3f', 1, 'Go::RectGrid::rawData()'], ['./class_go_1_1_rect_grid.html#a8838e86fa3c332093ed997b3233b7102', 1, 'Go::RectGrid::rawData() const ']],
 ['rawregistration', ['rawRegistration', ['./namespace_go.html#aac556b609d575079c24b6f3d3b768298', 1, 'Go']]
],
 ['rcoefs_5fbegin', ['rcoefs_begin', ['./class_go_1_1_spline_curve.html#aafe4f32a475ec2d3aad5d639a7ef8ca', 1,
 'Go::SplineCurve::rcoefs_begin()'], ['./class_go_1_1_spline_curve.html#a180d33d6fbf4c6f6fc258a40c16a3c19', 1,
 'Go::SplineCurve::rcoefs_begin() const '], ['./class_go_1_1_spline_surface.html#a187f0d825f895dfa7c60f82c12156c4b', 1, 'Go::SplineSurface::rcoefs_begin()'], ['./class_go_1_1_spline_surface.html#a1e69cbb383f4ec9e60e3f70fec6ed9c', 1, 'Go::SplineSurface::rcoefs_begin() const '], ['./class_go_1_1_spline_volume.html#a143461b2f89077c804bd5a5b81632714', 1, 'Go::SplineVolume::rcoefs_begin()'], ['./class_go_1_1_spline_volume.html#ab43772480d569bfb1cb88e57490ffalla', 1, 'Go::SplineVolume::rcoefs_begin() const ']],
 ['rcoefs_5fend', ['rcoefs_end', ['./class_go_1_1_spline_curve.html#a0ccc811f2d8121d2c89c280ba85efec', 1,
 'Go::SplineCurve::rcoefs_end()'], ['./class_go_1_1_spline_curve.html#a0074323f14b7204e29b3263f0b13efc0', 1,
 'Go::SplineCurve::rcoefs_end() const '], ['./class_go_1_1_spline_surface.html#a5cb25201024ac7b7aa784358b3c8722bc', 1,
 'Go::SplineSurface::rcoefs_end()'], ['./class_go_1_1_spline_surface.html#ab996b487c4746f2db272058be07fb52f', 1, 'Go::SplineSurface::rcoefs_end() const '], ['./class_go_1_1_spline_volume.html#a7d00b9d0862c03d1f5fc31948d9c5541', 1, 'Go::SplineVolume::rcoefs_end()'], ['./class_go_1_1_spline_volume.html#a6709620910ecc8c61bd990b0fcla29d9', 1, 'Go::SplineVolume::rcoefs_end() const ']],
 ['read', ['read', ['./class_go_1_1_bounded_curve.html#a3147ca8c72df42d4e2359b348dbc7803', 1,
 'Go::BoundedCurve::read()'], ['./class_go_1_1_bounded_surface.html#af40eb854cff758a1d69452c5fa7b046c', 1,
 'Go::BoundedSurface::read(std::istream &);'], ['./class_go_1_1_bounded_surface.html#a48659c768ba2ac439808ed09f997c4', 1,
 'Go::BoundedSurface::read(std::istream &); bool fix_trim_cvts'], ['./class_go_1_1_bspline_basis.html#a15751c6a8b9182e0a1f0dabe92af5643', 1, 'Go::BsplineBasis::read()'], ['./class_go_1_1_circle.html#a561b41187108e9b309d38f8241168a60', 1, 'Go::Circle::read()'], ['./class_go_1_1_composite_surface.html#af754175964543c6bfa6ba6b1cd2ec94c', 1, 'Go::CompositeSurface::read()'], ['./class_go_1_1_curve_on_surface.html#a87c10a0b1dea3892d3c5bcf8bb48f6ca', 1, 'Go::CurveOnSurface::read()'], ['./class_go_1_1_cylinder.html#a0ab5b3d9f2b85fb8c24bccc7716739cb7', 1, 'Go::Cylinder::read()'], ['./class_go_1_1_disc.html#a1e0d28eafcf4fa5b62179b8076d1e60c8', 1, 'Go::Disc::read()'], ['./class_go_1_1_ellipse.html#a14658f7ce525fb91719c53ddc23f7145', 1, 'Go::Ellipse::read()'], ['./class_go_1_1_hyperbola.html#a4e5bcc0aeb7aa02730a57e8e9419eb90', 1, 'Go::Hyperbola::read()'], ['./class_go_1_1_line.html#a97b0741988ff9650df1191dfb1fc770', 1, 'Go::Line::read()'], ['./class_go_1_1_line_cloud.html#a306ad4a1f5ab97ab1087360331a51a26', 1, 'Go::LineCloud::read()'], ['./class_go_1_1_object_header.html#a40795d9e4fa51548dfef133befd6085ac', 1, 'Go::ObjectHeader::read()'], ['./class_go_1_1_parabola.html#ace279c2aa49604f4566605f9ecef9ea', 1, 'Go::Parabola::read()'], ['./class_go_1_1_plane.html#a985561bf7b890544a1492d2e9dfa1262', 1, 'Go::Plane::read()'], ['./class_go_1_1_point_cloud.html#afc7e14c4ab1b737ec56314124a28d18', 1, 'Go::PointCloud::read()'], ['./class_go_1_1_rect_grid.html#ab674ee685f79f10cbf7513391c7b138f', 1, 'Go::RectGrid::read()'], ['./class_go_1_1_sphere.html#a6e8eed759544a2edd93421b797ad3782', 1, 'Go::Sphere::read()'], ['./class_go_1_1_spline_curve.html#a30aec6770815b9cc7aeb61fcd52a746', 1, 'Go::SplineCurve::read()'], ['./class_go_1_1_spline_surface.html#a9dfb256575fe822472430672a29883', 1, 'Go::SplineSurface::read()'], ['./class_go_1_1_streamable.html#a9dfb256575fe822472430672a29883', 1, 'Go::Streamable::read()'], ['./class_go_1_1_surface_of_linear_extrusion.html#a254286b8407ee48209ad91e70bc8607f', 1, 'Go::SurfaceOfLinearExtrusion::read()'], ['./class_go_1_1_surface_of_revolution.html#a9d608e73a1b0e974dbad7078a1ac9020', 1, 'Go::SurfaceOfRevolution::read()'], ['./class_go_1_1_torus.html#a1cbf7eca85e957f0479938079b7f1f07', 1, 'Go::Torus::read()'], ['./class_go_1_1_array.html#a199ce0e0e0e0e0e0e0e0e0e0e0e0e0e', 1, 'Go::Array::read()'], ['./class_go_1_1_bary_coord_system.html#a91eb73692e0ae0c157e5aa749a3352bd', 1, 'Go::BaryCoordSystem::read()'], ['./class_go_1_1_bounding_box.html#a357abf9122958f6d747d55883cf74277', 1, 'Go::BoundingBox::read()'], ['./class_go_1_1_composite_box.html#a963b13d6cb9266e8c9c0c05117869b85', 1, 'Go::CompositeBox::read()'], ['./class_go_1_1_direction_cone.html#a41d9c682879fec0ab8efae60c8b999e7', 1, 'Go::DirectionCone::read()'], ['./class_go_1_1_point.html#a4306540b18e7ad1f7059c3370b17c83', 1, 'Go::Point::read()'], ['./class_go_1_1_bernstein_multi.html#a7767c00e78e5e2ad5a0be24b605a2583', 1, 'Go::BernsteinMulti::read()'], ['./class_go_1_1_bernstein_poly.html#a64c8fff3352f3958366b142e4edaaaa9', 1, 'Go::BernsteinPoly::read()'],

```



```

../class_go_1_1_bernstein_tetrahedral_poly.html#af758b9a9c83aff8bd5c5f98906ddfb66',1,'Go::BernsteinTetrahedralPoly::read()
../class_go_1_1_bernstein_triangular_poly.html#addef2d12c934d32868035bbbe005c948',1,'
Go::BernsteinTriangularPoly::read()'],['../class_go_1_1_cylinder_int.html#ab9916e69cd910d06ad5f05b9f780c8fe',1,'Go::Cyl
../class_go_1_1_geo_tol.html#a9273319186522ead8f717dae87ae9b1f',1,'Go::GeoTol::read()'],[
../class_go_1_1_interpolated_intersection_curve.html#a91bc2564d6bc1f840f451e9850dcae5f',1,'
Go::InterpolatedIntersectionCurve::read()'],['../class_go_1_1_intersection_point.html#a37f273cf45131cb1fb650d7e434e163f
Go::IntersectionPoint::read()'],['../class_go_1_1_plane_int.html#aaffbfeab5d401db79d65ef540646358ce',1,'Go::PlaneInt::rea
../class_go_1_1_sphere_int.html#a0bfbbf9d7fd3639303e1deb0fda46d57',1,'Go::SphereInt::read()'],[
../class_go_1_1_l_r_b_spline2_d.html#a7e31a499ff5b658193b8cdeaa7b5657d',1,'Go::LRBSpline2D::read()'],[
../class_go_1_1_l_r_spline_surface.html#a55061b561c5706ba922d2d9de3f5d4c1',1,'Go::LRSplineSurface::read()'],[
../class_go_1_1_mesh2_d.html#aac6ae2f050d06a1632e21a1297ccc3cb',1,'Go::Mesh2D::read()'],[
../class_pr_mat.html#a137fb40cee88e91211c2df1a90a0e62c',1,'PrMat::read()'],['../class_pr_matrix.html#afb99calle79ef2180
PrMatrix::read()'],['../class_pr_mat_sparse.html#a6ae00848a5bb4ee4b6cc0460ec09721',1,'PrMatSparse::read()'],[
../class_pr_vec.html#a5bfbbe99e8a711b5d93908f4ce3255d8',1,'PrVec::read()'],[
../class_go_1_1_cone_volume.html#aace95efa8527024bd752680e37e92a9c',1,'Go::ConeVolume::read()'],[
../class_go_1_1_curve_on_volume.html#a791ab5ec72d311be1633713252faa70d',1,'Go::CurveOnVolume::read()'],[
../class_go_1_1_cylinder_volume.html#a065b3bab47de82c643de7ec5cb0f9f23',1,'Go::CylinderVolume::read()'],[
../class_go_1_1_parallelepiped.html#ada24cffb4496199778e5371de67f716a',1,'Go::Parallelepiped::read()'],[
../class_go_1_1_sphere_volume.html#a5c808dfd3cdab6319034a20cdc0b286b',1,'Go::SphereVolume::read()'],[
../class_go_1_1_spline_volume.html#a3b02ceb3clf8cbb4aca26alab1e17a8c',1,'Go::SplineVolume::read()'],[
../class_go_1_1_surface_on_volume.html#a432a8cb6e45c46e05f3a2d7f202366c7',1,'Go::SurfaceOnVolume::read()'],[
../class_go_1_1_torus_volume.html#a1257580f5c26ae9ee6378065bbcf9bb3',1,'Go::TorusVolume::read()']]],
['read_5fbin',['read_bin',['../class_go_1_1_bspline_basis.html#a3e7de9080ad961e3f397ad1ed6bbb8fe',1,'
Go::BsplineBasis']],
['read_5fgl_5fmatrices',['read_gl_matrices',['../gl_aux_8h.html#aabb2eae1c53cf1b829442db5524cd900',1,'
read_gl_matrices(FILE *f):

```

Definition at line 1 of file functions\_12.js.

## 30.1512 doc/html/search/functions\_13.js File Reference

### Variables

- var [searchData](#)
- [sisl h](#)

### 30.1512.1 Variable Documentation

#### 30.1512.1.1 [sisl h](#)

Definition at line 3 of file functions\_13.js.

#### 30.1512.1.2 [var searchData](#)

#### Initial value:

```

=
[
 ['s1001', ['s1001', ['../sisl_8h.html#a1d7b3b4fe11df91a4b23bdd6995fda11',1,'s1001():

```

Definition at line 1 of file functions\_13.js.

## 30.1513 doc/html/search/functions\_14.js File Reference

### Variables

- var [searchData](#)
- PrParamUtil [h](#)
- PrParamUtil std::vector & [lt](#)
- GeneralMesh & [gt](#)
- & [amp](#)
- meshes [const](#)

### 30.1513.1 Variable Documentation

#### 30.1513.1.1 & amp

Definition at line 18 of file functions\_14.js.

#### 30.1513.1.2 meshes const

#### Initial value:

```
=0'], ['./class_go_1_1_composite_model.html#a1546764e88cba9c13fa2bea819e9fe2c'
,1,'Go::CompositeModel::tessellate(double density, std::vector&
lt; shared_ptr<GeneralMesh > > & amp; meshes) const =0'], ['./
class_go_1_1_curve_model.html#a5aa6f99880fd5126485b39a59932a30f',1,'
Go::CurveModel::tessellate(std::vector< shared_ptr<GeneralMesh > > &
amp; meshes) const ', ['./class_go_1_1_curve_model.html#
a2cc71ae48a6d1024c8779c1fde08eb80',1,'Go::CurveModel::tessellate(int resolution[], std::vector<
shared_ptr<GeneralMesh > > & amp; meshes) const ', ['./
class_go_1_1_curve_model.html#a13361c5bf5d010aa4da56ae5569070e0',1,'
Go::CurveModel::tessellate(double density, std::vector< shared_ptr<
GeneralMesh > > & amp; meshes) const ', ['./class_go_1_1ft_curve_segment.html#
ad3a6be9011d0466686c7095dd6e54f85',1,'Go::ftCurveSegment::tessellate()'], ['./
class_go_1_1ft_curve.html#a3fa46cd0cd1b6301996463d9fc95589c',1,'
Go::ftCurve::tessellate(std::vector< shared_ptr<LineStrip &
gt; > & amp; meshes) const ', ['./class_go_1_1ft_curve.html#
a2305a44a658989280b9877ac04600a19',1,'Go::ftCurve::tessellate(int resolution, std::vector<
shared_ptr<LineStrip > > & amp; meshes) const ', ['./class_go_1_1ft_curve.html#
af4679709df9d47b8f07fb7dbf82737b5',1,'Go::ftCurve::tessellate(double density,
std::vector< shared_ptr<LineStrip > > & amp; meshes) const ', ['./
class_go_1_1_int_results_comp_cv.html#a0a2c6122f9a871bf7b9666d2e0ad2a2f',1,
'Go::IntResultsCompCv::tessellate(std::vector< shared_ptr<LineStrip >
> & amp; meshes, PointCloud3D & amp; points) const ', ['./
class_go_1_1_int_results_comp_cv.html#aaa85f9ea710d6c6700b49456212c73ba',1,
'Go::IntResultsCompCv::tessellate(int resolution, std::vector< shared_ptr<
LineStrip > > & amp; meshes, PointCloud3D & amp; points) const ', ['./
class_go_1_1_int_results_comp_cv.html#a679437896e0247ae64a4375234f1e270',1,
'Go::IntResultsCompCv::tessellate(double density, std::vector< shared_ptr<
LineStrip > > & amp; meshes, PointCloud3D & amp; points) const ', ['./
class_go_1_1_int_results_model.html#af758fa08ald9a2e291102486e4029128',1,'
Go::IntResultsModel::tessellate(std::vector< shared_ptr<
LineStrip > > & amp; meshes, PointCloud3D & amp; points) const =0'], ['./
class_go_1_1_int_results_model.html#a5cba20336f32b65cfd840edbf8f46bbd',1,'
Go::IntResultsModel::tessellate(int resolution, std::vector< shared_ptr<
LineStrip > > & amp; meshes, PointCloud3D & amp; points) const =0'], ['./
class_go_1_1_int_results_model.html#aa65af6225021c30ea430fc9287191cd9',1,'
Go::IntResultsModel::tessellate(double density, std::vector< shared_ptr<
LineStrip > > & amp; meshes, PointCloud3D & amp; points) const =0'], ['./
class_go_1_1_int_results_sf_model.html#ab7833b7cdf1ab93b3635e2d68cea77e4'
,1,'Go::IntResultsSfModel::tessellate(std::vector< shared_ptr<LineStrip
> > & amp; meshes, PointCloud3D & amp; points) const ', ['./
class_go_1_1_int_results_sf_model.html#a0482f111c47cb1d5aa53dd87f7fa59f5'
,1,'Go::IntResultsSfModel::tessellate(int resolution, std::vector<
shared_ptr<LineStrip > > & amp; meshes, PointCloud3D & amp; points) const ', ['./
class_go_1_1_int_results_sf_model.html#a9f75cafb391badc9153f52c01393e207'
,1,'Go::IntResultsSfModel::tessellate(double density, std::vector<
shared_ptr<LineStrip > > & amp; meshes, PointCloud3D & amp; points) const ', ['./
class_go_1_1_surface_model.html#a499dcd53fc489e79193ec08a40000321',1,'
```

```

Go::SurfaceModel::tessellate(std::vector<& shared_ptr<
lt; GeneralMesh > > & meshes) const ', ['../
class_go_1_1_surface_model.html#a0cfa68b53d7d7ba43270ac0370434da2',1,'
Go::SurfaceModel::tessellate(int uv_res, std::vector<& shared_ptr<
GeneralMesh > & meshes) const ', ['../class_go_1_1_surface_model.html#a388420dafc3080e6681f70b8eb0d4f33',1,'Go::SurfaceModel::tessellate(int resolution[],
std::vector<& shared_ptr< GeneralMesh > & meshes) const ', ['../
class_go_1_1_surface_model.html#a0c7f814bd5bbbba9a376c490f0f06a85',1,'
Go::SurfaceModel::tessellate(double density, std::vector<& shared_ptr<
GeneralMesh > & meshes) const ', ['../class_go_1_1_surface_model.html#a
d71862c93633533cf9b92fd3999c512c',1,'Go::SurfaceModel::tessellate(const std::vector<&
shared_ptr< ftFaceBase > & faces, std::vector<
lt; shared_ptr< GeneralMesh > & meshes) const ', ['../
class_go_1_1_surface_model.html#ab8a38d3b31d12e3db759bc29328dbcb9',1,'
Go::SurfaceModel::tessellate(const std::vector<& shared_ptr< ftFaceBase >
& faces, int resolution[], std::vector<& shared_ptr< GeneralMesh > & meshes) const
', ['../class_go_1_1_surface_model.html#a2ffe542d114f0ae72313b5f99fd2b3d9',1,'
Go::SurfaceModel::tessellate(const std::vector<& shared_ptr<
lt; ftFaceBase > & faces, double density, std::vector<& shared_ptr<
lt; GeneralMesh > & meshes) const ', ['../
class_go_1_1_curve_tessellator.html#a2903e0b478f9c9bc52de4b78a2e58554',1,'
Go::CurveTessellator::tessellate()', ['../
class_go_1_1_line_cloud_tessellator.html#a741142d2fb183073f3999ebdb0987490
',1,'Go::LineCloudTessellator::tessellate()', ['../
class_go_1_1_noop_tessellator.html#a098bad44d4975404a9ecbf097a895423',1,'
Go::NoopTessellator::tessellate()', ['../
class_go_1_1_parametric_surface_tessellator.html#
af0a157fa8039d15ed49d25eccf1d6d8b',1,'Go::ParametricSurfaceTessellator::tessellate()
'], ['../class_go_1_1_rectangular_surface_tessellator.html#
a98bc65dc65f847c17602033717c41843',1,'Go::RectangularSurfaceTessellator::tessellate
()', ['../class_go_1_1_rect_grid_tessellator.html#
a37c7daci717250aae0fe930161e4406b',1,'Go::RectGridTessellator::tessellate()', ['../
class_go_1_1_tessellator.html#ac990c28ea9a36e37b26edd30cc523ac3',1,'
Go::Tessellator::tessellate()', ['../
class_go_1_1_rectangular_volume_tessellator.html#
a81d00516682f244ebac9097b6de6da78',1,'Go::RectangularVolumeTessellator::tessellate()
'], ['../class_go_1_1_volume_model.html#aaf3da8a531cbdl1a8e65853e2933abe00',1,'
Go::VolumeModel::tessellate(std::vector<& shared_ptr<
lt; GeneralMesh > & meshes) const ', ['../
class_go_1_1_volume_model.html#aef65bfbbaa2e8f5a66fc0b23630861b3',1,'
Go::VolumeModel::tessellate(int resolution[], std::vector<& shared_ptr<
GeneralMesh > & meshes) const ', ['../class_go_1_1_volume_model.html#
a3e034d7fa26dd97bb9cb8733e50c82c8',1,'Go::VolumeModel::tessellate(double density,
std::vector<& shared_ptr< GeneralMesh > & meshes) const ']]],
['tessellatedctrpolygon', ['tessellatedCtrPolygon', ['../
class_go_1_1_composite_curve.html#accf52036d9edefe396c90df496e8ea2e',1,'
Go::CompositeCurve::tessellatedCtrPolygon(std::vector<& shared_ptr<
lt; LineCloud > & ctr_pol) const ', ['../class_go_1_1_composite_curve
.html#a98d7786acc00fe2d2b642beaff493324',1,'
Go::CompositeCurve::tessellatedCtrPolygon(const std::vector<&
shared_ptr< ParamCurve > & curves, std::vector<& shared_ptr<
lt; LineCloud > & ctr_pol) const ', ['../
class_go_1_1_composite_model.html#a60e568eb9b8531f2bc5de60acb9d8fd7',1,'
Go::CompositeModel::tessellatedCtrPolygon()', ['../
class_go_1_1_curve_model.html#a0bc9f19a78ec7aaac8b655ea7e25cb7b',1,'
Go::CurveModel::tessellatedCtrPolygon()', ['../
class_go_1_1_surface_model.html#a42d3923e99fa793ca34de335767c53f1',1,'
Go::SurfaceModel::tessellatedCtrPolygon(std::vector<& shared_ptr<
lt; LineCloud > & ctr_pol) const ', ['../
class_go_1_1_surface_model.html#a81fcl1beb1ddd467c6fd34f44a4351938',1,'
Go::SurfaceModel::tessellatedCtrPolygon(const std::vector<&
shared_ptr< ftFaceBase > & faces, std::vector<& shared_ptr< LineCloud > & ctr_pol)
const ', ['../class_go_1_1_volume_model.html#ad7cfeb7a9a77a4f4d1d40e7aedef0de3',1,'
Go::VolumeModel::tessellatedCtrPolygon()']]]],
['tessellator', ['tessellator', ['../class_data_handler.html#
ab7caedcbf1e13f7456ce14b84c3b6789',1,'DataHandler::tessellator()', ['../classgv_data.html#
a56fb6a11bc06f0998bd20a67a9274836',1,'gvData::tessellator()']]]],
['test1', ['test1', ['../example_8cpp.html#ac0697ad85fb379446222fcc2df02c778',1,'
example.cpp']]],
['test2', ['test2', ['../example_8cpp.html#ac7865afb3b9f6d710b1df18e501c94d6',1,'
example.cpp']]],
['test3', ['test3', ['../example_8cpp.html#a727a32bdd7e2491824ca4acd62be9b1a',1,'
example.cpp']]],
['test4', ['test4', ['../example_8cpp.html#acd897830273f9673b3894e831ceb3423',1,'
example.cpp']]],
['test5', ['test5', ['../example_8cpp.html#aad4ba83ae697b892cf4711aa7170c1f3',1,'
example.cpp']]],
['test_5fcyclic_5fknots', ['test_cyclic_knots', ['../sisl_p_8h.html#
aac1f746eeb564d7f5eb39e1c36b5b6d0',1,'test_cyclic_knots():#160

```

Definition at line 18 of file functions\_14.js.

## 30.1513.1.3 &amp;gt;

Definition at line 18 of file functions\_14.js.

## 30.1513.1.4 sisIP h

Definition at line 13 of file functions\_14.js.

## 30.1513.1.5 shared\_ptr&amp; lt

Definition at line 13 of file functions\_14.js.

## 30.1513.1.6 var searchData

## Initial value:

```
=
[
 ['t', ['t', ['../class_n_e_w_m_a_t_l_l_matrix_type.html#a87734af76e2bd752f88e7b4976b52bbb', 1,
 'NEWMAT::MatrixType::t()'], ['../class_n_e_w_m_a_t_l_l_matrix_band_width.html#a9282c4cccc63ddeadc55483508c5c2794', 1,
 'NEWMAT::MatrixBandWidth::t()'], ['../class_n_e_w_m_a_t_l_l_base_matrix.html#a9b2e4b5ac85906a6c0007c40df340b33', 1,
 'NEWMAT::BaseMatrix::t()']],
 ['t1', ['t1', ['../class_pr_triangle.html#a3c5e90139c8ad3fe42bf807f5a68b7ba', 1, 'PrTriangle::t1() const'], [
 '../class_pr_triangle.html#a3b7f193f51f1ba9aceeeae8d414fld3a', 1, 'PrTriangle::t1()']],
 ['t2', ['t2', ['../class_pr_triangle.html#a3b4affb990eb31ec311069ee83b5b01b', 1, 'PrTriangle::t2() const'], [
 '../class_pr_triangle.html#a41d3422b3e374cba8ed081021eb43fa7', 1, 'PrTriangle::t2()']],
 ['t3', ['t3', ['../class_pr_triangle.html#a9aa253de3e236af87f94b8dc5c5cf19f', 1, 'PrTriangle::t3() const'], [
 '../class_pr_triangle.html#ac1063043a62da70ab3a33dfad452e1f2', 1, 'PrTriangle::t3()']],
 ['tag', ['Tag', ['../class_n_e_w_m_a_t_l_l_general_matrix.html#ac86f41d9beeda0ba938e8f777167b47', 1,
 'NEWMAT::GeneralMatrix']],
 ['tangent', ['tangent', ['../class_go_1_lft_curve_segment.html#ad3cf5a268861e7ed344ccf146e59d0457', 1,
 'Go::ftCurveSegment::tangent()'], ['../class_go_1_lft_curve.html#af97541ce68c9b3a6769d16d38b09267c', 1,
 'Go::ftCurve::tangent()'], ['../class_go_1_lft_edge.html#a62badabelbae46b1ba238ead31889f51', 1, 'Go::ftEdge::tangent()'], [
 '../class_go_1_lft_edge_base.html#aed5b154957fdd7a97df9726053c32eef', 1, 'Go::ftEdgeBase::tangent()'], [
 '../class_go_1_lft_s_sf_edge.html#aec5af9042c2a159c2b8dce97f88eb4ad', 1, 'Go::ftSSFEdge::tangent()'], [
 '../class_go_1_ltp_edge.html#ad854b7324cfe2d5ef2edc4342d97c267', 1, 'Go::tpEdge::tangent()']],
 ['tangentCone', ['tangentCone', ['../class_go_1_l_bounded_surface.html#ac311070548b96d789d21758e5ee7ec88', 1,
 'Go::BoundedSurface::tangentCone()'], [
 '../class_go_1_l_composite_surface.html#a9bfe05b31ca0bab1be6adb707586a18f', 1, 'Go::CompositeSurface::tangentCone()'], [
 '../class_go_1_l_cylinder.html#a583676f3507e5b9e5cb8d3cd9f2772f0', 1,
 'Go::Cylinder::tangentCone()'], [
 '../class_go_1_l_disc.html#a7b3943b053c0a274af05b8d5284cc9', 1,
 'Go::Disc::tangentCone()'], [
 '../class_go_1_l_param_surface.html#a0892308334265519c050b32f050e065e', 1,
 'Go::ParamSurface::tangentCone()'], [
 '../class_go_1_l_plane.html#a08c1d6807c5866963922e8265393fa62', 1, 'Go::Plane::tangentCone()'], [
 '../class_go_1_l_sphere.html#a3f02936582bc72857d2af3b1b9962c63', 1, 'Go::Sphere::tangentCone()'], [
 '../class_go_1_l_spline_surface.html#ac4f32c873766da700a3f78ebf5d95c37', 1, 'Go::SplineSurface::tangentCone()'], [
 '../class_go_1_l_surface_of_linear_extrusion.html#ac9cf10e65169359291347f3569eeadb2', 1,
 'Go::SurfaceOfLinearExtrusion::tangentCone()'], [
 '../class_go_1_l_surface_of_revolution.html#a07bc005c7579121b29e4ef1e0ed', 1,
 'Go::SurfaceOfRevolution::tangentCone()'], [
 '../class_go_1_l_torus.html#ac8849f0b6a5a7083c8286ebf83d8346f', 1,
 'Go::Torus::tangentCone()'], [
 '../class_go_1_l_r_spline_surface.html#acde98e9907ecaa6b5ba085540f50db94', 1,
 'Go::LRSplineSurface::tangentCone()'], [
 '../class_go_1_l_cone_volume.html#a1f7d1a81eae6e98a3422a2320aa932bd', 1,
 'Go::ConeVolume::tangentCone()'], [
 '../class_go_1_l_cylinder_volume.html#a4d5146512b6324e7616b54c3c7fb3a51', 1,
 'Go::CylinderVolume::tangentCone()'], [
 '../class_go_1_l_parallelepiped.html#a65b0a65033c8c65f6a2884fe2e6a11ca', 1,
 'Go::Parallelepiped::tangentCone()'], [
 '../class_go_1_l_param_volume.html#af5b51bc450da49f76b73f038618a57b6', 1,
 'Go::ParamVolume::tangentCone()'], [
 '../class_go_1_l_sphere_volume.html#a6f5a787cfalabfd98de924e4dc36f262', 1,
 'Go::SphereVolume::tangentCone()'], [
 '../class_go_1_l_spline_volume.html#ac01dec30b44329e7cc9d97a02f9c85d7', 1,
 'Go::SplineVolume::tangentCone()'], [
 '../class_go_1_l_surface_on_volume.html#a346db16aaa51c7e3cf525d4f4918bc2d', 1,
 'Go::SurfaceOnVolume::tangentCone()'], [
 '../class_go_1_l_torus_volume.html#a4e5d384ca79038f368e5f447959407c2', 1,
 'Go::TorusVolume::tangentCone()']],
 ['tangentIsoriented', ['tangentIsOriented', [
 '../class_go_1_l_intersection_point.html#a6de8bbeb86de15714906fb08e8469a27', 1, 'Go::IntersectionPoint']],
 ['tangentPointingInwards', ['tangentPointingInwards', [
 '../class_go_1_l_intersection_point.html#a7d2d0f72efc426b982994f0355860fda', 1, 'Go::IntersectionPoint']],
 ['tanLenFromradius', ['tanLenFromRadius', ['../namespace_go.html#a87059cc6cddba0b64ae502b9c4d52a52', 1, 'Go']
]],
 ['tanthetaovertwo', ['tanThetaOverTwo', [
 '../class_pr_prm_sym_mean_value.html#a9f07830f6d01f9430204d6b090da8823', 1, 'PrPrmSymMeanValue::tanThetaOverTwo()'], [
 '../class_pr_prm_wachspress.html#a3b09c67987bea5e0a6f2ac95755ca258', 1, 'PrPrmWachspress::tanThetaOverTwo()'], [
 '../pr_param_util_8h.html#acf689ef17a246312b527f1c2b8fal266', 1, 'tanThetaOverTwo() :
]]
```

Definition at line 1 of file functions\_14.js.

## 30.1514 doc/html/search/functions\_15.js File Reference

### Variables

- var [searchData](#)

#### 30.1514.1 Variable Documentation

##### 30.1514.1.1 var searchData

Definition at line 1 of file functions\_15.js.

## 30.1515 doc/html/search/functions\_16.js File Reference

### Variables

- var [searchData](#)

#### 30.1515.1 Variable Documentation

##### 30.1515.1.1 var searchData

Definition at line 1 of file functions\_16.js.

## 30.1516 doc/html/search/functions\_17.js File Reference

### Variables

- var [searchData](#)
- [gl\\_aux](#) `cpp`

#### 30.1516.1 Variable Documentation

##### 30.1516.1.1 [gl\\_aux](#) `cpp`

Definition at line 21 of file functions\_17.js.

## 30.1516.1.2 var searchData

## Initial value:

```

=
[
 ['weakly_5fdecreasing', ['weakly_decreasing', ['../namespace_go.html#a9ef55b273f5d603669d73063e84a435c', 1, 'Go::weakly_decreasing(Iterator begin, Iterator end)'], ['../namespace_go.html#af88f2fc0b462e4d9bf12c1361cd362ec', 1, 'Go::weakly_decreasing(const Array &A)']], ['weakly_5fincreasing', ['weakly_increasing', ['../namespace_go.html#af16aaec72dabb981744e338aef8c9748', 1, 'Go::weakly_increasing(Iterator begin, Iterator end)'], ['../namespace_go.html#ad520388adbe4b3719116f02ba3be0145', 1, 'Go::weakly_increasing(const Array &A)']], ['weedoutclutterpoints', ['weedOutClutterPoints', ['../class_go_1_1_intersection_pool.html#a7fala9d9968caefa52bb731b2b461451', 1, 'Go::IntersectionPool']], ['weight', ['weight', ['../class_go_1_1_l_r_b_spline2_d.html#aab4dc5282223c9bd0d6a3e9ee255fcf2', 1, 'Go::LRBSpline2D::weight()'], ['../class_go_1_1_l_r_b_spline2_d.html#a51952c9c74df2333392d465f8eaeddeb', 1, 'Go::LRBSpline2D::weight() const']], ['weightedl2norm', ['weightedL2Norm', ['../class_pr_threshold.html#a49faf4a4837b4986080824fe64df929c', 1, 'PrThreshold']], ['wg', ['WG', ['../class_n_e_w_m_a_t_1_1_l_l_d_f_i.html#ae6376b9b2bed94af9ca3f16abcf7926b', 1, 'NEWMAT::LL_DF_I']], ['what', ['what', ['../class_r_b_d_c_o_m_m_o_n_1_1_base_exception.html#aec4c2009cf2c58c624938fa307369812', 1, 'RBD_COMMON::BaseException']], ['whichboundary', ['whichBoundary', ['../class_go_1_1_curve_on_surface.html#a2fd1a9d6379d50a33c445de1e5a2da13', 1, 'Go::CurveOnSurface::whichBoundary()'], ['../class_go_1_1_rect_domain.html#a846d3147d576b8c7b1ddad3a5d282ee7', 1, 'Go::RectDomain::whichBoundary()'], ['../class_go_1_1_eval_param_curve.html#a1312bd05d7825d583db7c4a8b5656da5', 1, 'Go::SurfaceOnVolume::whichBoundary()']], ['whichcell', ['whichCell', ['../class_pr_cell_structure.html#a029bd0e3caec3c0041f84d4fbc9fde07', 1, 'PrCellStructure']], ['width', ['width', ['../classgv_texture.html#aa9078e4da5c899ca528cf48791788f71', 1, 'gvTexture']], ['willnotconverge', ['WillNotConverge', ['../tmti_8cpp.html#a0f921978f9992fd411af3b20cca52c04', 1, 'tmti.cpp']], ['wireframe', ['wireframe', ['../classgv_view.html#a063782599eaf127077e7fa3a824db0e9', 1, 'gvView']], ['worstjointtype', ['WorstJointType', ['../class_go_1_1_ft_curve.html#a852166efc18b7f0e8e5dbc53a980bce', 1, 'Go::ftCurve']], ['worststatus', ['WorstStatus', ['../struct_go_1_1_face_connectivity.html#a71455fb59609e8d2356ad33f1bc637cb', 1, 'Go::FaceConnectivity::WorstStatus()'], ['../struct_go_1_1_tp_topological_info.html#a7ef71026619e890f6f54fd349f780c9', 1, 'Go::tpTopologicalInfo::WorstStatus()']], ['write', ['write', ['../class_go_1_1_ft_curve.html#a28e19a4ecfb16faf241ea71be6fd0c94', 1, 'Go::ftCurve::write()'], ['../class_go_1_1_ft_point_set.html#ab934467685f4f9d4ed2717adeeeaa80a', 1, 'Go::ftPointSet::write()'], ['../class_go_1_1_ft_surface.html#a9ae863e4a138a7c4f6583946d9e42309', 1, 'Go::ftSurface::write()'], ['../class_go_1_1_eval_curve.html#a4ff86322f9dc39ffcc4620ee477a2cele', 1, 'Go::EvalCurve::write()'], ['../class_go_1_1_eval_param_curve.html#a1e4877d32b0ce7cac403583c1710511c', 1, 'Go::EvalParamCurve::write()'], ['../class_go_1_1_eval_surface.html#a04f0ae2e984ad26c9fea650a5f0f54f5', 1, 'Go::EvalSurface::write()'], ['../class_go_1_1_bounded_curve.html#aceacbec37facea840a5fcc22768a2d51', 1, 'Go::BoundedCurve::write()'], ['../class_go_1_1_bounded_surface.html#a856ef400fd0c6d1cac0d7810dcae2895', 1, 'Go::BoundedSurface::write()'], ['../class_go_1_1_bspline_basis.html#af4fd78fee6e6e4b5d21a95b5b72643e9', 1, 'Go::BsplineBasis::write()'], ['../class_go_1_1_circle.html#a3c63191c45406a20fc2ab26416f141c7', 1, 'Go::Circle::write()'], ['../class_go_1_1_composite_surface.html#a709afaea27653a43009241ad48e3095c', 1, 'Go::CompositeSurface::write()'], ['../class_go_1_1_cone.html#ad26085ced7959788b713269ce155d7d7', 1, 'Go::Cone::write()'], ['../class_go_1_1_curve_on_surface.html#a7cfe9696a1e3b0254a1304f5594ba091', 1, 'Go::CurveOnSurface::write()'], ['../class_go_1_1_cylinder.html#a6e58f17e86868902f26a6e43727736b8', 1, 'Go::Disc::write()'], ['../class_go_1_1_ellipse.html#ad3984e926cad3799218ed3e9e3018eea', 1, 'Go::Ellipse::write()'], ['../class_go_1_1_hyperbola.html#abaa712776a6c103534bc95359a444cbf', 1, 'Go::Hyperbola::write()'], ['../class_go_1_1_line.html#a86230302a80829759af8cfe337bfe2c5', 1, 'Go::Line::write()'], ['../class_go_1_1_line_cloud.html#ad0108951f753e802c0deb441b83ec4b6', 1, 'Go::LineCloud::write()'], ['../class_go_1_1_object.html#a68d31e1edelf511c1ebc4fd4c5d47a65', 1, 'Go::ObjectHeader::write()'], ['../class_go_1_1_parabola.html#a68d31e1edelf511c1ebc4fd4c5d47a65', 1, 'Go::Parabola::write()'], ['../class_go_1_1_plane.html#ac2fea031c1034cd8f0b3b2c5a6880f95', 1, 'Go::Plane::write()'], ['../class_go_1_1_point_cloud.html#ad89fec212c58f01559025679842c0c1f', 1, 'Go::PointCloud::write()'], ['../class_go_1_1_rect_grid.html#aff9c01ccc35a6db444c68e8c62e31335', 1, 'Go::RectGrid::write()'], ['../class_go_1_1_sphere.html#aa7d16b5e1fb2b26c4534f923c2b4b2f7', 1, 'Go::Sphere::write()'], ['../class_go_1_1_spline_curve.html#a923930edaab997b412c189cf42fcfcf', 1, 'Go::SplineCurve::write()'], ['../class_go_1_1_spline_surface.html#abd6298134ce0e71ad562aac7bf320b0f', 1, 'Go::SplineSurface::write()'], ['../class_go_1_1_streamable.html#ae8180cc0ef0185251251e1e48714ad79', 1, 'Go::Streamable::write()'], ['../class_go_1_1_surface_of_linear_extrusion.html#a2c3db5e09491b23b5978063c3db8a01e', 1, 'Go::SurfaceOfLinearExtrusion::write()'], ['../class_go_1_1_surface_of_revolution.html#a6fb097b8370423f05f42b4b141583e5', 1, 'Go::SurfaceOfRevolution::write()'], ['../class_go_1_1_torus.html#ae82866130ed2fa422bd453ac87fea31e', 1, 'Go::Torus::write()'], ['../class_go_1_1_array.html#a521b4345783ea16d1e689408c464994b', 1, 'Go::Array::write()'], ['../class_go_1_1_bary_coord_system.html#alcab0342adaec3d4a49009166490663b', 1, 'Go::BaryCoordSystem::write()'], ['../class_go_1_1_boundingbox.html#abc9e743afa6168a129d765dclb44014e', 1, 'Go::BoundingBox::write()'], ['../class_go_1_1_composite_box.html#abc9e743afa6168a129d765dclb44014e', 1, 'Go::CompositeBox::write()'], ['../class_go_1_1_direction_cone.html#aa60780e28b9bec47b945b0bf5b1e8764', 1, 'Go::DirectionCone::write()'], ['../class_go_1_1_point.html#a0b569e805513613f8ad877b65756c523', 1, 'Go::Point::write()'], ['../class_go_1_1_rational.html#ac7152300ba2fd30d30012851ae15fef', 1, 'Go::Rational::write()'], ['../class_go_1_1_bernstein_multi.html#a6eab04ad3bead441043dd3671d901c1', 1, 'Go::BernsteinMulti::write()'], ['../class_go_1_1_bernstein_poly.html#aeebfff34e228191fac817504465db7e49', 1, 'Go::BernsteinPoly::write()'], ['../class_go_1_1_bernstein_tetrahedral_poly.html#ada755f1820d75b14415a7f269e8acb', 1, 'Go::BernsteinTetrahedralPoly::write()'], ['../class_go_1_1_bernstein_triangular_poly.html#ae9150fb59d5da6a36c8ab6510e8af', 1, 'Go::BernsteinTriangularPoly::write()'], ['../class_go_1_1_geo_tol.html#a442a98b53da78dc97f9d82c27e8fcel', 1, 'Go::GeoTol::write()'], ['../class_go_1_1_interpolated_intersection_curve.html#aaf65f6040f556d1b2c85846ae264c15b', 1, 'Go::InterpolatedIntersectionCurve::write()'], ['../class_go_1_1_intersection_point.html#a006203e0049242d58e006769b3d2d40', 1, 'Go::IntersectionPoint::write()'], ['../class_go_1_1_l_r_b_spline2_d.html#a449462ba765afcac0976afa68f2e2265', 1, 'Go::LRBSpline2D::write()'], ['../class_go_1_1_l_r_spline_surface.html#ab8bfbd2d2931febdf7b71f659f693b8', 1, 'Go::LRBSplineSurface::write()'], ['../class_go_1_1_cone_volume.html#ad8a963e0086079894fcb89e6d6eb86c7', 1, 'Go::Mesh2D::write()'],
]

```

```

Go::ConeVolume::write()'], ['./class_go_1_1_curve_on_volume.html#a7dbdc8777f92ae2326b8f31c4f75b3cc', 1, '
Go::CurveOnVolume::write()'], ['./class_go_1_1_cylinder_volume.html#a44d169907ccb3ff1a30daadd7bdc61bf', 1, '
Go::CylinderVolume::write()'], ['./class_go_1_1_parallelepiped.html#ada46eff243691531b4b0885515be7e1b', 1, '
Go::Parallelepiped::write()'], ['./class_go_1_1_sphere_volume.html#af39ffdf622b868ef6cc6d7915b28bade', 1, '
Go::SphereVolume::write()'], ['./class_go_1_1_spline_volume.html#a3306204b720e8fb1a3ce6c23b03ef5b0', 1, 'Go::SplineVolume
 ['./class_go_1_1_surface_on_volume.html#a86cc60540a2a14233bd2485824f5bbe4', 1, 'Go::SurfaceOnVolume::write()
], ['./class_go_1_1_torus_volume.html#a6add375e29268f3fcfa51fb5335401b4', 1, 'Go::TorusVolume::write()']]],
['write2d', ['write2D', ['./class_go_1_1_ft_point_set.html#adf8f9758c80d10b2eaf24dfe05f02468', 1, '
Go::ftPointSet']],
['write2dval', ['write2Dval', ['./class_go_1_1_ft_sample_point.html#a698415101e4d4bd3cca0ffddc6d8f282', 1, '
Go::ftSamplePoint::write2Dval()'], [
./class_go_1_1_ft_surface_set_point.html#a92b64340d0c99eb8523bf9dca4777ee9', 1, 'Go::ftSurfaceSetPoint::write2Dval()']]],
['write_5fbin', ['write_bin', ['./class_go_1_1_bspline_basis.html#ab5d495e50702146c985bbe5af2420c34', 1, '
Go::BsplineBasis']],
['write_5fgl_5fmatrices', ['write_gl_matrices', ['./gl__aux_8h.html#a9f517cb0717bfbcb4374fb1fad9a450', 1, '
write_gl_matrices(FILE *f):

```

Definition at line 1 of file functions\_17.js.

## 30.1517 doc/html/search/functions\_18.js File Reference

### Variables

- var [searchData](#)

### 30.1517.1 Variable Documentation

#### 30.1517.1.1 var searchData

#### Initial value:

```

=
[
 ['x', ['x', ['./classhetriang_1_1_dart.html#a54e0bbd87448ac2b3e9bddd8db856c59', 1, 'hetriang::Dart::x()'], [
./class_go_1_1_tttl_point.html#a2edfd5b3a2a64c8c3af3e02c06ae2357', 1, 'Go::tttlPoint::x()'], [
./classhetriang_1_1_node.html#a6d6619cfedaeafeaf9e468da7670da5f', 1, 'hetriang::Node::x()'], [
./class_go_1_1_array.html#a3c8aa548649a570e01713ce17c1d784c', 1, 'Go::Array::x() const'], [
./class_go_1_1_array.html#ac87b81b08bef59f57cc13c5531115110', 1, 'Go::Array::x()'], ['./class_pr_nested_node.html#a69747e
PrNestedNode::x() const'], ['./class_pr_nested_node.html#a613b6c4885ec0097d356410119e8a4b9', 1, 'PrNestedNode::x()
 ['./class_pr_node.html#a5a5865db2f78a4ccfd762634b4899e39', 1, 'PrNode::x() const'], [
./class_pr_node.html#af1089d11e7ee670efb2007351752c1b6', 1, 'PrNode::x()'], [
./class_pr_p_g_node.html#af56272e39fbc9e4b0273b83d9d9aa970', 1, 'PrPGNode::x() const'], ['./class_pr_p_g_node.html#a373e
PrPGNode::x()'], ['./classvector3t.html#a85c1f558e7e9d6800afa25b4d95b4e1f', 1, 'vector3t::x(void) const'], [
./classvector3t.html#a4e3bd29a46e1bc19dc4a06313829e6a', 1, 'vector3t::x(void)'], [
./classhed_1_1_dart.html#a39d20327f3eeaeefbc53f06f79f53d840', 1, 'hed::Dart::x()'], ['./classhed_1_1_node.html#ae5240e904
1, 'hed::Node::x()']]],
 ['xyzdata', ['xyzData', ['./class_pr_rectangular_grid__o_p.html#a13ccc3b90a2db2fd3622747f808ca320', 1, '
PrRectangularGrid_OP::xyzData()'], ['./class_pr_rectangular_grid__o_p.html#a2638cc8d320b621128c9f7c8147dcb7d',
1, 'PrRectangularGrid_OP::xyzData() const']]
]

```

Definition at line 1 of file functions\_18.js.

## 30.1518 doc/html/search/functions\_19.js File Reference

### Variables

- var [searchData](#)



### 30.1518.1 Variable Documentation

#### 30.1518.1.1 var searchData

##### Initial value:

```
=
[
 ['y', 'y', ['../classhetriang_l_1_dart.html#af210f3f33a71b195c745e54cd7165070', 1, 'hetriang::Dart::y()'], ['
 ../class_go_l_1ttl_point.html#aa68070e89bc646943dcfff514e3d194f', 1, 'Go::ttlPoint::y()'], ['
 ../classhetriang_l_1_node.html#a9e14e02fa7d2ec93a21f26108a6d1b36', 1, 'hetriang::Node::y()'], ['
 ../class_go_l_1_array.html#a7503e9570b4c3e5df9d7c6a2f2882df4', 1, 'Go::Array::y() const'], ['
 ../class_go_l_1_array.html#ad343c6549d1fd8aae5abfb88ae509cf6', 1, 'Go::Array::y()'], ['../class_pr_nested_node.html#ac2aa9
 PrNestedNode::y() const'], ['../class_pr_nested_node.html#a437a7f299f4aeb739d3c86fd434a5673', 1, 'PrNestedNode::y()']
], ['../class_pr_node.html#ae837126218bc3668de1896a232be37bf', 1, 'PrNode::y() const'], ['
 ../class_pr_node.html#a5aa29384b673ee37acbd1cf7f95059e5', 1, 'PrNode::y()'], ['
 ../class_pr_p_g_node.html#a73c1340c393a2d29be3ab2aca97dfb88', 1, 'PrPGNode::y() const'], ['../class_pr_p_g_node.html#acf6
 PrPGNode::y()'], ['../classvector3t.html#ab0f2c22060b6068c3104d8c6992a8b89', 1, 'vector3t::y(void) const'], ['
 ../classvector3t.html#af88d3df5d1b648785874485afc10a876', 1, 'vector3t::y(void)'], ['
 ../classhed_l_1_dart.html#a20849660f038888f05ba65daf0a99ae5', 1, 'hed::Dart::y()'], ['../classhed_l_1_node.html#ab71715188
 1, 'hed::Node::y()']]
]
```

Definition at line 1 of file functions\_19.js.

## 30.1519 doc/html/search/functions\_1a.js File Reference

### Variables

- var [searchData](#)

### 30.1519.1 Variable Documentation

#### 30.1519.1.1 var searchData

##### Initial value:

```
=
[
 ['z', 'z', ['../class_go_l_1ttl_point.html#ac3852f5d65cea6fc03ded418c52b1224', 1, 'Go::ttlPoint::z()'], ['
 ../classhetriang_l_1_node.html#a3a3e75bffc96f2230699b7ec641afa70', 1, 'hetriang::Node::z()'], ['
 ../class_go_l_1_array.html#aae59550f97c7b789f9fedfef6214fca4', 1, 'Go::Array::z() const'], ['
 ../class_go_l_1_array.html#ab13fabbb125a68879f994e0c0a492755b', 1, 'Go::Array::z()'], ['
 ../class_pr_nested_node.html#ae5e6fab6058c33f1c08c5e7b335502ba', 1, 'PrNestedNode::z() const'], ['../class_pr_nested_node
 PrNestedNode::z()'], ['../class_pr_node.html#ad6eec21be11c8ba6ee3a0e4fe9d4693a', 1, 'PrNode::z() const'], ['
 ../class_pr_node.html#a323e95d93b0be8200fbfb5cf0b98afc1', 1, 'PrNode::z()'], ['
 ../class_pr_p_g_node.html#a302fb7445d4022c990244d61768aaa22', 1, 'PrPGNode::z() const'], ['
 ../class_pr_p_g_node.html#aab8d9580b6112f5342804c503aa5aa78', 1, 'PrPGNode::z()'], ['../classvector3t.html#ac756398d08e462
 '], ['../classvector3t.html#a40e10484bca97074094c63c9ea3f458b', 1, 'vector3t::z(void)'], ['
 ../classhed_l_1_node.html#a812130c3e029cdec9bc9a47a1e53a7', 1, 'hed::Node::z()']],
 ['zero', 'zero', ['../class_go_l_1_array.html#a23d28fcf202baff8a7c9f1aa5d9bfdd0', 1, 'Go::Array::zero()'], ['
 ../class_go_l_1_matrix_x_d.html#ac360b2abe7e401969db7516ce266e416', 1, 'Go::MatrixXD::zero()'], ['
 ../class_matrix_row_col.html#a1f6eebf4201d2b85d99c9273cb05b235', 1, 'MatrixRowCol::Zero()'], ['
 ../class_rect_matrix_row_col.html#a935c0ec5f6ed575e62f2a52a3ce5ef12', 1, 'RectMatrixRowCol::Zero()']],
 ['zeroSegments', 'zeroSegments', ['../class_go_l_1_mesh2_d.html#a0a9bd1d4ae3374e0b17b60673b9507f4', 1, '
 Go::Mesh2D']]
]
```

Definition at line 1 of file functions\_1a.js.



## 30.1520 doc/html/search/functions\_1b.js File Reference

### Variables

- var [searchData](#)

#### 30.1520.1 Variable Documentation

##### 30.1520.1.1 var searchData

Definition at line 1 of file functions\_1b.js.

## 30.1521 doc/html/search/functions\_2.js File Reference

### Variables

- var [searchData](#)
- [jquery js](#)

#### 30.1521.1 Variable Documentation

##### 30.1521.1.1 jquery js

Definition at line 3 of file functions\_2.js.

##### 30.1521.1.2 var searchData

#### Initial value:

```
=
[
 ['b', ['b', ['../class_go_1_1_line2_d_int.html#a1bf14760b8c49e8b71cd24fa7a38cd35', 1, 'Go::Line2DInt::b()'], [
 '../class_go_1_1_plane_int.html#aa61eda35ce195ebbb59321069d0b20f1', 1, 'Go::PlaneInt::b()'], [
 '../jquery_8js.html#a2fa551895933fae935a0a6b87282241d', 1, 'b():
```

Definition at line 1 of file functions\_2.js.

## 30.1522 doc/html/search/functions\_3.js File Reference

### Variables

- var [searchData](#)
- cholesky [cpp](#)
- cholesky shared\_ptr & [lt](#)
- IntersectionPoint & [gt](#)
- & [amp](#)
- pnt [const](#)
- [dist](#)
- [cpos1](#)
- double [gpos2](#) []
- [pt\\_cv](#)
- [pt\\_su](#)
- bool [second\\_order](#)
- [basisValues](#)
- [basisDerivs\\_u](#)
- [basisDerivs\\_v](#)
- bool [evaluate\\_from\\_right](#)
- [sisl h](#)
- [sisl int nmb\\_cvs](#)
- [init\\_basis](#)
- double [tol](#)
- double int double double [degree](#)

### 30.1522.1 Variable Documentation

30.1522.1.1 [sisl int const BsplineBasis& amp](#) [[property](#)], [[bound](#)]

Definition at line 106 of file [functions\\_3.js](#).

30.1522.1.2 [basisDerivs\\_u](#)

Definition at line 170 of file [functions\\_3.js](#).

30.1522.1.3 [basisDerivs\\_v](#)

Definition at line 170 of file [functions\\_3.js](#).

30.1522.1.4 [basisValues](#)

Definition at line 170 of file [functions\\_3.js](#).

30.1522.1.5 [pnt const](#)

Definition at line 106 of file [functions\\_3.js](#).

## 30.1522.1.6 cpos1

Definition at line 119 of file functions\_3.js.

## 30.1522.1.7 newmat4 cpp

Definition at line 74 of file functions\_3.js.

## 30.1522.1.8 double int double double degree

## Initial value:

```
=3)'], ['../namespace_go_1_1_curve_creators.html#a8dc25e7453ec9dcab07d0f51a868dea5', 1, '
Go::CurveCreators::curveApprox(shared_ptr<t; ParamCurve &
gt; cvs[], int nmb_cvs, const BsplineBasis &init_basis, double
tol)'], ['../namespace_go_1_1_curve_creators.html#a7f3047d4b2671a0942d339addb2d3220', 1, '
Go::CurveCreators::curveApprox(shared_ptr<t; ParamCurve &
gt; cvs[], int nmb_cvs, double tol, double degree=3)']],
['curveboundeddomain', ['CurveBoundedDomain', ['../
class_go_1_1_curve_bounded_domain.html#a18a47f463be561c4b9159f9a48b75810',
1, 'Go::CurveBoundedDomain::CurveBoundedDomain()'], ['../
class_go_1_1_curve_bounded_domain.html#a64b7a200c859eed5197f0690322577f7',
1, 'Go::CurveBoundedDomain::CurveBoundedDomain(std::vector<
t; shared_ptr<t; CurveLoop &t; &t; loops)'], ['../
class_go_1_1_curve_bounded_domain.html#af9d5bb9de6ebc645029186e7d60e8639',
1, 'Go::CurveBoundedDomain::CurveBoundedDomain(shared_ptr<t;
CurveLoop &t; ccw_loop)']],
['curvecreationmethod', ['curveCreationMethod', ['../
class_go_1_1_curve_on_surface.html#a8ca063ce84c44382ec718fel7d149649', 1, '
Go::CurveOnSurface']],
['curvekinks', ['curveKinks', ['../namespace_go_1_1_geometry_tools.html#
adec58c3980fd345194b3d2c38cb5410f', 1, 'Go::GeometryTools']],
['curveloop', ['CurveLoop', ['../class_go_1_1_curve_loop.html#
ale1cb95e731e6406e2e85ef158474f74', 1, 'Go::CurveLoop::CurveLoop()'], ['../
class_go_1_1_curve_loop.html#ade5272d85ed158bc981182df67c4aef4', 1, '
Go::CurveLoop::CurveLoop(const std::vector<t; shared_ptr<
t; ParamCurve &t; &t; &t; curves, double space_epsilon)']],
['curvemodel', ['CurveModel', ['../class_go_1_1_curve_model.html#
a28e80476530ffdfef76de33d447d385f', 1, 'Go::CurveModel']],
['curveonelement', ['curveOnElement', ['../class_go_1_1_element2_d.html#
a30f6705a5cb2c1ecf2d342b20290b903', 1, 'Go::Element2D']],
['curveonsurface', ['CurveOnSurface', ['../class_go_1_1_curve_on_surface.html#
acd1e4079fbb5207f7b58ecf0f02b1e639', 1, 'Go::CurveOnSurface::CurveOnSurface()
'], ['../class_go_1_1_curve_on_surface.html#a36f4291f5a7488e0e7650d9e94ae8d7a',
1, 'Go::CurveOnSurface::CurveOnSurface(shared_ptr<t; ParamSurface &
gt; surf, shared_ptr<t; ParamCurve &t; curve, bool preferparameter)'], ['../
class_go_1_1_curve_on_surface.html#aac96baf244fd948975fb121ab9ba96ce', 1, '
Go::CurveOnSurface::CurveOnSurface(shared_ptr<t; ParamSurface &t; surf,
shared_ptr<t; ParamCurve &t; curve, int constdir, double constpar, int boundary)'], ['../
class_go_1_1_curve_on_surface.html#af983a62292c58fe6ad3cb8c64a607031', 1, '
Go::CurveOnSurface::CurveOnSurface(shared_ptr<t; ParamSurface &
gt; surf, int constdir, double constpar, double par1, double par2, int boundary)'], ['../
class_go_1_1_curve_on_surface.html#ad08771990f783498eb9ba840bfb5e01a', 1, '
Go::CurveOnSurface::CurveOnSurface(shared_ptr<t; ParamSurface &t; surf,
shared_ptr<t; ParamCurve &t; pcurve, shared_ptr<t; ParamCurve &t; spacecurve, bool preferparameter, int
ccm, int constdir, double constpar, int boundary, bool same_orientation)'], ['../
class_go_1_1_curve_on_surface.html#a213980bda5fd3946878a81793805a929', 1, '
Go::CurveOnSurface::CurveOnSurface(shared_ptr<t; ParamSurface &
gt; surf, shared_ptr<t; ParamCurve &t; pcurve, shared_ptr<t; ParamCurve &
gt; spacecurve, bool preferparameter, int ccm=0)'], ['../
class_go_1_1_curve_on_surface.html#a59f4bd292ec36e5ad97c8458088bb1f1', 1, '
Go::CurveOnSurface::CurveOnSurface(const CurveOnSurface &
&surf, surface_curve)']],
['curveonvolume', ['CurveOnVolume', ['../class_go_1_1_curve_on_volume.html#
a7f86d41b6721769dde6f4cce99b97272', 1, 'Go::CurveOnVolume::CurveOnVolume()'], ['../
class_go_1_1_curve_on_volume.html#a7c1f620e6b106f8d99660a7d0a56f622', 1, '
Go::CurveOnVolume::CurveOnVolume(shared_ptr<t; ParamVolume &
gt; vol, shared_ptr<t; ParamCurve &t; curve, bool preferparameter)'], ['../
class_go_1_1_curve_on_volume.html#a226398d5c1319bfb0cc72759849328f1', 1, '
Go::CurveOnVolume::CurveOnVolume(shared_ptr<t; ParamVolume &t; vol,
shared_ptr<t; ParamCurve &t; pcurve, shared_ptr<t; ParamCurve &t; spacecurve, bool preferparameter)'], ['../
class_go_1_1_curve_on_volume.html#a98e604b39927a02be47981cde9de89ea', 1, '
Go::CurveOnVolume::CurveOnVolume(const CurveOnVolume &
&volume_curve)']],
```

```
[
 ['curveresolutionsheet', ['CurveResolutionSheet', ['../
 class_curve_resolution_sheet.html#abc796a029f0fb49851c335326ad945eb', 1, '
 CurveResolutionSheet']],
 ['curvesum', ['curveSum', ['../namespace_go_1_1_geometry_tools.html#
 a0ce6c0626f27496cac39684f76e88d51', 1, 'Go::GeometryTools']],
 ['curvetesselator', ['CurveTesselator', ['../class_go_1_1_curve_tesselator.
 html#aefe53d0a6b1795aefd01855793a80ffb', 1, 'Go::CurveTesselator']],
 ['curvetype', ['curveType', ['../class_go_1_1_ft_curve.html#
 a81f5700c76fe33118e3f0903870b8d5d', 1, 'Go::ftCurve']],
 ['cvc1gl1discontinuity', ['cvC1G1Discontinuity', ['../
 class_go_1_1_face_set_quality.html#af9e36bad3c4c3ffee440ed3a72433140', 1, '
 Go::FaceSetQuality::cvC1G1Discontinuity()'], ['../
 class_go_1_1_model_quality.html#ac31b1f14d4d87271276fa5c048a23055', 1, '
 Go::ModelQuality::cvC1G1Discontinuity()']],
 ['cvcurvatureradius', ['cvCurvatureRadius', ['../class_go_1_1_face_set_quality
 .html#a09bf24af1506c5d26becdb67afc0fa8e', 1, 'Go::FaceSetQuality::cvCurvatureRadius
 ()'], ['../class_go_1_1_model_quality.html#a894e9d602c1ab82c6102b19f45498edf', 1, '
 Go::ModelQuality::cvCurvatureRadius()']],
 ['cvcvintersector', ['CvCvIntersector', ['../class_go_1_1_cv_cv_intersector.
 html#a537c5e9bc036e51d6cd9ef73868d1866', 1, 'Go::CvCvIntersector::CvCvIntersector
 (shared_ptr< ParamGeomInt > curve1, shared_ptr< ParamGeomInt > curve2, double
 epsge, Intersector *prev=0, int eliminated_parameter=-1, double eliminated_value=0)'], ['../
 class_go_1_1_cv_cv_intersector.html#ae8389b331a736a88309a72d3d17ac79e', 1, '
 Go::CvCvIntersector::CvCvIntersector(shared_ptr< ParamGeomInt >
 curve1, shared_ptr< ParamGeomInt > curve2, shared_ptr< GeoTol > epsge, Intersector *prev=0, int
 eliminated_parameter=-1, double eliminated_value=0)']],
 ['cvcptintersector', ['CvPtIntersector', ['../class_go_1_1_cv_pt_intersector.
 html#afeedc2980f108a85ccaee64927e647a8', 1, 'Go::CvPtIntersector::CvPtIntersector
 (shared_ptr< ParamGeomInt > obj1, shared_ptr< ParamGeomInt > obj2, shared_ptr&
 lt; GeoTol > epsge, Intersector *prev=0, int eliminated_parameter=-1, double eliminated_value=0
)'], ['../class_go_1_1_cv_pt_intersector.html#a2c2d2af25116cb8064076dc17a797c19
 ', 1, 'Go::CvPtIntersector::CvPtIntersector(shared_ptr< ParamGeomInt &
 gt; obj1, shared_ptr< ParamGeomInt > obj2, double epsge, Intersector *prev=0, int eliminated_parameter=
 -1, double eliminated_value=0)']],
 ['cvsetconstraint', ['cvSetConstraint', ['../struct_go_1_1cv_set_constraint.
 html#a941d33cdfeda3a108633b734725ebd83', 1, 'Go::cvSetConstraint']],
 ['cworientation', ['cwOrientation', ['../namespace_go_1_1cm_utils.html#
 a0e8d24cd5737086fc3e72524d818d2d6', 1, 'Go::cmUtils']],
 ['cworientation2', ['cwOrientation2', ['../namespace_go_1_1cm_utils.html#
 a08a484efc37055f021aa7673e9e751c6', 1, 'Go::cmUtils']],
 ['cylinder', ['Cylinder', ['../class_go_1_1_cylinder.html#
 a8b175477b2d696563063e44b56ae094f', 1, 'Go::Cylinder::Cylinder()'], ['../
 class_go_1_1_cylinder.html#a4dbfcfc7962f8c87d4e63ae2f93a3929', 1, '
 Go::Cylinder::Cylinder(double radius, Point location,
 Point z_axis, Point x_axis, bool isSwapped=false)']],
 ['cylinderint', ['CylinderInt', ['../class_go_1_1_cylinder_int.html#
 a2578453e0bbb9b5981ed8520be74eef4', 1, 'Go::CylinderInt::CylinderInt()'], ['../
 class_go_1_1_cylinder_int.html#a9067b1a615e16beb0781fd491f2393f0', 1, '
 Go::CylinderInt::CylinderInt(Point ax_pt, Point ax_dir, double radius
)']],
 ['cylindervolume', ['CylinderVolume', ['../class_go_1_1_cylinder_volume.html#
 a33ba420957082fdb531a2ffff19a610d6', 1, 'Go::CylinderVolume::CylinderVolume()
 '], ['../class_go_1_1_cylinder_volume.html#a067ee920433858475d5d9f144be8564b', 1, '
 Go::CylinderVolume::CylinderVolume(Point centre, double radius,
 Point normal, Point x_axis)']]
]
```

Definition at line 308 of file functions\_3.js.

### 30.1522.1.9 dist

Definition at line 111 of file functions\_3.js.

### 30.1522.1.10 bool evaluate\_from\_right

Definition at line 170 of file functions\_3.js.

### 30.1522.1.11 double gpos2[]

Definition at line 119 of file functions\_3.js.

**30.1522.1.12** `double& gt`

Definition at line 106 of file functions\_3.js.

**30.1522.1.13** `sisl h`

Definition at line 230 of file functions\_3.js.

**30.1522.1.14** `init_basis`

Definition at line 308 of file functions\_3.js.

**30.1522.1.15** `std::vector& lt [bound]`

Definition at line 74 of file functions\_3.js.

**30.1522.1.16** `double int nmb_cvs [property], [bound]`

Definition at line 230 of file functions\_3.js.

**30.1522.1.17** `pt_cv`

Definition at line 119 of file functions\_3.js.

**30.1522.1.18** `pt_su`

Definition at line 119 of file functions\_3.js.

**30.1522.1.19** `var searchData`

Definition at line 1 of file functions\_3.js.

## 30.1522.1.20 bool second\_order

## Initial value:

```

=false)'], ['../namespace_go_1_1_closest_point.html#afeee87804e5a35525c7017d7fcc95c03', 1, '
 Go::ClosestPoint::closestPtCurveSurf(ParamCurve *pcurve, ParamSurface *
 psurf, double aepsge, double astartl, double aend2[], double aendl, double aend2[], double anextl,
 double enext2[], double &cpos1, double gpos2[], double &
 dist, Point &pt_cv, Point &pt_su, int &istat, bool
 second_order=false)']],
['closestptsurfsurfplane', ['closestPtSurfSurfPlane', ['../
 namespace_go_1_1_closest_point.html#a9bbf94f33a5d46b1f7fda8aa0d6bea', 1, 'Go::ClosestPoint']],
['closestptsurfsurfplanefunctional', ['closestPtSurfPlaneFunctional', [
 '../namespace_go_1_1_closest_point.html#a789a3fc2d43b9234327ebf3bbd3e88c3', 1, '
 Go::ClosestPoint']],
['closestptsurfsurfplanegeometrical', ['closestPtSurfSurfPlaneGeometrical
 ', ['../namespace_go_1_1_closest_point.html#a6788ce489922788dc484c43db18c94d4', 1, '
 Go::ClosestPoint']],
['closestsigneddistances', ['closestSignedDistances', ['../
 namespace_go.html#a8c08c7f6202af07431b1c5a39241e4a2', 1, 'Go']],
['closestvectorsold', ['closestVectorsOld', ['../namespace_go.html#
 a26d674d4f146b5604e861ca42c7e4a1f', 1, 'Go']],
['closetoscaledpoint', ['closeToScaledPoint', ['../class_go_1_1_ft_line.html#
 a3d7867a61227b67f16c611e57fa38733', 1, 'Go::ftLine']],
['closetounderlyingboundary', ['closeToUnderlyingBoundary', ['../
 class_go_1_1_bounded_surface.html#aa88b5aba06b7762233032a979c88478a', 1, '
 Go::BoundedSurface']],
['cndiscontinuities', ['cNDiscontinuities', ['../class_go_1_1_bspline_basis.html#
 aef46283739907138c5e742e911ba8b76', 1, 'Go::BsplineBasis']],
['coef', ['Coef', ['../class_go_1_1_r_b_spline2_d.html#
 aa763179a5d7ae4d7eba2a7a9233695dd', 1, 'Go::LRBSpline2D::Coef()'], ['../
 class_go_1_1_r_b_spline2_d.html#af30001b192b985a780aad33e8ad1cb6', 1, '
 Go::LRBSpline2D::Coef() const']],
['coeffixed', ['coefFixed', ['../class_go_1_1_r_b_spline2_d.html#
 af4104a2d152fad9cb608843df01ec29a', 1, 'Go::LRBSpline2D']],
['coefs_5fbegin', ['coefs_begin', ['../class_go_1_1_point_sequence.html#aa10ccbc7ab05c9a6bfd342c3f7107f9f',
 1, 'Go::PointSequence::coefs_begin()'], ['
 ../class_go_1_1_point_sequence.html#af8497ebb399a76f8f0e7dd75c729a34c', 1, 'Go::PointSequence::coefs_begin() const'], ['
 ../class_go_1_1_spline_curve.html#aa599da56f08b4de93bdc63efe4dde54a', 1, 'Go::SplineCurve::coefs_begin()'], ['
 ../class_go_1_1_spline_curve.html#a39d56e8f5f60cb45e573f704f03b260a', 1, 'Go::SplineCurve::coefs_begin() const'], ['
 ../class_go_1_1_spline_surface.html#acfa63bacdec3037faafd2dc885cfac9f', 1, 'Go::SplineSurface::coefs_begin()'], ['
 ../class_go_1_1_spline_surface.html#a2a31976341ded957536a675cc7bfa248', 1, 'Go::SplineSurface::coefs_begin() const'], ['
 ../class_go_1_1_spline_volume.html#ad58da7291700f0575efa4c2762b216f2', 1, 'Go::SplineVolume::coefs_begin()'], ['
 ../class_go_1_1_spline_volume.html#a42fd5d06ccc6d1ee23863bbade866bad', 1, 'Go::SplineVolume::coefs_begin() const']],
['coefs_5fend', ['coefs_end', ['../class_go_1_1_point_sequence.html#aa54646ecec7c7152430533b333723d558', 1, '
 Go::PointSequence::coefs_end()'], ['../class_go_1_1_point_sequence.html#aa383d5038fecac721e34b4b133c4ee01', 1, '
 Go::PointSequence::coefs_end() const'], ['
 ../class_go_1_1_spline_curve.html#a3ab302156aa321d810c2ab6ba51c6c5a', 1, 'Go::SplineCurve::coefs_end()'], ['../class_go_1_1
 1, 'Go::SplineCurve::coefs_end() const'], ['
 ../class_go_1_1_spline_surface.html#a3b9e715b5cb0272bf1f3f0d84c78b066', 1, 'Go::SplineSurface::coefs_end()'], ['
 ../class_go_1_1_spline_surface.html#a30089d92c4050f5ca6a56faee540dlc2', 1, 'Go::SplineSurface::coefs_end() const'], ['
 ../class_go_1_1_spline_volume.html#ac5326b49b3f3c3e673fd38ceae8d0c38', 1, 'Go::SplineVolume::coefs_end()'], ['
 ../class_go_1_1_spline_volume.html#a3838b5e920ea61fa0619968bbelad1e7', 1, 'Go::SplineVolume::coefs_end() const']],
['coefs_affectingparam', ['coefsAffectingParam', ['
 ../class_go_1_1_bspline_basis.html#ac8837d81197dd7d714750dea076c6ea4', 1, 'Go::BsplineBasis']],
['coefsbegin', ['coefsBegin', ['../class_go_1_1_bernstein_multi.html#ac83c6c6bb479263b626223b20f8ac4fc', 1, '
 Go::BernsteinMulti::coefsBegin()'], ['../class_go_1_1_bernstein_multi.html#a47d1d868c38c56c6910757afc94f7f61'
 , 1, 'Go::BernsteinMulti::coefsBegin() const'], ['
 ../class_go_1_1_bernstein_poly.html#a7dac78a3d206b3f26b605a44d9686del', 1, 'Go::BernsteinPoly::coefsBegin()'], ['
 ../class_go_1_1_bernstein_poly.html#abb3ad9d6283bf1ab52b5a22b21e6b419', 1, 'Go::BernsteinPoly::coefsBegin() const']],
['coefsend', ['coefsEnd', ['../class_go_1_1_bernstein_multi.html#a624861788f13f29d36e992b650a2b2a7', 1, '
 Go::BernsteinMulti::coefsEnd()'], ['../class_go_1_1_bernstein_multi.html#a2c9cb647e5d8822c8379d92dfel158b6', 1, '
 Go::BernsteinMulti::coefsEnd() const'], ['
 ../class_go_1_1_bernstein_poly.html#ace3dd87d2de438862b417d535d9149ae', 1, 'Go::BernsteinPoly::coefsEnd()'], ['
 ../class_go_1_1_bernstein_poly.html#a50faef0b3b23a2758acea1598ba38756', 1, 'Go::BernsteinPoly::coefsEnd() const']],
['coeftimesgamma', ['coefTimesGamma', ['
 ../class_go_1_1_r_b_spline2_d.html#a3b075d57cb21d3a5efd1dd90e50d4e46', 1, 'Go::LRBSpline2D::coefTimesGamma()'], ['
 ../class_go_1_1_r_b_spline2_d.html#a499edbee7c32a5994813185955be2a48', 1, 'Go::LRBSpline2D::coefTimesGamma() const']],
['col_5fscheme_5fselect', ['col_scheme_select', ['
 ../classmaterial_appearance.html#a9b42c9958f67223ed59b7957e5460ccd', 1, 'material_appearance']],
['col_5fscheme_5fselected', ['col_scheme_selected', ['
 ../classmaterial_appearance.html#a6e3e961804250b37fb94fd4f8d9b91d4', 1, 'material_appearance']],
['colmsns', ['colmsns', ['../class_pr_mat.html#aed80d11429e1c2a0566db0d806d80f3f', 1, 'PrMat::colmsns()'], ['
 ../class_pr_matrix.html#a8281ab6b6c20077323e3365c566e3df3', 1, 'PrMatrix::colmsns()'], ['
 ../class_pr_mat_sparse.html#a22e218eabe43458adaf7acac161da74a', 1, 'PrMatSparse::colmsns()']],
['color', ['color', ['../classgv_data.html#a20b594c0ecdda0f29a2b0d67baf25c917', 1, 'gvData']],
['column', ['Column', ['../class_n_e_w_m_a_t_1_1_base_matrix.html#a2642e545626b302f4b7f22793f7a1b70', 1, '
 NEWMAT::BaseMatrix']],
['columnsv', ['Columns', ['../class_n_e_w_m_a_t_1_1_base_matrix.html#a09b153845997e716109fc128573fce84', 1, '
 NEWMAT::BaseMatrix']],
['columnvector', ['ColumnVector', ['
 ../class_n_e_w_m_a_t_1_1_column_vector.html#a5b0c165a9ecee3313dbc0255579e0e8e', 1, 'NEWMAT::ColumnVector::ColumnVector()'], ['
 ../class_n_e_w_m_a_t_1_1_column_vector.html#a625731389b21c037ea40516b2009fa30', 1, 'NEWMAT::ColumnVector::ColumnVector(Ar)'], ['
 ../class_n_e_w_m_a_t_1_1_column_vector.html#a7dcc6e2dce014f0be26ff94e79c33994', 1, 'NEWMAT::ColumnVector::ColumnVector(Co)']

```

```

 BaseMatrix & amp;)'], ['./class_n_e_w_m_a_t_l_1_column_vector.html#a23e4d4faf5af6441fd7203e90c308288', 1, '
 NEWMAT::ColumnVector::ColumnVector(const ColumnVector & amp; gm)']]],
 ['common_5fedge', ['common_edge', ['../pr_path_triangle_seq_8h.html#a85bcac6907dcecff7f8e855068efbb47', 1, '
 PrPathTriangleSeq.h']]],
 ['commonisolinks', ['commonIsoLinks', ['
 ../class_go_l_1_intersection_point.html#a5b3b1e1ba9f54f8eb5fa0f3b663b34b8', 1, 'Go::IntersectionPoint']]],
 ['commonsplinespace', ['commonSplineSpace', ['
 ../class_go_l_1_ft_surface.html#a12dcae6ce5e961c188aa65625c7d824e', 1, 'Go::ftSurface::commonSplineSpace()'], ['
 ../class_go_l_1_ft_volume.html#a0b8317992cd18e0ea62aecad584d36e9', 1, 'Go::ftVolume::commonSplineSpace()']]],
 ['commonvertex', ['commonVertex', ['../class_go_l_1_ft_edge.html#a9b13cbe314c8101502dd34c5c6376e5a', 1, '
 Go::ftEdge']]],
 ['compare', ['Compare', ['../namespace_n_e_w_m_a_t.html#a141c465f6d4643bdefd8e03865d244c5', 1, '
 NEWMAT::Compare()'], ['../newmat4_8cpp.html#afad59b9d5b29862e68144236e760ccb8', 1, 'Compare():

```

Definition at line 119 of file functions\_3.js.

### 30.1522.1.21 double int double tol

Definition at line 308 of file functions\_3.js.

## 30.1523 doc/html/search/functions\_4.js File Reference

### Variables

- var [searchData](#)
- [fft.cpp](#)

### 30.1523.1 Variable Documentation

#### 30.1523.1.1 fft.cpp

Definition at line 9 of file functions\_4.js.

#### 30.1523.1.2 var searchData

### Initial value:

```

=
[
 ['d', ['d', ['../class_go_l_1_plane_int.html#adbef9642db16186c9411dce8c3fd7c92', 1, 'Go::PlaneInt']]],
 ['dart', ['Dart', ['../classhetriang_l_1_dart.html#ab39198f2792ee004668b8fc7f6a2d54f', 1, '
 hetriang::Dart::Dart()'], ['../classhetriang_l_1_dart.html#ad19d89b2aed2b3a8d928be1625c41522', 1, 'hetriang::Dart::Dart(Edge
 *edge, bool dir=true)'], ['../classhetriang_l_1_dart.html#a38272a3789a247be3b18bd270c7f7e0e', 1, '
 hetriang::Dart::Dart(const Dart & amp; dart)'], ['../classhed_l_1_dart.html#a141548d2908ef5847fd81370d57c07a0', 1, '
 hed::Dart::Dart()'], ['../classhed_l_1_dart.html#a40d2832ae9a9d61a1ccl16c9b171a2d14', 1, 'hed::Dart::Dart(Edge
 *edge, bool
 dir=true)'], ['../classhed_l_1_dart.html#a75fe6236394e8e039alb3ad44582bf4f', 1, 'hed::Dart::Dart(const Dart
 & amp; dart)']]]],
 ['data', ['Data', ['../class_n_e_w_m_a_t_l_1_simple_int_array.html#a224921d28dcde7d9131ae08d30adba79', 1, '
 NEWMAT::SimpleIntArray::Data()'], ['
 ../class_n_e_w_m_a_t_l_1_simple_int_array.html#aa2caf98cf436c591d84b0c4ffa1f95d2', 1, 'NEWMAT::SimpleIntArray::Data() cons
 ../class_matrix_row_col.html#a9900af353a47350b3346718cd4a6007f', 1, 'MatrixRowCol::Data()']]],
 ['datahandler', ['DataHandler', ['../class_data_handler.html#a51d845c1d9bbeef44607ec7592ba3c1d', 1, '
 DataHandler']]],
 ['datahandlervolandlr', ['DataHandlerVolAndLR', ['
 ../class_data_handler_vol_and_l_r.html#aa9165829b2bcbfd4bbb5c2afbb44ea6c0', 1, 'DataHandlerVolAndLR']]],
 ['datapointsiz', ['dataPointSize', ['
 ../struct_go_l_1_l_s_smooth_data.html#a535cfb16e7a24d9389902429bbe5b64b', 1, 'Go::LSSmoothData']]],
 ['dct', ['DCT', ['../namespace_n_e_w_m_a_t.html#a8d6c3e8e2abaccf76b86956d403842cb', 1, 'NEWMAT::DCT()'], ['
 ../fft_8cpp.html#a924dff49d847f7fc827760a9adad7f75', 1, 'DCT():

```

Definition at line 1 of file functions\_4.js.

## 30.1524 doc/html/search/functions\_5.js File Reference

### Variables

- var [searchData](#)
- var [const](#)
- [sisIP h](#)
- [sisIP Point & amp](#)
- [pnt int nder](#)
- [pnt int std::vector & lt](#)
- [Point & gt](#)

### 30.1524.1 Variable Documentation

#### 30.1524.1.1 [der Point& amp](#) [bound]

Definition at line 84 of file [functions\\_5.js](#).

#### 30.1524.1.2 [pnt const](#) [bound]

### Initial value:

```
=0'], ['./class_go_1_1_eval_curve_set.html#a6cad51d0922a7dedac3a31b2026ab0a4', 1,
'Go::EvalCurveSet::eval(double t)=0'], ['./
class_go_1_1_eval_curve_set.html#af94dfe476d9a31a9283263e1c7eb8bb3', 1,
'Go::EvalCurveSet::eval(double t, int n, std::vector<lt; std::vector<
lt; Point > > & amp;der)=0'], ['./
class_go_1_1_eval_param_curve.html#a765082ce27f3a3ac77f9dbdal1001cf1', 1,
'Go::EvalParamCurve::eval(double t) const ', ['./
class_go_1_1_eval_param_curve.html#ae053eebea18f9dbe4a3289a7aeb2563d', 1,
'Go::EvalParamCurve::eval(double t, int n, Point der[]) const ', ['./
class_go_1_1_eval_surface.html#a2048831767bc0760266fb999eb2f1036c', 1,
'Go::EvalSurface::eval(double u, double v) const =0'], ['./
class_go_1_1_eval_surface.html#ac30c4ea3682ad3433ed510bebf64c968', 1,
'Go::EvalSurface::eval(double u, double v, int n, Point der[]) const =0'], ['./
class_go_1_1_int_crv_evaluator.html#ab9f4d31119d434a7399b4f57a99c7efb', 1,
'Go::IntCrvEvaluator::eval(double t)', ['./
class_go_1_1_int_crv_evaluator.html#a212ac4e6d0896e12fef2d5a0bd895e58', 1,
'Go::IntCrvEvaluator::eval(double t, int n, std::vector<lt; std::vector<
lt; Point > > & amp;der)', ['./class_go_1_1_lift_curve.html#
ae512f443f85b1704a433e11bb8782af5', 1, 'Go::LiftCurve::eval(double t) const ', ['./
class_go_1_1_lift_curve.html#a2cb2e79f40e4d61f22303b5bcf2aeba3', 1,
'Go::LiftCurve::eval(double t, int n, Point der[]) const ', ['./
class_go_1_1_project_curve.html#af98b1ced4a661a5876e60db1d659f4fa', 1,
'Go::ProjectCurve::eval(double t) const ', ['./
class_go_1_1_project_curve.html#a96186d4cf98720c6777bfd7686f27f49', 1,
'Go::ProjectCurve::eval(double t, Go::Point seed) const ', ['./
class_go_1_1_project_curve.html#aa04bfd8be7761c82b715e59ab2537bc3', 1,
'Go::ProjectCurve::eval(double t, int n, Go::Point der[]) const ', ['./
class_go_1_1_project_curve_and_cross_tan.html#
a2887b1adafc03d983eb8405ea079f828', 1, 'Go::ProjectCurveAndCrossTan::eval(double t)', ['./
class_go_1_1_project_curve_and_cross_tan.html#
afe33ce9e2590e0bfe7808194acce9ebb', 1, 'Go::ProjectCurveAndCrossTan::eval(double t, int n,
std::vector<lt; std::vector<lt; Go::Point > > & amp;ders)', ['./
class_go_1_1_project_intersection_curve.html#
a6c51a345c6bc7eef9ad51c6a6af1a6a7', 1, 'Go::ProjectIntersectionCurve::eval(double t) const ', ['./
class_go_1_1_project_intersection_curve.html#
ae83c55aba4c48f4f9e67ee10a469eebd', 1, 'Go::ProjectIntersectionCurve::eval(double t, int n,
Point der[]) const ', ['./class_go_1_1_smooth_transition.html#
a9dec775aeaa43960e2f31389536c020d', 1, 'Go::SmoothTransition::eval(double t)', ['./
class_go_1_1_smooth_transition.html#a3014fd12933c038db7eb267fbed90b50', 1,
'Go::SmoothTransition::eval(double t, int n, std::vector<lt; std::vector<
lt; Point > > & amp;der)', ['./class_go_1_1_space_int_crv.
html#ab9a922c9e907889610ba88f507340ed3', 1, 'Go::SpaceIntCrv::eval(double t) const ', ['./
class_go_1_1_space_int_crv.html#a52581fc9d01325d625eff7e90fd5ad23', 1,
'Go::SpaceIntCrv::eval(double t, int n, Point der[]) const ', ['./
class_go_1_1_trim_curve.html#a383077ca958c414fc6c39ecde49df68f', 1,
'
```



```

Go::TrimCurve::eval(double t)'],['../class_go_1_1_trim_curve.html
#a94b8b25ba0369b8bf01a5e18caa7da',1,'Go::TrimCurve::eval(double t, int n, std::vector&
lt; std::vector<Point > > > &der)'],['../
class_go_1_1_bezier_triangle.html#a04e7a7850a4292cd300aac604e17efec',1,'
Go::BezierTriangle::eval()'],['../
class_go_1_1_eval_functor_curve.html#adc7d79d2ad002096776641323d421fc0',1,'
Go::EvalFunctorCurve::eval(double t) const'],['../
class_go_1_1_eval_functor_curve.html#ad765b1fdca8d47430a90b05d08358c42',1,'
Go::EvalFunctorCurve::eval(double t, int n, Point der[]) const'],['../
class_go_1_1_eval_functor_surface.html#a4bba1fd389a8a21408c76279077d8c98'
,1,'Go::EvalFunctorSurface::eval(double u, double v) const'],['../
class_go_1_1_eval_functor_surface.html#a52a9888c724819231d867b5f8ecc9edb'
,1,'Go::EvalFunctorSurface::eval(double u, double v, int n,
Point der[]) const'],['../class_go_1_1_r_b_spline2_d.html#
af0a9b4ba330f30b6a7cceeaa8bd8e40b',1,'Go::LRBSpline2D::eval()'],['../
class_go_1_1_volume_parameter_curve.html#
a9fe5deedf6c920a026949df25a804eeb',1,'Go::VolumeParameterCurve::eval(double t) const'],['../
class_go_1_1_volume_parameter_curve.html#
a7a1967e05c656a34b29c9e455c6985d3',1,'Go::VolumeParameterCurve::eval(double t, int n,
Point der[]) const'],['../class_go_1_1_volume_space_curve.html#
a9c680f8db2892c4000e199d3591c4ec6',1,'Go::VolumeSpaceCurve::eval(double t) const'],['../
class_go_1_1_volume_space_curve.html#adf40f504a4ffcd83d09f88dd023deb69',1,'
Go::VolumeSpaceCurve::eval(double t, int n, Point der[]) const']]],
['eval_5f2_5fcrv',['eval_2_crv',['../sisl_p_8h.html#a08942c76767d8172cc58da500917c959'
,1,'eval_2_crv():

```

Definition at line 1 of file functions\_5.js.

### 30.1524.1.3 Point& gt

Definition at line 99 of file functions\_5.js.

### 30.1524.1.4 sislP h

Definition at line 84 of file functions\_5.js.

### 30.1524.1.5 pnt int std::vector& lt

Definition at line 99 of file functions\_5.js.

### 30.1524.1.6 pnt int nder

Definition at line 99 of file functions\_5.js.

### 30.1524.1.7 var searchData

#### Initial value:

```

=
[
 ['each',['each',['../jquery_8js.html#a871ff39db627c54c710a3e9909b8234c',1,'jquery.js']]],
 ['eatwhite',['eatwhite',['../namespace_go_1_1_utils.html#a43780996b9bc44a2d902858e953f8050',1,'Go::Utils'
]]],
 ['edge'

```

Definition at line 1 of file functions\_5.js.

## 30.1525 doc/html/search/functions\_6.js File Reference

### Variables

- var [searchData](#)
- [fft](#) [cpp](#)

### 30.1525.1 Variable Documentation

#### 30.1525.1.1 [fft](#) [cpp](#)

Definition at line 28 of file [functions\\_6.js](#).

#### 30.1525.1.2 [var searchData](#)

#### Initial value:

```
=
[
 ['face', ['face', ['../class_go_1_lft_cell.html#aa121981ee3b13d136e7d29b0ee4bcd77', 1, 'Go::ftCell::face()'],
 ['../class_go_1_lft_curve_segment.html#af474a0ed29e5e7fb6dfdb8f80e5f38fb', 1, 'Go::ftCurveSegment::face()'], ['
 ../class_go_1_lft_edge.html#a3f58aed1bbe6f007f2c5f345279ed2e2', 1, 'Go::ftEdge::face()'], ['
 ../class_go_1_lft_edge_base.html#a31c097d1aebfbb34aedb37b4d2d60491', 1, 'Go::ftEdgeBase::face()'], ['
 ../class_go_1_lft_point.html#aa32c53f685602d17fa5e4f9a1e1f0c4d', 1, 'Go::ftPoint::face()'], ['
 ../class_go_1_lft_sf_edge.html#a2659c6f1a432256dab88ba2d2b110b84', 1, 'Go::ftSSfEdge::face()'], ['
 ../class_go_1_lft_surface_set_point.html#a882e43b4b07c97b22f9979ab5d0099ee', 1, 'Go::ftSurfaceSetPoint::face()'], ['
 ../class_go_1_ltp_edge.html#a15014c47fb50b7246d467a7c877f22ed', 1, 'Go::tpEdge::face()']]],
 ['faceadjacency', ['FaceAdjacency', ['../class_go_1_l_face_adjacency.html#a3301eab6895130fcaal924fff513361e
 ', 1, 'Go::FaceAdjacency::FaceAdjacency(double tol_gap, double tol_neighbour, double tol_kink, double
 tol_bend)'], ['../class_go_1_l_face_adjacency.html#a586af6c2a4877a6fccadc369b1807d37', 1, '
 Go::FaceAdjacency::FaceAdjacency(const tpTolerances &tol)']]],
 ['facebox', ['faceBox', ['../class_go_1_lft_cell.html#a5e5ae761c933390e66693af8ae239122', 1, 'Go::ftCell']]],
 ['faceconnectivity', ['FaceConnectivity', ['
 ../struct_go_1_l_face_connectivity.html#a296535b83af35ef3587335903306feb4', 1, 'Go::FaceConnectivity']]],
 ['faceedgeidistance', ['faceEdgeDistance', ['
 ../class_go_1_l_face_set_quality.html#a49d75b7386f9ee37fae3811528a221ca', 1, 'Go::FaceSetQuality::faceEdgeDistance()'], ['
 ../class_go_1_l_model_quality.html#a9b312ebb24d3dc34c760ea3729a1ae16', 1, 'Go::ModelQuality::faceEdgeDistance()']]],
 ['facenormalconsistency', ['faceNormalConsistency', ['
 ../class_go_1_l_face_set_quality.html#ad1d8907e97bafed46bdc9f63c00537f83', 1, 'Go::FaceSetQuality::faceNormalConsistency()'], ['
 ../class_go_1_l_model_quality.html#a516625b64e95b6dfc521b621f7055daa', 1, 'Go::ModelQuality::faceNormalConsistency()']]],
 ['facenumber', ['faceNumber', ['
 ../class_go_1_l_vol_boundary_condition.html#a917407b5d8f423b026002d6b0f1daccl', 1, 'Go::VolBoundaryCondition::faceNumber()'],
 ['../class_go_1_l_vol_point_bd_cond.html#a64e1b684ea2f63c8eeb42ae6219cd38', 1, 'Go::VolPointBdCond::faceNumber()']]],
 ['faceparameter', ['faceParameter', ['../class_go_1_lft_edge.html#ab3e951684c4b19faa7e75b1ea3d600a4', 1, '
 Go::ftEdge::faceParameter()'], ['../class_go_1_l_curve_on_surface.html#a72cf294c8a3569e54f7469ae8ef30800', 1, '
 Go::CurveOnSurface::faceParameter()']]],
 ['facepositiondiscontinuity', ['facePositionDiscontinuity', ['
 ../class_go_1_l_face_set_quality.html#abf6c4e2597c6c988dba8e410cb297c19', 1, 'Go::FaceSetQuality::facePositionDiscontinuity()'],
 ['../class_go_1_l_model_quality.html#a6449726206f9f5782a3eef58fa44037', 1, 'Go::ModelQuality::facePositionDiscontinuity()']]],
 ['faces', ['faces', ['../class_go_1_l_vertex.html#aa4c5e30501c4d7bae0c10fclc103b237', 1, 'Go::Vertex::faces()
 const'], ['../class_go_1_l_vertex.html#afaa45ecc2c3a6ff2f839e150137c5540', 1, 'Go::Vertex::faces(Body *bd)
 const']]],
 ['facesetquality', ['FaceSetQuality', ['
 ../class_go_1_l_face_set_quality.html#ab92daefb5beae4cf2d11fc10d387f342', 1, 'Go::FaceSetQuality::FaceSetQuality(double g
 ../class_go_1_l_face_set_quality.html#a4108bfff3378f81f49c8b9ce4411fc4a4', 1, 'Go::FaceSetQuality::FaceSetQuality(const
 tpTolerances &tpTol, double approx)'], ['../class_go_1_l_face_set_quality.html#a80b64f2fd037fca7f37ab8d808456a1a', 1, '
 Go::FaceSetQuality::FaceSetQuality(shared_ptr< SurfaceModel > &sfmodel)']]],
 ['facesetrepair', ['FaceSetRepair', ['
 ../class_go_1_l_face_set_repair.html#a4ab24096dd61caa312afdd87bdad917c', 1, 'Go::FaceSetRepair::FaceSetRepair(shared_ptr<
 ../class_go_1_l_face_set_repair.html#af9a1e4c4cd65ea8fd3b6189f38bbca57c', 1, 'Go::FaceSetRepair::FaceSetRepair(shared_ptr<
 FaceSetQuality & > quality)']]],
 ['facesinplane', ['facesInPlane', ['../class_go_1_l_surface_model.html#a3694b0cdf7688cd5f832ab14c2558579', 1, '
 Go::SurfaceModel']]],
 ['facetangentdiscontinuity', ['faceTangentDiscontinuity', ['
 ../class_go_1_l_face_set_quality.html#a61a7a9c95e1fa425fc8b860813af8ae7', 1, 'Go::FaceSetQuality::faceTangentDiscontinuity()'],
 ['../class_go_1_l_model_quality.html#acf5f2662524a1715908ebb06b3723262', 1, 'Go::ModelQuality::faceTangentDiscontinuity()']]],
 ['facevertexdistance', ['faceVertexDistance', ['
 ../class_go_1_l_face_set_quality.html#ad832c76db79664cdeca3dec6235d35cd', 1, 'Go::FaceSetQuality::faceVertexDistance()'],
 ['../class_go_1_l_model_quality.html#afdad4f29b9c9b7f642cd6a1db3efc82', 1, 'Go::ModelQuality::faceVertexDistance()']]],
 ['factorial', ['factorial', ['../namespace_go.html#af2dale127f042c70cdb76c5b48158f8d', 1, 'Go']]],

```

```

['feedback', ['feedback', ['../classgv_view.html#a2ccd054a91d3f42596b3bfc1a752a3ee', 1, 'gvView']]],
['feedbackmode', ['feedbackmode', ['../classgv_view.html#af284c1223a03cf29d864b8bbb2314922', 1, 'gvView']]],
['fetchassharedptr', ['fetchAsSharedPtr', ['
 ../class_go_1_1_surface_model.html#aafd3dbc037ce0c761de79b15e6f38c20', 1, 'Go::SurfaceModel::fetchAsSharedPtr()'], ['
 ../class_go_1_1_volume_model.html#a963787be67a45107e3bbd39a0e1b37cc', 1, 'Go::VolumeModel::fetchAsSharedPtr()']]],
['fetchcompositecurves', ['fetchCompositeCurves', ['
 ../class_go_1_1_curve_model.html#a1f227f8d4d720f672925567f90eea029', 1, 'Go::CurveModel']]],
['fetchcorrespondingfaces', ['fetchCorrespondingFaces', ['
 ../class_go_1_1_ft_surface.html#a153b817ffbe94c0dc06432706c527a4e', 1, 'Go::ftSurface']]],
['fetchededgecorners', ['fetchEdgeCorners', ['
 ../namespace_go_1_1_l_r_surf_stitch.html#a448922c6cf420b46b0e5008d85734979', 1, 'Go::LRSurfStitch']]],
['fetchloops', ['fetchLoops', ['../class_go_1_1_intersection_pool.html#a66197c013cde173beb92852e6f9a28ce', 1,
 'Go::IntersectionPool']]],
['fetchsamplepoints', ['fetchSamplePoints', ['
 ../class_go_1_1_ft_surface_set.html#ac9988bb705b14ed04100f7ccb0f5965c', 1, 'Go::ftSurfaceSet::fetchSamplePoints()'], ['
 ../class_go_1_1_surface_model.html#a3a59e2b123bc5ec4e16a4a119b9c4eca', 1, 'Go::SurfaceModel::fetchSamplePoints()']]],
['fetchvxpntcorr', ['fetchVxPntCorr', ['
 ../class_go_1_1_regularize_face.html#a919c13841fade539b38a4de946ba5de8', 1, 'Go::RegularizeFace::fetchVxPntCorr()'], ['
 ../class_go_1_1_regularize_face_set.html#a8fd526b05f5e833e036b35954b16042b', 1, 'Go::RegularizeFaceSet::fetchVxPntCorr()']]],
['fft', ['FFT', ['../namespace_n_e_w_m_a_t.html#ac40dccb5f5f25e5280cb7acb5249ed01', 1, 'NEWMAT::FFT()'], ['
 ../fft_8cpp.html#a52ecc4818418779fbd2d2e134625ec7d', 1, 'FFT():

```

Definition at line 1 of file functions\_6.js.

## 30.1526 doc/html/search/functions\_7.js File Reference

### Variables

- var [searchData](#)
- generic\_graph\_algorithms\_implementation [h](#)
- generic\_graph\_algorithms\_implementation [const int xs](#)
- generic\_graph\_algorithms\_implementation [const int const int ys](#)
- generic\_graph\_algorithms\_implementation [const int const int const int x0](#)
- generic\_graph\_algorithms\_implementation [const int const int const int const int y0](#)
- generic\_graph\_algorithms\_implementation [const int const int const int const int const int doubleBuffer](#)
- generic\_graph\_algorithms\_implementation [const int const int const int const int const int const int two\\_sided](#)
- generic\_graph\_algorithms\_implementation [const int const int const int const int const int const int const int lighting](#)
- generic\_graph\_algorithms\_implementation [const int const int const int const int const int const int const int const int const int normalize](#)
- generic\_graph\_algorithms\_implementation [const int const int const int const int const int const int const int const int const int smooth](#)
- generic\_graph\_algorithms\_implementation [const int const int const int const int const int const int const int const int const int const int const double xtrans](#)
- generic\_graph\_algorithms\_implementation [const int const int const int const int const int const int const int const int const int const int const double const double ytrans](#)
- generic\_graph\_algorithms\_implementation [const int const int const int const int const int const int const int const int const int const int const double const double const double ztrans](#)
- generic\_graph\_algorithms\_implementation [const int const int const int const int const int const int const int const int const int const int const double const double const double const double xscale](#)
- generic\_graph\_algorithms\_implementation [const int const int const int const int const int const int const int const int const int const int const double const double const double const double const double const double yscale](#)
- generic\_graph\_algorithms\_implementation [const int const int const int const int const int const int const int const int const int const int const double const double const double const double const double const double const double zscale](#)
- generic\_graph\_algorithms\_implementation [const int const int const int const int const int const int const int const int const int const int const double const double const double const double const double const double const double int & amp](#)
- tx
- ty
- [const int texture\\_mode](#)
- [const int const char \\*const texfile =NULL\):&#160](#)
- [gl\\_aux cpp](#)

### 30.1526.1 Variable Documentation

30.1526.1.1 `int& amp` `[property]`, `[bound]`, `[constrained]`

Definition at line 35 of file `functions_7.js`.

30.1526.1.2 `gl_aux` `cpp`

Definition at line 559 of file `functions_7.js`.

30.1526.1.3 `gl_aux` `const int const int const int const int const int doubleBuffer` `[property]`, `[bound]`, `[constrained]`

Definition at line 35 of file `functions_7.js`.

30.1526.1.4 `generic_graph_algorithms_implementation` `h`

Definition at line 35 of file `functions_7.js`.

30.1526.1.5 `gl_aux` `const int const int const int const int const int const int const int lighting` `[property]`, `[bound]`, `[constrained]`

Definition at line 35 of file `functions_7.js`.

30.1526.1.6 `gl_aux` `const int const int const int const int const int const int const int const int normalize` `[property]`, `[bound]`, `[constrained]`

Definition at line 35 of file `functions_7.js`.

30.1526.1.7 `var searchData`

Definition at line 1 of file `functions_7.js`.

30.1526.1.8 `gl_aux` `const int const int const int const int const int const int const int const int const int smooth` `[property]`, `[bound]`, `[constrained]`

Definition at line 35 of file `functions_7.js`.

30.1526.1.9 `const int const char* const texfile =NULL);&#160`

Definition at line 559 of file `functions_7.js`.

30.1526.1.10 **const int texture\_mode**

Definition at line 559 of file functions\_7.js.

30.1526.1.11 **gl\_aux const int const int const int const int const int const int two\_sided** [property], [bound], [constrained]

Definition at line 35 of file functions\_7.js.

30.1526.1.12 **tx**

Definition at line 559 of file functions\_7.js.

30.1526.1.13 **ty**

Definition at line 559 of file functions\_7.js.

30.1526.1.14 **gl\_aux const int const int const int x0** [property], [bound], [constrained]

Definition at line 35 of file functions\_7.js.

30.1526.1.15 **gl\_aux const int xs** [property], [bound], [constrained]

Definition at line 35 of file functions\_7.js.

30.1526.1.16 **gl\_aux const int const int const int const int const int const int const int const int const int const double const double const double const double xscale** [property], [bound], [constrained]

Definition at line 35 of file functions\_7.js.

30.1526.1.17 **gl\_aux const int const int const int const int const int const int const int const int const int const int const double xtrans** [property], [bound], [constrained]

Definition at line 35 of file functions\_7.js.

30.1526.1.18 **gl\_aux const int const int const int const int y0** [property], [bound], [constrained]

Definition at line 35 of file functions\_7.js.

30.1526.1.19 **gl\_aux const int const int ys** [property], [bound], [constrained]

Definition at line 35 of file functions\_7.js.

30.1526.1.20 **gl\_aux const int const int const int const int const int const int const int const int const int const int const double const double const double const double const double const double yscale** [property], [bound], [constrained]

Definition at line 35 of file functions\_7.js.

30.1526.1.21 **gl\_aux const int const int const int const int const int const int const int const int const int const int const double const double ytrans** [property], [bound], [constrained]

Definition at line 35 of file functions\_7.js.

30.1526.1.22 **gl\_aux const int const int const int const int const int const int const int const int const int const int const int const double const double const double const double const double const double const double zscale** [property], [bound], [constrained]

Definition at line 35 of file functions\_7.js.

30.1526.1.23 **gl\_aux const int const int const int const int const int const int const int const int const int const int const int const double const double const double ztrans** [property], [bound], [constrained]

Definition at line 35 of file functions\_7.js.

## 30.1527 doc/html/search/functions\_8.js File Reference

### Variables

- var [searchData](#)
- [sisIP](#) h

### 30.1527.1 Variable Documentation

30.1527.1.1 [sisIP](#) h

Definition at line 61 of file functions\_8.js.

30.1527.1.2 [var searchData](#)

Definition at line 1 of file functions\_8.js.

## 30.1528 doc/html/search/functions\_9.js File Reference

### Variables

- var [searchData](#)
- [jquery](#) js

## 30.1528.1 Variable Documentation

### 30.1528.1.1 jquery.js

Definition at line 20 of file functions\_9.js.

### 30.1528.1.2 var searchData

#### Initial value:

```
=
[
 ['i', ['i', ['../class_n_e_w_m_a_t_l_l_matrix_type.html#acb7b50521ae4e68a7cf332d180a19bc8', 1,
 'NEWMAT::MatrixType::i()'], ['../class_n_e_w_m_a_t_l_l_base_matrix.html#a4227de5c79ebbe72cb883dc44a4adc1f', 1,
 'NEWMAT::BaseMatrix::i()']],
 ['id', ['id', ['../classhed_l_l_node.html#ae0ae30c021c0e665b409db77bf604afa', 1, 'hed::Node::id()'], ['
 ../classgv_paintable.html#a7a31121be20df12925c3aac0e838d598', 1, 'gvPaintable::id()']],
 ['identicalandembeddedfaces', ['identicalAndEmbeddedFaces', ['
 ../class_go_l_l_face_set_repair.html#ab30952f5cfc5281414f1c90af73de25e', 1, 'Go::FaceSetRepair::identicalAndEmbeddedFaces
 ../class_go_l_l_model_repair.html#a966573f66568b95f898e4e2d6bd0517f', 1, 'Go::ModelRepair::identicalAndEmbeddedFaces()']],
 ['identicalcvs', ['identicalCvs', ['../class_go_l_l_identity.html#acf3fc01c2e37aa833e398ad4d81d9fee', 1,
 'Go::Identity']],
 ['identicalorembdededges', ['identicalOrEmbeddedEdges', ['
 ../class_go_l_l_face_set_quality.html#adec9ce087118da601948c1d7c220f861', 1, 'Go::FaceSetQuality::identicalOrEmbeddedEdges
 ../class_go_l_l_model_quality.html#a58691538013438d3e8812f23a42b0bee', 1, 'Go::ModelQuality::identicalOrEmbeddedEdges()']],
 ['identicalorembdedfaces', ['identicalOrEmbeddedFaces', ['
 ../class_go_l_l_face_set_quality.html#aa7bb010c6e9b4040645015843d0d6c19', 1, 'Go::FaceSetQuality::identicalOrEmbeddedFaces
 ../class_go_l_l_model_repair.html#aabf8ba0d9a44fc491c8204a7d5754ba1', 1, 'Go::ModelQuality::identicalOrEmbeddedFaces()']],
 ['identicalsfs', ['identicalSfs', ['../class_go_l_l_identity.html#a14ba65d23ce10a0689f5f353aff2d25a', 1,
 'Go::Identity']],
 ['identicalvertices', ['identicalVertices', ['
 ../class_go_l_l_face_set_quality.html#a7d0ab3adb89a04a011120e8c4bdd9491', 1, 'Go::FaceSetQuality::identicalVertices()'], ['
 ../class_go_l_l_face_set_repair.html#a06b2af4b37d3fe6dd6c26fca17d150e3', 1, 'Go::FaceSetRepair::identicalVertices()'], ['
 ../class_go_l_l_model_quality.html#a415888bf6f57c852849e4f2ee719dfa6', 1, 'Go::ModelQuality::identicalVertices()'], ['
 ../class_go_l_l_model_repair.html#acb862290dbbc7a5a0045e771baa03c53', 1, 'Go::ModelRepair::identicalVertices()']],
 ['identify_5felements_5ffrom_5fmesh', ['identify_elements_from_mesh', ['
 ../namespace_go_l_l_l_r_spline_utils.html#a0df66adf1511debd6e5042d890989278', 1, 'Go::LRSplineUtils']],
 ['identify_5fpatch_5flower_5fleft', ['identify_patch_lower_left', ['
 ../namespace_go_l_l_mesh2_d_utils.html#a70ff355fcf2aa04dabab17c8387ce6d3', 1, 'Go::Mesh2DUtils']],
 ['identify_5fpatch_5fupper_5fright', ['identify_patch_upper_right', ['
 ../namespace_go_l_l_mesh2_d_utils.html#a118220e0b56c3f07a13dc8ba42a393cb', 1, 'Go::Mesh2DUtils']],
 ['identifybdpnts', ['identifyBdPnts', ['../class_go_l_lft_point_set.html#ab8e8c73cdf2c3a939c5574f487365330',
 1, 'Go::ftPointSet']],
 ['identifyloop', ['identifyLoop', ['../namespace_go_l_l_path.html#a12301ef59c50bee8128a06243290bdfb', 1,
 'Go::Path']],
 ['identity', ['identity', ['../class_go_l_l_matrix_x_d.html#a69ef9806148e42baf80b86cf813022b5', 1,
 'Go::MatrixXD']],
 ['identity_5felement', ['identity_element', ['../namespacestd.html#a78e37a8fb419e182957f4be0adac1d03', 1,
 'std::identity_element(plus<< _Tp >)', ['../namespacestd.html#a0f2e93c7362c26ce388c62a5abc1be52', 1,
 'std::identity_element(multiplies<< _Tp >)']],
 ['identitymatrix', ['IdentityMatrix', ['
 ../class_n_e_w_m_a_t_l_l_identity_matrix.html#acd9124ff1b21a69afd415a9e68d073b8', 1, 'NEWMAT::IdentityMatrix::IdentityMatr
 ../class_n_e_w_m_a_t_l_l_identity_matrix.html#ad9a24dff9b08f41cda062438666d942f', 1, 'NEWMAT::IdentityMatrix::IdentityMatr
 ../class_n_e_w_m_a_t_l_l_identity_matrix.html#a4f3706dc33b285d3a8706b945a4dfc17', 1,
 'NEWMAT::IdentityMatrix::IdentityMatrix(const IdentityMatrix &gm)', ['
 ../class_n_e_w_m_a_t_l_l_identity_matrix.html#aeed6ff0975f66d11b1ce9f2464e10b32', 1, 'NEWMAT::IdentityMatrix::IdentityMatr
 ['ieqnd', ['IEQND', ['../class_n_e_w_m_a_t_l_l_base_matrix.html#a3e76a73d4cb17962fc9d99e8ba8fec27', 1,
 'NEWMAT::BaseMatrix']],
 ['if', ['if', ['../jquery_8js.html#a9db6d45a025ad692282fe23e69eeba43', 1, 'if(!b.support.opacity):
```

Definition at line 1 of file functions\_9.js.

## 30.1529 doc/html/search/functions\_a.js File Reference

### Variables

- var [searchData](#)
- [jacobi.cpp](#)

### 30.1529.1 Variable Documentation

#### 30.1529.1.1 jacobi.cpp

Definition at line 3 of file functions\_a.js.

#### 30.1529.1.2 var searchData

##### Initial value:

```
=
[
 ['jacobi', ['Jacobi', ['../namespace_n_e_w_m_a_t.html#a2836cab2994eacfc0e7a3b5197d0e9a', 1, '
 NEWMAT::Jacobi(const SymmetricMatrix &, DiagonalMatrix &)', ['
 ../namespace_n_e_w_m_a_t.html#a556f131206374d64bb896c8f815e8fd6', 1, 'NEWMAT::Jacobi(const SymmetricMatrix &, Diagonal
 ../namespace_n_e_w_m_a_t.html#a84f3caafdd15a0749b7a11ff668866f7', 1, 'NEWMAT::Jacobi(const SymmetricMatrix
 &, DiagonalMatrix &, Matrix &)', ['../namespace_n_e_w_m_a_t.html#ab7fb7f360924503ff4e0a681602cbc0'
 , 1, 'NEWMAT::Jacobi(const SymmetricMatrix &, DiagonalMatrix &, SymmetricMatrix &, Matrix &,
 bool=true)'], ['../jacobi_8cpp.html#ad370e3456b419cfe2eae25f2300fd5b1', 1, 'Jacobi(const SymmetricMatrix &X,
 DiagonalMatrix &D, SymmetricMatrix &A, Matrix &V, bool eivec):
```

Definition at line 1 of file functions\_a.js.

## 30.1530 doc/html/search/functions\_b.js File Reference

### Variables

- var [searchData](#)
- [newmat6.cpp](#)

### 30.1530.1 Variable Documentation

#### 30.1530.1.1 newmat6.cpp

Definition at line 15 of file functions\_b.js.

#### 30.1530.1.2 var searchData

##### Initial value:



```

=
[
 ['key', ['Key', ['../classmaterial__appearance.html#a74882356eb34bffa5a8683cccbee97bc', 1, '
 material_appearance']]],
 ['knotepsilon', ['knotEpsilon', ['../class_go_1_1_go_tools.html#af44ef84796819b793f2b196a7b296b9c', 1, '
 Go::GoTools']]],
 ['knotinterval', ['knotInterval', ['../class_go_1_1_bspline_basis.html#a305a7e9c5ded401fbb4de72c0e9d2693', 1, '
 Go::BsplineBasis']]],
 ['knotintervalfuzzy', ['knotIntervalFuzzy', ['
 ../class_go_1_1_bspline_basis.html#a0ae41ca8567fa19eaccdc8b228f1580', 1, 'Go::BsplineBasis::knotIntervalFuzzy()'], ['
 ../class_go_1_1_param0_function_int.html#adf18f61ba3429e8337f680bc6ed9eb4a', 1, 'Go::Param0FunctionInt::knotIntervalFuzzy'],
 ../class_go_1_1_param1_function_int.html#a4cfe2132fa258b820edaf776f5383396', 1, 'Go::Param1FunctionInt::knotIntervalFuzzy'],
 ../class_go_1_1_param2_function_int.html#a751cf94d3a521e66fde43b34d920662a', 1, 'Go::Param2FunctionInt::knotIntervalFuzzy'],
 ../class_go_1_1_param_curve_int.html#ab7260dc6baf8fa87b13c773360cced24', 1, '
 Go::ParamCurveInt::knotIntervalFuzzy()'], ['../class_go_1_1_param_surface_int.html#af16dd54625073ebcccc4cac921c9d6ac2', 1, '
 Go::ParamSurfaceInt::knotIntervalFuzzy()'], ['../class_go_1_1_spline1_function_int.html#ae54cff5b82d26a4171300d8fbb3de2d', 1, '
 Go::Spline1FunctionInt::knotIntervalFuzzy()'], ['
 ../class_go_1_1_spline2_function_int.html#a7229f63cd797e93f4aed7b4f3f8650ba', 1, 'Go::Spline2FunctionInt::knotIntervalFuzzy'],
 ../class_go_1_1_spline_curve_int.html#aaccc36c235ff72d3989818aa3d99b024', 1, 'Go::SplineCurveInt::knotIntervalFuzzy()'], [
 ../class_go_1_1_spline_surface_int.html#ab48552a70547a940f282bf4f9eecd5ad', 1, 'Go::SplineSurfaceInt::knotIntervalFuzzy()'],
 ../class_go_1_1_mesh2_d.html#a7038b8a5c1ea106fac4c1b5b0bbddc7e', 1, 'Go::Mesh2D::knotIntervalFuzzy()']]],
 ['knotmultiplicities', ['knotMultiplicities', ['
 ../class_go_1_1_bspline_basis.html#a91f3e774d6f812df432c2f38fcf44f8e', 1, 'Go::BsplineBasis']]],
 ['knotmultiplicity', ['knotMultiplicity', ['
 ../class_go_1_1_bspline_basis.html#aca0cff53a80af7a8cfd0cc64acea7c95', 1, 'Go::BsplineBasis']]],
 ['knots', ['knots', ['../class_go_1_1_block_solution.html#af0bdaab56a91706e89bcafe49ad2ae8f', 1, '
 Go::BlockSolution::knots()'], ['../class_go_1_1_isogeometric_block.html#ad515166c60c7e702dc9f27d48f7122d3', 1, '
 Go::IsogeometricBlock::knots()'], ['../class_go_1_1_sf_solution.html#a74a9a0c6cc3dc7e3ddfcab6ac8031cdc', 1, '
 Go::SfSolution::knots()'], ['../class_go_1_1_vol_solution.html#a9d6a56296f7300e21f6f371dccc5d08a0', 1, '
 Go::VolSolution::knots()']]],
 ['knots_5fto_5finsert', ['knots_to_insert', ['
 ../namespace_go_1_1_r_spline_utils.html#a4f98f5746feb419caf049335e2bleda7', 1, 'Go::LRSplineUtils']]],
 ['knotsbegin', ['knotsBegin', ['../class_go_1_1_spline_curve.html#a9617b6d88db3ed4e11ed498a40d89581', 1, '
 Go::SplineCurve::knotsBegin()'], ['../class_go_1_1_spline_curve.html#ab7cfeb2dc285d0378062d55d18c0acac2', 1, '
 Go::SplineCurve::knotsBegin() const '], ['../class_go_1_1_mesh2_d.html#a752b5850413c524460efed453321a688', 1, '
 Go::Mesh2D::knotsBegin()']]],
 ['knotsend', ['knotsEnd', ['../class_go_1_1_spline_curve.html#acf20e5283b690c8f22aae9fb147fc210', 1, '
 Go::SplineCurve::knotsEnd()'], ['../class_go_1_1_spline_curve.html#a9fe54c39a94b2c2575f6d0511f721145', 1, '
 Go::SplineCurve::knotsEnd() const '], ['../class_go_1_1_mesh2_d.html#a1cedb03145e626d0bd834ca339f091db', 1, '
 Go::Mesh2D::knotsEnd()']]],
 ['knotspan', ['knotSpan', ['../class_go_1_1_spline_surface.html#a4ae188d833ba7e018a60361975ca1754', 1, '
 Go::SplineSurface::knotSpan()'], ['../class_go_1_1_spline_volume.html#aaffe28b54eaal1c97de1e2a3e2229b608f', 1, '
 Go::SplineVolume::knotSpan()']]],
 ['knotssimple', ['knotsSimple', ['../class_go_1_1_bspline_basis.html#aa2e75ab7996534ff328f96d5b2cfadde', 1, '
 Go::BsplineBasis']]],
 ['kp', ['KP', ['../class_n_e_w_m_a_t_l_1_matrix_type.html#a6a5c67f2185e7691d2d8e85e7f209130', 1, '
 NEWMAT::MatrixType::KP()'], ['../class_matrix_row_col.html#a8a788cf7179884e2100b77f845cea7dd', 1, 'MatrixRowCol::KP()'], [
 ../namespace_n_e_w_m_a_t_l.html#adac429a80804603f506371d42304b686', 1, 'NEWMAT::KP()'], [
 ../newmat_6_cpp.html#ae3166f1201bb79865ac1945a9ebc64d6', 1, 'KP():

```

Definition at line 1 of file functions\_b.js.

## 30.1531 doc/html/search/functions\_c.js File Reference

### Variables

- var [searchData](#)
- PrGeodesics [h](#)
- PrGeodesics [double epsge](#)
- PrGeodesics [double bool init\\_mba =false](#)
- PrGeodesics [double bool double mba\\_level =0.0](#)
- PrGeodesics [double bool double bool closest\\_dist =true](#)
- PrGeodesics [double bool double bool bool repar](#)

### 30.1531.1 Variable Documentation

#### 30.1531.1.1 PrGeodesics double bool double bool closest\_dist =true [bound]

Definition at line 88 of file functions\_c.js.

## 30.1531.1.2 PrGeodesics double epsge [bound]

Definition at line 17 of file functions\_c.js.

## 30.1531.1.3 PrGeodesics h

Definition at line 17 of file functions\_c.js.

## 30.1531.1.4 PrGeodesics double bool init\_mba =false [bound]

Definition at line 88 of file functions\_c.js.

## 30.1531.1.5 PrGeodesics double bool double mba\_level =0.0 [bound]

Definition at line 88 of file functions\_c.js.

## 30.1531.1.6 PrGeodesics double bool double bool bool repar [bound]

**Initial value:**

```
=false)']]],
 ['lrsurfsmoothls', ['LRSurfSmoothLS', ['../class_go_1_1_1_r_surf_smooth_1_s
.html#a8acba4937645e7d7c619ce67ae71c5b8', 1, 'Go::LRSurfSmoothLS::LRSurfSmoothLS
(shared_ptr<LRSplineSurface > surf, std::vector<int & coef_known)'], ['../
class_go_1_1_1_r_surf_smooth_1_s.html#ad3d1d6e980d0818742352b8719702ac8', 1,
'Go::LRSurfSmoothLS::LRSurfSmoothLS()']]],
 ['lssmoothdata', ['LSSmoothData', ['../struct_go_1_1_1_s_smooth_data.html#
adbe08d7ac0e51e41aad91021aaef8014', 1, 'Go::LSSmoothData']]],
 ['ltlexpoint', ['ltLexPoint', ['../main_8cpp.html#a3ad6217edc1cd915556be09a7a6e512', 1,
main.cpp']],
 ['lubksb', ['lubksb', ['../class_n_e_w_m_a_t_1_1_crout_matrix.html#
afcfb1bcd9b105d4be4f13b9dc85a0f98', 1, 'NEWMAT::CroutMatrix::lubksb()'], ['../
class_n_e_w_m_a_t_1_1_band_l_u_matrix.html#
aa6b6344001665e9241bd7a3e6564c7da', 1, 'NEWMAT::BandLUMatrix::lubksb()']]],
 ['ludcomp', ['LUDcomp', ['../namespace_go.html#a77c8de2bdc12a47b62ef5c306ac0e46d', 1,
Go']],
 ['lusolvesystem', ['LUSolveSystem', ['../namespace_go.html#
afd9ea9a56159c9a5841819ebf13280b8', 1, 'Go']]]
]
```

Definition at line 88 of file functions\_c.js.

## 30.1531.1.7 var searchData

## Initial value:

```

=
[
 ['labelbdynode', ['labelBdyNode', ['../class_pr_organized_points.html#acab5b170fefc327f7db0aac6dac3347b', 1,
 'PrOrganizedPoints']],
 ['labelnode', ['labelNode', ['../class_pr_organized_points.html#a02406e295c174e57548497a85e987988', 1,
 'PrOrganizedPoints::labelNode(int i, int ic, std::vector<int > &component) const'], ['
 ../class_pr_organized_points.html#ae7d58624e3dabfac3af4a976ecd46ca2', 1, 'PrOrganizedPoints::labelNode(int i, int ic, int
 &index, std::vector<int > &component, std::vector<int > &newIndex) const']],
 ['lackingparameter', ['lackingParameter', ['
 ../class_go_1_1_intersection_pool.html#a204c700b87c2881793dc64850c98ca03', 1, 'Go::IntersectionPool']],
 ['lackingparametervalue', ['lackingParameterValue', ['
 ../class_go_1_1_intersection_pool.html#afd672fd1d2375a2c42b178c9aa139363a', 1, 'Go::IntersectionPool']],
 ['largestmultinline', ['largestMultInLine', ['
 ../class_go_1_1_mesh2_d.html#a6f5984896001959aacf751265b29208c', 1, 'Go::Mesh2D']],
 ['last_5fnonlarger_5fknotvalue_5fix', ['last_nonlarger_knotvalue_ix', ['
 ../namespace_go_1_1_mesh2_d_utils.html#a8a567523d54d76c3a532ce9a9749f58e', 1, 'Go::Mesh2DUtils']],
 ['lastadded', ['lastAdded', ['../class_go_1_1ft_point_set.html#adb78acd38e793f16f100844363fec50d', 1,
 'Go::ftPointSet']],
 ['lastknotinterval', ['lastKnotInterval', ['
 ../class_go_1_1_bspline_basis.html#a5f899f19d8953ff88810150adc2777b7', 1, 'Go::BsplineBasis']],
 ['lastmeshvecix', ['lastMeshVecIx', ['../class_go_1_1_mesh2_d.html#ad5473996d004e4f93f357f2a1049333c', 1,
 'Go::Mesh2D']],
 ['leavelevel', ['leaveLevel', ['../class_pr_threshold.html#a816cbbad25faf259db527bf3c302bc5e', 1,
 'PrThreshold']],
 ['left', ['Left', ['../class_rect_matrix_row.html#a459e0abe296fe1d3dc3000eda2e5a8f4', 1,
 'RectMatrixRow::Left()'], ['../class_rect_matrix_col.html#a749656e672061aca5b6be61cb2cel1f2d', 1, 'RectMatrixCol::Left()']],
 ['length', ['length', ['../class_go_1_1_bounded_curve.html#a52dff472a3423061bf038ce69699b380', 1,
 'Go::BoundedCurve::length()'], ['../class_go_1_1_circle.html#a50ca38850dcfdadc2d7fd76e8571dd57', 1, 'Go::Circle::length()
 '], ['../class_go_1_1_curve_on_surface.html#a86222695e99d89c9e6931fd1c56fca69', 1, 'Go::CurveOnSurface::length()
 '], ['../class_go_1_1_ellipse.html#ab462f16f96d55a31758668ced9529eb2', 1, 'Go::Ellipse::length()'], ['
 ../class_go_1_1_hyperbola.html#adfe71112434a8ac87efe0d62aa0138b1', 1, 'Go::Hyperbola::length()'], ['
 ../class_go_1_1_line.html#a71b60c4fe361df79b87a187580b60723', 1, 'Go::Line::length()'], ['
 ../class_go_1_1_parabola.html#a0d5884e48828c244112446d4861468f5', 1, 'Go::Parabola::length()'], ['
 ../class_go_1_1_param_curve.html#a03c0c3d367d97ce42b0d71fff4f2e3ccc', 1, 'Go::ParamCurve::length(double tol)=0'], ['
 ../class_go_1_1_param_curve.html#ae67c1874cfe426ea988e51bae328bbf4', 1, 'Go::ParamCurve::length(double tol, double tstart,
 double tstop)'], ['../class_go_1_1_spline_curve.html#ac752958b03163ea580ff752d184f73bc', 1, 'Go::SplineCurve::length()'], ['
 ../class_go_1_1_array.html#a1602961ef1753005f8bb4602f2f6e6dd', 1, 'Go::Array::length()'], ['
 ../class_go_1_1_point.html#a4b22f412d01dfa566f973ae2f52ffbd5', 1, 'Go::Point::length()'], ['../class_vector3t.html#ad4f2c44
 vector3t::length()'], ['../class_go_1_1_curve_on_volume.html#ad58a92a8b5899727209bdbe9ee0fce21', 1,
 'Go::CurveOnVolume::length()'], ['../class_matrix_row_col.html#a0219c2695986db07f74871eb2ff6b24f', 1, 'MatrixRowCol::Length
 '], ['../class_matrix_row_col.html#a244b92e34d0e34fe4dff2cfff19ef23d', 1, 'MatrixRowCol::Length(int i)']],
 ['length2', ['length2', ['../class_go_1_1_array.html#a54cb2bbf81e87d380fb58716d11660ff', 1,
 'Go::Array::length2()'], ['../class_go_1_1_point.html#a9b757aefb2094e95efc92a2e46e0c108', 1, 'Go::Point::length2()
 ']],
 ['length_5ferror', ['Length_error', ['
 ../class_r_b_d___c_o_m_m_o_n_l_l_length_error.html#aaa6de69c775aee8e7fbf15521ffa5733', 1, 'RBD_COMMON::Length_error']],
 ['length_5fpolygonal_5fpath', ['length_polygonal_path', ['
 ../pr_geodesics_8h.html#ab184bcc756b99d5c32b7480c0aea576d', 1, 'length_polygonal_path(const vector<int > &vector,
 Vector3D &start, Vector3D &end)']],

```

Definition at line 1 of file functions\_c.js.

## 30.1532 doc/html/search/functions\_d.js File Reference

## Variables

- var [searchData](#)
- example [cpp](#)

## 30.1532.1 Variable Documentation

30.1532.1.1 example [sisl\\_view\\_demo.cpp](#)

Definition at line 9 of file functions\_d.js.

### 30.1532.1.2 var searchData

#### Initial value:

```
=
[
 ['m3_5fdet', ['m3_det', ['./gv_utilities_8h.html#a15f8dd61e74e6e8637e610a774df9607', 1, 'gvUtilities.h']],
 ['m3_5fidentity', ['m3_identity', ['./gv_utilities_8h.html#a6091a5b9ab587c9a4b692565e6e1d6c6', 1, 'gvUtilities.h']],
 ['m3_5finverse', ['m3_inverse', ['./gv_utilities_8h.html#a3347b853ddaa80d8f92360e764b433e8', 1, 'gvUtilities.h']],
 ['m4_5fdet', ['m4_det', ['./gv_utilities_8h.html#acdaa244d22eb4f9c9c0bda4b5d94e5b3', 1, 'gvUtilities.h']],
 ['m4_5finverse', ['m4_inverse', ['./gv_utilities_8h.html#a708f7916c5b2c579c1096435f8f481f4', 1, 'gvUtilities.h']],
 ['m4_5fsubmat', ['m4_submat', ['./gv_utilities_8h.html#a94b4064202801e77aa5bd9b975b37d0c', 1, 'gvUtilities.h']],
 ['main', ['main', ['./example_8cpp.html#ae66f6b31b5ad750f1fe042a706a4e3d4', 1, 'main():
```

Definition at line 1 of file functions\_d.js.

## 30.1533 doc/html/search/functions\_e.js File Reference

### Variables

- var [searchData](#)
- [sisl h](#)

### 30.1533.1 Variable Documentation

#### 30.1533.1.1 sisl h

Definition at line 21 of file functions\_e.js.

#### 30.1533.1.2 var searchData

#### Initial value:

```
=
[
 ['n1', ['n1', ['./class_pr_triangle.html#a96449b1505011a047203fc32e8f58df7', 1, 'PrTriangle::n1() const'], [
 ['./class_pr_triangle.html#a9acbd361babe718867639b75e9f88edd', 1, 'PrTriangle::n1()']],
 ['n2', ['n2', ['./class_pr_triangle.html#a28836174b41f58634b030ca78a9df0f8', 1, 'PrTriangle::n2() const'], [
 ['./class_pr_triangle.html#a74e37eb7e0d02c4f33bb80db4a282eb2', 1, 'PrTriangle::n2()']],
 ['n3', ['n3', ['./class_pr_triangle.html#ac5c483c7358d77184ae21b9120b664fe', 1, 'PrTriangle::n3() const'], [
 ['./class_pr_triangle.html#ad38195d658ed67a5493c31cb3bded64f', 1, 'PrTriangle::n3()']],
 ['n_5fboxes', ['n_boxes', [
 ['./class_go_1_lbox_structuring_1_1_bounding_box_structure.html#a60971e92b7351db884394bb924002f99', 1, 'Go::boxStructuring'],
 ['n_5fsurfaces', ['n_surfaces', [
 ['./class_go_1_lbox_structuring_1_1_bounding_box_structure.html#aa51b875860291f92adb86b4034bdef4e', 1, 'Go::boxStructuring'],
 ['n_5fvoxels_5fx', ['n_voxels_x', [
 ['./class_go_1_lbox_structuring_1_1_bounding_box_structure.html#af796d20691409c02d3d3831106917bba', 1, 'Go::boxStructuring'],
 ['n_5fvoxels_5fy', ['n_voxels_y', [
 ['./class_go_1_lbox_structuring_1_1_bounding_box_structure.html#a70d433f73cb7086fd851e190a3ede696', 1, 'Go::boxStructuring'],
 ['n_5fvoxels_5fz', ['n_voxels_z', [
 ['./class_go_1_lbox_structuring_1_1_bounding_box_structure.html#a6e6bec0d1412870fabcf13a14c244e3d', 1, 'Go::boxStructuring'],
 ['name', ['name', ['./classgv_group.html#a6032ce853c50bb71260ecf96d90720cc', 1, 'gvGroup']],
 ['narrowregion', ['narrowRegion', ['./class_go_1_l_face_set_quality.html#a57cdf98fc6cc7eb3b1e8c9890208c6d2', 1, 'Go::FaceSetQuality::narrowRegion()'], [
 ['./class_go_1_l_model_quality.html#a0279066977434f3f9abc88676a8b7d3a', 1, 'Go::ModelQuality::narrowRegion()']],
 ['ncols', ['Ncols', ['./class_n_e_w_m_a_t_1_1_general_matrix.html#a1f41c8345ef34da8de3ebce655874564', 1, 'NEWMAT::GeneralMatrix']],
```

```
['negadd', ['NegAdd', ['../class_n_e_w_m_a_t_l_l_general_matrix.html#a34d6cfbafa85353e633e591aea41b9ec', 1, '
 NEWMAT::GeneralMatrix::NegAdd(GeneralMatrix *, Real)'], ['
 ../class_n_e_w_m_a_t_l_l_general_matrix.html#a89c69d4e1cec85b64092c9849b564466', 1, 'NEWMAT::GeneralMatrix::NegAdd(Real)'],
 ../class_matrix_row_col.html#a41b3194775a1aldcfd560ec52b3eeffa', 1, 'MatrixRowCol::NegAdd()']]],
['negate', ['Negate', ['../class_n_e_w_m_a_t_l_l_general_matrix.html#aec996470a2a281abdafa83a04395ab15', 1, '
 NEWMAT::GeneralMatrix::Negate(GeneralMatrix *)'], ['
 ../class_n_e_w_m_a_t_l_l_general_matrix.html#a65732aecbb9e92b9fb3cfcf828542dca', 1, 'NEWMAT::GeneralMatrix::Negate()'], ['
 ../class_matrix_row_col.html#a85aedb535b4ac31d17b3991093eccdb3', 1, 'MatrixRowCol::Negate()'], ['
 ../class_rect_matrix_row_col.html#a5c6d68fb80759c68d09a0b5d12b84822', 1, 'RectMatrixRowCol::Negate()']]],
['negatedmatrix', ['NegatedMatrix', ['
 ../class_n_e_w_m_a_t_l_l_negated_matrix.html#a701a017652a4ec29860401e3f2292a30', 1, 'NEWMAT::NegatedMatrix']]],
['negativeproj', ['negativeProj', ['
 ../namespace_go_l_l_geometry_tools.html#a4a00d270de7dc569dd5649d41283b847', 1, 'Go::GeometryTools']]],
['negshiftedmatrix', ['NegShiftedMatrix', ['
 ../class_n_e_w_m_a_t_l_l_neg_shifted_matrix.html#a759d2cc85daee8a6f22f9f07559fc272', 1, 'NEWMAT::NegShiftedMatrix']]],
['new', ['New', ['../class_n_e_w_m_a_t_l_l_matrix_type.html#a7cec58150cf70497bbe0dc9807d8aa7d', 1, '
 NEWMAT::MatrixType::New() const'], ['../class_n_e_w_m_a_t_l_l_matrix_type.html#adab87f8bdd8f148819ea880e303f9ea8', 1, '
 NEWMAT::MatrixType::New(int, int, BaseMatrix *) const']]],
['new_5fparametrize3d', ['new_parametrize3d', ['
 ../class_pr_parametrize_int.html#ac6e89a09aba3bf0918ce57ed3a4d453a', 1, 'PrParametrizeInt']]],
['newbox', ['newbox', ['../sisl_8h.html#aa6461e5f7144bbaf22c9f76da8aalb57', 1, 'newbox():
```

Definition at line 1 of file functions\_e.js.

## 30.1534 doc/html/search/functions\_f.js File Reference

### Variables

- var [searchData](#)
- StreamUtils [h](#)
- StreamUtils int [idx](#)
- StreamUtils int int [idx const](#)
- StreamUtils int int [idx](#) int i int i
- StreamUtils int int [idx](#) int i int int i int int i int int num int [num](#)

### 30.1534.1 Variable Documentation

**30.1534.1.1 StreamUtils int int [idx](#) int i int int i int int i int int num int int num int int num int int num int int i int int i const**  
[bound]

Definition at line 4 of file functions\_f.js.

**30.1534.1.2 StreamUtils [h](#)**

Definition at line 4 of file functions\_f.js.

**30.1534.1.3 StreamUtils int int [idx](#) int i int int i int int i int int num int int num int int num int int num int int int i int i**  
[bound]

Definition at line 4 of file functions\_f.js.

**30.1534.1.4 StreamUtils int [idx](#)** [bound]

Definition at line 4 of file functions\_f.js.

30.1534.1.5 `StreamUtils` `int int idx int i int int i int int i int int num int int num int int num int int num int num` `[bound]`

Definition at line 4 of file `functions_f.js`.

30.1534.1.6 `var searchData`

**Initial value:**

```
=
[
 ['object', ['object', ['./classgv_data.html#a13c47f9abfb21d7d5ef58957b229f70e', 1, 'gvData::object(int i)'],
 ['./classgv_data.html#a28402d21f6e677b96197b32feb40d8e4', 1, 'gvData::object(int i) const ']]],
 ['object_5ffrom_5fstream', ['object_from_stream', ['./
 ../_stream_utils_8h.html#a4894fce509ab613c659adfb8c6e4f3a4', 1, 'object_from_stream(std::istream &is, T &obj):
```

Definition at line 1 of file `functions_f.js`.

## 30.1535 `doc/html/search/groups_0.js` File Reference

**Variables**

- `var searchData`

### 30.1535.1 Variable Documentation

30.1535.1.1 `var searchData`

**Initial value:**

```
=
[
 ['rbd_20common_20library', ['RBD common library', ['./group__rbd__common.html', 1, '']]
]
```

Definition at line 1 of file `groups_0.js`.

## 30.1536 `doc/html/search/namespaces_0.js` File Reference

**Variables**

- `var searchData`

### 30.1536.1 Variable Documentation

30.1536.1.1 `var searchData`

Definition at line 1 of file `namespaces_0.js`.

## 30.1537 doc/html/search/namespaces\_1.js File Reference

### Variables

- var [searchData](#)

#### 30.1537.1 Variable Documentation

##### 30.1537.1.1 var searchData

###### Initial value:

```
=
[
 ['hed', ['hed', ['../namespacehed.html', 1, '']]],
 ['hetriang', ['hetriang', ['../namespacehetriang.html', 1, '']]]
]
```

Definition at line 1 of file namespaces\_1.js.

## 30.1538 doc/html/search/namespaces\_2.js File Reference

### Variables

- var [searchData](#)

#### 30.1538.1 Variable Documentation

##### 30.1538.1.1 var searchData

###### Initial value:

```
=
[
 ['newmat', ['NEWMAT', ['../namespace_n_e_w_m_a_t.html', 1, '']]]
]
```

Definition at line 1 of file namespaces\_2.js.

## 30.1539 doc/html/search/namespaces\_3.js File Reference

### Variables

- var [searchData](#)

### 30.1539.1 Variable Documentation

#### 30.1539.1.1 var searchData

**Initial value:**

```
=
[
 ['rbd_5fcommon', ['RBD_COMMON', ['../namespace_r_b_d___c_o_m_m_o_n.html', 1, '']]],
 ['rbd_5flibraries', ['RBD_LIBRARIES', ['../namespace_r_b_d___l_i_b_r_a_r_i_e_s.html', 1, '']]]
]
```

Definition at line 1 of file namespaces\_3.js.

### 30.1540 doc/html/search/namespaces\_4.js File Reference

**Variables**

- var [searchData](#)

#### 30.1540.1 Variable Documentation

##### 30.1540.1.1 var searchData

**Initial value:**

```
=
[
 ['std', ['std', ['../namespacestd.html', 1, '']]]
]
```

Definition at line 1 of file namespaces\_4.js.

### 30.1541 doc/html/search/namespaces\_5.js File Reference

**Variables**

- var [searchData](#)

#### 30.1541.1 Variable Documentation

##### 30.1541.1.1 var searchData

**Initial value:**

```
=
[
 ['ttl', ['ttl', ['../namespacettl.html', 1, '']]],
 ['ttl_5fconstr', ['ttl_constr', ['../namespacettl__constr.html', 1, '']]],
 ['ttl_5futil', ['ttl_util', ['../namespacettl__util.html', 1, '']]]
]
```

Definition at line 1 of file namespaces\_5.js.



## 30.1542 doc/html/search/pages\_0.js File Reference

### Variables

- var [searchData](#)

#### 30.1542.1 Variable Documentation

##### 30.1542.1.1 var searchData

###### Initial value:

```
=
[
 ['application_20programming_20interface_20to_20ttl_20_28api_29', ['Application Programming Interface to
 TTL (API)', ['../api.html', 1, '']]]
]
```

Definition at line 1 of file pages\_0.js.

## 30.1543 doc/html/search/pages\_1.js File Reference

### Variables

- var [searchData](#)

#### 30.1543.1 Variable Documentation

##### 30.1543.1.1 var searchData

###### Initial value:

```
=
[
 ['bug_20list', ['Bug List', ['../bug.html', 1, '']]]
]
```

Definition at line 1 of file pages\_1.js.

## 30.1544 doc/html/search/pages\_2.js File Reference

### Variables

- var [searchData](#)

### 30.1544.1 Variable Documentation

#### 30.1544.1.1 var searchData

##### Initial value:

```
=
[
 ['example_20using_20ttl_20and_20the_20half_20edge_20data_20structure', ['Example using TTL and the
 half-edge data structure', ['../hesimplest.html', 1, '']]
]
```

Definition at line 1 of file pages\_2.js.

### 30.1545 doc/html/search/pages\_3.js File Reference

#### Variables

- var [searchData](#)

### 30.1545.1 Variable Documentation

#### 30.1545.1.1 var searchData

##### Initial value:

```
=
[
 ['gotools_20compositemodel', ['GoTools CompositeModel', ['../compositemodel.html', 1, '']],
 ['gotools_20core', ['GoTools Core', ['../geometry_doc.html', 1, '']],
 ['gotools_20example_20files', ['GoTools example files', ['../gotools_examples.html', 1, '']],
 ['gotools_20igeslib', ['GoTools Igeslib', ['../igeslib.html', 1, '']],
 ['gotools_20implicitization', ['GoTools Implicitization', ['../implicitization.html', 1, '']],
 ['gotools_20library', ['GoTools Library', ['../index.html', 1, '']],
 ['gotools_20intersections', ['GoTools Intersections', ['../intersections.html', 1, '']],
 ['gotools_20isogeometricmodel', ['GoTools IsogeometricModel', ['../isogeometric_model.html', 1, '']],
 ['gotools_20lrsplines2d', ['GoTools LRsplines2D', ['../lrsplines2d.html', 1, '']],
 ['gotools_20parametrization', ['GoTools Parametrization', ['../parametrization.html', 1, '']],
 ['gotools_20qualitymodule', ['GoTools QualityModule', ['../qualitymodule.html', 1, '']],
 ['gotools_20topology', ['GoTools Topology', ['../topology.html', 1, '']],
 ['gotools_20trivariate', ['GoTools Trivariate', ['../trivariate.html', 1, '']],
 ['gotools_20trivariatemodel', ['GoTools Trivariatemodel', ['../trivariatemodel.html', 1, '']],
 ['gotools_20viewlib', ['GoTools Viewlib', ['../viewlib.html', 1, '']]
]
```

Definition at line 1 of file pages\_3.js.

### 30.1546 doc/html/search/pages\_4.js File Reference

#### Variables

- var [searchData](#)

### 30.1546.1 Variable Documentation

#### 30.1546.1.1 var searchData

##### Initial value:

```
=
[
 ['the_20filter_20routines_20main_20page', ['The filter routines main page', ['./filters_mainpage.html', 1, '
 ']]],
 ['the_20half_20dedge_20data_20structure_20and_20adaption_20to_20ttl', ['The Half-Edge Data Structure and
 Adaption to TTL', ['./halfedge.html', 1, '']]],
 ['the_20g2_20dformat_20c_20gotools_20file_20format_20for_20geometry_20entities', ['The g2-format, GoTools
 file format for geometry entities', ['./streamable_doc.html', 1, '']]]
]
```

Definition at line 1 of file pages\_4.js.

## 30.1547 doc/html/search/related\_0.js File Reference

### Variables

- var [searchData](#)

### 30.1547.1 Variable Documentation

#### 30.1547.1.1 var searchData

##### Initial value:

```
=
[
 ['addedmatrix', ['AddedMatrix', [
 './class_n_e_w_m_a_t_l_l_base_matrix.html#aa69c602d29e1d8c3cddcb3279bab7fa8', 1, 'NEWMAT::BaseMatrix::AddedMatrix()'], [
 './class_n_e_w_m_a_t_l_l_general_matrix.html#aa69c602d29e1d8c3cddcb3279bab7fa8', 1, 'NEWMAT::GeneralMatrix::AddedMatrix()']
]]
]
```

Definition at line 1 of file related\_0.js.

## 30.1548 doc/html/search/related\_1.js File Reference

### Variables

- var [searchData](#)

## 30.1548.1 Variable Documentation

### 30.1548.1.1 var searchData

#### Initial value:

```
=
[
 ['bandmatrix', ['BandMatrix', ['../class_n_e_w_m_a_t_l_l_base_matrix.html#a86c113dda73d85b0f706f1978e86e541', 1, 'NEWMAT::BaseMatrix::BandMatrix()'], ['../class_n_e_w_m_a_t_l_l_general_matrix.html#a86c113dda73d85b0f706f1978e86e541', 1, 'NEWMAT::GeneralMatrix::BandMatrix()'], ['baseexception', ['BaseException', ['../class_r_b_d___o_m_m_o_n_l_l_tracer.html#acc29c9a717ece42fb764f4a48a04b6cb', 1, 'RBD_COMMON::Tracer']]], ['basematrix', ['BaseMatrix', ['../class_n_e_w_m_a_t_l_l_general_matrix.html#afcc50b811bd45d547cae53941018fdde', 1, 'NEWMAT::GeneralMatrix::BaseMatrix()'], ['../class_n_e_w_m_a_t_l_l_generic_matrix.html#afcc50b811bd45d547cae53941018fdde', 1, 'NEWMAT::GenericMatrix::BaseMatrix()'], ['../class_n_e_w_m_a_t_l_l_multiplied_matrix.html#afcc50b811bd45d547cae53941018fdde', 1, 'NEWMAT::MultipliedMatrix::BaseMatrix()'], ['../class_n_e_w_m_a_t_l_l_added_matrix.html#afcc50b811bd45d547cae53941018fdde', 1, 'NEWMAT::AddedMatrix::BaseMatrix()'], ['../class_n_e_w_m_a_t_l_l_s_p_matrix.html#afcc50b811bd45d547cae53941018fdde', 1, 'NEWMAT::SPMatrix::BaseMatrix()'], ['../class_n_e_w_m_a_t_l_l_k_p_matrix.html#afcc50b811bd45d547cae53941018fdde', 1, 'NEWMAT::KPMatrix::BaseMatrix()'], ['../class_n_e_w_m_a_t_l_l_concatenated_matrix.html#afcc50b811bd45d547cae53941018fdde', 1, 'NEWMAT::ConcatenatedMatrix::BaseMatrix()'], ['../class_n_e_w_m_a_t_l_l_stacked_matrix.html#afcc50b811bd45d547cae53941018fdde', 1, 'NEWMAT::StackedMatrix::BaseMatrix()'], ['../class_n_e_w_m_a_t_l_l_solved_matrix.html#afcc50b811bd45d547cae53941018fdde', 1, 'NEWMAT::SolvedMatrix::BaseMatrix()'], ['../class_n_e_w_m_a_t_l_l_subtracted_matrix.html#afcc50b811bd45d547cae53941018fdde', 1, 'NEWMAT::SubtractedMatrix::BaseMatrix()'], ['../class_n_e_w_m_a_t_l_l_shifted_matrix.html#afcc50b811bd45d547cae53941018fdde', 1, 'NEWMAT::ShiftedMatrix::BaseMatrix()'], ['../class_n_e_w_m_a_t_l_l_neg_shifted_matrix.html#afcc50b811bd45d547cae53941018fdde', 1, 'NEWMAT::NegShiftedMatrix::BaseMatrix()'], ['../class_n_e_w_m_a_t_l_l_scaled_matrix.html#afcc50b811bd45d547cae53941018fdde', 1, 'NEWMAT::ScaledMatrix::BaseMatrix()'], ['../class_n_e_w_m_a_t_l_l_negated_matrix.html#afcc50b811bd45d547cae53941018fdde', 1, 'NEWMAT::NegatedMatrix::BaseMatrix()'], ['../class_n_e_w_m_a_t_l_l_transposed_matrix.html#afcc50b811bd45d547cae53941018fdde', 1, 'NEWMAT::TransposedMatrix::BaseMatrix()'], ['../class_n_e_w_m_a_t_l_l_reversed_matrix.html#afcc50b811bd45d547cae53941018fdde', 1, 'NEWMAT::ReversedMatrix::BaseMatrix()'], ['../class_n_e_w_m_a_t_l_l_inverted_matrix.html#afcc50b811bd45d547cae53941018fdde', 1, 'NEWMAT::InvertedMatrix::BaseMatrix()'], ['../class_n_e_w_m_a_t_l_l_rowed_matrix.html#afcc50b811bd45d547cae53941018fdde', 1, 'NEWMAT::RowedMatrix::BaseMatrix()'], ['../class_n_e_w_m_a_t_l_l_coled_matrix.html#afcc50b811bd45d547cae53941018fdde', 1, 'NEWMAT::ColedMatrix::BaseMatrix()'], ['../class_n_e_w_m_a_t_l_l_diaged_matrix.html#afcc50b811bd45d547cae53941018fdde', 1, 'NEWMAT::DiagedMatrix::BaseMatrix()'], ['../class_n_e_w_m_a_t_l_l_mated_matrix.html#afcc50b811bd45d547cae53941018fdde', 1, 'NEWMAT::MatedMatrix::BaseMatrix()'], ['../class_n_e_w_m_a_t_l_l_return_matrix_x.html#afcc50b811bd45d547cae53941018fdde', 1, 'NEWMAT::ReturnMatrixX::BaseMatrix()'], ['../class_n_e_w_m_a_t_l_l_get_sub_matrix.html#afcc50b811bd45d547cae53941018fdde', 1, 'NEWMAT::GetSubMatrix::BaseMatrix()'], ['../class_n_e_w_m_a_t_l_l_linear_equation_solver.html#afcc50b811bd45d547cae53941018fdde', 1, 'NEWMAT::LinearEquationSolver']
]
```

Definition at line 1 of file related\_1.js.

## 30.1549 doc/html/search/related\_2.js File Reference

### Variables

- var [searchData](#)

## 30.1549.1 Variable Documentation

### 30.1549.1.1 var searchData

#### Initial value:

```

=
[
 ['coledmatrix', ['ColedMatrix', ['
 ../class_n_e_w_m_a_t_l_l_base_matrix.html#adfa43567d33d5b9a9882734cddc5e143', 1, 'NEWMAT::BaseMatrix::ColedMatrix()'], ['
 ../class_n_e_w_m_a_t_l_l_general_matrix.html#adfa43567d33d5b9a9882734cddc5e143', 1, 'NEWMAT::GeneralMatrix::ColedMatrix()'],
 ['columnvector', ['ColumnVector', ['
 ../class_n_e_w_m_a_t_l_l_base_matrix.html#a2a47af7bd92cf6d5a8d67ffa3957db2b', 1, 'NEWMAT::BaseMatrix::ColumnVector()'], ['
 ../class_n_e_w_m_a_t_l_l_general_matrix.html#a2a47af7bd92cf6d5a8d67ffa3957db2b', 1, 'NEWMAT::GeneralMatrix::ColumnVector()'],
 ['compare', ['Compare', ['../class_n_e_w_m_a_t_l_l_matrix_type.html#a6ba54373b9b6c990f86c20c24764f8a7', 1, '
 NEWMAT::MatrixType']]],
 ['complexscale', ['ComplexScale', ['../class_rect_matrix_row_col.html#ae131888d49cb5f48944aa641f5998af5', 1, '
 RectMatrixRowCol::ComplexScale()'], ['../class_rect_matrix_col.html#ae131888d49cb5f48944aa641f5998af5', 1, '
 RectMatrixCol::ComplexScale()']]],
 ['concatenatedmatrix', ['ConcatenatedMatrix', ['
 ../class_n_e_w_m_a_t_l_l_base_matrix.html#a18c145a190fdb6a65a61170be5585dc89', 1, 'NEWMAT::BaseMatrix::ConcatenatedMatrix()'],
 ../class_n_e_w_m_a_t_l_l_general_matrix.html#a18c145a190fdb6a65a61170be5585dc89', 1, 'NEWMAT::GeneralMatrix::ConcatenatedMatrix()'],
 ['constructintersectioncurve', ['constructIntersectionCurve', ['
 ../class_go_l_l_intersection_curve.html#a6ab34095fc2901f18aff0d440f5f9ea5', 1, 'Go::IntersectionCurve::constructIntersectionCurve()'],
 ../class_go_l_l_degenerated_intersection_curve.html#a6ab34095fc2901f18aff0d440f5f9ea5', 1, '
 Go::DegeneratedIntersectionCurve::constructIntersectionCurve()'], ['
 ../class_go_l_l_non_evaluable_intersection_curve.html#a6ab34095fc2901f18aff0d440f5f9ea5', 1, 'Go::NonEvaluableIntersectionCurve::constructIntersectionCurve()'],
 ../class_go_l_l_isoparametric_intersection_curve.html#a6ab34095fc2901f18aff0d440f5f9ea5', 1, '
 Go::IsoparametricIntersectionCurve::constructIntersectionCurve()'], ['
 ../class_go_l_l_interpolated_intersection_curve.html#a6ab34095fc2901f18aff0d440f5f9ea5', 1, 'Go::InterpolatedIntersectionCurve::constructIntersectionCurve()'],
 ['conv2', ['conv2', ['../classvector3t.html#a305b357d7ef8a52418e1e04c137ed1b8', 1, 'vector3t']]],
 ['cosangle', ['cosangle', ['../classvector3t.html#ac7f417d59352ac50bb3dce6c0c054879', 1, 'vector3t']]],
 ['croutmatrix', ['CroutMatrix', ['
 ../class_n_e_w_m_a_t_l_l_base_matrix.html#a260688c58e06fb98f68aa624c865c435', 1, 'NEWMAT::BaseMatrix::CroutMatrix()'], ['
 ../class_n_e_w_m_a_t_l_l_general_matrix.html#a260688c58e06fb98f68aa624c865c435', 1, 'NEWMAT::GeneralMatrix::CroutMatrix()'],
]
]

```

Definition at line 1 of file `related_2.js`.

## 30.1550 doc/html/search/related\_3.js File Reference

### Variables

- var [searchData](#)

### 30.1550.1 Variable Documentation

#### 30.1550.1.1 var searchData

#### Initial value:

```

=
[
 ['diagedmatrix', ['DiagedMatrix', ['
 ../class_n_e_w_m_a_t_l_l_base_matrix.html#ae98438838c66748c639f9646a8658a8f', 1, 'NEWMAT::BaseMatrix::DiagedMatrix()'], ['
 ../class_n_e_w_m_a_t_l_l_general_matrix.html#ae98438838c66748c639f9646a8658a8f', 1, 'NEWMAT::GeneralMatrix::DiagedMatrix()'],
 ['diagonalmatrix', ['DiagonalMatrix', ['
 ../class_n_e_w_m_a_t_l_l_base_matrix.html#a9abdda7fd7cf3908adadd1587f4ed77e', 1, 'NEWMAT::BaseMatrix::DiagonalMatrix()'],
 ../class_n_e_w_m_a_t_l_l_general_matrix.html#a9abdda7fd7cf3908adadd1587f4ed77e', 1, 'NEWMAT::GeneralMatrix::DiagonalMatrix()'],
 ['dotprod', ['DotProd', ['../class_matrix_row_col.html#a453803ca6490cb877d3a0dc3fe802c63', 1, 'MatrixRowCol']]],
 ['dotproduct', ['DotProduct', ['../class_n_e_w_m_a_t_l_l_matrix.html#a9e8ce3d9f963c09ee55fd8e84d540973', 1, '
 NEWMAT::Matrix']]]
]

```

Definition at line 1 of file `related_3.js`.

## 30.1551 doc/html/search/related\_4.js File Reference

### Variables

- var [searchData](#)

### 30.1551.1 Variable Documentation

#### 30.1551.1.1 var searchData

##### Initial value:

```
=
[
 ['facesetquality', ['FaceSetQuality', ['
 ../class_go_l_l_quality_results.html#ae820d513170db3b6b7ce7887b8a0d4dc', 1, 'Go::QualityResults']],
 ['facesetrepair', ['FaceSetRepair', ['
 ../class_go_l_l_quality_results.html#a18e8ba52e738ee5a12e03dab6f012c27', 1, 'Go::QualityResults']]
]
```

Definition at line 1 of file [related\\_4.js](#).

## 30.1552 doc/html/search/related\_5.js File Reference

### Variables

- var [searchData](#)

### 30.1552.1 Variable Documentation

#### 30.1552.1.1 var searchData

##### Initial value:

```
=
[
 ['generalmatrix', ['GeneralMatrix', ['
 ../class_n_e_w_m_a_t_l_l_base_matrix.html#abaf0969f44c36896ac567a558a330771', 1, 'NEWMAT::BaseMatrix::GeneralMatrix()'], [
 ../class_n_e_w_m_a_t_l_l_multiplied_matrix.html#abaf0969f44c36896ac567a558a330771', 1, 'NEWMAT::MultipliedMatrix::GeneralMatrix()'], [
 ../class_n_e_w_m_a_t_l_l_added_matrix.html#abaf0969f44c36896ac567a558a330771', 1, 'NEWMAT::AddedMatrix::GeneralMatrix()'], [
 ../class_n_e_w_m_a_t_l_l_s_p_matrix.html#abaf0969f44c36896ac567a558a330771', 1, 'NEWMAT::SPMatrix::GeneralMatrix()'], [
 ../class_n_e_w_m_a_t_l_l_k_p_matrix.html#abaf0969f44c36896ac567a558a330771', 1, 'NEWMAT::KPMatrix::GeneralMatrix()'], [
 ../class_n_e_w_m_a_t_l_l_concatenated_matrix.html#abaf0969f44c36896ac567a558a330771', 1, 'NEWMAT::ConcatenatedMatrix::GeneralMatrix()'], [
 ../class_n_e_w_m_a_t_l_l_stacked_matrix.html#abaf0969f44c36896ac567a558a330771', 1, 'NEWMAT::StackedMatrix::GeneralMatrix()'], [
 ../class_n_e_w_m_a_t_l_l_subtracted_matrix.html#abaf0969f44c36896ac567a558a330771', 1, 'NEWMAT::SubtractedMatrix::GeneralMatrix()'], [
 ../class_n_e_w_m_a_t_l_l_shifted_matrix.html#abaf0969f44c36896ac567a558a330771', 1, 'NEWMAT::ShiftedMatrix::GeneralMatrix()'], [
 ../class_n_e_w_m_a_t_l_l_neg_shifted_matrix.html#abaf0969f44c36896ac567a558a330771', 1, 'NEWMAT::NegShiftedMatrix::GeneralMatrix()'], [
 ../class_n_e_w_m_a_t_l_l_scaled_matrix.html#abaf0969f44c36896ac567a558a330771', 1, 'NEWMAT::ScaledMatrix::GeneralMatrix()'], [
 ../class_n_e_w_m_a_t_l_l_matrix_input.html#abaf0969f44c36896ac567a558a330771', 1, 'NEWMAT::MatrixInput::GeneralMatrix()']]],
 ['genericmatrix', ['GenericMatrix', ['
 ../class_n_e_w_m_a_t_l_l_base_matrix.html#a4ecabb4ff1be9d3dd3fd895b25fd7237', 1, 'NEWMAT::BaseMatrix::GenericMatrix()'], [
 ../class_n_e_w_m_a_t_l_l_general_matrix.html#a4ecabb4ff1be9d3dd3fd895b25fd7237', 1, 'NEWMAT::GeneralMatrix::GenericMatrix()'], [
 ../class_n_e_w_m_a_t_l_l_multiplied_matrix.html#a4ecabb4ff1be9d3dd3fd895b25fd7237', 1, 'NEWMAT::MultipliedMatrix::GenericMatrix()'], [
 ../class_n_e_w_m_a_t_l_l_added_matrix.html#a4ecabb4ff1be9d3dd3fd895b25fd7237', 1, 'NEWMAT::AddedMatrix::GenericMatrix()'], [
 ../class_n_e_w_m_a_t_l_l_s_p_matrix.html#a4ecabb4ff1be9d3dd3fd895b25fd7237', 1, 'NEWMAT::SPMatrix::GenericMatrix()'], [
 ../class_n_e_w_m_a_t_l_l_k_p_matrix.html#a4ecabb4ff1be9d3dd3fd895b25fd7237', 1, 'NEWMAT::KPMatrix::GenericMatrix()']]]]
```

```

NEWMAT::KPMatrix::GenericMatrix()'], ['./class_n_e_w_m_a_t_1_1_concatenated_matrix.html#a4ecabb4ff1be9d3dd3fd895b25fd7237',1,'NEWMAT::ConcatenatedMatrix::GenericMatrix()'], ['./class_n_e_w_m_a_t_1_1_stacked_matrix.html#a4ecabb4ff1be9d3dd3fd895b25fd7237',1,'NEWMAT::StackedMatrix::GenericMatrix()'], ['./class_n_e_w_m_a_t_1_1_subtracted_matrix.html#a4ecabb4ff1be9d3dd3fd895b25fd7237',1,'NEWMAT::SubtractedMatrix::GenericMatrix()'], ['./class_n_e_w_m_a_t_1_1_shifted_matrix.html#a4ecabb4ff1be9d3dd3fd895b25fd7237',1,'NEWMAT::ShiftedMatrix::GenericMatrix()'], ['./class_n_e_w_m_a_t_1_1_neg_shifted_matrix.html#a4ecabb4ff1be9d3dd3fd895b25fd7237',1,'NEWMAT::NegShiftedMatrix::GenericMatrix()'], ['./class_n_e_w_m_a_t_1_1_scaled_matrix.html#a4ecabb4ff1be9d3dd3fd895b25fd7237',1,'NEWMAT::ScaledMatrix::GenericMatrix()']]],
['getsubmatrix',['GetSubMatrix',['
./class_n_e_w_m_a_t_1_1_base_matrix.html#a99a4375b34d3548efc90d13431a4d54f',1,'NEWMAT::BaseMatrix::GetSubMatrix()'], ['./class_n_e_w_m_a_t_1_1_general_matrix.html#a99a4375b34d3548efc90d13431a4d54f',1,'NEWMAT::GeneralMatrix::GetSubMatrix()']]],
['setparameterdomain',['setParameterDomain',['
./class_go_1_1_bounded_surface.html#a9fa33bdf92930ca56ca7e882c8e0180a',1,'Go::BoundedSurface']]]
]

```

Definition at line 1 of file related\_5.js.

## 30.1553 doc/html/search/related\_6.js File Reference

### Variables

- var [searchData](#)

### 30.1553.1 Variable Documentation

#### 30.1553.1.1 var searchData

#### Initial value:

```

=
[
['intersectionpool',['IntersectionPool',['
./class_go_1_1_intersector.html#aaf8c0751b08d6f7243931a07947a594e',1,'Go::Intersector::IntersectionPool()'], ['./class_go_1_1_sf_sf_intersector.html#aaf8c0751b08d6f7243931a07947a594e',1,'Go::SfSfIntersector::IntersectionPool()']]],
['intersectoralqpar',['IntersectorAlgPar',['
./class_go_1_1_intersector_func_const.html#ae0bce107ca8435c4f869edb948b496cf',1,'Go::IntersectorFuncConst']]],
['invertedmatrix',['InvertedMatrix',['
./class_n_e_w_m_a_t_1_1_base_matrix.html#ae361bc8ebdc01f511bb97ab6516e2e34',1,'NEWMAT::BaseMatrix::InvertedMatrix()'], ['./class_n_e_w_m_a_t_1_1_general_matrix.html#ae361bc8ebdc01f511bb97ab6516e2e34',1,'NEWMAT::GeneralMatrix::InvertedMatrix()'], ['./class_n_e_w_m_a_t_1_1_solved_matrix.html#ae361bc8ebdc01f511bb97ab6516e2e34',1,'NEWMAT::SolvedMatrix::InvertedMatrix()']]]
]

```

Definition at line 1 of file related\_6.js.

## 30.1554 doc/html/search/related\_7.js File Reference

### Variables

- var [searchData](#)

### 30.1554.1 Variable Documentation

#### 30.1554.1.1 var searchData

##### Initial value:

```
=
[
 ['kp', ['KP', ['./class_n_e_w_m_a_t_l_1_k_p_matrix.html#a7b84f8e43bac4608bf225867306c9450', 1, '
 NEWMAT::KPMatrix']]],
 ['kpmatrix', ['KPMatrix', ['./class_n_e_w_m_a_t_l_1_base_matrix.html#ad4980f118f2c21b78c50f76b6891a166', 1,
 'NEWMAT::BaseMatrix::KPMatrix()'], ['./class_n_e_w_m_a_t_l_1_general_matrix.html#ad4980f118f2c21b78c50f76b6891a166', 1, 'NEWMAT::GeneralMatrix::KPMatrix()']]
]
```

Definition at line 1 of file related\_7.js.

### 30.1555 doc/html/search/related\_8.js File Reference

#### Variables

- var [searchData](#)

### 30.1555.1 Variable Documentation

#### 30.1555.1.1 var searchData

##### Initial value:

```
=
[
 ['linearequationsolver', ['LinearEquationSolver', ['./class_n_e_w_m_a_t_l_1_base_matrix.html#a44143f61f0e4670c28163293df71583f', 1, 'NEWMAT::BaseMatrix::LinearEquationSolver',
 './class_n_e_w_m_a_t_l_1_general_matrix.html#a44143f61f0e4670c28163293df71583f', 1, 'NEWMAT::GeneralMatrix::LinearEquationSolver']],
 ['lowerbandmatrix', ['LowerBandMatrix', ['./class_n_e_w_m_a_t_l_1_base_matrix.html#a4d3e29bfb45ba278e5257ede2b4e020d', 1, 'NEWMAT::BaseMatrix::LowerBandMatrix()'],
 './class_n_e_w_m_a_t_l_1_general_matrix.html#a4d3e29bfb45ba278e5257ede2b4e020d', 1, 'NEWMAT::GeneralMatrix::LowerBandMatrix']],
 ['lowertriangularmatrix', ['LowerTriangularMatrix', ['./class_n_e_w_m_a_t_l_1_base_matrix.html#aal7a07bb642f9d28f6a6b9f50c43c41a', 1, 'NEWMAT::BaseMatrix::LowerTriangularMatrix',
 './class_n_e_w_m_a_t_l_1_general_matrix.html#aal7a07bb642f9d28f6a6b9f50c43c41a', 1, 'NEWMAT::GeneralMatrix::LowerTriangularMatrix']]
]
```

Definition at line 1 of file related\_8.js.

### 30.1556 doc/html/search/related\_9.js File Reference

#### Variables

- var [searchData](#)



## 30.1556.1 Variable Documentation

### 30.1556.1.1 var searchData

#### Initial value:

```
=
[
 ['matedmatrix', ['MatedMatrix', ['
 ../class_n_e_w_m_a_t_l_l_base_matrix.html#a74e701028481006374af8538416b7217', 1, 'NEWMAT::BaseMatrix::MatedMatrix()'], ['
 ../class_n_e_w_m_a_t_l_l_general_matrix.html#a74e701028481006374af8538416b7217', 1, 'NEWMAT::GeneralMatrix::MatedMatrix()
 '], 1, 'NEWMAT::GeneralMatrix::Matrix()']], ['../class_n_e_w_m_a_t_l_l_base_matrix.html#a34913a9261681f734171a6da06bd56fe', 1, '
 NEWMAT::BaseMatrix::Matrix()'], ['../class_n_e_w_m_a_t_l_l_general_matrix.html#a34913a9261681f734171a6da06bd56fe
 ', 1, 'NEWMAT::GeneralMatrix::Matrix()']],
 ['modelquality', ['ModelQuality', ['../class_go_l_l_quality_results.html#a8b28ece74f17b35d5d93ee5b4e8bb3a1'
 , 1, 'Go::QualityResults']],
 ['modelrepair', ['ModelRepair', ['../class_go_l_l_quality_results.html#adbeec2dc762f5b47140d310dbba07cf8', 1
 , 'Go::QualityResults']],
 ['multipliedmatrix', ['MultipliedMatrix', ['
 ../class_n_e_w_m_a_t_l_l_base_matrix.html#adf9d5633a1bc685e4eacf58768b99201', 1, 'NEWMAT::BaseMatrix::MultipliedMatrix()
 ../class_n_e_w_m_a_t_l_l_general_matrix.html#adf9d5633a1bc685e4eacf58768b99201', 1, 'NEWMAT::GeneralMatrix::MultipliedMatr
 ']]]]
]
```

Definition at line 1 of file related\_9.js.

## 30.1557 doc/html/search/related\_a.js File Reference

### Variables

- var [searchData](#)

## 30.1557.1 Variable Documentation

### 30.1557.1.1 var searchData

#### Initial value:

```
=
[
 ['negatedmatrix', ['NegatedMatrix', ['
 ../class_n_e_w_m_a_t_l_l_base_matrix.html#a339246a27fdf960873e646538f0245ad', 1, 'NEWMAT::BaseMatrix::NegatedMatrix()'], ['
 ../class_n_e_w_m_a_t_l_l_general_matrix.html#a339246a27fdf960873e646538f0245ad', 1, 'NEWMAT::GeneralMatrix::NegatedMatrix
 '], 1, 'NEWMAT::GeneralMatrix::Matrix()']], ['negshiftedmatrix', ['NegShiftedMatrix', ['
 ../class_n_e_w_m_a_t_l_l_base_matrix.html#a437c0b9876f15d7f060627f8fc463313', 1, 'NEWMAT::BaseMatrix::NegShiftedMatrix()
 ../class_n_e_w_m_a_t_l_l_general_matrix.html#a437c0b9876f15d7f060627f8fc463313', 1, 'NEWMAT::GeneralMatrix::NegShiftedMatr
 '], 1, 'NEWMAT::GeneralMatrix::Matrix()']], ['nricmatrix', ['nricMatrix', ['../class_n_e_w_m_a_t_l_l_base_matrix.html#ab755e1343fa84fbbea94979d734af82a
 ', 1, 'NEWMAT::BaseMatrix::nricMatrix()'], ['
 ../class_n_e_w_m_a_t_l_l_general_matrix.html#ab755e1343fa84fbbea94979d734af82a', 1, 'NEWMAT::GeneralMatrix::nricMatrix()
 ']]]]
]
```

Definition at line 1 of file related\_a.js.

## 30.1558 doc/html/search/related\_b.js File Reference

### Variables

- var [searchData](#)
- & [lt](#)

### 30.1558.1 Variable Documentation

#### 30.1558.1.1 & It

Definition at line 6 of file related\_b.js.

#### 30.1558.1.2 var searchData

##### Initial value:

```
=
[
 ['operator_2a', ['operator*', ['
 ../class_n_e_w_m_a_t_l_1_scaled_matrix.html#ab1d04e45670e7e54332437ab81f69f69', 1, 'NEWMAT::ScaledMatrix::operator*()'], [
 vector3t::operator*()']],
 ['operator_2b', ['operator+', ['
 ../class_n_e_w_m_a_t_l_1_shifted_matrix.html#afc960ce11532a76ca986ec75542bca02', 1, 'NEWMAT::ShiftedMatrix::operator+()'], [
 vector3t::operator+()'], ['../classvector3t.html#a7572339e888c5262a1418d17f46c473c', 1, 'vector3t::operator+()']
]],
 ['operator_2d', ['operator-', ['
 ../class_n_e_w_m_a_t_l_1_neg_shifted_matrix.html#a12197e802fd1e5829c78adaccb57840', 1, 'NEWMAT::NegShiftedMatrix']],
 ['operator_3c_3c', ['operator<
```

Definition at line 1 of file related\_b.js.

## 30.1559 doc/html/search/related\_c.js File Reference

### Variables

- var [searchData](#)

### 30.1559.1 Variable Documentation

#### 30.1559.1.1 var searchData

##### Initial value:

```
=
[
 ['rectangular', ['Rectangular', ['
 ../class_n_e_w_m_a_t_l_1_matrix_type.html#a858f15c5cad7d2271c38bc54fbca9b5c', 1, 'NEWMAT::MatrixType']],
 ['returnmatrixx', ['ReturnMatrixX', ['
 ../class_n_e_w_m_a_t_l_1_base_matrix.html#aa5d5751eb6bb90341d5dd65e404ad716', 1, 'NEWMAT::BaseMatrix::ReturnMatrixX()'], [
 ../class_n_e_w_m_a_t_l_1_general_matrix.html#aa5d5751eb6bb90341d5dd65e404ad716', 1, 'NEWMAT::GeneralMatrix::ReturnMatrixX']
]],
 ['reversedmatrix', ['ReversedMatrix', ['
 ../class_n_e_w_m_a_t_l_1_base_matrix.html#a59874012f358719fe00fd48bb93d932', 1, 'NEWMAT::BaseMatrix::ReversedMatrix()'], [
 ../class_n_e_w_m_a_t_l_1_general_matrix.html#a59874012f358719fe00fd48bb93d932', 1, 'NEWMAT::GeneralMatrix::ReversedMatrix']
]],
 ['rotate', ['Rotate', ['../class_rect_matrix_row_col.html#a23acf0c66955f76462c923517f1f64be', 1, '
 RectMatrixRowCol::Rotate()'], ['../class_rect_matrix_col.html#a23acf0c66955f76462c923517f1f64be', 1, '
 RectMatrixCol::Rotate()']],
 ['rowedmatrix', ['RowedMatrix', ['
 ../class_n_e_w_m_a_t_l_1_base_matrix.html#a744f5ba295f31416bef930e9490866b8', 1, 'NEWMAT::BaseMatrix::RowedMatrix()'], [
 ../class_n_e_w_m_a_t_l_1_general_matrix.html#a744f5ba295f31416bef930e9490866b8', 1, 'NEWMAT::GeneralMatrix::RowedMatrix()']
]],
 ['rowvector', ['RowVector', ['../class_n_e_w_m_a_t_l_1_base_matrix.html#a575e505f29f178875eb92d4f000c4f30',
 1, 'NEWMAT::BaseMatrix::RowVector()'], [
 ../class_n_e_w_m_a_t_l_1_general_matrix.html#a575e505f29f178875eb92d4f000c4f30', 1, 'NEWMAT::GeneralMatrix::RowVector()']
]
]
```

Definition at line 1 of file related\_c.js.

## 30.1560 doc/html/search/related\_d.js File Reference

### Variables

- var [searchData](#)

### 30.1560.1 Variable Documentation

#### 30.1560.1.1 var searchData

#### Initial value:

```
=
[
 ['scaledmatrix', ['ScaledMatrix', ['
 ../class_n_e_w_m_a_t_l_l_base_matrix.html#a4e95604e6eadabba787161333e03d8ec', 1, 'NEWMAT::BaseMatrix::ScaledMatrix()'], ['
 ../class_n_e_w_m_a_t_l_l_general_matrix.html#a4e95604e6eadabba787161333e03d8ec', 1, 'NEWMAT::GeneralMatrix::ScaledMatrix()'],
 ['sfsselfintersector', ['SfSelfIntersector', ['
 ../class_go_l_l_sf_sf_intersector.html#a4ab896dd725973ca14b23d6247eeae9a', 1, 'Go::SfSfIntersector']],
 ['sfsfintersector', ['SfSfIntersector', ['
 ../class_go_l_l_intersection_pool.html#a3f55857a10bc333ba09f54817cfa86ed', 1, 'Go::IntersectionPool::SfSfIntersector()'], ['
 ../class_go_l_l_intersector.html#a3f55857a10bc333ba09f54817cfa86ed', 1, 'Go::Intersector::SfSfIntersector()']],
 ['shiftedmatrix', ['ShiftedMatrix', ['
 ../class_n_e_w_m_a_t_l_l_base_matrix.html#ae80d8e7b06578973d61d715f0aedf08', 1, 'NEWMAT::BaseMatrix::ShiftedMatrix()'], ['
 ../class_n_e_w_m_a_t_l_l_general_matrix.html#ae80d8e7b06578973d61d715f0aedf08', 1, 'NEWMAT::GeneralMatrix::ShiftedMatrix()'],
 ['solvedmatrix', ['SolvedMatrix', ['
 ../class_n_e_w_m_a_t_l_l_base_matrix.html#a4e4a4a256dec48fcd44bc30348ca4ee4', 1, 'NEWMAT::BaseMatrix::SolvedMatrix()'], ['
 ../class_n_e_w_m_a_t_l_l_general_matrix.html#a4e4a4a256dec48fcd44bc30348ca4ee4', 1, 'NEWMAT::GeneralMatrix::SolvedMatrix()'],
 ['sp', ['SP', ['../class_n_e_w_m_a_t_l_l_s_p_matrix.html#a2490eace871fe6269a3511e64cbba7e5', 1, '
 NEWMAT::SPMatrix']],
 ['spmatrix', ['SPMatrix', ['../class_n_e_w_m_a_t_l_l_base_matrix.html#a7ec92c6ad15f0f152685c44c5d92237f', 1, '
 NEWMAT::BaseMatrix::SPMatrix()'], ['
 ../class_n_e_w_m_a_t_l_l_general_matrix.html#a7ec92c6ad15f0f152685c44c5d92237f', 1, 'NEWMAT::GeneralMatrix::SPMatrix()']],
 ['stackedmatrix', ['StackedMatrix', ['
 ../class_n_e_w_m_a_t_l_l_base_matrix.html#a486cdaca18b02f92f6a7323ddc03a028', 1, 'NEWMAT::BaseMatrix::StackedMatrix()'], ['
 ../class_n_e_w_m_a_t_l_l_general_matrix.html#a486cdaca18b02f92f6a7323ddc03a028', 1, 'NEWMAT::GeneralMatrix::StackedMatrix()'],
 ['subtractedmatrix', ['SubtractedMatrix', ['
 ../class_n_e_w_m_a_t_l_l_base_matrix.html#aea3e12cdd2ab8257c89e397c8f79be73', 1, 'NEWMAT::BaseMatrix::SubtractedMatrix()'], ['
 ../class_n_e_w_m_a_t_l_l_general_matrix.html#aea3e12cdd2ab8257c89e397c8f79be73', 1, 'NEWMAT::GeneralMatrix::SubtractedMatrix()'],
 ['symmetricbandmatrix', ['SymmetricBandMatrix', ['
 ../class_n_e_w_m_a_t_l_l_base_matrix.html#aad958a8a0c4264f8b3a0972e8d40fc3d', 1, 'NEWMAT::BaseMatrix::SymmetricBandMatrix()'], ['
 ../class_n_e_w_m_a_t_l_l_general_matrix.html#aad958a8a0c4264f8b3a0972e8d40fc3d', 1, 'NEWMAT::GeneralMatrix::SymmetricBandMatrix()'],
 ['symmetricmatrix', ['SymmetricMatrix', ['
 ../class_n_e_w_m_a_t_l_l_base_matrix.html#a363bd98d69c9aceacb215c76cba271b4', 1, 'NEWMAT::BaseMatrix::SymmetricMatrix()'], ['
 ../class_n_e_w_m_a_t_l_l_general_matrix.html#a363bd98d69c9aceacb215c76cba271b4', 1, 'NEWMAT::GeneralMatrix::SymmetricMatrix()'],
]
```

Definition at line 1 of file [related\\_d.js](#).

## 30.1561 doc/html/search/related\_e.js File Reference

### Variables

- var [searchData](#)

### 30.1561.1 Variable Documentation

#### 30.1561.1.1 var searchData

##### Initial value:

```
=
[
 ['transposedmatrix', ['TransposedMatrix', ['
 ../class_n_e_w_m_a_t_l_l_base_matrix.html#a2c776a0940d99a0576d2a1bc8784101f', 1, 'NEWMAT::BaseMatrix::TransposedMatrix()'],
 ../class_n_e_w_m_a_t_l_l_general_matrix.html#a2c776a0940d99a0576d2a1bc8784101f', 1, 'NEWMAT::GeneralMatrix::TransposedMatr
]
```

Definition at line 1 of file related\_e.js.

### 30.1562 doc/html/search/related\_f.js File Reference

#### Variables

- var [searchData](#)

### 30.1562.1 Variable Documentation

#### 30.1562.1.1 var searchData

##### Initial value:

```
=
[
 ['upperbandmatrix', ['UpperBandMatrix', ['
 ../class_n_e_w_m_a_t_l_l_base_matrix.html#af6af70a4a2780f63283c5311240939a8', 1, 'NEWMAT::BaseMatrix::UpperBandMatrix()'],
 ../class_n_e_w_m_a_t_l_l_general_matrix.html#af6af70a4a2780f63283c5311240939a8', 1, 'NEWMAT::GeneralMatrix::UpperBandMatr
]
 ['uppertriangularmatrix', ['UpperTriangularMatrix', ['
 ../class_n_e_w_m_a_t_l_l_base_matrix.html#aee395f7c6d3ec58abdf66ce548376c20', 1, 'NEWMAT::BaseMatrix::UpperTriangularMatr
 ../class_n_e_w_m_a_t_l_l_general_matrix.html#aee395f7c6d3ec58abdf66ce548376c20', 1, 'NEWMAT::GeneralMatrix::UpperTriangul
]
```

Definition at line 1 of file related\_f.js.

### 30.1563 doc/html/search/search.js File Reference

#### Functions

- function [convertTold](#) (search)
- function [getXPos](#) (item)
- function [getYPos](#) (item)
- function [SearchBox](#) (name, resultsPath, inFrame, label)
- function [SearchResults](#) (name)
- function [setKeyActions](#) (elem, action)
- function [setClassAttr](#) (elem, attr)
- function [createResults](#) ()
- function [init\\_search](#) ()

### 30.1563.1 Function Documentation

#### 30.1563.1.1 function convertTold ( *search* )

Definition at line 1 of file search.js.

#### 30.1563.1.2 function createResults ( )

Definition at line 722 of file search.js.

#### 30.1563.1.3 function getXPos ( *item* )

Definition at line 24 of file search.js.

#### 30.1563.1.4 function getYPos ( *item* )

Definition at line 38 of file search.js.

#### 30.1563.1.5 function init\_search ( )

Definition at line 777 of file search.js.

#### 30.1563.1.6 function SearchBox ( *name*, *resultsPath*, *inFrame*, *label* )

Definition at line 59 of file search.js.

#### 30.1563.1.7 function SearchResults ( *name* )

Definition at line 404 of file search.js.

#### 30.1563.1.8 function setClassAttr ( *elem*, *attr* )

Definition at line 716 of file search.js.

#### 30.1563.1.9 function setKeyActions ( *elem*, *action* )

Definition at line 709 of file search.js.

## 30.1564 doc/html/search/searchdata.js File Reference

### Variables

- var [indexSectionsWithContent](#)
- var [indexSectionNames](#)
- var [indexSectionLabels](#)

### 30.1564.1 Variable Documentation

#### 30.1564.1.1 var indexSectionLabels

##### Initial value:

```
=
{
 0: "All",
 1: "Classes",
 2: "Namespaces",
 3: "Files",
 4: "Functions",
 5: "Variables",
 6: "Typedefs",
 7: "Enumerations",
 8: "Enumerator",
 9: "Friends",
 10: "Macros",
 11: "Modules",
 12: "Pages"
}
```

Definition at line 35 of file searchdata.js.

#### 30.1564.1.2 var indexSectionNames

##### Initial value:

```
=
{
 0: "all",
 1: "classes",
 2: "namespaces",
 3: "files",
 4: "functions",
 5: "variables",
 6: "typedefs",
 7: "enums",
 8: "enumvalues",
 9: "related",
 10: "defines",
 11: "groups",
 12: "pages"
}
```

Definition at line 18 of file searchdata.js.

### 30.1564.1.3 var indexSectionsWithContent

#### Initial value:

```
=
{
 0: "2_abcdefghijklmnopqrstuvwxyz~",
 1: "abcdefghijklmnopqrstuvwxyz",
 2: "ghrst",
 3: "2abcdefghijklmnopqrstuvwxyz",
 4: "_abcdefghijklmnopqrstuvwxyz~",
 5: "abcdefghijklmnopqrstuvwxyz",
 6: "abcdefghijklmnoprsv",
 7: "abcdefghijklmnoprstw",
 8: "abcdefghijklmnoprstuvwxyz",
 9: "abcdfgiklmnorstu",
 10: "_abcdefghijklmnoprstvw",
 11: "r",
 12: "abegt"
}
```

Definition at line 1 of file searchdata.js.

## 30.1565 doc/html/search/typedefs\_0.js File Reference

### Variables

- var [searchData](#)

### 30.1565.1 Variable Documentation

#### 30.1565.1.1 var searchData

#### Initial value:

```
=
[
 ['argument_5ftype', ['argument_type', ['
 ../class_go_1_1_crosses_value.html#aae9be45a2093dec324dea50127163874', 1, 'Go::CrossesValue::argument_type()'], ['
 ../class_go_1_1_test_in_domain.html#a5190b2f25aa4dd7d98df5a5ffc306291', 1, 'Go::TestInDomain::argument_type()']]
]
```

Definition at line 1 of file typedefs\_0.js.

## 30.1566 doc/html/search/typedefs\_1.js File Reference

### Variables

- var [searchData](#)

### 30.1566.1 Variable Documentation

#### 30.1566.1.1 var searchData

##### Initial value:

```
=
[
 ['barycoordsystem2d', ['BaryCoordSystem2D', ['./namespace_go.html#a12cca84099b202b44209f71f6c9dd20c', 1, 'Go'
]]],
 ['barycoordsystem3d', ['BaryCoordSystem3D', ['./namespace_go.html#a4fd447cf6cb7a456837fe15c599f8077', 1, 'Go'
]]],
 ['boundarypiece', ['BoundaryPiece', ['./namespace_go.html#a5ffb8f679caa5a03117315a175a828b5', 1, 'Go']]],
 ['bsplineindexmap', ['BsplineIndexMap', ['./namespace_go.html#a4b15ecc9104dfd8fd5bc9cd2a0e525f1', 1, 'Go']]],
 ['bsplinemap', ['BSplineMap', ['./class_go_1_1_l_r_spline_surface.html#a22753c1f579698a35503a78c18dbe3a6',
 1, 'Go::LRSplineSurface']]]
]
```

Definition at line 1 of file typedefs\_1.js.

### 30.1567 doc/html/search/typedefs\_2.js File Reference

#### Variables

- var [searchData](#)

### 30.1567.1 Variable Documentation

#### 30.1567.1.1 var searchData

##### Initial value:

```
=
[
 ['colcontainer', ['ColContainer', ['./gv_application_8h.html#a4fb71c3010e59d6637a56f27a1d9d57d', 1, '
 gvApplication.h']]],
 ['const_5fiteractor', ['const_iterator', ['./class_go_1_1_scratch_vect.html#a3ec1bec2fc393949be4a21c24d98f87a', 1, 'Go::ScratchVect']]]
]
```

Definition at line 1 of file typedefs\_2.js.

### 30.1568 doc/html/search/typedefs\_3.js File Reference

#### Variables

- var [searchData](#)
- T \* & gt



### 30.1568.1 Variable Documentation

#### 30.1568.1.1 T\* & gt

Definition at line 3 of file typedefs\_3.js.

#### 30.1568.1.2 var searchData

##### Initial value:

```
=
[
 ['difference_5ftype', ['difference_type', ['
 ../struct_go_1_lgo_iterator_traits.html#a60ff0cd0a7bea11271baf9e731e38668', 1, 'Go::go_iterator_traits::difference_type
 ../struct_go_1_lgo_iterator_traits_3_01_t_01_5_01_4.html#a5e7177e826be4b2f9f5d096498718dd0', 1, 'Go::go_iterator_traits
```

Definition at line 1 of file typedefs\_3.js.

## 30.1569 doc/html/search/typedefs\_4.js File Reference

### Variables

- var [searchData](#)

### 30.1569.1 Variable Documentation

#### 30.1569.1.1 var searchData

##### Initial value:

```
=
[
 ['elementmap', ['ElementMap', ['../class_go_1_l_l_r_spline_surface.html#ae21adad701eb6a1be6d9880485c1cd87',
 1, 'Go::LRSplineSurface']],
 ['exception', ['Exception', ['../namespace_r_b_d__c_o_m_m_o_n.html#abbff87b4655088a6fd923859d2a6a8d2', 1, '
 RBD_COMMON']]
]
```

Definition at line 1 of file typedefs\_4.js.

## 30.1570 doc/html/search/typedefs\_5.js File Reference

### Variables

- var [searchData](#)
- [s1786](#) c

### 30.1570.1 Variable Documentation

#### 30.1570.1.1 s1786 c

Definition at line 3 of file typedefs\_5.js.

#### 30.1570.1.2 var searchData

##### Initial value:

```
=
[
 ['fevalcproc', ['fevalcProc', ['../s1786_8c.html#a6f5d68d6da132de46dd79668ba91bb37', 1, 'fevalcProc() :
```

Definition at line 1 of file typedefs\_5.js.

## 30.1571 doc/html/search/typedefs\_6.js File Reference

### Variables

- var [searchData](#)

### 30.1571.1 Variable Documentation

#### 30.1571.1.1 var searchData

##### Initial value:

```
=
[
 ['genmesh', ['genMesh', ['../gv_parametric_surface_paintable_8h.html#a97b8df4fe8839d115decc99c9eda4a35', 1, '
 gvParametricSurfacePaintable.h']],
 ['graphiter', ['GraphIter', ['../namespace_go.html#a428f052620c22896093395bb8d6ddcae', 1, 'Go']]
]
```

Definition at line 1 of file typedefs\_6.js.

## 30.1572 doc/html/search/typedefs\_7.js File Reference

### Variables

- var [searchData](#)

### 30.1572.1 Variable Documentation

#### 30.1572.1.1 var searchData

**Initial value:**

```
=
[
 ['heaptype', ['HeapType', ['../_pr_dijkstra_8h.html#afb8a42e8b8299d2995279be8fdaa601d', 1, 'PrDijkstra.h']],
 ['heaptype2', ['HeapType2', ['../_pr_multi_dijkstra_8h.html#a07dcabcf75f750257e24dd71571ec318', 1, '
 PrMultiDijkstra.h']]]
]
```

Definition at line 1 of file typedefs\_7.js.

## 30.1573 doc/html/search/typedefs\_8.js File Reference

### Variables

- var [searchData](#)

### 30.1573.1 Variable Documentation

#### 30.1573.1.1 var searchData

**Initial value:**

```
=
[
 ['initproc', ['initProc', ['../sh1461_8c.html#a6d9a3f0bc40bd855afe60e6376874834', 1, 'sh1461.c']],
 ['iter'
]
```

Definition at line 1 of file typedefs\_8.js.

## 30.1574 doc/html/search/typedefs\_9.js File Reference

### Variables

- var [searchData](#)

### 30.1574.1 Variable Documentation

#### 30.1574.1.1 var searchData

**Initial value:**

```
=
[
 ['long_5freal', ['long_Real', ['../namespace_r_b_d__c_o_m_m_o_n.html#ab577ec287616d0a76ae90f9b579f43c1', 1, '
 'RBD_COMMON']]]
]
```

Definition at line 1 of file typedefs\_9.js.

## 30.1575 doc/html/search/typedefs\_a.js File Reference

### Variables

- var [searchData](#)

### 30.1575.1 Variable Documentation

#### 30.1575.1.1 var searchData

##### Initial value:

```
=
[
 ['matrix', ['Matrix', ['../class_go_1_1_coordinate_system.html#a01318b17e30343e7b38c6dd129c5ae57', 1, '
 Go::CoordinateSystem']],
 ['matrix3', ['MATRIX3', ['../gv_utilities_8h.html#a5277c85f00358938e80963e1f28bd448', 1, 'gvUtilities.h']],
 ['matrix4', ['MATRIX4', ['../gv_utilities_8h.html#a17ca410317577d99816160c68b070fc7', 1, 'gvUtilities.h']]]
]
```

Definition at line 1 of file typedefs\_a.js.

## 30.1576 doc/html/search/typedefs\_b.js File Reference

### Variables

- var [searchData](#)

### 30.1576.1 Variable Documentation

#### 30.1576.1.1 var searchData

##### Initial value:

```
=
[
 ['objcontainer', ['ObjContainer', ['../gv_application_8h.html#a930ed0f609554d54ba43b401b20d1f37', 1, '
 gvApplication.h']]]
]
```

Definition at line 1 of file typedefs\_b.js.

## 30.1577 doc/html/search/typedefs\_c.js File Reference

### Variables

- var [searchData](#)
- T \*& [gt](#)

### 30.1577.1 Variable Documentation

#### 30.1577.1.1 T\* & gt

Definition at line 6 of file typedefs\_c.js.

#### 30.1577.1.2 var searchData

##### Initial value:

```

=
[
 ['param_5ffloat_5ftype', ['param_float_type', ['
 ../class_go_1_1_l_r_spline_eval_grid.html#a054a8a1a64ac866831717daf34aef80c', 1, 'Go::LRSplineEvalGrid']]],
 ['pointcloud3d', ['PointCloud3D', ['../namespace_go.html#aca0adb8106a4ffc358eeffeeb11ef851', 1, 'Go']],
 ['pointcloud4d', ['PointCloud4D', ['../namespace_go.html#ac0a110c4920004f774150b3e9ccbfe64', 1, 'Go']],
 ['pointer', ['pointer', ['../struct_go_1_lgo__iterator__traits.html#abbc7c10f9ef14171c490acf122004524', 1, '
 Go::go_iterator_traits::pointer()'], ['
 ../struct_go_1_lgo__iterator__traits_3_01_t_01_5_01_4.html#a11eab6473cd0d43eb5df24534c0079b4', 1, 'Go::go_iterator_traits

```

Definition at line 1 of file typedefs\_c.js.

## 30.1578 doc/html/search/typedefs\_d.js File Reference

### Variables

- var [searchData](#)

### 30.1578.1 Variable Documentation

#### 30.1578.1.1 var searchData

##### Initial value:

```

=
[
 ['rank_5finfo', ['rank_info', ['../sisl_p_8h.html#ad075872fe80a1bd569afb76ea3f2ae5', 1, 'sislP.h']],
 ['real'

```

Definition at line 1 of file typedefs\_d.js.

## 30.1579 doc/html/search/typedefs\_e.js File Reference

### Variables

- var [searchData](#)

### 30.1579.1 Variable Documentation

#### 30.1579.1.1 var searchData

##### Initial value:

```
=
[
 ['short_5flist', ['short_list', ['../2dpoly__for__s2m_8h.html#a8b138f733256652f3d5086d3c2245247', 1, '
 2dpoly_for_s2m.h']],
 ['short_5flist_5fshort_5flist', ['short_list_short_list', ['
 ../2dpoly__for__s2m_8h.html#acf0d185d8fcc5467cf3c7db6bb337659', 1, '2dpoly_for_s2m.h']],
 ['sideconstraint', ['sideConstraint', ['../namespace_go.html#ab5e42d330d6c5ac6e51b604a52c0d4d7', 1, 'Go']],
 ['sideconstraintset', ['sideConstraintSet', ['../namespace_go.html#af57c06077dc46e0ee165b4f9c38280e', 1, 'Go
 ']]],
 ['simpleelement', ['simpleElement', ['../namespace_go.html#a046535108edecaad94e9960634734291', 1, 'Go']],
 ['sislbox', ['SISLbox', ['../sisl_8h.html#ab44a7bbd80df4092a56bf8251a49ebd7', 1, 'sisl.h']],
 ['sislcurve', ['SISLCurve', ['../sisl_8h.html#ae14493c7c28658c0d5d85868fb6f5c41', 1, 'sisl.h']],
 ['sisldir', ['SISLdir', ['../sisl_8h.html#a2dfffd3de5a9808bf9ca53c4b435e949d', 1, 'sisl.h']],
 ['sisledge', ['SISLEdge', ['../sisl_p_8h.html#a543d75a1190a84caf5c7775bf23fc037', 1, 'sislP.h']],
 ['sislintcurve', ['SISLIntcurve', ['../sisl_8h.html#a835d444731761a886799f1f94ca3381d', 1, 'sisl.h']],
 ['sislintdat', ['SISLIntdat', ['../sisl_p_8h.html#ad98eb9fe1088a0154e676d28889f008e', 1, 'sislP.h']],
 ['sislintlist', ['SISLIntlist', ['../sisl_p_8h.html#ae0e318a6057716c7d541e8818b713d7d', 1, 'sislP.h']],
 ['sislintpt', ['SISLIntpt', ['../sisl_p_8h.html#a96a4b4e682039a36b690f85ba24699af', 1, 'sislP.h']],
 ['sislintsurf', ['SISLIntsurf', ['../sisl_p_8h.html#a43f6c0ad06ddb8dff509780784397a4a', 1, 'sislP.h']],
 ['sislobject', ['SISLObject', ['../sisl_p_8h.html#a03e483ef55998a5bade12cea81ba2561', 1, 'sislP.h']],
 ['sislpoint', ['SISLPoint', ['../sisl_p_8h.html#a77a679c87802f0bc0cb13c871692de90', 1, 'sislP.h']],
 ['sislptedge', ['SISLPtedge', ['../sisl_p_8h.html#ae836f16d4793e103a7ceb97dabe2f6b4', 1, 'sislP.h']],
 ['sislsurf', ['SISLSurf', ['../sisl_8h.html#af69927d4ce24935580acae03f3b61048', 1, 'sisl.h']],
 ['sisltrack', ['SISLTrack', ['../sisl_p_8h.html#ad674ab4ceceb5c56cedf406b84bd315d', 1, 'sislP.h']],
 ['sisltrimpar', ['SISLTrimpar', ['../sisl_p_8h.html#a374219250c15b439345efce1fd3ac19e', 1, 'sislP.h']]
]
```

Definition at line 1 of file `typedefs_e.js`.

### 30.1580 doc/html/search/typedefs\_f.js File Reference

#### Variables

- var [searchData](#)
- T\* & [gt](#)

#### 30.1580.1 Variable Documentation

##### 30.1580.1.1 T\* & gt

Definition at line 3 of file `typedefs_f.js`.

##### 30.1580.1.2 var searchData

##### Initial value:

```
=
[
 ['value_5ftype', ['value_type', ['
 ../struct_go_1_lgo_iterator_traits.html#aca25ce118093b91e456e2982fb419bb8', 1, 'Go::go_iterator_traits::value_type()'],
 ../struct_go_1_lgo_iterator_traits_3_01_t_01_5_01_4.html#a0ee41a5f38a4f3fe31c01d158afdall1', 1, 'Go::go_iterator_traits::value_type()']]]
]
```

Definition at line 1 of file `typedefs_f.js`.

## 30.1581 doc/html/search/variables\_0.js File Reference

### Variables

- var [searchData](#)
- mouse [cpp](#)

### 30.1581.1 Variable Documentation

#### 30.1581.1.1 mouse cpp

Definition at line 14 of file variables\_0.js.

#### 30.1581.1.2 var searchData

### Initial value:

```
=
[
 ['a', ['a', ['../class_n_e_w_m_a_t_l_l_simple_int_array.html#af8aa7920093ee4bde2aa7dfb272cb403', 1, '
 NEWMAT::SimpleIntArray']],
 ['a_5f', ['a', ['../class_pr_mat.html#adc5bf006d2b3c60eb45bc6f374992a3d', 1, 'PrMat::a_()', ['
 ../class_pr_vec.html#afda074f05b8bb9e81086ba29f31f2762', 1, 'PrVec::a_()', ['
 ../class_go_l_l_solve_c_g.html#a3dela62612cb883f2f7af53f13057fd7', 1, 'Go::SolveCG::A_()']],
 ['aang', ['aang', ['../struct_s_i_s_ldir.html#aa799f5940d835206e64ba17ac835a34e', 1, 'SISLdir']],
 ['accumulated_5ferror_5f', ['accumulated_error_', ['
 ../struct_go_l_l_l_s_smooth_data.html#alfff92be11bc1ff771731c50a17120dba', 1, 'Go::LSSmoothData']],
 ['actionform', ['actionForm', ['../classgv_application.html#ae8bb2d69bd0bb028897047e28f062f77', 1, '
 gvApplication']],
 ['ad', ['aD', ['../jquery_8js.html#ad223f5fba68c41c1236671ac5c5b0fcb', 1, 'jquery.js']],
 ['additional_5fcorner_5fpts_5f', ['additional_corner_pts_', ['
 ../class_go_l_l_ft_surface_set.html#a15f9f8d23a9bc31138cc584a65515e61', 1, 'Go::ftSurfaceSet']],
 ['adist', ['adist', ['../struct_s_i_s_l_intpt.html#a8693e35392ccdbf93e6669a2bcdc4af', 1, 'SISLIntpt']],
 ['adjacency_5ffound_5f', ['adjacency_found_', ['
 ../struct_go_l_l_adjacency_info.html#a3ac7dc366593e8dacac6ecfc3dab3e16', 1, 'Go::AdjacencyInfo::adjacency_found_()'], ['
 ../struct_go_l_l_volume_adjacency_info.html#a1456ba763024182354bdc7b0293a8006', 1, 'Go::VolumeAdjacencyInfo::adjacency_fo
 ['algobj_5fint_5f', ['algobj_int_', ['
 ../class_go_l_l_intersector_alg_par.html#a6e26ce4d645ba59581bcfd4863522bb8', 1, 'Go::IntersectorAlgPar']],
 ['allneighbours_5f', ['allNeighbours_', ['
 ../class_pr_parametrize_int.html#a376adea9f4931c23acf517e5d06bf7d7', 1, 'PrParametrizeInt']],
 ['allow_5fzrot', ['allow_zrot', ['../mouse_8h.html#a506eb7695cd87667f501cfd89fe29839', 1, 'allow_zrot():
```

Definition at line 1 of file variables\_0.js.

## 30.1582 doc/html/search/variables\_1.js File Reference

### Variables

- var [searchData](#)

### 30.1582.1 Variable Documentation

#### 30.1582.1.1 var searchData

Definition at line 1 of file variables\_1.js.

## 30.1583 doc/html/search/variables\_10.js File Reference

### Variables

- var [searchData](#)

### 30.1583.1 Variable Documentation

#### 30.1583.1.1 var searchData

##### Initial value:

```
=
[
 ['quads_5f', ['quads_', ['../classgv_quads_paintable.html#a80b1bdf3b938aee5229b96ba7a5959b0', 1, '
 gvQuadsPaintable']],
 ['queue_5f', ['queue_', ['../class_dijkstra.html#af2a9d922323cb2849a2256b53f3df903', 1, 'Dijkstra::queue_()']
 , ['../class_multi_dijkstra.html#a0c4bdac62aa04c86ef015a9c2c3a5507', 1, 'MultiDijkstra::queue_()']]
]
```

Definition at line 1 of file variables\_10.js.

## 30.1584 doc/html/search/variables\_11.js File Reference

### Variables

- var [searchData](#)

### 30.1584.1 Variable Documentation

#### 30.1584.1.1 var searchData

##### Initial value:

```
=
[
 ['r1_5f', ['r1_', ['../class_go_1_1_ellipse.html#a773aaeb5188d2e6d4e9034fbd4a30123', 1, 'Go::Ellipse::r1_()']
 , ['../class_go_1_1_hyperbola.html#ac624bd16d4a32d9b74361c141421dc81', 1, 'Go::Hyperbola::r1_()']],
 ['r2_5f', ['r2_', ['../class_go_1_1_ellipse.html#a2c90fff391d88f4349a96482c1e4d0fc', 1, 'Go::Ellipse::r2_()']
 , ['../class_go_1_1_hyperbola.html#aa0d052580ed57b97c4974580cdd4733c', 1, 'Go::Hyperbola::r2_()']],
 ['r_5f', ['r_', ['../struct_go_1_1_param_surface_1_degenerate__info.html#af2004242eeb813ed444c5fdb9b265071
 ', 1, 'Go::ParamSurface::degenerate_info']],
 ['r_5fpath_5f', ['r_path_', ['../class_path_type.html#a6698f5a904d409bc6779015e5ef36aca', 1, 'PathType']],
 ['radius_5f', ['radius_', ['../class_go_1_1_circle.html#a9c349501a00ac666fb501c5a3360c0f9', 1, '
 Go::Circle::radius_()'], ['../class_go_1_1_cone.html#a5bd969d6fb325d1dcc994007031ef23f', 1, 'Go::Cone::radius_()'], ['
 ../class_go_1_1_cylinder.html#a1fc104ecd7893897a5fbbdb94e3f5f3', 1, 'Go::Cylinder::radius_()'], ['
 ../class_go_1_1_sphere.html#a4e42c3da8d108c5ea687dccc4cf4652cc', 1, 'Go::Sphere::radius_()']],
 ['rational_5f', ['rational_', ['../class_go_1_1_smooth_surf.html#a1bc65bd1716252f9b3c695cf922be1f0', 1, '
 Go::SmoothSurf']],
 ['rcoef', ['rcoef', ['../struct_s_i_s_1_curve.html#a61db2260a12fff723c165b0c2a42d2382', 1, 'SISLCurve::rcoef()
 '], ['../struct_s_i_s_1_surf.html#a03e61d32e600e946e2ba0a15ee9181a6', 1, 'SISLSurf::rcoef()']],
 ['recdel', ['recdel', ['../struct_go_1_1_i_g_e_header.html#a3d57b617cb021be08b63a5f2f7e9ae8c', 1, '
 Go::IGESheader']],
 ['receiving_5fssystem_5fprodid', ['receiving_system_prodid', ['
 ../struct_go_1_1_i_g_e_header.html#a3fd3c246e86e529940fdff520775eed8', 1, 'Go::IGESheader']],
 ['refcount', ['refcount', ['../class_handle_id.html#aa6fcc3f78cf1734743a9c0a4c85e731d', 1, 'HandleId']],
 ['rescaling_5f', ['rescaling_', ['
 ../struct_go_1_1_registration_result.html#a507fbaaf0d1eab2a0512885808a182b3', 1, 'Go::RegistrationResult']],
]
```



```
['result_5ftype_5f', ['result_type_', ['
 ../struct_go_1_1_registration_result.html#a0c2bd34528ab3fdee72d2ea9fb2ac10d', 1, 'Go::RegistrationResult']]],
['results_5f', ['results_', ['../class_go_1_1_model_quality.html#a2b8e1a1ad7ba37f0157c6e09901cb969', 1,
 Go::ModelQuality::results_()'], ['../class_go_1_1_model_repair.html#a4e0074ebccc8f570fa3b15aeb7c53560', 1,
 Go::ModelRepair::results_()']]]],
['rgba', ['rgba', ['../structgv_color.html#ac897899ae348ba30cfd7bf158ec967f8', 1, 'gvColor']]],
['right_5fobj_5f1', ['right_obj_1', ['../struct_s_i_s_l_intpt.html#aa0709dea69d6e8d3cfdef881fdca74f9', 1,
 SISLIntpt']]],
['right_5fobj_5f2', ['right_obj_2', ['../struct_s_i_s_l_intpt.html#af5c85d9c7929f4c1f9320c1e307e3ff8', 1,
 SISLIntpt']]],
['rot_5fangle_5f', ['rot_angle_', ['../struct_go_1_1_rotation_info.html#a3db6ec36284fa5acb073cf25ec8d62',
 1, 'Go::RotationInfo']]],
['rot_5faxis_5f', ['rot_axis_', ['../struct_go_1_1_rotation_info.html#a2798f84d097d7cf10a29384d43f4be10', 1,
 'Go::RotationInfo']]],
['rotation_5fmatrix_5f', ['rotation_matrix_', ['
 ../struct_go_1_1_registration_result.html#a1a9d598ac5b634684c0195091c3e4b0c', 1, 'Go::RegistrationResult']]],
['rowcol', ['rowcol', ['../class_matrix_row_col.html#ae8586ad3df230474f71d91c9b3304154', 1, 'MatrixRowCol']]]
]
```

Definition at line 1 of file variables\_11.js.

## 30.1585 doc/html/search/variables\_12.js File Reference

### Variables

- var [searchData](#)

### 30.1585.1 Variable Documentation

#### 30.1585.1.1 var searchData

Definition at line 1 of file variables\_12.js.

## 30.1586 doc/html/search/variables\_13.js File Reference

### Variables

- var [searchData](#)

### 30.1586.1 Variable Documentation

#### 30.1586.1.1 var searchData

#### Initial value:

```
=
[
 ['t_5f', ['t_', ['../struct_go_1_1_param_surface_1_1degenerate__info.html#aa4a1b5a04e6f9581c46bf8c50ec6e408
 ', 1, 'Go::ParamSurface::degenerate_info::t_()'], ['
 ../class_pr_filterbank.html#a8ba3a2c1b525d44832be09fc86afe136', 1, 'PrFilterbank::t_()'], ['../class_pr_threshold.html#a56
 PrThreshold::t_()']]]],
 ['table_5f', ['table_', ['../class_go_1_1tp_topology_table.html#a144bdf017ad7970a1a1d85386652fbd7', 1,
 Go::tpTopologyTable']]],
 ['tag'
```

Definition at line 1 of file variables\_13.js.

## 30.1587 doc/html/search/variables\_14.js File Reference

### Variables

- var [searchData](#)

### 30.1587.1 Variable Documentation

#### 30.1587.1.1 var searchData

#### Initial value:

```
=
[
 ['u_5f', ['u_', ['../class_go_1_lft_point.html#a3a514101ecd669aeb1feb67bf095a176', 1, 'Go::ftPoint::u_()'], ['
 ../class_pr_prm_shp_pres.html#ae67db719b7c94fed334b2d6711c224d7', 1, 'PrPrmShpPres::u_()']],
 ['u_5fmax', ['u_max', ['
 ../struct_go_1_l_l_r_spline_surface_1_l_b_s_key.html#a5cb5e8c6fbbfe331603f97c20d1ccc69', 1, 'Go::LRSplineSurface::BSKey']],
 ['u_5fmin', ['u_min', ['
 ../struct_go_1_l_l_r_spline_surface_1_l_b_s_key.html#a59296b191fcad4e9c7c7034a2967ae3f', 1, 'Go::LRSplineSurface::BSKey::
 ../struct_go_1_l_l_r_spline_surface_1_l_elem_key.html#a585692254199d6a43db9f961acdf51c5', 1, 'Go::LRSplineSurface::ElemKey']],
 ['u_5fmult1', ['u_mult1', ['
 ../struct_go_1_l_l_r_spline_surface_1_l_b_s_key.html#a0dd8a6a75f9f5f7404616f5c970c9285', 1, 'Go::LRSplineSurface::BSKey']],
 ['u_5fmult2', ['u_mult2', ['
 ../struct_go_1_l_l_r_spline_surface_1_l_b_s_key.html#ac4082082e914b3bf54b9cc1115f4bfd2', 1, 'Go::LRSplineSurface::BSKey']],
 ['underlying_5fface_5f', ['underlying_face_', ['
 ../class_go_1_lft_curve_segment.html#a032a49673a073b3ddf2dc3f073d2c4cd', 1, 'Go::ftCurveSegment']],
 ['unit_5fdescription', ['unit_description', ['
 ../struct_go_1_l_l_i_g_e_sheader.html#a438baf6ecd3e387591afb93736b68bfa', 1, 'Go::IGESheader']],
 ['unit_5fflag', ['unit_flag', ['../struct_go_1_l_l_i_g_e_sheader.html#a11ale75ae4fd53f5e72c2f30ba80bc87', 1, '
 Go::IGESheader']],
 ['unitx_5f', ['unitx_', ['../classgv_view.html#aec8d22e5e735f138a4d68059e14b0653', 1, 'gvView']],
 ['unity_5f', ['unity_', ['../classgv_view.html#a0acc955f7dc6577247d77966286835c4', 1, 'gvView']],
 ['upper', ['upper', ['../class_n_e_w_m_a_t_l_l_matrix_band_width.html#a6b13bddfdaa1318d25610484b8504e74', 1, '
 NEWMAT::MatrixBandWidth::upper()'], ['
 ../class_n_e_w_m_a_t_l_l_band_matrix.html#ad6f9d1c9d3e2e49f5bd4b3205b7ff68a', 1, 'NEWMAT::BandMatrix::upper()']],
 ['ures_5f', ['ures_', ['../classgv_resolution_dialog.html#a80928c1c8369d3a57c58dc7c3f45b4e4', 1, '
 gvResolutionDialog']],
 ['use_5fcount', ['use_count', ['../struct_s_i_s_l_surf.html#ac378f7abff1662f5b75820dc4de4c2ca', 1, 'SISLSurf'
]]],
 ['uslide_5f', ['uslide_', ['../classgv_resolution_dialog.html#ad1d66a6de7b9177c1a871720cdeda823', 1, '
 gvResolutionDialog']],
 ['uv_5f', ['uv_', ['../class_go_1_lft_sample_point.html#a10e7313208a2bb894f16ed8308e4a657', 1, '
 Go::ftSamplePoint']]
]
```

Definition at line 1 of file variables\_14.js.

## 30.1588 doc/html/search/variables\_15.js File Reference

### Variables

- var [searchData](#)

## 30.1588.1 Variable Documentation

### 30.1588.1.1 var searchData

#### Initial value:

```
=
[
 ['v_5f', ['v_', ['../class_go_1_1ft_point.html#ae66d28de47384fb0fd2f113281a83848', 1, 'Go::ftPoint::v_()'], ['
 ../class_pr_prm_shp_pres.html#a1b91934fd02891b562a62a7def09b645', 1, 'PrPrmShpPres::v_()']],
 ['v_5fmax', ['v_max', ['
 ../struct_go_1_1_l_r_spline_surface_1_1_b_s_key.html#a6751c5eed5c5aa610e8dac5f3558b4702', 1, 'Go::LRSplineSurface::BSKey']],
 ['v_5fmin', ['v_min', ['
 ../struct_go_1_1_l_r_spline_surface_1_1_b_s_key.html#a917055d1f2ba2783786ba6345d73cbd8', 1, 'Go::LRSplineSurface::BSKey']],
 ['v_5fmult1', ['v_mult1', ['
 ../struct_go_1_1_l_r_spline_surface_1_1_b_s_key.html#ac8f082b1e3b79de0d2531198fd3648af', 1, 'Go::LRSplineSurface::BSKey']],
 ['v_5fmult2', ['v_mult2', ['
 ../struct_go_1_1_l_r_spline_surface_1_1_b_s_key.html#a04b64c2a027326729a138282d8015159', 1, 'Go::LRSplineSurface::BSKey']],
 ['vecl_5f', ['vecl_', ['../class_go_1_1_circle.html#a97750bd585d4b0ff6db5823e54ee6ef9', 1, '
 Go::Circle::vecl_()'], ['../class_go_1_1_ellipse.html#a7823410f8d1df9190ee4de6ebc0d8066', 1, 'Go::Ellipse::vecl_()'], ['
 ../class_go_1_1_hyperbola.html#a2114f3b3f1a9364090d14ed763cda4a0', 1, 'Go::Hyperbola::vecl_()'], ['
 ../class_go_1_1_parabola.html#acd2810f249bce3e2d3cd2d6a68af7b3', 1, 'Go::Parabola::vecl_()'], ['
 ../class_go_1_1_plane.html#abcffa580eef7b1e36b79a492dde66feb', 1, 'Go::Plane::vecl_()']],
 ['vec2_5f', ['vec2_', ['../class_go_1_1_circle.html#ada637174c69fc9e212011e9562b90c5a', 1, '
 Go::Circle::vec2_()'], ['../class_go_1_1_ellipse.html#a42184e9707b1a7b9607e525016ae8730', 1, 'Go::Ellipse::vec2_()'], ['
 ../class_go_1_1_hyperbola.html#a102ce0310d2723234bed346ced8fd40f', 1, 'Go::Hyperbola::vec2_()'], ['
 ../class_go_1_1_parabola.html#a50c8d1ac6330013d73578d96d76a7fd', 1, 'Go::Parabola::vec2_()'], ['
 ../class_go_1_1_plane.html#acd191a7cb7d89ba4bd5d3011121a0759', 1, 'Go::Plane::vec2_()']],
 ['vert_5ftranslation_5f', ['vert_translation_', ['
 ../class_go_1_1_general_mesh.html#a1ac92dd244f23728a0a59d9e8517ed8', 1, 'Go::GeneralMesh::vert_translation_()'], ['
 ../classgv_point_cloud_paintable.html#a377d65530f2f4b7fb754d9b6232171a4', 1, 'gvPointCloudPaintable::vert_translation_()']],
 ['vertex_5f', ['vertex_', ['../class_edge_type.html#a90544457b28ef5bc656953195dd129f0', 1, 'EdgeType']],
 ['vertices_5f', ['vertices_', ['
 ../class_go_1_1_composite_model_file_handler.html#a53723c2dab9c1d5falc75a5cec117c25', 1, 'Go::CompositeModelFileHandler']],
 ['view', ['view_', ['../struct_go_1_1_i_g_e_sdirentry.html#a302fe7f59f59804f3b0b81c258dfe234', 1, '
 Go::IGESdirentry']],
 ['view_5f', ['view_', ['../classgv_application.html#a9fdb679574dd8740e2d0621fe6d9e90a', 1, 'gvApplication']],
 ['view_5fmenu_5f', ['view_menu_', ['../classgv_application.html#a6c70c7760fa87d3a1f92a76699c3ce52', 1, '
 gvApplication']],
 ['viewfield_5f', ['viewfield_', ['../classgv_camera.html#aae3344bf9ae0bc20cc5cf5ac0a8425ee', 1, 'gvCamera']],
 ['viewheight_5f', ['viewheight_', ['../classgv_camera.html#abb88858c5588187a48fbc5f9346d73e9', 1, 'gvCamera']],
 ['viewwidth_5f', ['viewwidth_', ['../classgv_camera.html#a74aac1e98e45e0f99fea252e987205bc', 1, 'gvCamera']],
 ['viewx_5f', ['viewx_', ['../classgv_camera.html#ab8ea6a94ed38592cdc57876cd8b40139', 1, 'gvCamera']],
 ['viewy_5f', ['viewy_', ['../classgv_camera.html#a49837938b247011aab8949070b9713f3', 1, 'gvCamera']],
 ['visible_5f', ['visible_', ['../classgv_paintable.html#a629c8d39d5467e02154457e1c7dde41', 1, 'gvPaintable']],
 ['vlist', ['vlist_', ['../struct_s_i_s_l_intdat.html#aad63f961bf24bb54af199e9018af1979', 1, 'SISLIntdat']],
 ['volume_5f', ['volume_', ['../class_go_1_1_curve_on_volume.html#aed6a72f4c6073d4b30cf0c36c4e35587', 1, '
 Go::CurveOnVolume']],
 ['vpoint', ['vpoint_', ['../struct_s_i_s_l_intdat.html#aedd63d0f344d9481f9d86c3bbff5e1e', 1, 'SISLIntdat']],
 ['vres_5f', ['vres_', ['../classgv_resolution_dialog.html#a09508da0c8d1c00b9da9493802a5be51', 1, '
 gvResolutionDialog']],
 ['vslide_5f', ['vslide_', ['../classgv_resolution_dialog.html#a967e9ab226b8ee2a6ed004306ef8d6e9', 1, '
 gvResolutionDialog']]
]
```

Definition at line 1 of file variables\_15.js.

## 30.1589 doc/html/search/variables\_16.js File Reference

### Variables

- var [searchData](#)

### 30.1589.1 Variable Documentation

#### 30.1589.1.1 var searchData

##### Initial value:

```
=
[
 ['weights_5f', ['weights_', ['../class_pr_parametrize_int.html#ae212bbd37bc09f289a0dbe076dab9b77', 1, 'PrParametrizeInt']],
 ['wg'
```

Definition at line 1 of file variables\_16.js.

### 30.1590 doc/html/search/variables\_17.js File Reference

#### Variables

- var [searchData](#)

### 30.1590.1 Variable Documentation

#### 30.1590.1.1 var searchData

##### Initial value:

```
=
[
 ['x', ['x', ['../class_r_b_d__c_o_m_m_o_n_l_l_r_l__r1.html#a4812b378711ea5f78a5441b96134f569', 1, 'RBD_COMMON::R1_R1']],
 ['x_5faxis_5f', ['x_axis_', ['../class_go_1_1_cone.html#a97719276e5e89fc874cee2df6327c46d', 1, 'Go::Cone::x_axis_()'], ['../class_go_1_1_cylinder.html#aa070cf5deafb3498f2601d9a2847ffc5', 1, 'Go::Cylinder::x_axis_()'], ['../class_go_1_1_sphere.html#a6865f0ab0d027141c49dceb412a613b4', 1, 'Go::Sphere::x_axis_()'], ['../class_go_1_1_torus.html#ae07499b7a7a24f9c29bcf451ef48b14b', 1, 'Go::Torus::x_axis_()']],
 ['xrot'
```

Definition at line 1 of file variables\_17.js.

### 30.1591 doc/html/search/variables\_18.js File Reference

#### Variables

- var [searchData](#)

### 30.1591.1 Variable Documentation

#### 30.1591.1.1 var searchData

##### Initial value:

```
=
[
 ['y_5faxis_5f', ['y_axis_', ['../class_go_1_1_cone.html#a04725fd933cb2a6d53a58ae1ebd24f11', 1, 'Go::Cone::y_axis_()'], ['../class_go_1_1_cylinder.html#aeaf20ee6ba23475fb4926e3c31274742', 1, 'Go::Cylinder::y_axis_()'], ['../class_go_1_1_sphere.html#a3b495bbd4eaa7a34ea93ad464b7c6c5a', 1, 'Go::Sphere::y_axis_()'], ['../class_go_1_1_torus.html#a045f285b84522d667f44f32cfc428810', 1, 'Go::Torus::y_axis_()']], ['yrot'
```

Definition at line 1 of file variables\_18.js.

## 30.1592 doc/html/search/variables\_19.js File Reference

### Variables

- var [searchData](#)

### 30.1592.1 Variable Documentation

#### 30.1592.1.1 var searchData

##### Initial value:

```
=
[
 ['z', ['z', ['../jquery_8js.html#adc18d83abfd9f87d396e8fd6b6ac0fe1', 1, 'jquery.js']], ['z_5faxis_5f', ['z_axis_', ['../class_go_1_1_cone.html#a3612c2930148f97c8bec289873b38969', 1, 'Go::Cone::z_axis_()'], ['../class_go_1_1_cylinder.html#ab641d4a68415d0a4b42ef7a1f6ea066a', 1, 'Go::Cylinder::z_axis_()'], ['../class_go_1_1_sphere.html#aca74e80cf0cd2925fa715b9ac5bc30e9', 1, 'Go::Sphere::z_axis_()'], ['../class_go_1_1_torus.html#a9baa1d5ae9abfb4ffe3bfc9658f90139', 1, 'Go::Torus::z_axis_()']], ['zrot'
```

Definition at line 1 of file variables\_19.js.

## 30.1593 doc/html/search/variables\_2.js File Reference

### Variables

- var [searchData](#)

### 30.1593.1 Variable Documentation

#### 30.1593.1.1 var searchData

Definition at line 1 of file variables\_2.js.

## 30.1594 doc/html/search/variables\_3.js File Reference

### Variables

- var [searchData](#)

### 30.1594.1 Variable Documentation

#### 30.1594.1.1 var searchData

Definition at line 1 of file variables\_3.js.

## 30.1595 doc/html/search/variables\_4.js File Reference

### Variables

- var [searchData](#)

### 30.1595.1 Variable Documentation

#### 30.1595.1.1 var searchData

Definition at line 1 of file variables\_4.js.

## 30.1596 doc/html/search/variables\_5.js File Reference

### Variables

- var [searchData](#)

## 30.1596.1 Variable Documentation

## 30.1596.1.1 var searchData

## Initial value:

```

=
[
 ['f', ['f', ['../class_n_e_w_m_a_t_l_1_shifted_matrix.html#ab5a72558c082fafa0a11a708df102156', 1, '
 NEWMAT::ShiftedMatrix']],
 ['f_5f', ['f_', ['../class_go_1_1_parabola.html#a72ad82de25d1106b5120bec75efcab10', 1, 'Go::Parabola']],
 ['face_5f', ['face_', ['../struct_go_1_1_sample_point_data.html#a465f4888fbb14ceecbcee972212ab7c6', 1, '
 Go::SamplePointData']],
 ['face_5fboxes_5f', ['face_boxes_', ['../class_go_1_1ft_cell.html#a9947e2f9b0e6a893dc537205c65c8442', 1, '
 Go::ftCell']],
 ['face_5fchecked_5f', ['face_checked_', ['
 ../class_go_1_1_surface_model.html#adc1642ae9fa9022fd01fa84cb124c43f', 1, 'Go::SurfaceModel']],
 ['face_5fpar_5f', ['face_par_', ['../struct_go_1_1_sample_point_data.html#a76366a7f8bc6787cd792f4802a3de4ae
 ', 1, 'Go::SamplePointData']],
 ['faces2_5f', ['faces2_', ['
 ../class_go_1_1_composite_model_file_handler.html#adaff19f041113e05f5accf5f53040b20', 1, 'Go::CompositeModelFileHandler']],
 ['faces_5f', ['faces_', ['../class_go_1_1ft_cell.html#a1818339a8ddd910cf35a2953c311b4af', 1, '
 Go::ftCell::faces_()'], ['../class_go_1_1_composite_model_file_handler.html#a3a495e311f122a041cddae2f4917dfe5', 1, '
 Go::CompositeModelFileHandler::faces_()'], ['../class_go_1_1ft_surface_set.html#a8a324f6042d0a525ee108e4fafa7d0c1', 1, '
 Go::ftSurfaceSet::faces_()'], ['../class_go_1_1_surface_model.html#a4b5ecf76e4cc91ac7f8886d5fa0cfe3c', 1, '
 Go::SurfaceModel::faces_()']],
 ['factor_5f', ['factor_', ['../struct_go_1_1side_constraint.html#a96f76c21dd13a3cbf2cc957b645894c7', 1, '
 Go::sideConstraint::factor_()'], ['../struct_go_1_1side_constraint_set.html#af7f3d4a1dd8b604945f562e958684188', 1, '
 Go::sideConstraintSet::factor_()'], ['../struct_go_1_1_alg2_d_elem.html#af716c43434f3b63fe96ac23765c33a58', 1, '
 Go::Alg2DElem::factor_()'], ['../struct_go_1_1_alg3_d_elem.html#aa1beae70718c23de5e2b5bfd258b3998', 1, '
 Go::Alg3DElem::factor_()']],
 ['false', ['false_', ['../boolean_8h.html#aa9a4c04243afddc8f2274245a5d39635', 1, 'boolean.h']],
 ['feedback_5fmode_5f', ['feedback_mode_', ['../classgv_view.html#a680a104c5df070cdf3496de05a947090', 1, '
 gvView']],
 ['finetex_5f', ['fineTex_', ['../classgv_view.html#ac480b98c2f1b29febdb96a6e7df3972b', 1, 'gvView']],
 ['first_5f', ['first_', ['../class_go_1_1ft_point_set.html#a5c1c2f93ee3fd048bfb0f84bc4dlaa0', 1, '
 Go::ftPointSet']],
 ['flags_5f', ['flags_', ['../class_dijkstra.html#a21e1b614db2ae36d5af65313b0780334', 1, 'Dijkstra::flags_()'],
 ['../class_multi_dijkstra.html#a0d6da2fa60a1854c199275fc1218e63f', 1, 'MultiDijkstra::flags_()']],
 ['focal_5fpoint_5f', ['focal_point_', ['../classgv_camera.html#ae4a942409e3ad7fd12c744ce9e94b4ef', 1, '
 gvCamera']],
 ['focus_5fon_5forigin_5f', ['focus_on_origin_', ['../classgv_view.html#a531e25ac8d4e553ccd64c8c405a9c9b0', 1, '
 gvView']],
 ['form', ['form', ['../struct_go_1_1_i_g_e_sdirentry.html#a8606d475e0ffca150e9946c5308de47e', 1, '
 Go::IGESdirentry']],
 ['form_5f', ['form_', ['../classgv_action_sheet.html#aeaf5f0304b7c950f719f027fd6999792', 1, 'gvActionSheet']]
],
 ['fractionrendered_5f', ['fractionrendered_', ['
 ../classgv_line_cloud_paintable.html#a090b222902dd687ec00d31dda642327c', 1, 'gvLineCloudPaintable::fractionrendered_()'],
 ../classgv_point_cloud_paintable.html#a1a860092e7b53459ea85b06b3b605f49', 1, 'gvPointCloudPaintable::fractionrendered_()'],
 ['frames_5fwithout_5fmovement', ['frames_without_movement', ['
 ../sisl_view_demo_8cpp.html#a184c97d15071596397d8f23902ec3fa9', 1, 'sisl_view_demo.cpp']],
 ['func_5fint_5f', ['func_int_', ['
 ../class_go_1_1_intersector_func_const.html#abc8f68ae80af7bc1a741a24c728d61aa', 1, 'Go::IntersectorFuncConst']],
 ['funnel_5f', ['funnel_', ['../class_path_type.html#aa494c02211096f0fb748cc6e41002b01', 1, 'PathType']]
]

```

Definition at line 1 of file variables\_5.js.

## 30.1597 doc/html/search/variables\_6.js File Reference

## Variables

- var [searchData](#)

## 30.1597.1 Variable Documentation

### 30.1597.1.1 var searchData

#### Initial value:

```
=
[
 ['g_5f', ['g_', ['../class_pr_parametrize_bdy.html#aflee487b52611457d3a81147f130021a', 1, 'PrParametrizeBdy::g()'], ['../class_pr_parametrize_int.html#a36919da9715f5f1625e062c6e6b27176', 1, 'PrParametrizeInt::g()'], ['gap', 'gap', ['../struct_go_1_ltp_tolerances.html#a62617afddb73267fdc536da40b3932cc', 1, 'Go::tpTolerances']],
 ['gauss_5fpar1_5f', ['gauss_par1_', ['../struct_go_1_lpre_evaluation_sf.html#a0393353862a463947fde80079d490cce', 1, 'Go::preEvaluationSf::gauss_par1()'], ['../struct_go_1_lpre_evaluation_vol.html#a9b92c9130ab7bdf6f0a264caf05a876', 1, 'Go::preEvaluationVol::gauss_par1()']],
 ['gauss_5fpar2_5f', ['gauss_par2_', ['../struct_go_1_lpre_evaluation_sf.html#aceld1bd2b24ca88754fdeb7ea99bd4d9', 1, 'Go::preEvaluationSf::gauss_par2()'], ['../struct_go_1_lpre_evaluation_vol.html#a8a9d745ba22ef7316928f5708f7c2218', 1, 'Go::preEvaluationVol::gauss_par2()']],
 ['gauss_5fpar3_5f', ['gauss_par3_', ['../struct_go_1_lpre_evaluation_vol.html#a12bc6c681b912d957caef1efa6c1fed5', 1, 'Go::preEvaluationVol']],
 ['geo_5fdata_5f1', ['geo_data_1', ['../struct_s_i_s_l_intpt.html#ad4083a64bc31371f487e7943157aa92b', 1, 'SISLIntpt']],
 ['geo_5fdata_5f2', ['geo_data_2', ['../struct_s_i_s_l_intpt.html#ae48801801ddc36807cf6885b69c9002d', 1, 'SISLIntpt']],
 ['geo_5ftrack_5f2d_5f1', ['geo_track_2d_1', ['../struct_s_i_s_l_intpt.html#acde072bd6f03b86ec5b894fc7f65144d', 1, 'SISLIntpt']],
 ['geo_5ftrack_5f2d_5f2', ['geo_track_2d_2', ['../struct_s_i_s_l_intpt.html#a08a00fed068ef573184fc50537408adc', 1, 'SISLIntpt']],
 ['geo_5ftrack_5f3d', ['geo_track_3d', ['../struct_s_i_s_l_intpt.html#a643bd8d87fb2dc110022a57994c42e74', 1, 'SISLIntpt']],
 ['geom_5fobjects2_5f', ['geom_objects2_', ['../class_go_1_l_composite_model_file_handler.html#a9bf5291a1f1aee5a9c54939077faald8', 1, 'Go::CompositeModelFileHandler']],
 ['geom_5fobjects_5f', ['geom_objects_', ['../class_go_1_l_composite_model_file_handler.html#a99d727511608d3d40666537f0cd0ac49', 1, 'Go::CompositeModelFileHandler']],
 ['geomobj_5f', ['geomobj_', ['../class_go_1_lft_group_geom.html#a777549837a91518c25e875647a5b4280', 1, 'Go::ftGroupGeom']],
 ['get_5fclick_5fmode_5f', ['get_click_mode_', ['../classgv_view.html#af4a6ddfe77052651d7292dd41846e0ad', 1, 'gvView']],
 ['ghost_5fpoints_5f', ['ghost_points_', ['../struct_go_1_l_l_s_smooth_data.html#a60a41bf1e1bfe6360960ec740d1dd55e', 1, 'Go::LSSmoothData']],
 ['gl_5finitialized_5f', ['gl_initialized_', ['../classgv_view.html#a491842e1784e955d1e5566823620e50a', 1, 'gvView']],
 ['gm', ['gm', ['../class_n_e_w_m_a_t_l_l_shifted_matrix.html#a19701ca94daecd6cdf5f137df145de8f', 1, 'NEWMAT::ShiftedMatrix::gm()'], ['../class_n_e_w_m_a_t_l_l_negated_matrix.html#a52bb6ea0a318be83c029bfa48346884a', 1, 'NEWMAT::NegatedMatrix::gm()'], ['../class_matrix_row_col.html#ac4f08deedb6355adb62e04f7acf7fc17', 1, 'MatrixRowCol::gm()']],
 ['gml', ['gml', ['../class_n_e_w_m_a_t_l_l_multiplied_matrix.html#a7803a6d673c5d8761701a9bccbe8c91d', 1, 'NEWMAT::MultipliedMatrix']],
 ['gm2', ['gm2', ['../class_n_e_w_m_a_t_l_l_multiplied_matrix.html#ab52429edb38c4019ebec3accd2806115', 1, 'NEWMAT::MultipliedMatrix']],
 ['gmat_5f', ['gmat_', ['../class_go_1_l_smooth_surf.html#af59b5eec6dee89daff37ea7bb017223', 1, 'Go::SmoothSurf::gmat()'], ['../class_go_1_l_smooth_surf_set.html#ad1e20276ff25cfd6a017b252adf9a0d', 1, 'Go::SmoothSurfSet::gmat()']],
 ['graph_5f', ['graph_', ['../class_go_1_lft_chart_surface.html#ae9f9c4172b0540eca8a2a4d19bc15d6b', 1, 'Go::ftChartSurface::graph()'], ['../class_dijkstra.html#a35ec62ac1573a9a6a74b4b53fd70dad', 1, 'Dijkstra::graph()'], ['../class_multi_dijkstra.html#a01cba4633b91a539cb4b663016db6d70', 1, 'MultiDijkstra::graph()']],
 ['grid_5fdistr_5ffunctions_5f', ['grid_distr_functions_', ['../class_go_1_lft_chart_surface.html#a3d6a267675eca4961d6f0e572e02a5d5', 1, 'Go::ftChartSurface']],
 ['grid_5fpts_5f', ['grid_pts_', ['../class_go_1_lft_chart_surface.html#a016eedbcc6a8bb3c6e73b983d89a08d0', 1, 'Go::ftChartSurface']],
 ['gright_5f', ['gright_', ['../class_go_1_l_smooth_surf.html#a33241dea8f6b1dad9b700de910a5bae', 1, 'Go::SmoothSurf::gright()'], ['../class_go_1_l_smooth_surf_set.html#abb8d17ab393b7f693037d20085d694cd', 1, 'Go::SmoothSurfSet::gright()']],
 ['group_5fmenu_5f', ['group_menu_', ['../classgv_application.html#a9471d39cad8ee5486beada02c57265f2', 1, 'gvApplication']],
 ['groups', ['groups', ['../structrank__info.html#a73fee02c95e70e58861325236285ad37', 1, 'rank_info']]
]
```

Definition at line 1 of file variables\_6.js.

## 30.1598 doc/html/search/variables\_7.js File Reference

### Variables

- var [searchData](#)



### 30.1598.1 Variable Documentation

#### 30.1598.1.1 var searchData

**Initial value:**

```
=
[
 ['heap_5f', ['heap_', ['../class_pr_thin.html#a6c903d183c116bcf9c8efde3a7e1d1d1', 1, 'PrThin']],
 ['highest_5fface_5fchecked_5f', ['highest_face_checked_', ['
 ../class_go_1_1_surface_model.html#ab7a360504bcdb200709539d79cab886b', 1, 'Go::SurfaceModel']]]]
]
```

Definition at line 1 of file variables\_7.js.

## 30.1599 doc/html/search/variables\_8.js File Reference

### Variables

- var [searchData](#)

### 30.1599.1 Variable Documentation

#### 30.1599.1.1 var searchData

Definition at line 1 of file variables\_8.js.

## 30.1600 doc/html/search/variables\_9.js File Reference

### Variables

- var [searchData](#)

### 30.1600.1 Variable Documentation

#### 30.1600.1.1 var searchData

**Initial value:**

```
=
[
 ['jcol_5f', ['jcol_', ['../class_go_1_1_solve_c_g.html#a5718e8d392c16f9adfc2b1c2147b8c8f', 1, 'Go::SolveCG']],
],
 ['joint_5f', ['joint_', ['../class_go_1_1_ft_curve_segment.html#afc24962e36a118cdb6a9950374410b03', 1, '
 Go::ftCurveSegment']]],
 ['jon_5ftimer', ['jon_timer', ['../aux2_8cpp.html#ab6b3a0f92f7304c85e25c9422c4c959c', 1, 'aux2.cpp']]]]
]
```

Definition at line 1 of file variables\_9.js.

## 30.1601 doc/html/search/variables\_a.js File Reference

### Variables

- var [searchData](#)

### 30.1601.1 Variable Documentation

#### 30.1601.1.1 var searchData

#### Initial value:

```
=
[
 ['k', ['k', ['../jquery_8js.html#ab26645c014aa005ecedef329ecf58c99', 1, 'jquery.js']],
 ['kdim_5f', ['kdim_', ['../class_go_1_1_smooth_surf.html#aecbfa2bdb4142f2207c88249237cf358', 1, '
 Go::SmoothSurf::kdim_()'], ['../class_go_1_1_smooth_surf_set.html#a16650b6975a4aaf03bf49ccd70a973e1', 1, '
 Go::SmoothSurfSet::kdim_()']],
 ['key_5f', ['key_', ['../class_heap_node.html#ae07076e227f6a24b9a70d2dff7031110', 1, 'HeapNode::key_()'], ['
 ../class_heap_node2.html#a5834899269c09b811d1f2cf807e41e22', 1, 'HeapNode2::key_()']],
 ['kink', ['kink', ['../struct_go_1_1_tp_tolerances.html#a587c6924aaf46d129e70defebf9cdf73', 1, '
 Go::tpTolerances']],
 ['kk1_5f', ['kk1_', ['../class_go_1_1_smooth_surf.html#ad616047930638a6351bdd0b2e82dc9d1', 1, 'Go::SmoothSurf
 ']],
 ['kk2_5f', ['kk2_', ['../class_go_1_1_smooth_surf.html#a1e00323a454408b86c724dd46fbc1b9c', 1, 'Go::SmoothSurf
 ']],
 ['kn1_5f', ['kn1_', ['../class_go_1_1_smooth_surf.html#a548dd3b669c60ffd8f971feac58a2053', 1, 'Go::SmoothSurf
 ']],
 ['kn2_5f', ['kn2_', ['../class_go_1_1_smooth_surf.html#a1ab34f0fee55320b22914774b1e943c6', 1, 'Go::SmoothSurf
 ']],
 ['kncond_5f', ['kncond_', ['../class_go_1_1_smooth_surf.html#a63899626d5ffddfc45f9d6aa22f44dd4', 1, '
 Go::SmoothSurf::kncond_()'], ['../class_go_1_1_smooth_surf_set.html#aa530cc21c18294d89efb3e986b5f1ad3', 1, '
 Go::SmoothSurfSet::kncond_()']],
 ['knconstraint_5f', ['knconstraint_', ['../class_go_1_1_smooth_surf.html#a9a80424597864d0adf36ef300ca2a41a'
 , 1, 'Go::SmoothSurf::knconstraint_()'], ['
 ../class_go_1_1_smooth_surf_set.html#ab4b15fa0722035377b4651a00d147e5f', 1, 'Go::SmoothSurfSet::knconstraint_()']],
 ['kpointer_5f', ['kpointer_', ['../class_go_1_1_smooth_surf.html#a475e1db2336e9bc042f739d4a74a2917', 1, '
 Go::SmoothSurf']],
 ['kval', ['kval', ['
 ../struct_go_1_1_l_r_spline_surface_1_1_refinement2_d.html#ad68d760801298c71f45dd219074943ea', 1, 'Go::LRSplineSurface::R
]
```

Definition at line 1 of file variables\_a.js.

## 30.1602 doc/html/search/variables\_b.js File Reference

### Variables

- var [searchData](#)
- mouse [cpp](#)

### 30.1602.1 Variable Documentation

#### 30.1602.1.1 mouse cpp

Definition at line 14 of file variables\_b.js.

## 30.1602.1.2 var searchData

## Initial value:

```
=
[
 ['l', ['L', ['../jquery_8js.html#a38ee4c0b5f4fe2a18d0c783af540d253', 1, 'jquery.js']]],
 ['l_5f', ['l', ['../struct_go_l_l_param_surface_1_degenerate__info.html#a9e9e4ddd3a9bad4fc495e1ca428d1c4e9', 1, 'Go::ParamSurface::degenerate_info']]],
 ['l_5fpath_5f', ['l_path_', ['../class_path_type.html#a3849d634c4440974a3e8664f0adf2c88', 1, 'PathType']]],
 ['label_5f', ['label_', ['../class_multi_dijkstra.html#a7e2736e3c6d24b2a5b451c5d6f4b02ff', 1, 'MultiDijkstra']]],
 ['label_5fdisplay', ['label_display', ['../struct_go_l_l_i_g_e_sdirentry.html#a2b8d7c0041720a2cc10d1eab59d69171', 1, 'Go::IGESdirentry']]],
 ['large_5fdistance_5f', ['large_distance_', ['../class_dijkstra.html#af7d4f8f644cdf1330bca5ce819fa58a2', 1, 'Dijkstra::large_distance_()'], ['../class_multi_dijkstra.html#a4cc650703fd03dae2d075afb06563c2f', 1, 'MultiDijkstra::large_distance_()']]],
 ['last', ['last', ['../class_r_b_d__c_o_m_m_o_n_l_l_tracer.html#a93a99ea96d8fc44d27b51629c2071cd1', 1, 'RBD_COMMON::Tracer']]],
 ['last_5fchange_5f', ['last_change_', ['../struct_go_l_l_registration_result.html#a7df6e84c504a75d83f47aa2de92d747d', 1, 'Go::RegistrationResult']]],
 ['last_5ffile_5fname_5f', ['last_file_name_', ['../classgv_application.html#a69d06231d8c542521ae2e638c7033a23', 1, 'gvApplication']]],
 ['last_5fmouse_5fpos_5f', ['last_mouse_pos_', ['../classgv_view.html#a08505548abel1069e5600ae0b17271a7', 1, 'gvView']]],
 ['last_5fnewton_5fiteration_5f', ['last_newton_iteration_', ['../struct_go_l_l_registration_result.html#a0955123270a8c94b76919e794f6e8e67', 1, 'Go::RegistrationResult']]],
 ['last_5fx', ['last_x', ['../mouse_8h.html#abbf254766520cad4d7fe189eb184194c', 1, 'last_x(): ']]]
]
```

Definition at line 1 of file variables\_b.js.

## 30.1603 doc/html/search/variables\_c.js File Reference

## Variables

- var [searchData](#)
- mouse [cpp](#)

## 30.1603.1 Variable Documentation

## 30.1603.1.1 mouse cpp

Definition at line 39 of file variables\_c.js.

## 30.1603.1.2 var searchData

Definition at line 1 of file variables\_c.js.

## 30.1604 doc/html/search/variables\_d.js File Reference

## Variables

- var [searchData](#)

### 30.1604.1 Variable Documentation

#### 30.1604.1.1 var searchData

Definition at line 1 of file variables\_d.js.

## 30.1605 doc/html/search/variables\_e.js File Reference

### Variables

- var [searchData](#)

### 30.1605.1 Variable Documentation

#### 30.1605.1.1 var searchData

#### Initial value:

```
=
[
 ['ol', ['ol', ['../struct_s_i_s_l_object.html#aa40c25309fff2ffd852103f2af7acd1f', 1, 'SISLObject']]],
 ['obj_5fint_5f', ['obj_int_', ['../class_go_1_1_intersector2_obj.html#ae87a91f8f4f95a31dc6a07678eabb132', 1,
 'Go::Intersector2Obj']]],
 ['object_5fcolors_5f', ['object_colors_', ['../classgv_data.html#a4efc6f51ba7d6d6fbad2afe862d7aac', 1, '
 gvData']]],
 ['object_5fmenu_5f', ['object_menu_', ['../classgv_application.html#af224575df535f0d2d9cce3dd6201d86e', 1, '
 gvApplication']]],
 ['objects_5f', ['objects_', ['../classgv_data.html#ac73db2f252c3d82c40f4b8049f6b9c2c', 1, 'gvData']]],
 ['omega_5f', ['omega_', ['../class_go_1_1_solve_c_g.html#a895bee6e49c4b8bfefdde37741693433', 1, 'Go::SolveCG'
]]],
 ['onlyoldfft', ['OnlyOldFFT', ['
 ../class_n_e_w_m_a_t_l_l_f_f_t_controller.html#aa5d0a0c2808fe16510e683f0f8283a95', 1, 'NEWMAT::FFT_Controller']]],
 ['opp_5f', ['opp_', ['../struct_go_1_1cv_set_constraint.html#a1596f6a4ffd640d10699ac01d5b179a4', 1, '
 Go::cvSetConstraint']]],
 ['organisation', ['organisation', ['../struct_go_1_1_i_g_e_sheader.html#a28f1fe259948ad7be77a9233b9a31e58',
 1, 'Go::IGESheader']]],
 ['orientation_5f', ['orientation_', ['
 ../struct_go_1_1_composite_model_file_handler_1_lsfcvinfo.html#a37202c0c673bb47d999ee7391da3dd92', 1, 'Go::CompositeModel'
]]],
 ['orientation_5finconsist_5f', ['orientation_inconsist_', ['
 ../class_go_1_1tp_topology_table.html#af611ebf3e740eef65964c53e827cb463', 1, 'Go::tpTopologyTable']]],
 ['orienteddomain_5f', ['orientedDomain_', ['../class_go_1_1_cone.html#a7fdf1219e6a965d4a1d45090ded24d11', 1,
 'Go::Cone::orientedDomain()'], ['../class_go_1_1_cylinder.html#aa8367eca5239f784a015d95f6a4a9df3', 1, '
 Go::Cylinder::orientedDomain()'], ['../class_go_1_1_plane.html#a33aa83fede5fa4caf78aba009d52f749', 1, '
 Go::Plane::orientedDomain()'], ['../class_go_1_1_sphere.html#a8a94511a68da84ba37b7a156e7e47f46', 1, '
 Go::Sphere::orientedDomain()'], ['../class_go_1_1_torus.html#ac69373ff0df20e0598490de1e8f86479', 1, 'Go::Torus::orientedD
 '
]]],
 ['orig_5fsurf_5f', ['orig_surf_', ['../class_go_1_1ft_smooth_surf.html#a99cfc4c9accc4aa03bb3a1d080c0d734', 1, '
 'Go::ftSmoothSurf']]],
 ['original_5ffilename', ['original_filename', ['
 ../struct_go_1_1_i_g_e_sheader.html#aa26adfff0cf4e3200a5153af8a748e1e', 1, 'Go::IGESheader']]],
 ['origine_5f', ['origine_', ['../class_unf_node_type.html#a6473043efe97dd34ceb497ee310efb63', 1, 'UnfNodeType'
]]],
 ['outside', ['outside', ['../struct_go_1_1_cached_interval.html#aa3934c163e9e6f7234e8b62e28c01fe5', 1, '
 Go::CachedInterval']]]
]
```

Definition at line 1 of file variables\_e.js.

## 30.1606 doc/html/search/variables\_f.js File Reference

### Variables

- var [searchData](#)
- [gl\\_aux](#) `cpp`

### 30.1606.1 Variable Documentation

#### 30.1606.1.1 `gl_aux` `cpp`

Definition at line 48 of file `variables_f.js`.

#### 30.1606.1.2 `var searchData`

Definition at line 1 of file `variables_f.js`.

## 30.1607 doc/html/sh1260\_8c.js File Reference

### Variables

- `var sh1260_8c`

### 30.1607.1 Variable Documentation

#### 30.1607.1.1 `var sh1260_8c`

##### Initial value:

```
=
[
 ["SH1260", "sh1260_8c.html#abe5f60c3fac3e45a4405c2e84275e337", null],
 ["sh1260", "sh1260_8c.html#a8362f46565d3a936348c4dedd1e67d3e", null]
]
```

Definition at line 1 of file `sh1260_8c.js`.

## 30.1608 doc/html/sh1261\_8c.js File Reference

### Variables

- `var sh1261_8c`

### 30.1608.1 Variable Documentation

#### 30.1608.1.1 `var sh1261_8c`

##### Initial value:

```
=
[
 ["SH1261", "sh1261_8c.html#a99a509c3d36da28fa785ffded0aca919", null],
 ["sh1261", "sh1261_8c.html#a46dd0fc16373e6d975ad69e306ed98a2", null]
]
```

Definition at line 1 of file `sh1261_8c.js`.

## 30.1609 doc/html/sh1262\_8c.js File Reference

### Variables

- var [sh1262\\_8c](#)

### 30.1609.1 Variable Documentation

#### 30.1609.1.1 var sh1262\_8c

##### Initial value:

```
=
[
 ["SH1262", "sh1262_8c.html#aa4845fb276046edaaabe9bc64f18fa42", null],
 ["sh1262", "sh1262_8c.html#aa57527251fed4b3aca7e2f92dfed5c4b", null]
]
```

Definition at line 1 of file sh1262\_8c.js.

## 30.1610 doc/html/sh1263\_8c.js File Reference

### Variables

- var [sh1263\\_8c](#)

### 30.1610.1 Variable Documentation

#### 30.1610.1.1 var sh1263\_8c

##### Initial value:

```
=
[
 ["SH1263", "sh1263_8c.html#aa989423354156a3b02b46b941fcf3c2e", null],
 ["sh1263", "sh1263_8c.html#a9b5ccb8970b802a6bd77f5b43cd2962e", null]
]
```

Definition at line 1 of file sh1263\_8c.js.

## 30.1611 doc/html/sh1365\_8c.js File Reference

### Variables

- var [sh1365\\_8c](#)

### 30.1611.1 Variable Documentation

#### 30.1611.1.1 var sh1365\_8c

**Initial value:**

```
=
[
 ["SH1365", "sh1365_8c.html#a305dc6a00b79de55ebaaf313302bbe26", null],
 ["sh1365", "sh1365_8c.html#a7db1ce4d838b67c7811ff17f10cde851", null]
]
```

Definition at line 1 of file sh1365\_8c.js.

## 30.1612 doc/html/sh1369\_8c.js File Reference

### Variables

- var [sh1369\\_8c](#)

### 30.1612.1 Variable Documentation

#### 30.1612.1.1 var sh1369\_8c

**Initial value:**

```
=
[
 ["SH1369", "sh1369_8c.html#aecff2d90ea6f5cbf78e043f719bc0ad7", null],
 ["sh1369", "sh1369_8c.html#aa4e206f7275423c07544bc63965c32ad", null]
]
```

Definition at line 1 of file sh1369\_8c.js.

## 30.1613 doc/html/sh1371\_8c.js File Reference

### Variables

- var [sh1371\\_8c](#)

### 30.1613.1 Variable Documentation

#### 30.1613.1.1 var sh1371\_8c

**Initial value:**

```
=
[
 ["SH1371", "sh1371_8c.html#adcc1c9251c9be4978f34e7c36a6e1a99", null],
 ["sh1371", "sh1371_8c.html#aa78c74e33f7a0ce1d74d152fda8be064", null]
]
```

Definition at line 1 of file sh1371\_8c.js.

## 30.1614 doc/html/sh1372\_8c.js File Reference

### Variables

- var [sh1372\\_8c](#)

#### 30.1614.1 Variable Documentation

##### 30.1614.1.1 var sh1372\_8c

###### Initial value:

```
=
[
 ["SH1372", "sh1372_8c.html#a794ffa03ab41b372531107ac16e14808", null],
 ["sh1372", "sh1372_8c.html#a3506e2ece13ed65351981e872250c872", null]
]
```

Definition at line 1 of file sh1372\_8c.js.

## 30.1615 doc/html/sh1373\_8c.js File Reference

### Variables

- var [sh1373\\_8c](#)

#### 30.1615.1 Variable Documentation

##### 30.1615.1.1 var sh1373\_8c

###### Initial value:

```
=
[
 ["SH1373", "sh1373_8c.html#a64655426580e2f263f3eb2e8010b93a4", null],
 ["sh1373", "sh1373_8c.html#ac79dddce554c1e8e268701f957ef8d1", null]
]
```

Definition at line 1 of file sh1373\_8c.js.

## 30.1616 doc/html/sh1374\_8c.js File Reference

### Variables

- var [sh1374\\_8c](#)



### 30.1616.1 Variable Documentation

#### 30.1616.1.1 var sh1374\_8c

**Initial value:**

```
=
[
 ["SH1374", "sh1374_8c.html#a4c65a85d1c85e267ee8796a8ca84e352", null],
 ["sh1374", "sh1374_8c.html#a160347d59a1ae6c1c7604489cae8d421", null]
]
```

Definition at line 1 of file sh1374\_8c.js.

## 30.1617 doc/html/sh1375\_8c.js File Reference

### Variables

- var [sh1375\\_8c](#)

### 30.1617.1 Variable Documentation

#### 30.1617.1.1 var sh1375\_8c

**Initial value:**

```
=
[
 ["SH1375", "sh1375_8c.html#a26b1fd1ed78e0f27a5d0d55126ea2ad4", null],
 ["sh1375", "sh1375_8c.html#a3258c9f42e1344006302641c8e854f23", null]
]
```

Definition at line 1 of file sh1375\_8c.js.

## 30.1618 doc/html/sh1460\_8c.js File Reference

### Variables

- var [sh1460\\_8c](#)

### 30.1618.1 Variable Documentation

#### 30.1618.1.1 var sh1460\_8c

**Initial value:**

```
=
[
 ["SH1460", "sh1460_8c.html#a390d45a22918c5fbeleca4506cb0806d1", null],
 ["fevalmidProc", "sh1460_8c.html#ad75504a98b85ca170c1038cb52b6ae3f", null],
 ["fshapeProc", "sh1460_8c.html#a23a1402d8877b20ca4dec1197f85940c", null],
 ["sh1460", "sh1460_8c.html#a62f89b5fa4886ce4180901dc831cd1c1", null]
]
```

Definition at line 1 of file sh1460\_8c.js.

## 30.1619 doc/html/sh1461\_8c.js File Reference

### Variables

- var [sh1461\\_8c](#)

### 30.1619.1 Variable Documentation

#### 30.1619.1.1 var sh1461\_8c

##### Initial value:

```
=
[
 ["SH1461", "sh1461_8c.html#a659bf97e55788e16f2b68b5bdb99f8fa", null],
 ["fshapeProc", "sh1461_8c.html#a23a1402d8877b20ca4dec1197f85940c", null],
 ["initProc", "sh1461_8c.html#a6d9a3f0bc40bd855afe60e6376874834", null],
 ["sh1461", "sh1461_8c.html#a737aa3f1124035f889ce642747d7e446", null]
]
```

Definition at line 1 of file sh1461\_8c.js.

## 30.1620 doc/html/sh1462\_8c.js File Reference

### Variables

- var [sh1462\\_8c](#)

### 30.1620.1 Variable Documentation

#### 30.1620.1.1 var sh1462\_8c

##### Initial value:

```
=
[
 ["SH1462", "sh1462_8c.html#a1d1411d6ef8dbc2ebc01dbcad9fcd384", null],
 ["fshapeProc", "sh1462_8c.html#a23a1402d8877b20ca4dec1197f85940c", null],
 ["sh1462", "sh1462_8c.html#ae140df5610ea0cbcc92a89c5c457f8af", null]
]
```

Definition at line 1 of file sh1462\_8c.js.

## 30.1621 doc/html/sh1463\_8c.js File Reference

### Variables

- var [sh1463\\_8c](#)

### 30.1621.1 Variable Documentation

#### 30.1621.1.1 var sh1463\_8c

**Initial value:**

```
=
[
 ["SH1463", "sh1463_8c.html#a5968742ce2b7a8eabec0a90ac63afc06", null],
 ["fshapeProc", "sh1463_8c.html#a23a1402d8877b20ca4dec1197f85940c", null],
 ["sh1463", "sh1463_8c.html#af792c61e308e48ae80a25568d9bf088c", null]
]
```

Definition at line 1 of file sh1463\_8c.js.

## 30.1622 doc/html/sh1464\_8c.js File Reference

### Variables

- var [sh1464\\_8c](#)

### 30.1622.1 Variable Documentation

#### 30.1622.1.1 var sh1464\_8c

**Initial value:**

```
=
[
 ["SH1464", "sh1464_8c.html#a79b0e269d264c1924d95779ffe90dd60", null],
 ["fshapeProc", "sh1464_8c.html#a23a1402d8877b20ca4dec1197f85940c", null],
 ["sh1464", "sh1464_8c.html#aad008c2f86869eb80850f5bc3ed599f0", null]
]
```

Definition at line 1 of file sh1464\_8c.js.

## 30.1623 doc/html/sh1465\_8c.js File Reference

### Variables

- var [sh1465\\_8c](#)

### 30.1623.1 Variable Documentation

#### 30.1623.1.1 var sh1465\_8c

**Initial value:**

```
=
[
 ["SH1465", "sh1465_8c.html#a2b8a452cc5bd01b5f01b514442506f50", null],
 ["fshapeProc", "sh1465_8c.html#a23a1402d8877b20ca4dec1197f85940c", null],
 ["sh1465", "sh1465_8c.html#a122be3d1a9ec3e190df13e8c1326cace", null]
]
```

Definition at line 1 of file sh1465\_8c.js.

### 30.1624 doc/html/sh1466\_8c.js File Reference

**Variables**

- var [sh1466\\_8c](#)

#### 30.1624.1 Variable Documentation

##### 30.1624.1.1 var sh1466\_8c

**Initial value:**

```
=
[
 ["SH1466", "sh1466_8c.html#a6558865b299f80f4f025ce4e93a6ceae", null],
 ["sh1466", "sh1466_8c.html#a8c80931206fdc217ec8a018fe596fef6", null]
]
```

Definition at line 1 of file sh1466\_8c.js.

### 30.1625 doc/html/sh1467\_8c.js File Reference

**Variables**

- var [sh1467\\_8c](#)

#### 30.1625.1 Variable Documentation

##### 30.1625.1.1 var sh1467\_8c

**Initial value:**

```
=
[
 ["SH1467", "sh1467_8c.html#a1d0a80034ec2b8d43b80fb5d851a0697", null],
 ["sh1467", "sh1467_8c.html#a989ec87a690d7c2f606d1350af881fa7", null]
]
```

Definition at line 1 of file sh1467\_8c.js.

## 30.1626 doc/html/sh1502\_8c.js File Reference

### Variables

- var [sh1502\\_8c](#)

#### 30.1626.1 Variable Documentation

##### 30.1626.1.1 var sh1502\_8c

###### Initial value:

```
=
[
 ["SH1502", "sh1502_8c.html#a77974de8a546cd7fd5b04e857a7d2c2c", null],
 ["sh1502", "sh1502_8c.html#ac627be8cf40a48ae96f758b19ee87b10", null]
]
```

Definition at line 1 of file sh1502\_8c.js.

## 30.1627 doc/html/sh1503\_8c.js File Reference

### Variables

- var [sh1503\\_8c](#)

#### 30.1627.1 Variable Documentation

##### 30.1627.1.1 var sh1503\_8c

###### Initial value:

```
=
[
 ["SH1503", "sh1503_8c.html#a1449809df05ba26721f71c4a1dae41fa", null],
 ["sh1503", "sh1503_8c.html#adfe7973592d367603215eccal7dcd10b", null]
]
```

Definition at line 1 of file sh1503\_8c.js.

## 30.1628 doc/html/sh1510\_8c.js File Reference

### Variables

- var [sh1510\\_8c](#)

### 30.1628.1 Variable Documentation

#### 30.1628.1.1 var sh1510\_8c

**Initial value:**

```
=
[
 ["SH1510", "sh1510_8c.html#ab9dc6b621a55229d321a371427ebf481", null],
 ["sh1510", "sh1510_8c.html#ae917347e4401ed2d74a38eeb007474a6", null]
]
```

Definition at line 1 of file sh1510\_8c.js.

### 30.1629 doc/html/sh1511\_8c.js File Reference

**Variables**

- var [sh1511\\_8c](#)

#### 30.1629.1 Variable Documentation

##### 30.1629.1.1 var sh1511\_8c

**Initial value:**

```
=
[
 ["SH1511", "sh1511_8c.html#a365a01a237c007e42add6fe64dc3b3d5", null],
 ["sh1511", "sh1511_8c.html#ad23cffd359644069b7dc53d48ca5ff42", null]
]
```

Definition at line 1 of file sh1511\_8c.js.

### 30.1630 doc/html/sh1761\_8c.js File Reference

**Variables**

- var [sh1761\\_8c](#)

#### 30.1630.1 Variable Documentation

##### 30.1630.1.1 var sh1761\_8c

**Initial value:**

```
=
[
 ["SH1761", "sh1761_8c.html#a403dc78655e2f8bd272df85bf01aad5f", null],
 ["sh1761", "sh1761_8c.html#a9e614a6235f0ddff0557c04167d3151a", null]
]
```

Definition at line 1 of file sh1761\_8c.js.

## 30.1631 doc/html/sh1762\_8c.js File Reference

### Variables

- var [sh1762\\_8c](#)

#### 30.1631.1 Variable Documentation

##### 30.1631.1.1 var sh1762\_8c

###### Initial value:

```
=
[
 ["SH1762", "sh1762_8c.html#a45c9c699ca514464cceb3777aa77295c", null],
 ["sh1762", "sh1762_8c.html#a778406bc5487cf97f09bc3f9a57b572f", null]
]
```

Definition at line 1 of file sh1762\_8c.js.

## 30.1632 doc/html/sh1779\_8c.js File Reference

### Variables

- var [sh1779\\_8c](#)

#### 30.1632.1 Variable Documentation

##### 30.1632.1.1 var sh1779\_8c

###### Initial value:

```
=
[
 ["SH1779", "sh1779_8c.html#a5c4ff4aa2d70f71b3f1581864480753d", null],
 ["sh1779", "sh1779_8c.html#a556c59fca88dc63f9034707c338372d3", null]
]
```

Definition at line 1 of file sh1779\_8c.js.

## 30.1633 doc/html/sh1779\_\_at\_8c.js File Reference

### Variables

- var [sh1779\\_\\_at\\_8c](#)

### 30.1633.1 Variable Documentation

30.1633.1.1 `var sh1779__at_8c`

**Initial value:**

```
=
[
 ["SH1779_AT", "sh1779__at_8c.html#aealf61ef0d72e41ba6536b91aeddffa3f", null],
 ["sh1779_at", "sh1779__at_8c.html#aa6a1f7ac01c38e3e99808f4c53d50df2", null]
]
```

Definition at line 1 of file sh1779\_\_at\_8c.js.

## 30.1634 doc/html/sh1780\_8c.js File Reference

### Variables

- `var sh1780_8c`

### 30.1634.1 Variable Documentation

30.1634.1.1 `var sh1780_8c`

**Initial value:**

```
=
[
 ["SH1780", "sh1780_8c.html#aa765c64e5a9014ac6df3571bdlee12b0", null],
 ["sh1780", "sh1780_8c.html#a315c6078afaa9b07373dbae20679115b", null]
]
```

Definition at line 1 of file sh1780\_8c.js.

## 30.1635 doc/html/sh1780\_\_at\_8c.js File Reference

### Variables

- `var sh1780__at_8c`

### 30.1635.1 Variable Documentation

30.1635.1.1 `var sh1780__at_8c`

**Initial value:**

```
=
[
 ["SH1780_AT", "sh1780__at_8c.html#aa54a684d7c33261f9682f6d0e59155bf", null],
 ["sh1780_at", "sh1780__at_8c.html#ac7bac4645fc95d09e2c98cd07987a656", null]
]
```

Definition at line 1 of file sh1780\_\_at\_8c.js.



## 30.1636 doc/html/sh1781\_8c.js File Reference

### Variables

- var [sh1781\\_8c](#)

#### 30.1636.1 Variable Documentation

##### 30.1636.1.1 var sh1781\_8c

###### Initial value:

```
=
[
 ["SH1781", "sh1781_8c.html#afbf59192fdd9ee79c689db0895d9722f", null],
 ["sh1781", "sh1781_8c.html#a973391a93780bb9a609b840c4dd8d7eb", null]
]
```

Definition at line 1 of file sh1781\_8c.js.

## 30.1637 doc/html/sh1781\_\_at\_8c.js File Reference

### Variables

- var [sh1781\\_\\_at\\_8c](#)

#### 30.1637.1 Variable Documentation

##### 30.1637.1.1 var sh1781\_\_at\_8c

###### Initial value:

```
=
[
 ["SH1781_AT", "sh1781__at_8c.html#ad30ab2e8ad73184a8e83f9d593916aec", null],
 ["sh1781_at", "sh1781__at_8c.html#a77f75556737c687f76620ee2c8d84e3e", null]
]
```

Definition at line 1 of file sh1781\_\_at\_8c.js.

## 30.1638 doc/html/sh1782\_8c.js File Reference

### Variables

- var [sh1782\\_8c](#)

### 30.1638.1 Variable Documentation

#### 30.1638.1.1 var sh1782\_8c

**Initial value:**

```
=
[
 ["SH1782", "sh1782_8c.html#ac555cd187c66475a5f4481b5f80ad308", null],
 ["sh1782", "sh1782_8c.html#a30aa2182ab1bb55905aa267dbf81f696", null]
]
```

Definition at line 1 of file sh1782\_8c.js.

### 30.1639 doc/html/sh1783\_8c.js File Reference

**Variables**

- var [sh1783\\_8c](#)

#### 30.1639.1 Variable Documentation

##### 30.1639.1.1 var sh1783\_8c

**Initial value:**

```
=
[
 ["SH1783", "sh1783_8c.html#ab7efd95528b4e99c58fa6ce3b8f96eb2", null],
 ["fevalProc", "sh1783_8c.html#a4a4af84d2cfe37d7fbd54664e8ed45f5", null],
 ["sh1783", "sh1783_8c.html#ae45da0720fb215c8c012bc7e0977163d", null]
]
```

Definition at line 1 of file sh1783\_8c.js.

### 30.1640 doc/html/sh1784\_8c.js File Reference

**Variables**

- var [sh1784\\_8c](#)

#### 30.1640.1 Variable Documentation

##### 30.1640.1.1 var sh1784\_8c

**Initial value:**

```
=
[
 ["SH1784", "sh1784_8c.html#a72bbccf449f305fb42ce095de29267df", null],
 ["fevalcProc", "sh1784_8c.html#a6f5d68d6da132de46dd79668ba91bb37", null],
 ["sh1784", "sh1784_8c.html#ac7302e661a4b5849fadfdc011c4b15eb", null]
]
```

Definition at line 1 of file sh1784\_8c.js.

## 30.1641 doc/html/sh1786\_8c.js File Reference

### Variables

- var [sh1786\\_8c](#)

#### 30.1641.1 Variable Documentation

##### 30.1641.1.1 var sh1786\_8c

###### Initial value:

```
=
[
 ["SH1786", "sh1786_8c.html#a4e6cc8c34fceb6ccb3ca8acd3efb4751", null],
 ["sh1786", "sh1786_8c.html#a0ae84082ea71cf254404f77cfee1127f", null]
]
```

Definition at line 1 of file sh1786\_8c.js.

## 30.1642 doc/html/sh1787\_8c.js File Reference

### Variables

- var [sh1787\\_8c](#)

#### 30.1642.1 Variable Documentation

##### 30.1642.1.1 var sh1787\_8c

###### Initial value:

```
=
[
 ["SH1787", "sh1787_8c.html#a343d9842ed1789fb2ff06fbbdbb9fdb9", null],
 ["sh1787", "sh1787_8c.html#a28c411b099be86c8309b28ccca60b2e7", null]
]
```

Definition at line 1 of file sh1787\_8c.js.

## 30.1643 doc/html/sh1790\_8c.js File Reference

### Variables

- var [sh1790\\_8c](#)

### 30.1643.1 Variable Documentation

#### 30.1643.1.1 var sh1790\_8c

**Initial value:**

```
=
[
 ["SH1790", "sh1790_8c.html#a49d05ff9e45e3342e1ddb5fd453cf4d", null],
 ["sh1790", "sh1790_8c.html#ad96d9ffc6d64fedb2906c630081dfd7c", null]
]
```

Definition at line 1 of file sh1790\_8c.js.

### 30.1644 doc/html/sh1830\_8c.js File Reference

**Variables**

- var [sh1830\\_8c](#)

#### 30.1644.1 Variable Documentation

##### 30.1644.1.1 var sh1830\_8c

**Initial value:**

```
=
[
 ["SH1830", "sh1830_8c.html#a716fe99613d568ebf73b617b59a5e154", null],
 ["sh1830", "sh1830_8c.html#aa8fee20fb84faf51b8637a3d77062070", null]
]
```

Definition at line 1 of file sh1830\_8c.js.

### 30.1645 doc/html/sh1831\_8c.js File Reference

**Variables**

- var [sh1831\\_8c](#)

#### 30.1645.1 Variable Documentation

##### 30.1645.1.1 var sh1831\_8c

**Initial value:**

```
=
[
 ["SH1831", "sh1831_8c.html#aa0251ca46661459ecbb46b85e6736786", null],
 ["sh1831", "sh1831_8c.html#a4c451867aec5246e1a2db31f018c3e82", null]
]
```

Definition at line 1 of file sh1831\_8c.js.

## 30.1646 doc/html/sh1834\_8c.js File Reference

### Variables

- var [sh1834\\_8c](#)

#### 30.1646.1 Variable Documentation

##### 30.1646.1.1 var sh1834\_8c

###### Initial value:

```
=
[
 ["SH1834", "sh1834_8c.html#a9996ad38a319ccde558e99f4aec5301f", null],
 ["sh1834", "sh1834_8c.html#a81c1699cd3f0178b5a743b5b7867fac2", null]
]
```

Definition at line 1 of file sh1834\_8c.js.

## 30.1647 doc/html/sh1839\_8c.js File Reference

### Variables

- var [sh1839\\_8c](#)

#### 30.1647.1 Variable Documentation

##### 30.1647.1.1 var sh1839\_8c

###### Initial value:

```
=
[
 ["SH1839", "sh1839_8c.html#a36cc0c25253eeb609bcfc94b16a7e8c8", null],
 ["sh1839", "sh1839_8c.html#ad7d4cfccf8bafaf60d82e7179aa88cbf", null]
]
```

Definition at line 1 of file sh1839\_8c.js.

## 30.1648 doc/html/sh1850\_8c.js File Reference

### Variables

- var [sh1850\\_8c](#)

### 30.1648.1 Variable Documentation

#### 30.1648.1.1 var sh1850\_8c

**Initial value:**

```
=
[
 ["SH1850", "sh1850_8c.html#a7519f66203d5f4b10f67aa52dac07493", null],
 ["sh1850", "sh1850_8c.html#ae9189f0e1d8afa324dbe96a53976c23e", null]
]
```

Definition at line 1 of file sh1850\_8c.js.

### 30.1649 doc/html/sh1851\_8c.js File Reference

**Variables**

- var [sh1851\\_8c](#)

#### 30.1649.1 Variable Documentation

##### 30.1649.1.1 var sh1851\_8c

**Initial value:**

```
=
[
 ["SH1851", "sh1851_8c.html#a0bf9334d9decdbbcbb3a9bcfe3bf1ff8", null],
 ["sh1851", "sh1851_8c.html#a8c8eac78eedb5a0eac9ae1b79afb98e0", null]
]
```

Definition at line 1 of file sh1851\_8c.js.

### 30.1650 doc/html/sh1852\_8c.js File Reference

**Variables**

- var [sh1852\\_8c](#)

#### 30.1650.1 Variable Documentation

##### 30.1650.1.1 var sh1852\_8c

**Initial value:**

```
=
[
 ["SH1852", "sh1852_8c.html#ac9383810836e865b354eb1f0b44b186a", null],
 ["sh1852", "sh1852_8c.html#a2c4a688b84b978cbeb7d429bba3ba128", null]
]
```

Definition at line 1 of file sh1852\_8c.js.

## 30.1651 doc/html/sh1853\_8c.js File Reference

### Variables

- var [sh1853\\_8c](#)

#### 30.1651.1 Variable Documentation

##### 30.1651.1.1 var sh1853\_8c

###### Initial value:

```
=
[
 ["SH1853", "sh1853_8c.html#ac8526ca756e8fde0f8a10d0fd16fefef1", null],
 ["sh1853", "sh1853_8c.html#af8236e75afcf2144054e46838adb0160", null]
]
```

Definition at line 1 of file sh1853\_8c.js.

## 30.1652 doc/html/sh1854\_8c.js File Reference

### Variables

- var [sh1854\\_8c](#)

#### 30.1652.1 Variable Documentation

##### 30.1652.1.1 var sh1854\_8c

###### Initial value:

```
=
[
 ["SH1854", "sh1854_8c.html#aa01b6886ebcfaf0dd554058fa8d97b2a", null],
 ["sh1854", "sh1854_8c.html#a631efd14d23a5cf5ba71a9ce25f88774", null]
]
```

Definition at line 1 of file sh1854\_8c.js.

## 30.1653 doc/html/sh1855\_8c.js File Reference

### Variables

- var [sh1855\\_8c](#)

### 30.1653.1 Variable Documentation

#### 30.1653.1.1 var sh1855\_8c

**Initial value:**

```
=
[
 ["SH1855", "sh1855_8c.html#a80cd0fa6142cc1da5860271348591d3f", null],
 ["sh1855", "sh1855_8c.html#ac2d90ba6b91002c3664e947b70ea7eea", null]
]
```

Definition at line 1 of file sh1855\_8c.js.

### 30.1654 doc/html/sh1856\_8c.js File Reference

**Variables**

- var [sh1856\\_8c](#)

#### 30.1654.1 Variable Documentation

##### 30.1654.1.1 var sh1856\_8c

**Initial value:**

```
=
[
 ["SH1856", "sh1856_8c.html#a6dbd5266deb9a6140646d51ef21cf193", null],
 ["sh1856", "sh1856_8c.html#a8cce63cd9ca4218da6c6b20cea437758", null]
]
```

Definition at line 1 of file sh1856\_8c.js.

### 30.1655 doc/html/sh1857\_8c.js File Reference

**Variables**

- var [sh1857\\_8c](#)

#### 30.1655.1 Variable Documentation

##### 30.1655.1.1 var sh1857\_8c

**Initial value:**

```
=
[
 ["SH1857", "sh1857_8c.html#aa75942263b3eaf58a8b10dba16c0db03", null],
 ["sh1857", "sh1857_8c.html#a3fe3901bc3a92cble48f7d0ed94a9dd2", null]
]
```

Definition at line 1 of file sh1857\_8c.js.



## 30.1656 doc/html/sh1858\_8c.js File Reference

### Variables

- var [sh1858\\_8c](#)

#### 30.1656.1 Variable Documentation

##### 30.1656.1.1 var sh1858\_8c

###### Initial value:

```
=
[
 ["SH1858", "sh1858_8c.html#ac0e1fb7c55652142418fe285ab73c06e", null],
 ["sh1858", "sh1858_8c.html#a8c7a713b3333ceaf5da5e1f310dcef67", null]
]
```

Definition at line 1 of file sh1858\_8c.js.

## 30.1657 doc/html/sh1859\_8c.js File Reference

### Variables

- var [sh1859\\_8c](#)

#### 30.1657.1 Variable Documentation

##### 30.1657.1.1 var sh1859\_8c

###### Initial value:

```
=
[
 ["SH1859", "sh1859_8c.html#abce061fa2e9913a10704e881d7199cf3", null],
 ["sh1859", "sh1859_8c.html#a961fce72214749443c5178dc8875b013", null]
]
```

Definition at line 1 of file sh1859\_8c.js.

## 30.1658 doc/html/sh1860\_8c.js File Reference

### Variables

- var [sh1860\\_8c](#)

### 30.1658.1 Variable Documentation

30.1658.1.1 `var sh1860_8c`

**Initial value:**

```
=
[
 ["SH1860", "sh1860_8c.html#aaa09af9fee31161cd844037f78eaeafa6", null],
 ["sh1860", "sh1860_8c.html#a92ac57b7209272badddc40e0a6deb49e6", null]
]
```

Definition at line 1 of file sh1860\_8c.js.

### 30.1659 doc/html/sh1870\_8c.js File Reference

#### Variables

- `var sh1870_8c`

### 30.1659.1 Variable Documentation

30.1659.1.1 `var sh1870_8c`

**Initial value:**

```
=
[
 ["SH1870", "sh1870_8c.html#a5bd1095e55392ddc1ea0c8187e0ec831", null],
 ["sh1870", "sh1870_8c.html#a3433bc26e5f23f954f11c698f4da003c", null]
]
```

Definition at line 1 of file sh1870\_8c.js.

### 30.1660 doc/html/sh1871\_8c.js File Reference

#### Variables

- `var sh1871_8c`

### 30.1660.1 Variable Documentation

30.1660.1.1 `var sh1871_8c`

**Initial value:**

```
=
[
 ["SH1871", "sh1871_8c.html#a7385e94b25613c47ebca2c709360b208", null],
 ["sh1871", "sh1871_8c.html#a0fe08167ea693b69601f5f7aac8586d0", null]
]
```

Definition at line 1 of file sh1871\_8c.js.

## 30.1661 doc/html/sh1922\_8c.js File Reference

### Variables

- var [sh1922\\_8c](#)

#### 30.1661.1 Variable Documentation

##### 30.1661.1.1 var sh1922\_8c

###### Initial value:

```
=
[
 ["SH1922", "sh1922_8c.html#af74228ff227487ab117f29f718e43c05", null],
 ["sh1922", "sh1922_8c.html#a99193e2bc2d2f7cbbb97af0c6895fbf8", null]
]
```

Definition at line 1 of file sh1922\_8c.js.

## 30.1662 doc/html/sh1923\_8c.js File Reference

### Variables

- var [sh1923\\_8c](#)

#### 30.1662.1 Variable Documentation

##### 30.1662.1.1 var sh1923\_8c

###### Initial value:

```
=
[
 ["SH1923", "sh1923_8c.html#af12c33e7c62867ce6a830eed2daffd1b", null],
 ["sh1923", "sh1923_8c.html#a8b53e796129b2bcf8d7cf5a93886afa2", null]
]
```

Definition at line 1 of file sh1923\_8c.js.

## 30.1663 doc/html/sh1924\_8c.js File Reference

### Variables

- var [sh1924\\_8c](#)

### 30.1663.1 Variable Documentation

#### 30.1663.1.1 var sh1924\_8c

**Initial value:**

```
=
[
 ["SH1924", "sh1924_8c.html#af94b3812baf8311584fe9c785e6c9b3e", null],
 ["sh1924", "sh1924_8c.html#acce5409e230e16f6cea80f2bb1cea286", null]
]
```

Definition at line 1 of file sh1924\_8c.js.

### 30.1664 doc/html/sh1925\_8c.js File Reference

**Variables**

- var [sh1925\\_8c](#)

#### 30.1664.1 Variable Documentation

##### 30.1664.1.1 var sh1925\_8c

**Initial value:**

```
=
[
 ["SH1925", "sh1925_8c.html#a312ba7342834a317edc97d59a9a0fd69", null],
 ["sh1925", "sh1925_8c.html#a154cc85e5c541c4a18c0cf2367399f6a", null]
]
```

Definition at line 1 of file sh1925\_8c.js.

### 30.1665 doc/html/sh1926\_8c.js File Reference

**Variables**

- var [sh1926\\_8c](#)

#### 30.1665.1 Variable Documentation

##### 30.1665.1.1 var sh1926\_8c

**Initial value:**

```
=
[
 ["SH1926", "sh1926_8c.html#a16a0d903aaf024ef1e0911d714782c52", null],
 ["sh1926", "sh1926_8c.html#a6dc81c77048256e9bb951cd2a71df8bb", null]
]
```

Definition at line 1 of file sh1926\_8c.js.

## 30.1666 doc/html/sh1927\_8c.js File Reference

### Variables

- var [sh1927\\_8c](#)

#### 30.1666.1 Variable Documentation

##### 30.1666.1.1 var sh1927\_8c

###### Initial value:

```
=
[
 ["SH1927", "sh1927_8c.html#ae87559c8f9b5dad2046eedd1808132ef", null],
 ["sh1927", "sh1927_8c.html#af3c3a1d60c7a62d775e5ce219e9a4788", null]
]
```

Definition at line 1 of file sh1927\_8c.js.

## 30.1667 doc/html/sh1928\_8c.js File Reference

### Variables

- var [sh1928\\_8c](#)

#### 30.1667.1 Variable Documentation

##### 30.1667.1.1 var sh1928\_8c

###### Initial value:

```
=
[
 ["SH1928", "sh1928_8c.html#a626eeb2d321e7388b527867f3f0a2920", null],
 ["sh1928", "sh1928_8c.html#ab4b70ecf3f02797353206a27012ddf78", null]
]
```

Definition at line 1 of file sh1928\_8c.js.

## 30.1668 doc/html/sh1929\_8c.js File Reference

### Variables

- var [sh1929\\_8c](#)

### 30.1668.1 Variable Documentation

#### 30.1668.1.1 var sh1929\_8c

**Initial value:**

```
=
[
 ["SH1929", "sh1929_8c.html#af72b31ab3405cc474e3b99b60f96bf31", null],
 ["sh1929", "sh1929_8c.html#a6ea3f3d035e7d8c6068cbb2625476f4e", null]
]
```

Definition at line 1 of file sh1929\_8c.js.

## 30.1669 doc/html/sh1930\_8c.js File Reference

**Variables**

- var [sh1930\\_8c](#)

### 30.1669.1 Variable Documentation

#### 30.1669.1.1 var sh1930\_8c

**Initial value:**

```
=
[
 ["SH1930", "sh1930_8c.html#add3f30fdee1d4bb8ee6556070d9cea1b", null],
 ["sh1930", "sh1930_8c.html#a9be863d38cdb94a1ab1425e564416695", null]
]
```

Definition at line 1 of file sh1930\_8c.js.

## 30.1670 doc/html/sh1992\_8c.js File Reference

**Variables**

- var [sh1992\\_8c](#)

### 30.1670.1 Variable Documentation

#### 30.1670.1.1 var sh1992\_8c

**Initial value:**

```
=
[
 ["SH1992", "sh1992_8c.html#af7a78b760687df7844111762cdb9ed92", null],
 ["sh1992", "sh1992_8c.html#a0d0544d9e8d03bb70a08751df2961c0d", null],
 ["sh1992cu", "sh1992_8c.html#ab752ae0d476864e79e9df62dced89af4", null],
 ["sh1992su", "sh1992_8c.html#aae575a79958c0f2be983ec3bd1133dcc", null]
]
```

Definition at line 1 of file sh1992\_8c.js.

## 30.1671 doc/html/sh1993\_8c.js File Reference

### Variables

- var [sh1993\\_8c](#)

### 30.1671.1 Variable Documentation

#### 30.1671.1.1 var sh1993\_8c

##### Initial value:

```
=
[
 ["SH1993", "sh1993_8c.html#a6dd20704053d2b2c9c9b0d0d5eb17efc", null],
 ["sh1993", "sh1993_8c.html#ad63d49b9a1e9a553de1b944bc470f9a5", null]
]
```

Definition at line 1 of file sh1993\_8c.js.

## 30.1672 doc/html/sh1994\_8c.js File Reference

### Variables

- var [sh1994\\_8c](#)

### 30.1672.1 Variable Documentation

#### 30.1672.1.1 var sh1994\_8c

##### Initial value:

```
=
[
 ["SH1994", "sh1994_8c.html#ae9c8bbc735555df7a77da4b64b674dbb", null],
 ["sh1994", "sh1994_8c.html#a30cd0efe00663a3f1297dc075c75e6bd", null]
]
```

Definition at line 1 of file sh1994\_8c.js.

## 30.1673 doc/html/sh6clvert\_8c.js File Reference

### Variables

- var [sh6clvert\\_8c](#)

### 30.1673.1 Variable Documentation

#### 30.1673.1.1 var sh6clvert\_8c

**Initial value:**

```
=
[
 ["SH6CLOSEVERT", "sh6clvert_8c.html#a5227e99e2de698f09e9acd2ed98ae6ff", null],
 ["sh6closevert", "sh6clvert_8c.html#ae0a1c74f12691152583d06755013dic0", null]
]
```

Definition at line 1 of file sh6clvert\_8c.js.

### 30.1674 doc/html/sh6comedg\_8c.js File Reference

**Variables**

- var [sh6comedg\\_8c](#)

#### 30.1674.1 Variable Documentation

##### 30.1674.1.1 var sh6comedg\_8c

**Initial value:**

```
=
[
 ["SH6COMEDG", "sh6comedg_8c.html#ad241a689937846a48631d61ed46d7fe0", null],
 ["sh6comedg", "sh6comedg_8c.html#a52c7e299d3f419ecad93806da7eea4e8", null]
]
```

Definition at line 1 of file sh6comedg\_8c.js.

### 30.1675 doc/html/sh6connect\_8c.js File Reference

**Variables**

- var [sh6connect\\_8c](#)

#### 30.1675.1 Variable Documentation

##### 30.1675.1.1 var sh6connect\_8c

**Initial value:**

```
=
[
 ["SH6CONNECT", "sh6connect_8c.html#ace90b1a6fae52ce173ae85a20ddb3538", null],
 ["sh6connect", "sh6connect_8c.html#afef2f709a2884e4f4b3bf4ea8e38f14b", null]
]
```

Definition at line 1 of file sh6connect\_8c.js.



## 30.1676 doc/html/sh6count\_8c.js File Reference

### Variables

- var [sh6count\\_8c](#)

#### 30.1676.1 Variable Documentation

##### 30.1676.1.1 var sh6count\_8c

###### Initial value:

```
=
[
 ["SH6COUNT", "sh6count_8c.html#ab2a46a5d2645dd540fbb96a0b0411797", null],
 ["sh6count", "sh6count_8c.html#a98bf7925021a09295ae7eadf00b59c0c", null]
]
```

Definition at line 1 of file sh6count\_8c.js.

## 30.1677 doc/html/sh6cvvert\_8c.js File Reference

### Variables

- var [sh6cvvert\\_8c](#)

#### 30.1677.1 Variable Documentation

##### 30.1677.1.1 var sh6cvvert\_8c

###### Initial value:

```
=
[
 ["SH6CVVERT", "sh6cvvert_8c.html#a92816bd70078361ef5f19a407af0e409", null],
 ["sh6cvvert", "sh6cvvert_8c.html#ab54d30b3f595bb7924867fd11ac10f83", null]
]
```

Definition at line 1 of file sh6cvvert\_8c.js.

## 30.1678 doc/html/sh6degen\_8c.js File Reference

### Variables

- var [sh6degen\\_8c](#)

### 30.1678.1 Variable Documentation

#### 30.1678.1.1 var sh6degen\_8c

**Initial value:**

```
=
[
 ["SH6DEGEN", "sh6degen_8c.html#af06233628f26ea80c2f3ca3c5ee31a34", null],
 ["sh6degen", "sh6degen_8c.html#ac0ab97e7a25be84d0f417e0038bbfad", null]
]
```

Definition at line 1 of file sh6degen\_8c.js.

### 30.1679 doc/html/sh6disconn\_8c.js File Reference

**Variables**

- var [sh6disconn\\_8c](#)

#### 30.1679.1 Variable Documentation

##### 30.1679.1.1 var sh6disconn\_8c

**Initial value:**

```
=
[
 ["SH6DISCONNECT", "sh6disconn_8c.html#a6822c427e0681b12b461ffdb1125c936", null],
 ["sh6disconnect", "sh6disconn_8c.html#a4280f5995d0545e323364876d341f107", null]
]
```

Definition at line 1 of file sh6disconn\_8c.js.

### 30.1680 doc/html/sh6edgpnt\_8c.js File Reference

**Variables**

- var [sh6edgpnt\\_8c](#)

#### 30.1680.1 Variable Documentation

##### 30.1680.1.1 var sh6edgpnt\_8c

**Initial value:**

```
=
[
 ["SH6EDGPOINT", "sh6edgpnt_8c.html#ac0616b535e8c07fa6b52b8221e1dfcbd", null],
 ["sh6edgpnt", "sh6edgpnt_8c.html#a063f6a5ac7607baed46419da60462e20", null]
]
```

Definition at line 1 of file sh6edgpnt\_8c.js.

## 30.1681 doc/html/sh6edgred\_8c.js File Reference

### Variables

- var [sh6edgred\\_8c](#)

### 30.1681.1 Variable Documentation

#### 30.1681.1.1 var sh6edgred\_8c

##### Initial value:

```
=
[
 ["SH6EDGRED", "sh6edgred_8c.html#a6d94485f308672b6b1e84ad4dd9c8d0f", null],
 ["sh6edgred", "sh6edgred_8c.html#a5784faf01a49282b79f84246473b4977", null]
]
```

Definition at line 1 of file sh6edgred\_8c.js.

## 30.1682 doc/html/sh6evalint\_8c.js File Reference

### Variables

- var [sh6evalint\\_8c](#)

### 30.1682.1 Variable Documentation

#### 30.1682.1.1 var sh6evalint\_8c

##### Initial value:

```
=
[
 ["SH6EVALINT", "sh6evalint_8c.html#a22566f7a761d6b2b14061fc6ed897d80", null],
 ["sh6evalint", "sh6evalint_8c.html#a2e82ea60c713dc49fdec64f870b96ff9", null]
]
```

Definition at line 1 of file sh6evalint\_8c.js.

## 30.1683 doc/html/sh6floop\_8c.js File Reference

### Variables

- var [sh6floop\\_8c](#)

### 30.1683.1 Variable Documentation

#### 30.1683.1.1 var sh6floop\_8c

**Initial value:**

```
=
[
 ["SH6FLOOP", "sh6floop_8c.html#a175aa1040ebef5216a87c37658eb05f0", null],
 ["sh6floop", "sh6floop_8c.html#a169dda0490387ca78847f0ca5617cb8e", null]
]
```

Definition at line 1 of file sh6floop\_8c.js.

## 30.1684 doc/html/sh6fndsplt\_8c.js File Reference

### Variables

- var [sh6fndsplt\\_8c](#)

### 30.1684.1 Variable Documentation

#### 30.1684.1.1 var sh6fndsplt\_8c

**Initial value:**

```
=
[
 ["SH6FINDSPLIT", "sh6fndsplt_8c.html#a150d3c90bfb69e79b6346d239c30ef33", null],
 ["sh6fndsplt", "sh6fndsplt_8c.html#ade8683fc91ba82736e1e595177aa58e5", null]
]
```

Definition at line 1 of file sh6fndsplt\_8c.js.

## 30.1685 doc/html/sh6getgeom\_8c.js File Reference

### Variables

- var [sh6getgeom\\_8c](#)

### 30.1685.1 Variable Documentation

#### 30.1685.1.1 var sh6getgeom\_8c

**Initial value:**

```
=
[
 ["SH6GETGEOM", "sh6getgeom_8c.html#a37bee6dfbabecbf0c7c198177678be1d", null],
 ["sh6getgeom", "sh6getgeom_8c.html#af28bfbbd6a4afcd7d8c8f082c83517e3", null]
]
```

Definition at line 1 of file sh6getgeom\_8c.js.

## 30.1686 doc/html/sh6getlist\_8c.js File Reference

### Variables

- var [sh6getlist\\_8c](#)

### 30.1686.1 Variable Documentation

#### 30.1686.1.1 var sh6getlist\_8c

##### Initial value:

```
=
[
 ["SH6GETLIST", "sh6getlist_8c.html#aa4c1fe9f2baf29a43e3e15ded921b8ce", null],
 ["sh6getlist", "sh6getlist_8c.html#a701601b3179395dbef1fba6a1174b4b9", null]
]
```

Definition at line 1 of file sh6getlist\_8c.js.

## 30.1687 doc/html/sh6getmain\_8c.js File Reference

### Variables

- var [sh6getmain\\_8c](#)

### 30.1687.1 Variable Documentation

#### 30.1687.1.1 var sh6getmain\_8c

##### Initial value:

```
=
[
 ["SH6GETMAIN", "sh6getmain_8c.html#aab361fcb31f359af7765faad37b1bd16", null],
 ["sh6getmain", "sh6getmain_8c.html#a1088de9c49ea57dac9a91e60144e9653", null]
]
```

Definition at line 1 of file sh6getmain\_8c.js.

## 30.1688 doc/html/sh6getnbrs\_8c.js File Reference

### Variables

- var [sh6getnbrs\\_8c](#)

### 30.1688.1 Variable Documentation

#### 30.1688.1.1 var sh6getnbrs\_8c

**Initial value:**

```
=
[
 ["SH6GETNHBRS", "sh6getnbrs_8c.html#abf7483ae7a4cee5a62cf35ce07bfc0c", null],
 ["sh6getnhbrs", "sh6getnbrs_8c.html#a10b2ee815b92a90badd1046a4dd502c1", null]
]
```

Definition at line 1 of file sh6getnbrs\_8c.js.

### 30.1689 doc/html/sh6getnext\_8c.js File Reference

**Variables**

- var [sh6getnext\\_8c](#)

#### 30.1689.1 Variable Documentation

##### 30.1689.1.1 var sh6getnext\_8c

**Initial value:**

```
=
[
 ["SH6GETNEXT", "sh6getnext_8c.html#a929dbdfcf2eaae318872b1e61de0fb47", null],
 ["sh6getnext", "sh6getnext_8c.html#a560b19880f6f11e8da518646a1cab30c", null]
]
```

Definition at line 1 of file sh6getnext\_8c.js.

### 30.1690 doc/html/sh6getothr\_8c.js File Reference

**Variables**

- var [sh6getothr\\_8c](#)

#### 30.1690.1 Variable Documentation

##### 30.1690.1.1 var sh6getothr\_8c

**Initial value:**

```
=
[
 ["SH6GETOTHER", "sh6getothr_8c.html#a0342d42ed3aa44625153f965e7dde713", null],
 ["sh6getother", "sh6getothr_8c.html#a3a2d2afb4562ba2486792649b715b521", null]
]
```

Definition at line 1 of file sh6getothr\_8c.js.

## 30.1691 doc/html/sh6getprev\_8c.js File Reference

### Variables

- var [sh6getprev\\_8c](#)

#### 30.1691.1 Variable Documentation

##### 30.1691.1.1 var sh6getprev\_8c

###### Initial value:

```
=
[
 ["SH6GETPREV", "sh6getprev_8c.html#a53218e5a6b559ald1ce235baa7b254c6", null],
 ["sh6getprev", "sh6getprev_8c.html#a4d238313c7573a17b9e841bc0adb2090", null]
]
```

Definition at line 1 of file sh6getprev\_8c.js.

## 30.1692 doc/html/sh6gettop\_8c.js File Reference

### Variables

- var [sh6gettop\\_8c](#)

#### 30.1692.1 Variable Documentation

##### 30.1692.1.1 var sh6gettop\_8c

###### Initial value:

```
=
[
 ["SH6GETTOP", "sh6gettop_8c.html#a7f39191e5b2203c31e5a6ffb7e364021", null],
 ["sh6gettop", "sh6gettop_8c.html#aa2508df0a2eff7c6d8ceaa6a52bd744d", null]
]
```

Definition at line 1 of file sh6gettop\_8c.js.

## 30.1693 doc/html/sh6idaledg\_8c.js File Reference

### Variables

- var [sh6idaledg\\_8c](#)

### 30.1693.1 Variable Documentation

#### 30.1693.1.1 var sh6idaledg\_8c

**Initial value:**

```
=
[
 ["SH6ALLEDG", "sh6idaledg_8c.html#acf94af63060e9567e4d49db89db7b13b", null],
 ["sh6idalledg", "sh6idaledg_8c.html#a6610136782c8defe16dae71c5231409f", null]
]
```

Definition at line 1 of file sh6idaledg\_8c.js.

### 30.1694 doc/html/sh6idcon\_8c.js File Reference

**Variables**

- var [sh6idcon\\_8c](#)

#### 30.1694.1 Variable Documentation

##### 30.1694.1.1 var sh6idcon\_8c

**Initial value:**

```
=
[
 ["SH6IDCON", "sh6idcon_8c.html#a5fbebfc7b402cb6142bc5ea95368b8bb", null],
 ["sh6idcon", "sh6idcon_8c.html#a4b027920c19388f0ac0799f844bfa0ef", null]
]
```

Definition at line 1 of file sh6idcon\_8c.js.

### 30.1695 doc/html/sh6idfcros\_8c.js File Reference

**Variables**

- var [sh6idfcros\\_8c](#)

#### 30.1695.1 Variable Documentation

##### 30.1695.1.1 var sh6idfcros\_8c

**Initial value:**

```
=
[
 ["SH6IDFCROSS", "sh6idfcros_8c.html#a0f29889c1eb0bf509830f00dc12a3452", null],
 ["sh6idfcross", "sh6idfcros_8c.html#a5ec1be74fc2e952f177cbc8f863474d5", null]
]
```

Definition at line 1 of file sh6idfcros\_8c.js.



## 30.1696 doc/html/sh6idget\_8c.js File Reference

### Variables

- var [sh6idget\\_8c](#)

#### 30.1696.1 Variable Documentation

##### 30.1696.1.1 var sh6idget\_8c

###### Initial value:

```
=
[
 ["SH6IDGET", "sh6idget_8c.html#a1a448888fd638900a23f6fc89795b3a0", null],
 ["sh6idget", "sh6idget_8c.html#aa89a6ed7aafad9a960377d2d5c0407a0", null]
]
```

Definition at line 1 of file sh6idget\_8c.js.

## 30.1697 doc/html/sh6idkpt\_8c.js File Reference

### Variables

- var [sh6idkpt\\_8c](#)

#### 30.1697.1 Variable Documentation

##### 30.1697.1.1 var sh6idkpt\_8c

###### Initial value:

```
=
[
 ["S6IDKPT", "sh6idkpt_8c.html#a9cdb66d9ecc5c082ad0c17786c154aa6", null],
 ["sh6idkpt", "sh6idkpt_8c.html#a678f398e2200778c38fe506190e1bbfd", null]
]
```

Definition at line 1 of file sh6idkpt\_8c.js.

## 30.1698 doc/html/sh6idlis\_8c.js File Reference

### Variables

- var [sh6idlis\\_8c](#)

## 30.1698.1 Variable Documentation

### 30.1698.1.1 var sh6idlis\_8c

#### Initial value:

```
=
[
 ["SH6IDLIS", "sh6idlis_8c.html#a5bd4828e80bc86a29dc330f3160f9fea", null],
 ["sh6idlis", "sh6idlis_8c.html#a90e02bc8d1f5f19acdfa5d7389bec9d7", null]
]
```

Definition at line 1 of file sh6idlis\_8c.js.

## 30.1699 doc/html/sh6idnpt\_8c.js File Reference

### Variables

- var [sh6idnpt\\_8c](#)

## 30.1699.1 Variable Documentation

### 30.1699.1.1 var sh6idnpt\_8c

#### Initial value:

```
=
[
 ["SH6IDNPT", "sh6idnpt_8c.html#aafffb8ea92f09285023ecd4fb4c3ed136", null],
 ["sh6idnpt", "sh6idnpt_8c.html#a251e74969631ebd5ce4744d5fb349b91", null]
]
```

Definition at line 1 of file sh6idnpt\_8c.js.

## 30.1700 doc/html/sh6idnwun\_8c.js File Reference

### Variables

- var [sh6idnwun\\_8c](#)

## 30.1700.1 Variable Documentation

### 30.1700.1.1 var sh6idnwun\_8c

#### Initial value:

```
=
[
 ["SH6IDNEWUNITE", "sh6idnwun_8c.html#ab94658f22c740aa76cef51baed32a1df", null],
 ["sh6idnewunite", "sh6idnwun_8c.html#aad62920d09ed3a7c9be024a685dff0a5", null]
]
```

Definition at line 1 of file sh6idnwun\_8c.js.

## 30.1701 doc/html/sh6idput\_8c.js File Reference

### Variables

- var [sh6idput\\_8c](#)

#### 30.1701.1 Variable Documentation

##### 30.1701.1.1 var sh6idput\_8c

###### Initial value:

```
=
[
 ["SH6IDPUT", "sh6idput_8c.html#ae4cb4da36de5c8e3250525773587fef3", null],
 ["sh6idput", "sh6idput_8c.html#a2eaa0322fe6925265b6c965e7792e40c", null]
]
```

Definition at line 1 of file sh6idput\_8c.js.

## 30.1702 doc/html/sh6idrcros\_8c.js File Reference

### Variables

- var [sh6idrcros\\_8c](#)

#### 30.1702.1 Variable Documentation

##### 30.1702.1.1 var sh6idrcros\_8c

###### Initial value:

```
=
[
 ["SH6IDRMCROSS", "sh6idrcros_8c.html#a505181fcb578f3c8dad95445c56293b9", null],
 ["sh6idrmercross", "sh6idrcros_8c.html#a1e59d65f0d73bbc7abc5baad47c0b3cd", null]
]
```

Definition at line 1 of file sh6idrcros\_8c.js.

## 30.1703 doc/html/sh6idsplit\_8c.js File Reference

### Variables

- var [sh6idsplit\\_8c](#)

### 30.1703.1 Variable Documentation

#### 30.1703.1.1 var sh6idsplit\_8c

**Initial value:**

```
=
[
 ["S6IDSPLIT", "sh6idsplit_8c.html#a089b8657d965865812fd7f9f31840eb0", null],
 ["sh6idsplit", "sh6idsplit_8c.html#a876f109f74a62a01383807cf5d168481", null]
]
```

Definition at line 1 of file sh6idsplit\_8c.js.

## 30.1704 doc/html/sh6idunite\_8c.js File Reference

### Variables

- var [sh6idunite\\_8c](#)

### 30.1704.1 Variable Documentation

#### 30.1704.1.1 var sh6idunite\_8c

**Initial value:**

```
=
[
 ["SH6IDUNITE", "sh6idunite_8c.html#aeb00fecb2ecd4af82bb8a391ab301556", null],
 ["sh6idunite", "sh6idunite_8c.html#a407c30e3d3d4603bd1496501bd423d0d", null]
]
```

Definition at line 1 of file sh6idunite\_8c.js.

## 30.1705 doc/html/sh6insert\_8c.js File Reference

### Variables

- var [sh6insert\\_8c](#)

### 30.1705.1 Variable Documentation

#### 30.1705.1.1 var sh6insert\_8c

**Initial value:**

```
=
[
 ["SH6INSERT", "sh6insert_8c.html#a3d952f2286e68e4a84337019217bcb7e", null],
 ["sh6insert", "sh6insert_8c.html#a87a721d19918aa75a2ea25dda10f6bc3", null]
]
```

Definition at line 1 of file sh6insert\_8c.js.

## 30.1706 doc/html/sh6inspnt\_8c.js File Reference

### Variables

- var [sh6inspnt\\_8c](#)

#### 30.1706.1 Variable Documentation

##### 30.1706.1.1 var sh6inspnt\_8c

###### Initial value:

```
=
[
 ["SH6INSERTPT", "sh6inspnt_8c.html#a2d6b7e67bdab7a0a4095601dfe75f4bd", null],
 ["sh6insertpt", "sh6inspnt_8c.html#ab65110f3152221cecd83680be891d96d", null]
]
```

Definition at line 1 of file sh6inspnt\_8c.js.

## 30.1707 doc/html/sh6iscnect\_8c.js File Reference

### Variables

- var [sh6iscnect\\_8c](#)

#### 30.1707.1 Variable Documentation

##### 30.1707.1.1 var sh6iscnect\_8c

###### Initial value:

```
=
[
 ["SH6ISCONNECT", "sh6iscnect_8c.html#aa08554fd31fb6ee673f8c241c6a4f4e1", null],
 ["sh6isconnect", "sh6iscnect_8c.html#a893c64add08451455091285683a69a3f", null]
]
```

Definition at line 1 of file sh6iscnect\_8c.js.

## 30.1708 doc/html/sh6ishelp\_8c.js File Reference

### Variables

- var [sh6ishelp\\_8c](#)

### 30.1708.1 Variable Documentation

#### 30.1708.1.1 var sh6ishelp\_8c

**Initial value:**

```
=
[
 ["SH6ISHELP", "sh6ishelp_8c.html#a7d84d068ec4d570e1f30886cc3731b7c", null],
 ["sh6ishelp", "sh6ishelp_8c.html#ae13c6d0c9a348ab2c9f2a79b56a7da80", null]
]
```

Definition at line 1 of file sh6ishelp\_8c.js.

### 30.1709 doc/html/sh6isinsid\_8c.js File Reference

**Variables**

- var [sh6isinsid\\_8c](#)

#### 30.1709.1 Variable Documentation

##### 30.1709.1.1 var sh6isinsid\_8c

**Initial value:**

```
=
[
 ["SH6ISINSIDE", "sh6isinsid_8c.html#a5aca7b2e618fe3e238d16ed1f9833cbc", null],
 ["sh6isinside", "sh6isinsid_8c.html#a97a97c24c868e99ce24b72d33ee9182e", null]
]
```

Definition at line 1 of file sh6isinsid\_8c.js.

### 30.1710 doc/html/sh6ismain\_8c.js File Reference

**Variables**

- var [sh6ismain\\_8c](#)

#### 30.1710.1 Variable Documentation

##### 30.1710.1.1 var sh6ismain\_8c

**Initial value:**

```
=
[
 ["SH6ISMAIN", "sh6ismain_8c.html#ab87337baae0ac9a4799691c6a7174640", null],
 ["sh6ismain", "sh6ismain_8c.html#a5c9d73ced459c88d59e78e6fe5298c3c", null]
]
```

Definition at line 1 of file sh6ismain\_8c.js.

## 30.1711 doc/html/sh6nmbhelp\_8c.js File Reference

### Variables

- var [sh6nmbhelp\\_8c](#)

#### 30.1711.1 Variable Documentation

##### 30.1711.1.1 var sh6nmbhelp\_8c

###### Initial value:

```
=
[
 ["SH6NMBHELP", "sh6nmbhelp_8c.html#ad4e3d0fa18290a69d3cecd82a715fffc", null],
 ["sh6nmbhelp", "sh6nmbhelp_8c.html#a42c189b49bbef14f74a0bf39a809652a", null]
]
```

Definition at line 1 of file sh6nmbhelp\_8c.js.

## 30.1712 doc/html/sh6nmbmain\_8c.js File Reference

### Variables

- var [sh6nmbmain\\_8c](#)

#### 30.1712.1 Variable Documentation

##### 30.1712.1.1 var sh6nmbmain\_8c

###### Initial value:

```
=
[
 ["SH6NMBMAIN", "sh6nmbmain_8c.html#a7126024b5ab5517d48337605da3d3e6a", null],
 ["sh6nmbmain", "sh6nmbmain_8c.html#ad1abddf30f2de712bbbd5aa54e6444d9", null]
]
```

Definition at line 1 of file sh6nmbmain\_8c.js.

## 30.1713 doc/html/sh6ptobj\_8c.js File Reference

### Variables

- var [sh6ptobj\\_8c](#)

### 30.1713.1 Variable Documentation

#### 30.1713.1.1 var sh6ptobj\_8c

**Initial value:**

```
=
[
 ["SH6PTOBJ", "sh6ptobj_8c.html#ab62d8f2f795be727b089fd2260e5805d", null],
 ["sh6ptobj", "sh6ptobj_8c.html#a47eda2808eeb05906685d99931a48660", null]
]
```

Definition at line 1 of file sh6ptobj\_8c.js.

### 30.1714 doc/html/sh6ptouchp\_8c.js File Reference

**Variables**

- var [sh6ptouchp\\_8c](#)

#### 30.1714.1 Variable Documentation

##### 30.1714.1.1 var sh6ptouchp\_8c

**Initial value:**

```
=
[
 ["SH6PUTTOUCH", "sh6ptouchp_8c.html#aa92f8bbe693f23b85ee24880538ee02a", null],
 ["sh6puttouch", "sh6ptouchp_8c.html#a6683809dd520ceba30f5090847b8a5cd", null]
]
```

Definition at line 1 of file sh6ptouchp\_8c.js.

### 30.1715 doc/html/sh6putsing\_8c.js File Reference

**Variables**

- var [sh6putsing\\_8c](#)

#### 30.1715.1 Variable Documentation

##### 30.1715.1.1 var sh6putsing\_8c

**Initial value:**

```
=
[
 ["SH6PUTSING", "sh6putsing_8c.html#af86f9ba2a053b56a5dea9e15abf27681", null],
 ["sh6putsing", "sh6putsing_8c.html#a5760688066a9dcba72cd41923f9dc444", null]
]
```

Definition at line 1 of file sh6putsing\_8c.js.



## 30.1716 doc/html/sh6red\_8c.js File Reference

### Variables

- var [sh6red\\_8c](#)

#### 30.1716.1 Variable Documentation

##### 30.1716.1.1 var sh6red\_8c

###### Initial value:

```
=
[
 ["SH6RED", "sh6red_8c.html#a4b51da6c9ddf99d067c6c323ffbdbb5a", null],
 ["sh6red", "sh6red_8c.html#a070ccccb1eb32adb8d21f08f230bb4e0", null]
]
```

Definition at line 1 of file sh6red\_8c.js.

## 30.1717 doc/html/sh6remcon\_8c.js File Reference

### Variables

- var [sh6remcon\\_8c](#)

#### 30.1717.1 Variable Documentation

##### 30.1717.1.1 var sh6remcon\_8c

###### Initial value:

```
=
[
 ["SH6REMCON", "sh6remcon_8c.html#ac0815d6d1d054bbb1906c2dd663a2f2d", null],
 ["sh6remcon", "sh6remcon_8c.html#a33753b225608455f10e902fc90b206e2", null]
]
```

Definition at line 1 of file sh6remcon\_8c.js.

## 30.1718 doc/html/sh6rempt\_8c.js File Reference

### Variables

- var [sh6rempt\\_8c](#)

### 30.1718.1 Variable Documentation

#### 30.1718.1.1 var sh6rempt\_8c

**Initial value:**

```
=
[
 ["SH6REMOVEPT", "sh6rempt_8c.html#ab91875a9a4cfe8ece303409ed266d714", null],
 ["sh6removept", "sh6rempt_8c.html#ad6c0d64e4c17fc4671b3a4f894afcbdd", null]
]
```

Definition at line 1 of file sh6rempt\_8c.js.

### 30.1719 doc/html/sh6sepcrv\_8c.js File Reference

**Variables**

- var [sh6sepcrv\\_8c](#)

#### 30.1719.1 Variable Documentation

##### 30.1719.1.1 var sh6sepcrv\_8c

**Initial value:**

```
=
[
 ["SH6SEPCRV", "sh6sepcrv_8c.html#abc90a2ce940bf936f50554984aeb384a", null],
 ["sh6sepcrv", "sh6sepcrv_8c.html#a93d2c11fcd92267a1cdfa67e5a81ddb9", null]
]
```

Definition at line 1 of file sh6sepcrv\_8c.js.

### 30.1720 doc/html/sh6setcnsd\_8c.js File Reference

**Variables**

- var [sh6setcnsd\\_8c](#)

#### 30.1720.1 Variable Documentation

##### 30.1720.1.1 var sh6setcnsd\_8c

**Initial value:**

```
=
[
 ["SH6SETCNSDIR", "sh6setcnsd_8c.html#a351b19abf17457b755b0cb8648dc2fffb", null],
 ["sh6setcnsdir", "sh6setcnsd_8c.html#a267a4676be5c9b7dbeaf2ebd89783b15", null]
]
```

Definition at line 1 of file sh6setcnsd\_8c.js.

## 30.1721 doc/html/sh6setdir\_8c.js File Reference

### Variables

- var [sh6setdir\\_8c](#)

#### 30.1721.1 Variable Documentation

##### 30.1721.1.1 var sh6setdir\_8c

###### Initial value:

```
=
[
 ["SH6SETDIR", "sh6setdir_8c.html#ac729135f6795117ad6b44ddec47ddd1", null],
 ["sh6setdir", "sh6setdir_8c.html#a20e29afe34cc0d2e5533f8ca9e351023", null]
]
```

Definition at line 1 of file sh6setdir\_8c.js.

## 30.1722 doc/html/sh6settop\_8c.js File Reference

### Variables

- var [sh6settop\\_8c](#)

#### 30.1722.1 Variable Documentation

##### 30.1722.1.1 var sh6settop\_8c

###### Initial value:

```
=
[
 ["SH6SETTOP", "sh6settop_8c.html#aa394d04421e09b58bf88fa2f0c08686a", null],
 ["sh6settop", "sh6settop_8c.html#a59c3d57c15d5812d124ea6b2a046735b", null]
]
```

Definition at line 1 of file sh6settop\_8c.js.

## 30.1723 doc/html/sh6spltgeo\_8c.js File Reference

### Variables

- var [sh6spltgeo\\_8c](#)

### 30.1723.1 Variable Documentation

#### 30.1723.1.1 var sh6spltgeo\_8c

**Initial value:**

```
=
[
 ["SH6SPLITGEOM", "sh6spltgeo_8c.html#acceleba6d4c4b8854d4ecd0574ca76dc", null],
 ["sh6spltgeom", "sh6spltgeo_8c.html#ab099e556161318dca26f3291a6e50799", null]
]
```

Definition at line 1 of file sh6spltgeo\_8c.js.

### 30.1724 doc/html/sh6tohelp\_8c.js File Reference

**Variables**

- var [sh6tohelp\\_8c](#)

#### 30.1724.1 Variable Documentation

##### 30.1724.1.1 var sh6tohelp\_8c

**Initial value:**

```
=
[
 ["SH6TOHELP", "sh6tohelp_8c.html#a607b71db79c735bc8ac56e30e5c6ac94", null],
 ["sh6tohelp", "sh6tohelp_8c.html#a848d1506b88f48c0d8749390bbccd017", null]
]
```

Definition at line 1 of file sh6tohelp\_8c.js.

### 30.1725 doc/html/sh6tomain\_8c.js File Reference

**Variables**

- var [sh6tomain\\_8c](#)

#### 30.1725.1 Variable Documentation

##### 30.1725.1.1 var sh6tomain\_8c

**Initial value:**

```
=
[
 ["SH6TOMAIN", "sh6tomain_8c.html#abc7e258f890a0ed91c5d7c4296abec7", null],
 ["sh6tomain", "sh6tomain_8c.html#ada68312b3b987f0c4eb10da02e52244d", null]
]
```

Definition at line 1 of file sh6tomain\_8c.js.

## 30.1726 doc/html/sh6topohlp\_8c.js File Reference

### Variables

- var [sh6topohlp\\_8c](#)

#### 30.1726.1 Variable Documentation

##### 30.1726.1.1 var sh6topohlp\_8c

###### Initial value:

```
=
[
 ["SH6GETTOPHLP", "sh6topohlp_8c.html#ac0b07e537df497e13122959615d41990", null],
 ["sh6gettophlp", "sh6topohlp_8c.html#a6627cb084123ddaa28012f1f36935882", null]
]
```

Definition at line 1 of file sh6topohlp\_8c.js.

## 30.1727 doc/html/sh6trmlist\_8c.js File Reference

### Variables

- var [sh6trmlist\\_8c](#)

#### 30.1727.1 Variable Documentation

##### 30.1727.1.1 var sh6trmlist\_8c

###### Initial value:

```
=
[
 ["SH6TRIMLIST", "sh6trmlist_8c.html#acc3c264459a941f3386916ceb56004e1", null],
 ["sh6trimlist", "sh6trmlist_8c.html#a5406954936b747638dfce401afbef2c", null]
]
```

Definition at line 1 of file sh6trmlist\_8c.js.

## 30.1728 doc/html/sh\_\_1d\_\_div\_8c.js File Reference

### Variables

- var [sh\\_\\_1d\\_\\_div\\_8c](#)

### 30.1728.1 Variable Documentation

#### 30.1728.1.1 var sh\_\_1d\_\_div\_\_8c

**Initial value:**

```
=
[
 ["SH_1D_DIV", "sh__1d__div__8c.html#a556babe68df3aa043c3b79df48cda50e", null],
 ["sh_1d_div", "sh__1d__div__8c.html#a1f72e0258e406ddc9ae28aa342e7fbb8", null]
]
```

Definition at line 1 of file sh\_\_1d\_\_div\_\_8c.js.

### 30.1729 doc/html/sh\_\_div\_\_crv\_\_8c.js File Reference

**Variables**

- var [sh\\_\\_div\\_\\_crv\\_\\_8c](#)

#### 30.1729.1 Variable Documentation

##### 30.1729.1.1 var sh\_\_div\_\_crv\_\_8c

**Initial value:**

```
=
[
 ["SH_DIV_CRV", "sh__div__crv__8c.html#aa005bac3d55af3b8228ef651f8d5ed82", null],
 ["sh_div_crv", "sh__div__crv__8c.html#ab1f9c6fab7b315df15cfa6b77aba28cb", null]
]
```

Definition at line 1 of file sh\_\_div\_\_crv\_\_8c.js.

### 30.1730 doc/html/sh\_\_set\_\_at\_\_8c.js File Reference

**Variables**

- var [sh\\_\\_set\\_\\_at\\_\\_8c](#)

#### 30.1730.1 Variable Documentation

##### 30.1730.1.1 var sh\_\_set\_\_at\_\_8c

**Initial value:**

```
=
[
 ["SH_SET_AT", "sh__set__at__8c.html#a3255a53266073c94bb1b30926bf66da9", null],
 ["sh_set_at", "sh__set__at__8c.html#a66d841775e76cce7990f5bc50196a694", null]
]
```

Definition at line 1 of file sh\_\_set\_\_at\_\_8c.js.

## 30.1731 doc/html/shape\_8c.js File Reference

### Variables

- var [shape\\_8c](#)

#### 30.1731.1 Variable Documentation

##### 30.1731.1.1 var shape\_8c

###### Initial value:

```
=
[
 ["SHAPE", "shape_8c.html#ac87c8082751a389b26904e86cb59dc1c", null],
 ["shape", "shape_8c.html#a7c4d1336dd8bb444d0807203f8241d89", null]
]
```

Definition at line 1 of file shape\_8c.js.

## 30.1732 doc/html/shcheckput\_8c.js File Reference

### Variables

- var [shcheckput\\_8c](#)

#### 30.1732.1 Variable Documentation

##### 30.1732.1.1 var shcheckput\_8c

###### Initial value:

```
=
[
 ["SHCHECKPUT", "shcheckput_8c.html#a9e36e2212f708937f0ed5b77f2cf4730", null],
 ["shcheckput", "shcheckput_8c.html#a26524e868944c4a184a5bd544ae9faa2", null]
]
```

Definition at line 1 of file shcheckput\_8c.js.

## 30.1733 doc/html/shchecktyp\_8c.js File Reference

### Variables

- var [shchecktyp\\_8c](#)

### 30.1733.1 Variable Documentation

#### 30.1733.1.1 var shchecktyp\_8c

**Initial value:**

```
=
[
 ["SHCHECKTYPE", "shchecktyp_8c.html#a90290aaaf635725f0b96e187b828558f", null],
 ["shchecktype", "shchecktyp_8c.html#a19d45c66afba74b4d0d5f4bb0a32105", null]
]
```

Definition at line 1 of file shchecktyp\_8c.js.

## 30.1734 doc/html/shcsfsing\_8c.js File Reference

### Variables

- var [shcsfsing\\_8c](#)

### 30.1734.1 Variable Documentation

#### 30.1734.1.1 var shcsfsing\_8c

**Initial value:**

```
=
[
 ["SHCSFSING", "shcsfsing_8c.html#a58cafdbd0467b9f0d5a6466753504fcc", null],
 ["shcsfsing", "shcsfsing_8c.html#ade962d53c8a17ac6235e6c983bee5fe6", null]
]
```

Definition at line 1 of file shcsfsing\_8c.js.

## 30.1735 doc/html/shdivsurf\_8c.js File Reference

### Variables

- var [shdivsurf\\_8c](#)

### 30.1735.1 Variable Documentation

#### 30.1735.1.1 var shdivsurf\_8c

**Initial value:**

```
=
[
 ["SH_DIV_SURF", "shdivsurf_8c.html#afa7e7bbd7549f685c9a4c65c5584c8f4", null],
 ["sh_div_surf", "shdivsurf_8c.html#abd93a9592b45b9316e65da391a7b5f1c", null]
]
```

Definition at line 1 of file shdivsurf\_8c.js.



## 30.1736 doc/html/shevalc\_8c.js File Reference

### Variables

- var [shevalc\\_8c](#)

#### 30.1736.1 Variable Documentation

##### 30.1736.1.1 var shevalc\_8c

###### Initial value:

```
=
[
 ["SHEVALC", "shevalc_8c.html#a2614bb5f46ad539324752297dd163f55", null],
 ["shevalc", "shevalc_8c.html#a385976a259db5e930009ff66013579b8", null]
]
```

Definition at line 1 of file shevalc\_8c.js.

## 30.1737 doc/html/shmkhlppts\_8c.js File Reference

### Variables

- var [shmkhlppts\\_8c](#)

#### 30.1737.1 Variable Documentation

##### 30.1737.1.1 var shmkhlppts\_8c

###### Initial value:

```
=
[
 ["SHMKHLPPTS", "shmkhlppts_8c.html#abf8f635b6fdaff6166dd237c3421981c", null],
 ["shmkhlppts", "shmkhlppts_8c.html#a3ed280ab29326295f9f85b00dcbdd107", null]
]
```

Definition at line 1 of file shmkhlppts\_8c.js.

## 30.1738 doc/html/shsing\_8c.js File Reference

### Variables

- var [shsing\\_8c](#)

### 30.1738.1 Variable Documentation

#### 30.1738.1.1 var shsing\_8c

**Initial value:**

```
=
[
 ["SHSING", "shsing_8c.html#aa91694f277641baba1e47820d9a2bda7", null],
 ["shsing", "shsing_8c.html#a9cdb51640398934a2c752f4796650119", null]
]
```

Definition at line 1 of file shsing\_8c.js.

### 30.1739 doc/html/sisl\_8h.js File Reference

**Variables**

- var [sisl\\_8h](#)

#### 30.1739.1 Variable Documentation

##### 30.1739.1.1 var sisl\_8h

Definition at line 1 of file sisl\_8h.js.

### 30.1740 doc/html/sisl\_\_aux\_8cpp.js File Reference

**Variables**

- var [sisl\\_\\_aux\\_8cpp](#)

#### 30.1740.1 Variable Documentation

##### 30.1740.1.1 var sisl\_\_aux\_8cpp

**Initial value:**

```
=
[
 ["DBGqqq", "sisl__aux_8cpp.html#a22e3c68f7ee44d24da231675c4aceafa", null],
 ["compute_surface_normals", "sisl__aux_8cpp.html#a8308e4ab9871c36e1c464ec66a38330e", null],
 ["lower_degree_and_subdivide", "sisl__aux_8cpp.html#a41370e2009c0559374f5b2a8fac856c3", null],
 ["lower_degree_to_linear", "sisl__aux_8cpp.html#aa5f98ab5409ba1865a5085a4538alb1c", null]
]
```

Definition at line 1 of file sisl\_\_aux\_8cpp.js.

## 30.1741 doc/html/sisl\_\_aux\_8h.js File Reference

### Variables

- var [sisl\\_\\_aux\\_8h](#)

### 30.1741.1 Variable Documentation

#### 30.1741.1.1 var sisl\_\_aux\_8h

#### Initial value:

```
=
[
 ["SISL_AUX_H_INCLUDED", "sisl__aux_8h.html#aff90e9e70b6fd8aa2943b9bca77603eb", null],
 ["compute_surface_normals", "sisl__aux_8h.html#a8308e4ab9871c36e1c464ec66a38330e", null],
 ["lower_degree_and_subdivide", "sisl__aux_8h.html#a41370e2009c0559374f5b2a8fac856c3", null],
 ["lower_degree_to_linear", "sisl__aux_8h.html#aa5f98ab5409ba1865a5085a4538a1b1c", null]
]
```

Definition at line 1 of file sisl\_\_aux\_8h.js.

## 30.1742 doc/html/sisl\_\_file\_\_io\_8h.js File Reference

### Variables

- var [sisl\\_\\_file\\_\\_io\\_8h](#)

### 30.1742.1 Variable Documentation

#### 30.1742.1.1 var sisl\_\_file\_\_io\_8h

#### Initial value:

```
=
[
 ["STD_FILE", "sisl__file__io_8h.html#af318346325017ad6fc3de1d8b3f04b63", null],
 ["curve_to_file", "sisl__file__io_8h.html#a2009d55672284872828775f9028501cf", null],
 ["file_to_obj", "sisl__file__io_8h.html#a33acd8174723671557c66157c7c497bc", null],
 ["get_next_surface", "sisl__file__io_8h.html#acd2574f50f6a00a582b57459d000548e", null],
 ["get_sisl_surfaces", "sisl__file__io_8h.html#a97db5bc2e2c26fcdecbb4e833e93962b", null],
 ["read_non_comment", "sisl__file__io_8h.html#a4ebf0dcd87434f41757f2b67961a45ef", null],
 ["surface_to_file", "sisl__file__io_8h.html#adb534c8afd8a0abd4cae8f7ff6b5731", null]
]
```

Definition at line 1 of file sisl\_\_file\_\_io\_8h.js.

## 30.1743 doc/html/sisl\_\_view\_\_demo\_8cpp.js File Reference

### Variables

- var [sisl\\_\\_view\\_\\_demo\\_8cpp](#)

### 30.1743.1 Variable Documentation

#### 30.1743.1.1 var sisl\_\_view\_\_demo\_8cpp

Definition at line 1 of file sisl\_\_view\_\_demo\_8cpp.js.

## 30.1744 doc/html/sisl\_p\_8h.js File Reference

### Variables

- var [sisl\\_p\\_8h](#)

### 30.1744.1 Variable Documentation

#### 30.1744.1.1 var sisl\_p\_8h

Definition at line 1 of file sisl\_p\_8h.js.

## 30.1745 doc/html/sl\_\_ex\_8cpp.js File Reference

### Variables

- var [sl\\_\\_ex\\_8cpp](#)

### 30.1745.1 Variable Documentation

#### 30.1745.1.1 var sl\_\_ex\_8cpp

#### Initial value:

```
=
[
 ["Cube", "class_cube.html", null],
 ["WANT_MATH", "sl__ex_8cpp.html#a8335b327d416f961a5dbaa0e058cc515", null],
 ["WANT_STREAM", "sl__ex_8cpp.html#a6ed6c2d6e68d8f0e7900326b4e30850c", null],
 ["main", "sl__ex_8cpp.html#ae66f6b31b5ad750f1fe042a706a4e3d4", null]
]
```

Definition at line 1 of file sl\_\_ex\_8cpp.js.

## 30.1746 doc/html/sleep\_8h.js File Reference

### Variables

- var [sleep\\_8h](#)

### 30.1746.1 Variable Documentation

30.1746.1.1 var sleep\_8h

**Initial value:**

```
=
[
 ["msleep", "sleep_8h.html#a6d5214d1f7774c98bbfb5a2f18a4d95c", null]
]
```

Definition at line 1 of file sleep\_8h.js.

## 30.1747 doc/html/solution\_8cpp.js File Reference

### Variables

- var [solution\\_8cpp](#)

### 30.1747.1 Variable Documentation

30.1747.1.1 var solution\_8cpp

**Initial value:**

```
=
[
 ["WANT_MATH", "solution_8cpp.html#a8335b327d416f961a5dbaa0e058cc515", null],
 ["WANT_STREAM", "solution_8cpp.html#a6ed6c2d6e68d8f0e7900326b4e30850c", null],
 ["square", "solution_8cpp.html#ae0c42a7620957500708a95cbb31e4260", null]
]
```

Definition at line 1 of file solution\_8cpp.js.

## 30.1748 doc/html/sort\_8cpp.js File Reference

### Variables

- var [sort\\_8cpp](#)

### 30.1748.1 Variable Documentation

#### 30.1748.1.1 var sort\_8cpp

**Initial value:**

```
=
[
 ["DoSimpleSort", "sort_8cpp.html#a8435a3581eb620a85d630b3d5ae61936", null],
 ["MaxDepth", "sort_8cpp.html#aba605615b0d8658dbbe798b26420fa7e", null],
 ["REPORT", "sort_8cpp.html#a787099914a94ed31fa544b985b55752f", null],
 ["WANT_MATH", "sort_8cpp.html#a8335b327d416f961a5dbaa0e058cc515", null],
 ["SortAscending", "sort_8cpp.html#ac9ddf9f13cd8b252a2712023d6a13f82", null],
 ["SortDescending", "sort_8cpp.html#a074fccd66ff446e8b7f3358a13fecb29", null],
 ["SortSV", "sort_8cpp.html#a20f7a2b9175cfc10ff1befacff795c", null],
 ["SortSV", "sort_8cpp.html#aa8f0accc1aabaaf5a211314aeaf5559b", null]
]
```

Definition at line 1 of file sort\_8cpp.js.

### 30.1749 doc/html/spli\_\_silh\_8c.js File Reference

#### Variables

- var [spli\\_\\_silh\\_8c](#)

#### 30.1749.1 Variable Documentation

##### 30.1749.1.1 var spli\_\_silh\_8c

**Initial value:**

```
=
[
 ["SPLI_SILH", "spli__silh_8c.html#ad5c5406b54fc3870162a123c96b07cab", null],
 ["spli_silh", "spli__silh_8c.html#ae3b475e63e628683bdd68b1c7046a182", null]
]
```

Definition at line 1 of file spli\_\_silh\_8c.js.

### 30.1750 doc/html/spline2mesh\_8h.js File Reference

#### Variables

- var [spline2mesh\\_8h](#)

### 30.1750.1 Variable Documentation

30.1750.1.1 var spline2mesh\_8h

#### Initial value:

```
=
[
 ["make_trimmed_mesh", "spline2mesh_8h.html#a27667fd082f69d891d6ef61d01812909", null]
]
```

Definition at line 1 of file spline2mesh\_8h.js.

## 30.1751 doc/html/struct\_go\_1\_1\_adjacency\_info.js File Reference

### Variables

- var [struct\\_go\\_1\\_1\\_adjacency\\_info](#)

### 30.1751.1 Variable Documentation

30.1751.1.1 var struct\_go\_1\_1\_adjacency\_info

#### Initial value:

```
=
[
 ["AdjacencyInfo", "struct_go_1_1_adjacency_info.html#ad3c0a4063f137288f0a9ac36f544298d", null],
 ["adjacency_found_", "struct_go_1_1_adjacency_info.html#a3ac7dc366593e8dacac6ecfc3dab3e16", null],
 ["bd_idx_1_", "struct_go_1_1_adjacency_info.html#a135e418f5acb664e9418a53950280d55", null],
 ["bd_idx_2_", "struct_go_1_1_adjacency_info.html#a81130b05146075f1811048f327f60df9", null],
 ["corner_failed_", "struct_go_1_1_adjacency_info.html#a972fd35f9c95bfb83228b3c9c5a030ca", null],
 ["same_orient_", "struct_go_1_1_adjacency_info.html#acad312b3c03024d7d1c1326aac59873a", null]
]
```

Definition at line 1 of file struct\_go\_1\_1\_adjacency\_info.js.

## 30.1752 doc/html/struct\_go\_1\_1\_alg2\_d\_elem.js File Reference

### Variables

- var [struct\\_go\\_1\\_1\\_alg2\\_d\\_elem](#)

### 30.1752.1 Variable Documentation

#### 30.1752.1.1 var struct\_go\_1\_1\_alg2\_d\_elem

##### Initial value:

```
=
[
 ["Alg2DElem", "struct_go_1_1_alg2_d_elem.html#a250274f1d6b660e316ea7afe05650c01", null],
 ["degrees_", "struct_go_1_1_alg2_d_elem.html#a522c5b04daa09ae4c08cb4cff3eafdcc", null],
 ["factor_", "struct_go_1_1_alg2_d_elem.html#af716c43434f3b63fe96ac23765c33a58", null]
]
```

Definition at line 1 of file struct\_go\_1\_1\_alg2\_d\_elem.js.

### 30.1753 doc/html/struct\_go\_1\_1\_alg3\_d\_elem.js File Reference

#### Variables

- var [struct\\_go\\_1\\_1\\_alg3\\_d\\_elem](#)

#### 30.1753.1 Variable Documentation

##### 30.1753.1.1 var struct\_go\_1\_1\_alg3\_d\_elem

##### Initial value:

```
=
[
 ["Alg3DElem", "struct_go_1_1_alg3_d_elem.html#a11a5c8e48159b5217cbc5ef0f3cddac8", null],
 ["degrees_", "struct_go_1_1_alg3_d_elem.html#a8a59cc9695d937aed4138522a05c96b6", null],
 ["factor_", "struct_go_1_1_alg3_d_elem.html#aa1beae70718c23de5e2b5bfd258b3998", null]
]
```

Definition at line 1 of file struct\_go\_1\_1\_alg3\_d\_elem.js.

### 30.1754 doc/html/struct\_go\_1\_1\_basis\_derivs.js File Reference

#### Variables

- var [struct\\_go\\_1\\_1\\_basis\\_derivs](#)



### 30.1754.1 Variable Documentation

#### 30.1754.1.1 var struct\_go\_1\_1\_basis\_derivs

##### Initial value:

```
=
[
 ["prepareDerivs", "struct_go_1_1_basis_derivs.html#a0f2d4956f345c6766520c5aec29e427d", null],
 ["basisDerivs_u", "struct_go_1_1_basis_derivs.html#a5f4e2a29371152a6434851fbfc0a6746", null],
 ["basisDerivs_v", "struct_go_1_1_basis_derivs.html#a75a1a2eb190559faf63e1c670c2b5284", null],
 ["basisDerivs_w", "struct_go_1_1_basis_derivs.html#a1b29f5f3ff4ba78f72923493f040adbb", null],
 ["basisValues", "struct_go_1_1_basis_derivs.html#ad574c9bd6638fac3def9a7c08770ee2d", null],
 ["left_idx", "struct_go_1_1_basis_derivs.html#ac2eecec7e9fb61617f3b459d979088b7", null],
 ["param", "struct_go_1_1_basis_derivs.html#a2e5e57b7537c7d0477e5fdcd1629a59c", null]
]
```

Definition at line 1 of file struct\_go\_1\_1\_basis\_derivs.js.

## 30.1755 doc/html/struct\_go\_1\_1\_basis\_derivs2.js File Reference

### Variables

- var [struct\\_go\\_1\\_1\\_basis\\_derivs2](#)

### 30.1755.1 Variable Documentation

#### 30.1755.1.1 var struct\_go\_1\_1\_basis\_derivs2

##### Initial value:

```
=
[
 ["prepareDerivs", "struct_go_1_1_basis_derivs2.html#a5cd8af2b43d26016c8e873b0b10286bb", null],
 ["basisDerivs_u", "struct_go_1_1_basis_derivs2.html#a3e86cd088c2275d67b028312d4194ccc", null],
 ["basisDerivs_uu", "struct_go_1_1_basis_derivs2.html#a4996c2b8943eb42e5a9fd7f8c1c68f2b", null],
 ["basisDerivs_uv", "struct_go_1_1_basis_derivs2.html#a05d1d272ca15c51f176630dff370b1c7", null],
 ["basisDerivs_uw", "struct_go_1_1_basis_derivs2.html#aa6b76d8fd5df9f04ec5393980543d3ec", null],
 ["basisDerivs_v", "struct_go_1_1_basis_derivs2.html#a15bc9a9f8c705547e215271f694b4f2b", null],
 ["basisDerivs_vv", "struct_go_1_1_basis_derivs2.html#a44960d0fc0cb8a5ea1801938577d85d7", null],
 ["basisDerivs_vw", "struct_go_1_1_basis_derivs2.html#a76278e7a79fac6a1d2a8c80f6f4cd9fb", null],
 ["basisDerivs_w", "struct_go_1_1_basis_derivs2.html#aa12e98b99c909c6e921bbba216f677e9", null],
 ["basisDerivs_ww", "struct_go_1_1_basis_derivs2.html#ac730fbecf4031715f1af594bf2eeeb5c", null],
 ["basisValues", "struct_go_1_1_basis_derivs2.html#a4bc4bdc3686322b8efe061781413d467", null],
 ["left_idx", "struct_go_1_1_basis_derivs2.html#aec82985f5d7d85022bf5b6e15396730a", null],
 ["param", "struct_go_1_1_basis_derivs2.html#a4dc5f908e2c48928ee48af3d45fdaadf", null]
]
```

Definition at line 1 of file struct\_go\_1\_1\_basis\_derivs2.js.

## 30.1756 doc/html/struct\_go\_1\_1\_basis\_derivs\_sf.js File Reference

### Variables

- var [struct\\_go\\_1\\_1\\_basis\\_derivs\\_sf](#)

### 30.1756.1 Variable Documentation

#### 30.1756.1.1 var struct\_go\_1\_1\_basis\_derivs\_sf

##### Initial value:

```
=
[
 ["prepareDerivs", "struct_go_1_1_basis_derivs_sf.html#aaldfdeda83e25fc3c15663dc42c516cd", null],
 ["basisDerivs_u", "struct_go_1_1_basis_derivs_sf.html#a7a26e86e9156363bdae38d7eb091b5a5", null],
 ["basisDerivs_v", "struct_go_1_1_basis_derivs_sf.html#add12858f4dea006a23fbb9193cd3bddb", null],
 ["basisValues", "struct_go_1_1_basis_derivs_sf.html#ab16e0b1de9155d2e66761345cb31e3a8", null],
 ["left_idx", "struct_go_1_1_basis_derivs_sf.html#a1adf51c6d8e7b08c87e953d84a09217f", null],
 ["param", "struct_go_1_1_basis_derivs_sf.html#a67167cf9e44d225ea7f31d55f9921c0d", null]
]
```

Definition at line 1 of file struct\_go\_1\_1\_basis\_derivs\_sf.js.

### 30.1757 doc/html/struct\_go\_1\_1\_basis\_derivs\_sf2.js File Reference

#### Variables

- var [struct\\_go\\_1\\_1\\_basis\\_derivs\\_sf2](#)

### 30.1757.1 Variable Documentation

#### 30.1757.1.1 var struct\_go\_1\_1\_basis\_derivs\_sf2

##### Initial value:

```
=
[
 ["prepareDerivs", "struct_go_1_1_basis_derivs_sf2.html#a676603203e8a459dd9494bdf183555b", null],
 ["basisDerivs_u", "struct_go_1_1_basis_derivs_sf2.html#a3d72d6a39c1f7034e6a6d7b061dfac14", null],
 ["basisDerivs_uu", "struct_go_1_1_basis_derivs_sf2.html#a13170a3e3019f68c407dfa0f8040dba8", null],
 ["basisDerivs_uv", "struct_go_1_1_basis_derivs_sf2.html#adc9789db3b1bc8eaa37c146755415591", null],
 ["basisDerivs_v", "struct_go_1_1_basis_derivs_sf2.html#a9e59c18dea79f0b30601a610d1e17cd4", null],
 ["basisDerivs_vv", "struct_go_1_1_basis_derivs_sf2.html#ab4001ded4fa3b3b4405cbbe257dd1ce3", null],
 ["basisValues", "struct_go_1_1_basis_derivs_sf2.html#a93947150bf557685ebafdb390db38917", null],
 ["left_idx", "struct_go_1_1_basis_derivs_sf2.html#a5ff58e0e70be8c4e2c38df1c41f33161", null],
 ["param", "struct_go_1_1_basis_derivs_sf2.html#a53dbe3c720404d69680df2b5feae9a93", null]
]
```

Definition at line 1 of file struct\_go\_1\_1\_basis\_derivs\_sf2.js.

### 30.1758 doc/html/struct\_go\_1\_1\_basis\_pts.js File Reference

#### Variables

- var [struct\\_go\\_1\\_1\\_basis\\_pts](#)

### 30.1758.1 Variable Documentation

#### 30.1758.1.1 var struct\_go\_1\_1\_basis\_pts

##### Initial value:

```
=
[
 ["preparePts", "struct_go_1_1_basis_pts.html#ad22db843371b8f397257dbe583843b7e", null],
 ["basisValues", "struct_go_1_1_basis_pts.html#aba6acc133e5a05541107092f6a9bca7c", null],
 ["left_idx", "struct_go_1_1_basis_pts.html#alc795dfc9bb560f5d484f091a1474bd3", null],
 ["param", "struct_go_1_1_basis_pts.html#ad97d1b920547ff7d174645fb4dfa2f5c", null]
]
```

Definition at line 1 of file struct\_go\_1\_1\_basis\_pts.js.

## 30.1759 doc/html/struct\_go\_1\_1\_basis\_pts\_sf.js File Reference

### Variables

- var [struct\\_go\\_1\\_1\\_basis\\_pts\\_sf](#)

### 30.1759.1 Variable Documentation

#### 30.1759.1.1 var struct\_go\_1\_1\_basis\_pts\_sf

##### Initial value:

```
=
[
 ["preparePts", "struct_go_1_1_basis_pts_sf.html#a86dd096c1aefe0d735b37d93fa535b41", null],
 ["basisValues", "struct_go_1_1_basis_pts_sf.html#a31b96b97399bfa287f882cdb2164a059", null],
 ["left_idx", "struct_go_1_1_basis_pts_sf.html#a2c610e84482885c898916dc71eb2d8a7", null],
 ["param", "struct_go_1_1_basis_pts_sf.html#a84abbd4aec60b7c33f66c390a5cd6f9", null]
]
```

Definition at line 1 of file struct\_go\_1\_1\_basis\_pts\_sf.js.

## 30.1760 doc/html/struct\_go\_1\_1\_boundary\_function\_int.js File Reference

### Variables

- var [struct\\_go\\_1\\_1\\_boundary\\_function\\_int](#)

### 30.1760.1 Variable Documentation

#### 30.1760.1.1 var struct\_go\_1\_1\_boundary\_function\_int

##### Initial value:

```
=
[
 ["BoundaryFunctionInt", "struct_go_1_1_boundary_function_int.html#adde591970ca49de228bf50daf99ae383",
 null],
 ["getDir", "struct_go_1_1_boundary_function_int.html#abbcc046f35a5b9fcd639cf7f12d1f90", null],
 ["getObject", "struct_go_1_1_boundary_function_int.html#adba90dced2c9bbd2c547e00676fafac7", null],
 ["getPar", "struct_go_1_1_boundary_function_int.html#aca6592096415555d4b1568a9c86743bd", null],
 ["bd_obj_", "struct_go_1_1_boundary_function_int.html#a0b0876977cae00b3366392a785271dc9", null],
 ["par_", "struct_go_1_1_boundary_function_int.html#abdd6fb0794b689f8cedab585558fb705", null],
 ["pardir_", "struct_go_1_1_boundary_function_int.html#adcac7d7de248a75f2430289d36c4e4d", null]
]
```

Definition at line 1 of file struct\_go\_1\_1\_boundary\_function\_int.js.

### 30.1761 doc/html/struct\_go\_1\_1\_boundary\_geom\_int.js File Reference

#### Variables

- var [struct\\_go\\_1\\_1\\_boundary\\_geom\\_int](#)

#### 30.1761.1 Variable Documentation

##### 30.1761.1.1 var struct\_go\_1\_1\_boundary\_geom\_int

##### Initial value:

```
=
[
 ["BoundaryGeomInt", "struct_go_1_1_boundary_geom_int.html#a882a22882af88ef60332f51546381b8a", null],
 ["~BoundaryGeomInt", "struct_go_1_1_boundary_geom_int.html#a8aec94df6c1451ed33falaf838634a7", null],
 ["getDir", "struct_go_1_1_boundary_geom_int.html#a99fe848182e9a3f5e0f7d7d2b4d91a73", null],
 ["getObject", "struct_go_1_1_boundary_geom_int.html#ab7bc4834231e9fe90d2665ca75793f10", null],
 ["getPar", "struct_go_1_1_boundary_geom_int.html#a1fa41fdd02c95da5cb423308f2be1cbf", null],
 ["bd_obj_", "struct_go_1_1_boundary_geom_int.html#a4e614dd0ecf36ec3ba697525a61acc2d", null],
 ["par_", "struct_go_1_1_boundary_geom_int.html#aad567caa842e72da59abef6979c8766b", null],
 ["pardir_", "struct_go_1_1_boundary_geom_int.html#a024ae519753767d06f28b20feb866c07", null]
]
```

Definition at line 1 of file struct\_go\_1\_1\_boundary\_geom\_int.js.

### 30.1762 doc/html/struct\_go\_1\_1\_boundary\_intersection\_data.js File Reference

#### Variables

- var [struct\\_go\\_1\\_1\\_boundary\\_intersection\\_data](#)

### 30.1762.1 Variable Documentation

#### 30.1762.1.1 var struct\_go\_1\_1\_boundary\_intersection\_data

**Initial value:**

```
=
[
 ["dir", "struct_go_1_1_boundary_intersection_data.html#ad6374760b4612e464ee5a17614d28f98", null],
 ["par", "struct_go_1_1_boundary_intersection_data.html#a2f9c90e69885fa15d27e41a8995bb6f3", null],
 ["pts", "struct_go_1_1_boundary_intersection_data.html#ac7fcf20e08a0bde5c3c3d3d622bbd4c2", null]
]
```

Definition at line 1 of file struct\_go\_1\_1\_boundary\_intersection\_data.js.

## 30.1763 doc/html/struct\_go\_1\_1\_cached\_interval.js File Reference

### Variables

- var [struct\\_go\\_1\\_1\\_cached\\_interval](#)

### 30.1763.1 Variable Documentation

#### 30.1763.1.1 var struct\_go\_1\_1\_cached\_interval

**Initial value:**

```
=
[
 ["CachedInterval", "struct_go_1_1_cached_interval.html#a2a79622b639df3527ba3d5fe75b6564e", null],
 ["cached", "struct_go_1_1_cached_interval.html#a8170fc35bc28a1bc0ec21ca682fa3467", null],
 ["inside", "struct_go_1_1_cached_interval.html#add52690f177dc786745baab34218d43e", null],
 ["outside", "struct_go_1_1_cached_interval.html#aa3934c163e9e6f7234e8b62e28c01fe5", null]
]
```

Definition at line 1 of file struct\_go\_1\_1\_cached\_interval.js.

## 30.1764 doc/html/struct\_go\_1\_1\_composite\_model\_file\_handler\_1\_1sfvinfo.js File Reference

### Variables

- var [struct\\_go\\_1\\_1\\_composite\\_model\\_file\\_handler\\_1\\_1sfvinfo](#)

### 30.1764.1 Variable Documentation

#### 30.1764.1.1 var struct\_go\_1\_1\_composite\_model\_file\_handler\_1\_1sfcvinfo

##### Initial value:

```
=
[
 ["sfcvinfo", "
 struct_go_1_1_composite_model_file_handler_1_1sfcvinfo.html#a177513f5a9a1909739ce7b596ae09151", null],
 ["sfcvinfo", "
 struct_go_1_1_composite_model_file_handler_1_1sfcvinfo.html#a9e3c53fc0515becef648e9d76361d936", null],
 ["bd_", "struct_go_1_1_composite_model_file_handler_1_1sfcvinfo.html#a896c96f83432fabe1857f3fa0c967471", null],
 ["ccm_", "
 struct_go_1_1_composite_model_file_handler_1_1sfcvinfo.html#a973e16399c81b5c4b7048218a994c0c1", null],
 ["constdir_", "
 struct_go_1_1_composite_model_file_handler_1_1sfcvinfo.html#ac85589591c5e53b310729228778b9cb6", null],
 ["constpar_", "
 struct_go_1_1_composite_model_file_handler_1_1sfcvinfo.html#a743d5484d4cbacb67b40895d038d6d5f", null],
 ["infoiset_", "
 struct_go_1_1_composite_model_file_handler_1_1sfcvinfo.html#a86fcad4cbb8e0d381a9fd62b15168932", null],
 ["orientation_", "
 struct_go_1_1_composite_model_file_handler_1_1sfcvinfo.html#a37202c0c673bb47d999ee7391da3dd92", null]
]
```

Definition at line 1 of file struct\_go\_1\_1\_composite\_model\_file\_handler\_1\_1sfcvinfo.js.

### 30.1765 doc/html/struct\_go\_1\_1\_entity\_list.js File Reference

#### Variables

- var [struct\\_go\\_1\\_1\\_entity\\_list](#)

### 30.1765.1 Variable Documentation

#### 30.1765.1.1 var struct\_go\_1\_1\_entity\_list

##### Initial value:

```
=
[
 ["EntityList", "struct_go_1_1_entity_list.html#ad199ed8f5f536681e40c02ed7193f42d", null],
 ["validEntity", "struct_go_1_1_entity_list.html#a4579a25ef0627fd75736eec3082a5fc3", null]
]
```

Definition at line 1 of file struct\_go\_1\_1\_entity\_list.js.

### 30.1766 doc/html/struct\_go\_1\_1\_face\_connectivity.js File Reference

#### Variables

- var [struct\\_go\\_1\\_1\\_face\\_connectivity](#)

### 30.1766.1 Variable Documentation

#### 30.1766.1.1 var struct\_go\_1\_1\_face\_connectivity

##### Initial value:

```
=
[
 ["FaceConnectivity", "struct_go_1_1_face_connectivity.html#a296535b83af35ef3587335903306feb4", null],
 ["BestStatus", "struct_go_1_1_face_connectivity.html#a70242d99cd485ceb0947580499029539", null],
 ["setEdges", "struct_go_1_1_face_connectivity.html#adf71b98305e2832964bd274299548736", null],
 ["WorstStatus", "struct_go_1_1_face_connectivity.html#a71455fb59609e8d2356ad33f1bc637cb", null],
 ["e1_", "struct_go_1_1_face_connectivity.html#a42876bc3d54e64697555761d3667382e", null],
 ["e2_", "struct_go_1_1_face_connectivity.html#a56d1d7a434f3fd07682e85488b369fb8", null],
 ["parameters_", "struct_go_1_1_face_connectivity.html#a2ed15f49224afe9774dca01f42adbc3d", null],
 ["status_", "struct_go_1_1_face_connectivity.html#ab76cb8807d8e5fc05e146bb22494969c", null]
]
```

Definition at line 1 of file struct\_go\_1\_1\_face\_connectivity.js.

## 30.1767 doc/html/struct\_go\_1\_1\_factorial.js File Reference

### Variables

- var [struct\\_go\\_1\\_1\\_factorial](#)

### 30.1767.1 Variable Documentation

#### 30.1767.1.1 var struct\_go\_1\_1\_factorial

##### Initial value:

```
=
[
 ["value", "
 struct_go_1_1_factorial.html#a579daed3b4275abe013c8f88077f888fa4acbf42abf2d3a50f62969f536each98", null]
]
```

Definition at line 1 of file struct\_go\_1\_1\_factorial.js.

## 30.1768 doc/html/struct\_go\_1\_1\_factorial\_3\_011\_01\_4.js File Reference

### Variables

- var [struct\\_go\\_1\\_1\\_factorial\\_3\\_011\\_01\\_4](#)

### 30.1768.1 Variable Documentation

#### 30.1768.1.1 var struct\_go\_1\_1\_factorial\_3\_011\_01\_4

##### Initial value:

```
=
[
 ["value", "
 struct_go_1_1_factorial_3_011_01_4.html#a654cb8f1088a190314946d0cb460482caf04561aa2fcbf1fb66fbc964a355d268", null]
]
```

Definition at line 1 of file struct\_go\_1\_1\_factorial\_3\_011\_01\_4.js.

### 30.1769 doc/html/struct\_go\_1\_1\_g\_pos.js File Reference

#### Variables

- var [struct\\_go\\_1\\_1\\_g\\_pos](#)

### 30.1769.1 Variable Documentation

#### 30.1769.1.1 var struct\_go\_1\_1\_g\_pos

##### Initial value:

```
=
[
 ["GPos", "struct_go_1_1_g_pos.html#ae910793fecbd0454cdd204d0caa3a4d2", null],
 ["GPos", "struct_go_1_1_g_pos.html#aee17bf04a62e40f1c61fe4ec6f0da6cb", null],
 ["ix", "struct_go_1_1_g_pos.html#a0895731fb21626c1164e1f6fa046adef", null],
 ["mult", "struct_go_1_1_g_pos.html#a4f709223efab431e6971b4c8339195b2", null]
]
```

Definition at line 1 of file struct\_go\_1\_1\_g\_pos.js.

### 30.1770 doc/html/struct\_go\_1\_1\_i\_g\_e\_sdirentry.js File Reference

#### Variables

- var [struct\\_go\\_1\\_1\\_i\\_g\\_e\\_sdirentry](#)



## 30.1770.1 Variable Documentation

### 30.1770.1.1 var struct\_go\_1\_1\_i\_g\_e\_sdirentry

#### Initial value:

```
=
[
 ["color", "struct_go_1_1_i_g_e_sdirentry.html#a4d990a00123c4704611b9ba8f27b989b", null],
 ["entity_label", "struct_go_1_1_i_g_e_sdirentry.html#a5480c7dc7f12734b1e313bef389b1712", null],
 ["entity_number", "struct_go_1_1_i_g_e_sdirentry.html#a9e49664cd4d7ed47a39c36b0b142667f", null],
 ["entity_type_number", "struct_go_1_1_i_g_e_sdirentry.html#a917759db22246f21ef576f1dae2227a8", null],
 ["form", "struct_go_1_1_i_g_e_sdirentry.html#a8606d475e0ffca150e9946c5308de47e", null],
 ["label_display", "struct_go_1_1_i_g_e_sdirentry.html#a2b8d7c0041720a2cc10d1eab59d69171", null],
 ["level", "struct_go_1_1_i_g_e_sdirentry.html#a837cf114d978fcc9afcebbfdb3669fd6", null],
 ["line_count", "struct_go_1_1_i_g_e_sdirentry.html#ab4ddf1370937d01fdd225c185bc7c258", null],
 ["line_font_pattern", "struct_go_1_1_i_g_e_sdirentry.html#a90c7e2e78d871d26f616c1d1f8cffc99", null],
 ["line_weight", "struct_go_1_1_i_g_e_sdirentry.html#acbc23ab052dfcfa1cb983ff880d06b9", null],
 ["param_data_start", "struct_go_1_1_i_g_e_sdirentry.html#a3e3ad50a649cfde1254cb2ded3d1976a", null],
 ["status", "struct_go_1_1_i_g_e_sdirentry.html#a0f36ffdbe266da4804cf0fa10745c7de", null],
 ["structure", "struct_go_1_1_i_g_e_sdirentry.html#a6093dbd62cd1555ef8eb137416f1b3e0", null],
 ["trans_matrix", "struct_go_1_1_i_g_e_sdirentry.html#ae98cd224456e7828d236ee28ce49b5df", null],
 ["view", "struct_go_1_1_i_g_e_sdirentry.html#a302fe7f59f59804f3b0b81c258dfe234", null]
]
```

Definition at line 1 of file struct\_go\_1\_1\_i\_g\_e\_sdirentry.js.

## 30.1771 doc/html/struct\_go\_1\_1\_i\_g\_e\_sheader.js File Reference

### Variables

- var [struct\\_go\\_1\\_1\\_i\\_g\\_e\\_sheader](#)

## 30.1771.1 Variable Documentation

### 30.1771.1.1 var struct\_go\_1\_1\_i\_g\_e\_sheader

Definition at line 1 of file struct\_go\_1\_1\_i\_g\_e\_sheader.js.

## 30.1772 doc/html/struct\_go\_1\_1\_int\_pt\_info.js File Reference

### Variables

- var [struct\\_go\\_1\\_1\\_int\\_pt\\_info](#)

### 30.1772.1 Variable Documentation

#### 30.1772.1.1 var struct\_go\_1\_1\_int\_pt\_info

##### Initial value:

```
=
[
 ["IntPtInfo", "struct_go_1_1_int_pt_info.html#add87e1a4de4b7f7a58cbab079cc68a06", null],
 ["direction", "struct_go_1_1_int_pt_info.html#ae3a23ef20b202e17ce73a7c7b0ad632c", null],
 ["is_ok", "struct_go_1_1_int_pt_info.html#a495860fb45a3d2a1cf2d2f07d3800484", null],
 ["location", "struct_go_1_1_int_pt_info.html#a88f8a5994f8e3764ac5f4214cbdf6cb5", null],
 ["neighbours", "struct_go_1_1_int_pt_info.html#a22aa9d2d7a95251c6039aa2b68c384a0", null],
 ["singularity_type", "struct_go_1_1_int_pt_info.html#a60186b83a5dadcca48c41b2b42175476", null]
]
```

Definition at line 1 of file struct\_go\_1\_1\_int\_pt\_info.js.

### 30.1773 doc/html/struct\_go\_1\_1\_l\_r\_spline\_surface\_1\_1\_b\_s\_key.js File Reference

#### Variables

- var [struct\\_go\\_1\\_1\\_l\\_r\\_spline\\_surface\\_1\\_1\\_b\\_s\\_key](#)

### 30.1773.1 Variable Documentation

#### 30.1773.1.1 var struct\_go\_1\_1\_l\_r\_spline\_surface\_1\_1\_b\_s\_key

##### Initial value:

```
=
[
 ["operator<", "struct_go_1_1_l_r_spline_surface_1_1_b_s_key.html#a32e81f8d2b5f0fd3352d8e22b9f233a1", null],
 ["u_max", "struct_go_1_1_l_r_spline_surface_1_1_b_s_key.html#a5cb5e8c6fbbfe331603f97c20d1ccc69", null],
 ["u_min", "struct_go_1_1_l_r_spline_surface_1_1_b_s_key.html#a59296b191fcad4e9c7c7034a2967ae3f", null],
 ["u_mult1", "struct_go_1_1_l_r_spline_surface_1_1_b_s_key.html#a0dd8a6a75f9f5f7404616f5c970c9285", null],
 ["u_mult2", "struct_go_1_1_l_r_spline_surface_1_1_b_s_key.html#ac4082082e914b3bf54b9cc1115f4bfd2", null],
 ["v_max", "struct_go_1_1_l_r_spline_surface_1_1_b_s_key.html#a6751c5eed5caa610e8dac5f3558b4702", null],
 ["v_min", "struct_go_1_1_l_r_spline_surface_1_1_b_s_key.html#a917055d1f2ba2783786ba6345d73cbd8", null],
 ["v_mult1", "struct_go_1_1_l_r_spline_surface_1_1_b_s_key.html#ac8f082b1e3b79de0d2531198fd3648af", null],
 ["v_mult2", "struct_go_1_1_l_r_spline_surface_1_1_b_s_key.html#a04b64c2a027326729a138282d8015159", null]
]
```

Definition at line 1 of file struct\_go\_1\_1\_l\_r\_spline\_surface\_1\_1\_b\_s\_key.js.

### 30.1774 doc/html/struct\_go\_1\_1\_l\_r\_spline\_surface\_1\_1\_elem\_key.js File Reference

#### Variables

- var [struct\\_go\\_1\\_1\\_l\\_r\\_spline\\_surface\\_1\\_1\\_elem\\_key](#)

### 30.1774.1 Variable Documentation

#### 30.1774.1.1 var struct\_go\_1\_1\_l\_r\_spline\_surface\_1\_1\_elem\_key

##### Initial value:

```
=
[
 ["operator<", "struct_go_1_1_l_r_spline_surface_1_1_elem_key.html#a4b8aa0d0e5013360361c758df0f5e0cf",
 null],
 ["u_min", "struct_go_1_1_l_r_spline_surface_1_1_elem_key.html#a585692254199d6a43db9f961acdf51c5", null
],
 ["v_min", "struct_go_1_1_l_r_spline_surface_1_1_elem_key.html#a9f214ce7d7d8e22bd73c4f1bb3c93f42", null
]
]
```

Definition at line 1 of file struct\_go\_1\_1\_l\_r\_spline\_surface\_1\_1\_elem\_key.js.

## 30.1775 doc/html/struct\_go\_1\_1\_l\_r\_spline\_surface\_1\_1\_refinement2\_d.js File Reference

### Variables

- var [struct\\_go\\_1\\_1\\_l\\_r\\_spline\\_surface\\_1\\_1\\_refinement2\\_d](#)

### 30.1775.1 Variable Documentation

#### 30.1775.1.1 var struct\_go\_1\_1\_l\_r\_spline\_surface\_1\_1\_refinement2\_d

##### Initial value:

```
=
[
 ["setVal", "struct_go_1_1_l_r_spline_surface_1_1_refinement2_d.html#aa26cc499b466c314bd246bd4418e87ef",
 null],
 ["d", "struct_go_1_1_l_r_spline_surface_1_1_refinement2_d.html#a8a68737e378b8daee92599fcb690ef3c",
 null],
 ["end", "struct_go_1_1_l_r_spline_surface_1_1_refinement2_d.html#a79c566a0c23670665f282314f2909e47",
 null],
 ["kval", "struct_go_1_1_l_r_spline_surface_1_1_refinement2_d.html#ad68d760801298c71f45dd219074943ea",
 null],
 ["multiplicity", "
 struct_go_1_1_l_r_spline_surface_1_1_refinement2_d.html#a462ad5c139fa0447bd322d7ff2268b73", null],
 ["start", "struct_go_1_1_l_r_spline_surface_1_1_refinement2_d.html#a325d8e12827d93a7cbe7bae48c39a6da",
 null]
]
```

Definition at line 1 of file struct\_go\_1\_1\_l\_r\_spline\_surface\_1\_1\_refinement2\_d.js.

## 30.1776 doc/html/struct\_go\_1\_1\_l\_r\_spline\_surface\_1\_1\_double\_pair\_hash.js File Reference

### Variables

- var [struct\\_go\\_1\\_1\\_l\\_r\\_spline\\_surface\\_1\\_1\\_double\\_pair\\_hash](#)

### 30.1776.1 Variable Documentation

30.1776.1.1 `var struct_go_1_1_l_r_spline_surface_1_1double__pair__hash`

#### Initial value:

```
=
[
 ["operator()", "
 struct_go_1_1_l_r_spline_surface_1_1double__pair__hash.html#a7e8b524336976f752599f2cb567bcd37", null]
]
```

Definition at line 1 of file `struct_go_1_1_l_r_spline_surface_1_1double__pair__hash.js`.

### 30.1777 `doc/html/struct_go_1_1_l_r_spline_utils_1_1support__compare.js` File Reference

#### Variables

- var [struct\\_go\\_1\\_1\\_l\\_r\\_spline\\_utils\\_1\\_1support\\_\\_compare](#)

### 30.1777.1 Variable Documentation

30.1777.1.1 `var struct_go_1_1_l_r_spline_utils_1_1support__compare`

#### Initial value:

```
=
[
 ["operator()", "
 struct_go_1_1_l_r_spline_utils_1_1support__compare.html#a28edd5aeea0ffbf79bea57b17574c74a", null]
]
```

Definition at line 1 of file `struct_go_1_1_l_r_spline_utils_1_1support__compare.js`.

### 30.1778 `doc/html/struct_go_1_1_l_s_smooth_data.js` File Reference

#### Variables

- var [struct\\_go\\_1\\_1\\_l\\_s\\_smooth\\_data](#)

### 30.1778.1 Variable Documentation

30.1778.1.1 `var struct_go_1_1_l_s_smooth_data`

Definition at line 1 of file `struct_go_1_1_l_s_smooth_data.js`.

## 30.1779 doc/html/struct\_go\_1\_1\_param\_surface\_1\_1degenerate\_\_info.js File Reference

### Variables

- var [struct\\_go\\_1\\_1\\_param\\_surface\\_1\\_1degenerate\\_\\_info](#)

### 30.1779.1 Variable Documentation

#### 30.1779.1.1 var struct\_go\_1\_1\_param\_surface\_1\_1degenerate\_\_info

##### Initial value:

```
=
[
 ["degenerate_info", "
 struct_go_1_1_param_surface_1_1degenerate__info.html#af7e36e8c2ef05ede0e1128b9e7095678", null],
 ["b_", "struct_go_1_1_param_surface_1_1degenerate__info.html#af0a777570c58379b45b0dd42f44abcf5", null
],
 ["is_set_", "struct_go_1_1_param_surface_1_1degenerate__info.html#a11677d293ea8af0c68160fdb3c842d5",
 null],
 ["l_", "struct_go_1_1_param_surface_1_1degenerate__info.html#a9e9e4dd3a9bad4fc495e1ca428d1c4e9", null
],
 ["r_", "struct_go_1_1_param_surface_1_1degenerate__info.html#af2004242eeb813ed444c5fdb9b265071", null
],
 ["t_", "struct_go_1_1_param_surface_1_1degenerate__info.html#aa4a1b5a04e6f9581c46bf8c50ec6e408", null
],
 ["tol_", "struct_go_1_1_param_surface_1_1degenerate__info.html#a55c0c7e01d4007c150c56dbafb798269",
 null]
]
```

Definition at line 1 of file `struct_go_1_1_param_surface_1_1degenerate__info.js`.

## 30.1780 doc/html/struct\_go\_1\_1\_registration\_input.js File Reference

### Variables

- var [struct\\_go\\_1\\_1\\_registration\\_input](#)

### 30.1780.1 Variable Documentation

#### 30.1780.1.1 var struct\_go\_1\_1\_registration\_input

##### Initial value:

```
=
[
 ["RegistrationInput", "struct_go_1_1_registration_input.html#afc39aa78123a30d05592b6980fbc482", null
],
 ["setToleranceWeights", "struct_go_1_1_registration_input.html#aca7253fe61b9355e8f84b2eb49985c02",
 null],
 ["area_tolerance_sq", "struct_go_1_1_registration_input.html#a9a3cc58d9786bf20ffbf59f98ac216e6", null
],
 ["calculate_tolerance_weights_", "
 struct_go_1_1_registration_input.html#aabe8f93a69c69d7bf91694bf6ecd4014", null],
 ["max_newton_iterations_", "struct_go_1_1_registration_input.html#abf0cd5901fd84ee604bc3c8bf7158fc1",
 null],
 ["max_solve_iterations_", "struct_go_1_1_registration_input.html#afd763fcedbcb0fc83fb44cfff6e4289",
 null],
 ["multi_core_", "struct_go_1_1_registration_input.html#a77cfc967fa9e21ea7a54dbd74b5babc4", null],
 ["newton_tolerance_", "struct_go_1_1_registration_input.html#ab16f5ac069894e7e8ca7373961e30238", null
],
 ["solve_tolerance_", "struct_go_1_1_registration_input.html#a58504d45397361df5e1e68b7882140d2", null]
 ,
 ["tolerance_weight_rescale_", "struct_go_1_1_registration_input.html#a9d3f27ba63e821fcb235938a203b2f0e
 ", null],
 ["tolerance_weight_rotation_", "
 struct_go_1_1_registration_input.html#a371aa86034f5a693f8dfd9229539b643", null],
 ["tolerance_weight_translation_", "
 struct_go_1_1_registration_input.html#a0208ec7d77cf7d0dbe724f362552ec56", null]
]
```

Definition at line 1 of file `struct_go_1_1_registration_input.js`.

## 30.1781 doc/html/struct\_go\_1\_1\_registration\_result.js File Reference

### Variables

- var [struct\\_go\\_1\\_1\\_registration\\_result](#)

### 30.1781.1 Variable Documentation

#### 30.1781.1.1 var struct\_go\_1\_1\_registration\_result

##### Initial value:

```
=
[
 ["ok", "struct_go_1_1_registration_result.html#a957cba48cdf7bb610efc628421dd67ec", null],
 ["last_change_", "struct_go_1_1_registration_result.html#a7df6e84c504a75d83f47aa2de92d747d", null],
 ["last_newton_iteration_", "struct_go_1_1_registration_result.html#a0955123270a8c94b76919e794f6e8e67",
 null],
 ["rescaling_", "struct_go_1_1_registration_result.html#a507fbaaf0d1eab2a0512885808a182b3", null],
 ["result_type_", "struct_go_1_1_registration_result.html#a0c2bd34528ab3fdee72d2ea9fb2ac10d", null],
 ["rotation_matrix_", "struct_go_1_1_registration_result.html#a1a9d598ac5b634684c0195091c3e4b0c", null
],
 ["solve_result_", "struct_go_1_1_registration_result.html#aca96630179c46e97c522fa12f6aa4ff5", null],
 ["translation_", "struct_go_1_1_registration_result.html#a7f3b4d4a4b1643d5c69ab233464af84e", null]
]
```

Definition at line 1 of file struct\_go\_1\_1\_registration\_result.js.

## 30.1782 doc/html/struct\_go\_1\_1\_rotation\_info.js File Reference

### Variables

- var [struct\\_go\\_1\\_1\\_rotation\\_info](#)

### 30.1782.1 Variable Documentation

#### 30.1782.1.1 var struct\_go\_1\_1\_rotation\_info

##### Initial value:

```
=
[
 ["center_pt_", "struct_go_1_1_rotation_info.html#ae1728f3027b87e4ba67a13cfda4a5bb0", null],
 ["rot_angle_", "struct_go_1_1_rotation_info.html#a3db6ec36284fa5acbac073cf25ec8d62", null],
 ["rot_axis_", "struct_go_1_1_rotation_info.html#a2798f84d097d7cfe10a29384d43f4be10", null]
]
```

Definition at line 1 of file struct\_go\_1\_1\_rotation\_info.js.

## 30.1783 doc/html/struct\_go\_1\_1\_sample\_point\_data.js File Reference

### Variables

- var [struct\\_go\\_1\\_1\\_sample\\_point\\_data](#)

### 30.1783.1 Variable Documentation

#### 30.1783.1.1 var struct\_go\_1\_1\_sample\_point\_data

##### Initial value:

```
=
[
 ["SamplePointData", "struct_go_1_1_sample_point_data.html#a85126d355b2454994df7a2469d5a6175", null],
 ["SamplePointData", "struct_go_1_1_sample_point_data.html#ab68887e84ad4ecc105ba26ea657e8ae5", null],
 ["edge_", "struct_go_1_1_sample_point_data.html#abba15d6076f86102c17f7c4bb857ac13", null],
 ["edge_par_", "struct_go_1_1_sample_point_data.html#ae50653d346158f4c60b5d7d1a4438422", null],
 ["face_", "struct_go_1_1_sample_point_data.html#a465f4888fbb14ceecbcee972212ab7c6", null],
 ["face_par_", "struct_go_1_1_sample_point_data.html#a76366a7f8bc6787cd792f4802a3de4ae", null],
 ["mean_curvature_", "struct_go_1_1_sample_point_data.html#ae6f344ac2642a13b3cc703b439739f37", null],
 ["norm_", "struct_go_1_1_sample_point_data.html#ab59472473d4e3cdcec26b12b2de3c2fd", null],
 ["pos_", "struct_go_1_1_sample_point_data.html#ac7d9f75b41ce1554ff0b13852f5ef1d2", null]
]
```

Definition at line 1 of file struct\_go\_1\_1\_sample\_point\_data.js.

## 30.1784 doc/html/struct\_go\_1\_1\_second\_order\_properties.js File Reference

### Variables

- var [struct\\_go\\_1\\_1\\_second\\_order\\_properties](#)

### 30.1784.1 Variable Documentation

#### 30.1784.1.1 var struct\_go\_1\_1\_second\_order\_properties

##### Initial value:

```
=
[
 ["SecondOrderProperties", "
 struct_go_1_1_second_order_properties.html#acec4d7b81e300b5447c044f4689aa8ca", null],
 ["clear", "struct_go_1_1_second_order_properties.html#a7969d702b4b5d1cb8ae6e6f4654f57e", null],
 ["singularity_info_is_cached", "
 struct_go_1_1_second_order_properties.html#a14908f83790648e620a16d33ecafcf62", null],
 ["singularity_type", "struct_go_1_1_second_order_properties.html#a14b5ad356212956de4932d1f3bcb3957",
 null],
 ["tangent_2d_1", "struct_go_1_1_second_order_properties.html#a2a04d22fd412a880614e77120022ed82", null
],
 ["tangent_2d_2", "struct_go_1_1_second_order_properties.html#ae150a2eb73072238b54e2c3f5e130dff", null
],
 ["tangent_2d_is_cached", "struct_go_1_1_second_order_properties.html#a43338efc4bd449d02635099c6ae9e082",
 null],
 ["tangent_3d", "struct_go_1_1_second_order_properties.html#a186f4ccbf51d36ce342d1ab3cala7eda", null],
 ["tangent_is_oriented", "struct_go_1_1_second_order_properties.html#a77419f5e3c6b905374de860cf8713de6",
 null]
]
```

Definition at line 1 of file struct\_go\_1\_1\_second\_order\_properties.js.

## 30.1785 doc/html/struct\_go\_1\_1\_sing\_box.js File Reference

### Variables

- var [struct\\_go\\_1\\_1\\_sing\\_box](#)

### 30.1785.1 Variable Documentation

#### 30.1785.1.1 var struct\_go\_1\_1\_sing\_box

##### Initial value:

```
=
[
 ["SingBox", "struct_go_1_1_sing_box.html#aa633222dc53a7a26ea0c855c6f804ba9", null],
 ["box_limit_", "struct_go_1_1_sing_box.html#aac7ab7ae3c058c0230309d6e24c76d2a", null],
 ["sing_", "struct_go_1_1_sing_box.html#a1e5835e588eea5b841cc21965d97831e", null]
]
```

Definition at line 1 of file struct\_go\_1\_1\_sing\_box.js.

## 30.1786 doc/html/struct\_go\_1\_1\_sing\_union.js File Reference

### Variables

- var [struct\\_go\\_1\\_1\\_sing\\_union](#)

### 30.1786.1 Variable Documentation

#### 30.1786.1.1 var struct\_go\_1\_1\_sing\_union

##### Initial value:

```
=
[
 ["SingUnion", "struct_go_1_1_sing_union.html#a0cc6c1cea085b254f191595693633466", null],
 ["isInside", "struct_go_1_1_sing_union.html#a59d8fee9a4c801e785eb6c7b779fc224", null],
 ["limit_", "struct_go_1_1_sing_union.html#a2dda72e7981d4b7aa75e3b07afeb3432", null],
 ["singbox_idx_", "struct_go_1_1_sing_union.html#a35dc77ec020eaeef2962f15e78cf5721f", null]
]
```

Definition at line 1 of file struct\_go\_1\_1\_sing\_union.js.

## 30.1787 doc/html/struct\_go\_1\_1\_volume\_adjacency\_info.js File Reference

### Variables

- var [struct\\_go\\_1\\_1\\_volume\\_adjacency\\_info](#)



### 30.1787.1 Variable Documentation

#### 30.1787.1.1 var struct\_go\_1\_1\_volume\_adjacency\_info

##### Initial value:

```
=
[
 ["VolumeAdjacencyInfo", "struct_go_1_1_volume_adjacency_info.html#af0cfdbdd8a7245df440439472f668b50",
 null],
 ["adjacency_found_", "struct_go_1_1_volume_adjacency_info.html#a1456ba763024182354bdc7b0293a8006",
 null],
 ["bd_idx_1_", "struct_go_1_1_volume_adjacency_info.html#a6d9d4da5f8f22f015a36c802615320f8", null],
 ["bd_idx_2_", "struct_go_1_1_volume_adjacency_info.html#a14b33072c37d65cde9c321b836171f48", null],
 ["corner_adjacency_", "struct_go_1_1_volume_adjacency_info.html#a52c4593e325975e3359fa00a34d92f3e",
 null],
 ["corner_failed_", "struct_go_1_1_volume_adjacency_info.html#a33b458ffd3d40e178ff5b2199590532f", null
],
 ["edg_idx_1_", "struct_go_1_1_volume_adjacency_info.html#ab19327268c61359eb382df49d03f9095", null],
 ["edg_idx_2_", "struct_go_1_1_volume_adjacency_info.html#a41ce7795661cc2ff8be7e16bcd260678", null],
 ["same_dir_order_", "struct_go_1_1_volume_adjacency_info.html#a521c75be572f2eec3873445975f31268", null
],
 ["same_orient_edge_", "struct_go_1_1_volume_adjacency_info.html#a555f87a35995e3a6867ee22872a98363",
 null],
 ["same_orient_u_", "struct_go_1_1_volume_adjacency_info.html#a1df26ac9019c9c2371c158ca1567a037", null
],
 ["same_orient_v_", "struct_go_1_1_volume_adjacency_info.html#ae7fc007c2677c37c0a4f7cf7755c974a", null
]
]
```

Definition at line 1 of file struct\_go\_1\_1\_volume\_adjacency\_info.js.

## 30.1788 doc/html/struct\_go\_1\_1cv\_set\_constraint.js File Reference

### Variables

- var [struct\\_go\\_1\\_1cv\\_set\\_constraint](#)

### 30.1788.1 Variable Documentation

#### 30.1788.1.1 var struct\_go\_1\_1cv\_set\_constraint

##### Initial value:

```
=
[
 ["cvSetConstraint", "struct_go_1_1cv_set_constraint.html#a941d33cdfeda3a108633b734725ebd83", null],
 ["cv1_der_", "struct_go_1_1cv_set_constraint.html#ac3cfb977e815ff0b74cda8d69f2598f3", null],
 ["cv1_id_", "struct_go_1_1cv_set_constraint.html#a14db96d1c4a8ee7c54b7715215854e06", null],
 ["cv1_par_", "struct_go_1_1cv_set_constraint.html#a46e18bfc5fc509398e237166bb5abc97", null],
 ["cv2_der_", "struct_go_1_1cv_set_constraint.html#aa8d9cb5e25786b1fa9ef39009b6b0311", null],
 ["cv2_id_", "struct_go_1_1cv_set_constraint.html#aa0816bc54f1d293792cee4d60c6d7b35", null],
 ["cv2_par_", "struct_go_1_1cv_set_constraint.html#ad571b0402688a16f6e83c69dae7c8940", null],
 ["opp_", "struct_go_1_1cv_set_constraint.html#a1596f6a4ffd640d10699ac01d5b179a4", null]
]
```

Definition at line 1 of file struct\_go\_1\_1cv\_set\_constraint.js.

## 30.1789 doc/html/struct\_go\_1\_1go\_iterator\_traits.js File Reference

### Variables

- var [struct\\_go\\_1\\_1go\\_iterator\\_traits](#)

### 30.1789.1 Variable Documentation

#### 30.1789.1.1 var struct\_go\_1\_1go\_iterator\_traits

##### Initial value:

```
=
[
 ["difference_type", "struct_go_1_1go_iterator_traits.html#a60ff0cd0a7bea11271baf9e731e38668", null],
 ["pointer", "struct_go_1_1go_iterator_traits.html#abbc7c10f9ef14171c490acf122004524", null],
 ["reference", "struct_go_1_1go_iterator_traits.html#a5740f64a72fc01a7253afa774d04a402", null],
 ["value_type", "struct_go_1_1go_iterator_traits.html#aca25ce118093b91e456e2982fb419bb8", null]
]
```

Definition at line 1 of file `struct_go_1_1go_iterator_traits.js`.

## 30.1790 doc/html/struct\_go\_1\_1go\_iterator\_traits\_3\_01\_t\_01\_5\_01\_4.js File Reference

### Variables

- var [struct\\_go\\_1\\_1go\\_iterator\\_traits\\_3\\_01\\_t\\_01\\_5\\_01\\_4](#)

### 30.1790.1 Variable Documentation

#### 30.1790.1.1 var struct\_go\_1\_1go\_iterator\_traits\_3\_01\_t\_01\_5\_01\_4

##### Initial value:

```
=
[
 ["difference_type", "
 struct_go_1_1go_iterator_traits_3_01_t_01_5_01_4.html#a5e7177e826be4b2f9f5d096498718dd0", null],
 ["pointer", "struct_go_1_1go_iterator_traits_3_01_t_01_5_01_4.html#a11eab6473cd0d43eb5df24534c0079b4",
 null],
 ["reference", "
 struct_go_1_1go_iterator_traits_3_01_t_01_5_01_4.html#a99b5a8308fd9f139997304d6342b6adb", null],
 ["value_type", "
 struct_go_1_1go_iterator_traits_3_01_t_01_5_01_4.html#a0ee41a5f38a4f3fe31c01d158afda111", null]
]
```

Definition at line 1 of file `struct_go_1_1go_iterator_traits_3_01_t_01_5_01_4.js`.

## 30.1791 doc/html/struct\_go\_1\_lgo\_iterator\_traits\_3\_01const\_01\_t\_01\_5\_01\_4.js File Reference

### Variables

- var [struct\\_go\\_1\\_lgo\\_iterator\\_traits\\_3\\_01const\\_01\\_t\\_01\\_5\\_01\\_4](#)

### 30.1791.1 Variable Documentation

#### 30.1791.1.1 var struct\_go\_1\_lgo\_iterator\_traits\_3\_01const\_01\_t\_01\_5\_01\_4

#### Initial value:

```
=
[
 ["difference_type", "
 struct_go_1_lgo_iterator_traits_3_01const_01_t_01_5_01_4.html#ad52128b11f892bd42c90da2176966d3d", null],
 ["pointer", "
 struct_go_1_lgo_iterator_traits_3_01const_01_t_01_5_01_4.html#a99812d2ef18fb8567a95f3c4a25c64f5", null],
 ["reference", "
 struct_go_1_lgo_iterator_traits_3_01const_01_t_01_5_01_4.html#a4578230c2e878ae79d7b879193a662ba", null],
 ["value_type", "
 struct_go_1_lgo_iterator_traits_3_01const_01_t_01_5_01_4.html#a84cddaaf55d0866599d519ad2c571901", null]
]
```

Definition at line 1 of file `struct_go_1_lgo_iterator_traits_3_01const_01_t_01_5_01_4.js`.

## 30.1792 doc/html/struct\_go\_1\_lpre\_evaluation\_sf.js File Reference

### Variables

- var [struct\\_go\\_1\\_lpre\\_evaluation\\_sf](#)

### 30.1792.1 Variable Documentation

#### 30.1792.1.1 var struct\_go\_1\_lpre\_evaluation\_sf

#### Initial value:

```
=
[
 ["basisvals_u_", "struct_go_1_lpre_evaluation_sf.html#a91dd426c6c5337ed9ed5f56eb322d84e", null],
 ["basisvals_v_", "struct_go_1_lpre_evaluation_sf.html#aabcb7e6e143886ca99b43f4e224bb1e9", null],
 ["deriv_u_", "struct_go_1_lpre_evaluation_sf.html#ac123341fd9aa000ce921267dd70b42cf", null],
 ["deriv_v_", "struct_go_1_lpre_evaluation_sf.html#a7b0efd0fe6f2368a385e8c73fdf7975", null],
 ["gauss_par1_", "struct_go_1_lpre_evaluation_sf.html#a0393353862a463947fde80079d490cce", null],
 ["gauss_par2_", "struct_go_1_lpre_evaluation_sf.html#ace1d1bd2b24ca88754fdeb7ea99bd4d9", null],
 ["left_u_", "struct_go_1_lpre_evaluation_sf.html#a59a9f4aafc91f9e82df45ac7ea6aae60", null],
 ["left_v_", "struct_go_1_lpre_evaluation_sf.html#a89358ac9fc24cade5a89b2a369fde41f", null],
 ["points_", "struct_go_1_lpre_evaluation_sf.html#a23a1a7948ea2467a97973836670e0a2f", null]
]
```

Definition at line 1 of file `struct_go_1_lpre_evaluation_sf.js`.

## 30.1793 doc/html/struct\_go\_1\_1pre\_evaluation\_vol.js File Reference

### Variables

- var [struct\\_go\\_1\\_1pre\\_evaluation\\_vol](#)

### 30.1793.1 Variable Documentation

#### 30.1793.1.1 var struct\_go\_1\_1pre\_evaluation\_vol

#### Initial value:

```
=
[
 ["basisvals_u_", "struct_go_1_1pre_evaluation_vol.html#a5e7c77da854e4afde03dc5ebe4c6d697", null],
 ["basisvals_v_", "struct_go_1_1pre_evaluation_vol.html#a3a7eb7d9e0d2572af2b4a8dc2573ac3b", null],
 ["basisvals_w_", "struct_go_1_1pre_evaluation_vol.html#a5e77e4b4dc72a43f05cb9343b61a2f7c", null],
 ["deriv_u_", "struct_go_1_1pre_evaluation_vol.html#aa89c476b1df345003c8b41ccc01f9649", null],
 ["deriv_v_", "struct_go_1_1pre_evaluation_vol.html#ac26d34eb8669c7a737d0523869c570b9", null],
 ["deriv_w_", "struct_go_1_1pre_evaluation_vol.html#a8d8118d56133deda0a60613a91d6a52b", null],
 ["gauss_par1_", "struct_go_1_1pre_evaluation_vol.html#a9b92c9130ab7bdf6f0a264cafb05a876", null],
 ["gauss_par2_", "struct_go_1_1pre_evaluation_vol.html#a8a9d745ba22ef7316928f5708f7c2218", null],
 ["gauss_par3_", "struct_go_1_1pre_evaluation_vol.html#a12bc6c681b912d957caef1efa6c1fed5", null],
 ["left_u_", "struct_go_1_1pre_evaluation_vol.html#a85883b21ed633ed6d067a28b9025f289", null],
 ["left_v_", "struct_go_1_1pre_evaluation_vol.html#a3df4e48b49594c6845f9d72536d0f387", null],
 ["left_w_", "struct_go_1_1pre_evaluation_vol.html#a69ef813656841df09ebb1b90c2e5df57", null],
 ["points_", "struct_go_1_1pre_evaluation_vol.html#a1b69a5ee428f6e0a0106cb92cdcaedab", null]
]
```

Definition at line 1 of file struct\_go\_1\_1pre\_evaluation\_vol.js.

## 30.1794 doc/html/struct\_go\_1\_1raw\_\_pointer\_\_comp.js File Reference

### Variables

- var [struct\\_go\\_1\\_1raw\\_\\_pointer\\_\\_comp](#)

### 30.1794.1 Variable Documentation

#### 30.1794.1.1 var struct\_go\_1\_1raw\_\_pointer\_\_comp

#### Initial value:

```
=
[
 ["operator()", "struct_go_1_1raw__pointer__comp.html#a49d48faead3948813f516529ac678a47", null]
]
```

Definition at line 1 of file struct\_go\_1\_1raw\_\_pointer\_\_comp.js.

## 30.1795 doc/html/struct\_go\_1\_1side\_constraint.js File Reference

### Variables

- var [struct\\_go\\_1\\_1side\\_constraint](#)

### 30.1795.1 Variable Documentation

#### 30.1795.1.1 var struct\_go\_1\_1side\_constraint

##### Initial value:

```
=
[
 ["constant_term", "struct_go_1_1side_constraint.html#a1c4ce153decf0be47e64049a0f03b997", null],
 ["dim", "struct_go_1_1side_constraint.html#a1952d607d06bc2bbc6b18732cc74babd", null],
 ["factor_", "struct_go_1_1side_constraint.html#a96f76c21dd13a3cbf2cc957b645894c7", null]
]
```

Definition at line 1 of file struct\_go\_1\_1side\_constraint.js.

## 30.1796 doc/html/struct\_go\_1\_1side\_constraint\_set.js File Reference

### Variables

- var [struct\\_go\\_1\\_1side\\_constraint\\_set](#)

### 30.1796.1 Variable Documentation

#### 30.1796.1.1 var struct\_go\_1\_1side\_constraint\_set

##### Initial value:

```
=
[
 ["sideConstraintSet", "struct_go_1_1side_constraint_set.html#ae8dfdf0b9ffe1f86ae409754713ebd22", null],
 ["sideConstraintSet", "struct_go_1_1side_constraint_set.html#a5b126e87b30c488581a328382f536bac", null],
 ["constant_term", "struct_go_1_1side_constraint_set.html#a5a19524ec281c4cec61a09477417ce95", null],
 ["dim_", "struct_go_1_1side_constraint_set.html#aa5df7e5ef2bc0cf501591605a651e431", null],
 ["factor_", "struct_go_1_1side_constraint_set.html#af7f3d4a1dd8b604945f562e958684188", null]
]
```

Definition at line 1 of file struct\_go\_1\_1side\_constraint\_set.js.

## 30.1797 doc/html/struct\_go\_1\_1tp\_tolerances.js File Reference

### Variables

- var [struct\\_go\\_1\\_1tp\\_tolerances](#)

### 30.1797.1 Variable Documentation

#### 30.1797.1.1 var struct\_go\_1\_1tp\_tolerances

##### Initial value:

```
=
[
 ["tpTolerances", "struct_go_1_1tp_tolerances.html#a9e03114371499d3134c72633caf9b82a", null],
 ["tpTolerances", "struct_go_1_1tp_tolerances.html#aalb9ae0e412a9e7817c072a517b3b103", null],
 ["bend", "struct_go_1_1tp_tolerances.html#a961c956c4e5955fc2d627f5d48e06b35", null],
 ["gap", "struct_go_1_1tp_tolerances.html#a62617afddb73267fdc536da40b3932cc", null],
 ["kink", "struct_go_1_1tp_tolerances.html#a587c6924aaf46d129e70defebf9cdf73", null],
 ["neighbour", "struct_go_1_1tp_tolerances.html#a76bf81fb08ac0b307500369354355d9c", null]
]
```

Definition at line 1 of file struct\_go\_1\_1tp\_tolerances.js.

### 30.1798 doc/html/struct\_go\_1\_1tp\_topological\_info.js File Reference

#### Variables

- var [struct\\_go\\_1\\_1tp\\_topological\\_info](#)

#### 30.1798.1 Variable Documentation

##### 30.1798.1.1 var struct\_go\_1\_1tp\_topological\_info

##### Initial value:

```
=
[
 ["BestStatus", "struct_go_1_1tp_topological_info.html#aeccd70e94d436846ae5eb57c4d3805e2", null],
 ["WorstStatus", "struct_go_1_1tp_topological_info.html#a7ef71026619e890f6f54f1d349f780c9", null],
 ["parameters_", "struct_go_1_1tp_topological_info.html#a45ee7f65fe4e8c01afcc093f7cb9088", null],
 ["status_", "struct_go_1_1tp_topological_info.html#ac068190d400ab2256663bcbe8a54ff92", null]
]
```

Definition at line 1 of file struct\_go\_1\_1tp\_topological\_info.js.

### 30.1799 doc/html/struct\_s\_i\_s\_l\_curve.js File Reference

#### Variables

- var [struct\\_s\\_i\\_s\\_l\\_curve](#)

### 30.1799.1 Variable Documentation

#### 30.1799.1.1 var struct\_s\_i\_s\_l\_curve

**Initial value:**

```
=
[
 ["cuopen", "struct_s_i_s_l_curve.html#a6c031e11c3badca1643217b2ec055bc1", null],
 ["ecoeff", "struct_s_i_s_l_curve.html#ad8783e20499cc2468b34e22d322354ad", null],
 ["et", "struct_s_i_s_l_curve.html#a6660a910d17845be1d4e770742bda5e2", null],
 ["icopy", "struct_s_i_s_l_curve.html#a20f717dfda7626d2381b6d852aac3b0", null],
 ["idim", "struct_s_i_s_l_curve.html#a6227a5fa011a17a04876d75673a6188d", null],
 ["ik", "struct_s_i_s_l_curve.html#aa89b9c10e7939117d38d4bb8e0b659b5", null],
 ["ikind", "struct_s_i_s_l_curve.html#a7a1be0fc6cfa3e8df203cdf9f550beda", null],
 ["in", "struct_s_i_s_l_curve.html#abf20462b97b644fdfcbbad9607f8f63a", null],
 ["pbox", "struct_s_i_s_l_curve.html#a9e80a5alc9a09ed706b3cb0bdcc3f5f5", null],
 ["pdir", "struct_s_i_s_l_curve.html#adf101061eed3968878f896139fed9e14", null],
 ["rcoef", "struct_s_i_s_l_curve.html#a61db2260a12ff723c165b0c2a42d2382", null]
]
```

Definition at line 1 of file struct\_s\_i\_s\_l\_curve.js.

## 30.1800 doc/html/struct\_s\_i\_s\_l\_edge.js File Reference

### Variables

- var [struct\\_s\\_i\\_s\\_l\\_edge](#)

### 30.1800.1 Variable Documentation

#### 30.1800.1.1 var struct\_s\_i\_s\_l\_edge

**Initial value:**

```
=
[
 ["iedge", "struct_s_i_s_l_edge.html#aae56bb9770ce1480fd04280c562556bf", null],
 ["ipoint", "struct_s_i_s_l_edge.html#af46a5cdf70574154600e1cf366fffc6", null],
 ["prpt", "struct_s_i_s_l_edge.html#abaa384b7e839e6b170334e4f406942c4", null]
]
```

Definition at line 1 of file struct\_s\_i\_s\_l\_edge.js.

## 30.1801 doc/html/struct\_s\_i\_s\_l\_intcurve.js File Reference

### Variables

- var [struct\\_s\\_i\\_s\\_l\\_intcurve](#)

### 30.1801.1 Variable Documentation

#### 30.1801.1.1 var struct\_s\_i\_s\_l\_intcurve

##### Initial value:

```
=
[
 ["epar1", "struct_s_i_s_l_intcurve.html#ad9c824225a104dcc0bfd21c778aac565", null],
 ["epar2", "struct_s_i_s_l_intcurve.html#aea825119f2d53a057a79777548dc66c6", null],
 ["ipar1", "struct_s_i_s_l_intcurve.html#aec61aedccef3c0c2c0912eef5056e59e", null],
 ["ipar2", "struct_s_i_s_l_intcurve.html#adfe4c670ba809912ae82093c0bd60225", null],
 ["ipoint", "struct_s_i_s_l_intcurve.html#a1da825a8df3499ee26bd5cd65d0439a8", null],
 ["itype", "struct_s_i_s_l_intcurve.html#afaae2668e42179480627d91d97631bf2", null],
 ["pgeom", "struct_s_i_s_l_intcurve.html#acee034e9a8a833d22d2a497558d45663", null],
 ["ppar1", "struct_s_i_s_l_intcurve.html#a6369dff5ebf4f62874ba17544a478885", null],
 ["ppar2", "struct_s_i_s_l_intcurve.html#a1daa3d431470166aa9af7c87caacca68", null],
 ["pretop", "struct_s_i_s_l_intcurve.html#a6102f396edc113322ce79c5fe8822d21", null]
]
```

Definition at line 1 of file struct\_s\_i\_s\_l\_intcurve.js.

### 30.1802 doc/html/struct\_s\_i\_s\_l\_intdat.js File Reference

#### Variables

- var [struct\\_s\\_i\\_s\\_l\\_intdat](#)

### 30.1802.1 Variable Documentation

#### 30.1802.1.1 var struct\_s\_i\_s\_l\_intdat

##### Initial value:

```
=
[
 ["ilist", "struct_s_i_s_l_intdat.html#ab2cb4908e33d27f064f48e4e9a96ea9e", null],
 ["ilmax", "struct_s_i_s_l_intdat.html#a0dacecddb448c7f346d8937ad9369e26", null],
 ["ipmax", "struct_s_i_s_l_intdat.html#a88191ca61bd1eae0a977c6d50fc144fd", null],
 ["ipoint", "struct_s_i_s_l_intdat.html#ae67dc7c802f091b5e23eaa9dalb35f46", null],
 ["vlist", "struct_s_i_s_l_intdat.html#aad63f961bf24bb54af199e9018af1979", null],
 ["vpoint", "struct_s_i_s_l_intdat.html#aedd63d0f344d9481f9d86c3bbffb5e1e", null]
]
```

Definition at line 1 of file struct\_s\_i\_s\_l\_intdat.js.

### 30.1803 doc/html/struct\_s\_i\_s\_l\_intlist.js File Reference

#### Variables

- var [struct\\_s\\_i\\_s\\_l\\_intlist](#)



### 30.1803.1 Variable Documentation

#### 30.1803.1.1 var struct\_s\_i\_s\_l\_intlist

##### Initial value:

```
=
[
 ["ind_first", "struct_s_i_s_l_intlist.html#a483154e00a22842259c47aed57eaea26", null],
 ["ind_last", "struct_s_i_s_l_intlist.html#acb27a34ec1638818320ac5793de6b28c", null],
 ["inumb", "struct_s_i_s_l_intlist.html#a232b0af3a54fd02c78ae6a7478dd151c", null],
 ["itype", "struct_s_i_s_l_intlist.html#ae29a8433d0703e6c774036328f9871a3", null],
 ["pfirst", "struct_s_i_s_l_intlist.html#a9ace40ac92695d31a2d0a6ebdba82318", null],
 ["plast", "struct_s_i_s_l_intlist.html#aaafac3dd38fc98f21d924fb1c733de075", null],
 ["pretop", "struct_s_i_s_l_intlist.html#a12aaf5c24722d9858919016d04d36b01", null]
]
```

Definition at line 1 of file struct\_s\_i\_s\_l\_intlist.js.

## 30.1804 doc/html/struct\_s\_i\_s\_l\_intpt.js File Reference

### Variables

- var struct\_s\_i\_s\_l\_intpt

### 30.1804.1 Variable Documentation

#### 30.1804.1.1 var struct\_s\_i\_s\_l\_intpt

##### Initial value:

```
=
[
 ["adist", "struct_s_i_s_l_intpt.html#a8693e35392ccbd93e6669a2bc4af", null],
 ["curve_dir", "struct_s_i_s_l_intpt.html#acadef71b3d0b7eb0095296f7763ce3c", null],
 ["edge_1", "struct_s_i_s_l_intpt.html#a46dc1e8c213299854359e559636234b9", null],
 ["edge_2", "struct_s_i_s_l_intpt.html#a6a017813f6d6c57e694fdd3332a8a233", null],
 ["epar", "struct_s_i_s_l_intpt.html#a28d46346449e16d0205947d09df8dba9", null],
 ["evaluated", "struct_s_i_s_l_intpt.html#aef820650f42a3fba2c5d01e23280985", null],
 ["geo_data_1", "struct_s_i_s_l_intpt.html#ad4083a64bc31371f487e7943157aa92b", null],
 ["geo_data_2", "struct_s_i_s_l_intpt.html#ae48801801ddc36807cf6885b69c9002d", null],
 ["geo_track_2d_1", "struct_s_i_s_l_intpt.html#acde072bd6f03b86ec5b894fc7f65144d", null],
 ["geo_track_2d_2", "struct_s_i_s_l_intpt.html#a08a00fed068ef573184fc50537408adc", null],
 ["geo_track_3d", "struct_s_i_s_l_intpt.html#a643bd8d87fb2dc110022a57994c42e74", null],
 ["iinter", "struct_s_i_s_l_intpt.html#aac443e9a9052df80f6e7e9f279f659e8", null],
 ["ipar", "struct_s_i_s_l_intpt.html#a52f2220278fcd7a4c02b1ab8762c6fb0", null],
 ["iside_1", "struct_s_i_s_l_intpt.html#a940f033ab6b38b4ecda0f8f606612090", null],
 ["iside_2", "struct_s_i_s_l_intpt.html#af639507dc79ebcda9af979665553e300", null],
 ["left_obj_1", "struct_s_i_s_l_intpt.html#a2c0713f85b7f0da8ad86ecf649dbf700", null],
 ["left_obj_2", "struct_s_i_s_l_intpt.html#a93bcf9e32b6579e98653418d40d8f079", null],
 ["marker", "struct_s_i_s_l_intpt.html#ad5c2518818bd8c1dfdf535ddc983a4ae", null],
 ["no_of_curves", "struct_s_i_s_l_intpt.html#aa666df488d76385c8080a785750274205", null],
 ["no_of_curves_alloc", "struct_s_i_s_l_intpt.html#a2f2bb287bcee40c0818c85ee80f442d0", null],
 ["pcurve", "struct_s_i_s_l_intpt.html#alcbl189b587de134efaa749f4ffd4eea", null],
 ["pnext", "struct_s_i_s_l_intpt.html#aad3e6c2126ca13639265e9c0bbd6124e", null],
 ["right_obj_1", "struct_s_i_s_l_intpt.html#aa0709dea69d6e8d3cdef881fdca74f9", null],
 ["right_obj_2", "struct_s_i_s_l_intpt.html#af5c85d9c7929f4c1f9320c1e307e3ff8", null],
 ["size_1", "struct_s_i_s_l_intpt.html#afe1e88d67af0b704c1260873ab60498e", null],
 ["size_2", "struct_s_i_s_l_intpt.html#a12b741d83ed7441e8e3d7317b86fd3a9", null],
 ["trim", "struct_s_i_s_l_intpt.html#ab6312ad6e3683f8c19abd76dd83632e6", null]
]
```

Definition at line 1 of file struct\_s\_i\_s\_l\_intpt.js.

## 30.1805 doc/html/struct\_s\_i\_s\_l\_intsurf.js File Reference

### Variables

- var [struct\\_s\\_i\\_s\\_l\\_intsurf](#)

### 30.1805.1 Variable Documentation

#### 30.1805.1.1 var struct\_s\_i\_s\_l\_intsurf

##### Initial value:

```
=
[
 ["const_par", "struct_s_i_s_l_intsurf.html#af4e7155eed736b7736ef8525bb1fd7b4", null],
 ["epar", "struct_s_i_s_l_intsurf.html#a91a596112825df965f624f036dd86bcf", null],
 ["ipar", "struct_s_i_s_l_intsurf.html#a7693f0a7f5897a7486e4981b8da7625c", null],
 ["ipoint", "struct_s_i_s_l_intsurf.html#a5f1b0420969e65fdd533234f6013e557", null]
]
```

Definition at line 1 of file struct\_s\_i\_s\_l\_intsurf.js.

## 30.1806 doc/html/struct\_s\_i\_s\_l\_object.js File Reference

### Variables

- var [struct\\_s\\_i\\_s\\_l\\_object](#)

### 30.1806.1 Variable Documentation

#### 30.1806.1.1 var struct\_s\_i\_s\_l\_object

##### Initial value:

```
=
[
 ["c1", "struct_s_i_s_l_object.html#a26da748cce4938c2115bd695d6810f0e", null],
 ["edg", "struct_s_i_s_l_object.html#a3b5ff84d39e82a228481c7806a424ee7", null],
 ["iobj", "struct_s_i_s_l_object.html#a7a8b2316f85a32bc3a5e6b3afc22015f", null],
 ["o1", "struct_s_i_s_l_object.html#aa40c25309fff2ffd852103f2af7acd1f", null],
 ["p1", "struct_s_i_s_l_object.html#ad43b93d19cac19ebe52bcd81e357e8b9", null],
 ["psimple", "struct_s_i_s_l_object.html#a8958916a7eb1f5c2bd6b1a79947e8e23", null],
 ["s1", "struct_s_i_s_l_object.html#a932bc80aef806de501a097405e6c5422", null]
]
```

Definition at line 1 of file struct\_s\_i\_s\_l\_object.js.

## 30.1807 doc/html/struct\_s\_i\_s\_l\_point.js File Reference

### Variables

- var [struct\\_s\\_i\\_s\\_l\\_point](#)

### 30.1807.1 Variable Documentation

#### 30.1807.1.1 var struct\_s\_i\_s\_l\_point

**Initial value:**

```
=
[
 ["ec", "struct_s_i_s_l_point.html#aaab08c937d85029f5d86c469a21bf45d", null],
 ["eccoef", "struct_s_i_s_l_point.html#aa674e75fad6c907c12ade842728d6eb4", null],
 ["icopy", "struct_s_i_s_l_point.html#abb277b543c1cfce82467725ead2544c2", null],
 ["idim", "struct_s_i_s_l_point.html#abef8b5e785c40e2db1b1d5ebdc708287", null],
 ["pbox", "struct_s_i_s_l_point.html#af4a8804cfc17dcac6e545a95226e4d6e", null]
]
```

Definition at line 1 of file struct\_s\_i\_s\_l\_point.js.

## 30.1808 doc/html/struct\_s\_i\_s\_l\_ptedge.js File Reference

### Variables

- var [struct\\_s\\_i\\_s\\_l\\_ptedge](#)

### 30.1808.1 Variable Documentation

#### 30.1808.1.1 var struct\_s\_i\_s\_l\_ptedge

**Initial value:**

```
=
[
 ["pnext", "struct_s_i_s_l_ptedge.html#a52cbd7e01e8626e196f1dce846185514", null],
 ["ppt", "struct_s_i_s_l_ptedge.html#af9a9f4641766f6729fdb34fbec63cca", null]
]
```

Definition at line 1 of file struct\_s\_i\_s\_l\_ptedge.js.

## 30.1809 doc/html/struct\_s\_i\_s\_l\_surf.js File Reference

### Variables

- var [struct\\_s\\_i\\_s\\_l\\_surf](#)

## 30.1809.1 Variable Documentation

### 30.1809.1.1 var struct\_s\_i\_s\_l\_surf

#### Initial value:

```
=
[
 ["cuopen_1", "struct_s_i_s_l_surf.html#aa837a027392850c3d24bbbc819d6ae8b", null],
 ["cuopen_2", "struct_s_i_s_l_surf.html#ae881c7eaa509bf8650b08315e4b691dc", null],
 ["ecoeff", "struct_s_i_s_l_surf.html#af123d50549798ffca464196d4bc6321e", null],
 ["et1", "struct_s_i_s_l_surf.html#aceffa0c397a6adb7aff68fa85830bb48", null],
 ["et2", "struct_s_i_s_l_surf.html#a1bb1b5ad1da78a4bb16b14e047404395", null],
 ["icopy", "struct_s_i_s_l_surf.html#aa626e2db7658d0247f79c931ac118e56", null],
 ["idim", "struct_s_i_s_l_surf.html#a019bcfe66a11fd362682b27769997b", null],
 ["ik1", "struct_s_i_s_l_surf.html#a40d082323ddfaecf33747d9d52db7465", null],
 ["ik2", "struct_s_i_s_l_surf.html#a76c8a3b086cb88f5b447f777bf00575d", null],
 ["ikind", "struct_s_i_s_l_surf.html#ad57401abd890ea793717dfad7d2e397b", null],
 ["in1", "struct_s_i_s_l_surf.html#a5a21e30e8a619feb2a6d7052a64857c9", null],
 ["in2", "struct_s_i_s_l_surf.html#ab35ab0913115ac994be9dd162ec45a0e", null],
 ["pbox", "struct_s_i_s_l_surf.html#ab2988bc8ac978910d425f474aea279f1", null],
 ["pdir", "struct_s_i_s_l_surf.html#a34f3fabfde05d92c10fac011c96b1bf8", null],
 ["rcoef", "struct_s_i_s_l_surf.html#a03e61d32e600e946e2ba0a15ee9181a6", null],
 ["use_count", "struct_s_i_s_l_surf.html#ac378f7abff1662f5b75820dc4de4c2ca", null]
]
```

Definition at line 1 of file struct\_s\_i\_s\_l\_surf.js.

## 30.1810 doc/html/struct\_s\_i\_s\_l\_track.js File Reference

### Variables

- var [struct\\_s\\_i\\_s\\_l\\_track](#)

## 30.1810.1 Variable Documentation

### 30.1810.1.1 var struct\_s\_i\_s\_l\_track

#### Initial value:

```
=
[
 ["eimpli", "struct_s_i_s_l_track.html#a0ea37cfe8212572b39b2e2b39b14bd11", null],
 ["exact", "struct_s_i_s_l_track.html#af27f30a02b7252048f0d6dce8771b2ea", null],
 ["ideg", "struct_s_i_s_l_track.html#ae8ac7e2bcd66072d9b91a1b98721e45e", null],
 ["pcurve_2d_1", "struct_s_i_s_l_track.html#a44e41e7c1d8aa4dd95d2ef2f7f9085da", null],
 ["pcurve_2d_2", "struct_s_i_s_l_track.html#a04c8a8213a0b4e9607b881c8b6702919", null],
 ["pcurve_3d", "struct_s_i_s_l_track.html#a88180d2b237d4041a635394845adb2e4", null],
 ["pretop", "struct_s_i_s_l_track.html#acb3175d087b67041f33bcd4dfcaa68a", null],
 ["psurf_1", "struct_s_i_s_l_track.html#acda4e9721862e8182c74a4c1874b0df7", null],
 ["psurf_2", "struct_s_i_s_l_track.html#acafb7a77a39354dd4fc8d9a9129bc5d4", null],
 ["sing_end", "struct_s_i_s_l_track.html#a28a8ff25f23b0d72dcf47b2b20e7e4e8", null],
 ["sing_start", "struct_s_i_s_l_track.html#a2656d1a08c1ec0e31cdaa8ac44454396", null],
 ["turned", "struct_s_i_s_l_track.html#ae1386a7490108e4a47e7156a8652b3df", null]
]
```

Definition at line 1 of file struct\_s\_i\_s\_l\_track.js.

## 30.1811 doc/html/struct\_s\_i\_s\_l\_trimpar.js File Reference

### Variables

- var [struct\\_s\\_i\\_s\\_l\\_trimpar](#)

#### 30.1811.1 Variable Documentation

##### 30.1811.1.1 var struct\_s\_i\_s\_l\_trimpar

#### Initial value:

```
=
[
 ["parindex", "struct_s_i_s_l_trimpar.html#aff23388e1f39cd924a8848aef7445592", null],
 ["ptindex", "struct_s_i_s_l_trimpar.html#a4bcaeb998bbf486ac7ca78906e3f68fd", null]
]
```

Definition at line 1 of file struct\_s\_i\_s\_l\_trimpar.js.

## 30.1812 doc/html/struct\_s\_i\_s\_lbox.js File Reference

### Variables

- var [struct\\_s\\_i\\_s\\_lbox](#)

#### 30.1812.1 Variable Documentation

##### 30.1812.1.1 var struct\_s\_i\_s\_lbox

#### Initial value:

```
=
[
 ["e2max", "struct_s_i_s_lbox.html#a669492a1c5428ae5d107e54a63052903", null],
 ["e2min", "struct_s_i_s_lbox.html#ae83586cc85d8c03f69f07976bdd48e20", null],
 ["emax", "struct_s_i_s_lbox.html#ae747be1534c35d809a4f6570df2a2645", null],
 ["emin", "struct_s_i_s_lbox.html#a51efab2a865658401a8431c21b150e51", null],
 ["etol", "struct_s_i_s_lbox.html#a96ff59a1a0103bb30f2c5f0d7e326b0b", null],
 ["imax", "struct_s_i_s_lbox.html#a1367186d47af3984e0f08f432e64b09b", null],
 ["imin", "struct_s_i_s_lbox.html#a1c59e9aab0ab10792f48f641b7ee9e3c", null]
]
```

Definition at line 1 of file struct\_s\_i\_s\_lbox.js.

## 30.1813 doc/html/struct\_s\_i\_s\_ldir.js File Reference

### Variables

- var [struct\\_s\\_i\\_s\\_ldir](#)

### 30.1813.1 Variable Documentation

#### 30.1813.1.1 var struct\_s\_i\_s\_ldir

##### Initial value:

```
=
[
 ["aang", "struct_s_i_s_ldir.html#aa799f5940d835206e64ba17ac835a34e", null],
 ["ecoeff", "struct_s_i_s_ldir.html#a9cae019213555ec9a755671f7c12ba98", null],
 ["esmooth", "struct_s_i_s_ldir.html#a97387e3b6a63720b32d8461c373045ec", null],
 ["igtpti", "struct_s_i_s_ldir.html#a890b7a9164c51f468b8e5e5092f01f2a", null]
]
```

Definition at line 1 of file struct\_s\_i\_s\_ldir.js.

### 30.1814 doc/html/structgv\_color.js File Reference

#### Variables

- var [structgv\\_color](#)

#### 30.1814.1 Variable Documentation

##### 30.1814.1.1 var structgv\_color

##### Initial value:

```
=
[
 ["gvColor", "structgv_color.html#ae6f09f93165d1b5ca8cf0bd452c97d6c", null],
 ["gvColor", "structgv_color.html#a3735cfc761ad806cf3b250881e1aa946", null],
 ["gvColor", "structgv_color.html#a0ac1453e77267e2226953a9bec725832", null],
 ["gvColor", "structgv_color.html#a456ade3b05ee26a0fc65d0614f7b7839", null],
 ["operator=", "structgv_color.html#a27a0054113071b8b40361a3c95d70326", null],
 ["rgba", "structgv_color.html#ac897899ae348ba30cfd7bf158ec967f8", null]
]
```

Definition at line 1 of file structgv\_color.js.

### 30.1815 doc/html/structhed\_1\_1\_t\_t\_traits.js File Reference

#### Variables

- var [structhed\\_1\\_1\\_t\\_t\\_traits](#)

### 30.1815.1 Variable Documentation

#### 30.1815.1.1 var structhed\_1\_1\_t\_t\_ltraits

**Initial value:**

```
=
[
 ["real_type", "structhed_1_1_t_t_ltraits.html#a106cb14e7c6eedbd6be052303fa0afc5", null]
]
```

Definition at line 1 of file structhed\_1\_1\_t\_t\_ltraits.js.

## 30.1816 doc/html/structhetriang\_1\_1\_t\_t\_ltraits.js File Reference

### Variables

- var [structhetriang\\_1\\_1\\_t\\_t\\_ltraits](#)

### 30.1816.1 Variable Documentation

#### 30.1816.1.1 var structhetriang\_1\_1\_t\_t\_ltraits

**Initial value:**

```
=
[
 ["real_type", "structhetriang_1_1_t_t_ltraits.html#abf42554b5b7d9797367a39319c888f97", null]
]
```

Definition at line 1 of file structhetriang\_1\_1\_t\_t\_ltraits.js.

## 30.1817 doc/html/structrank\_\_info.js File Reference

### Variables

- var [structrank\\_\\_info](#)

### 30.1817.1 Variable Documentation

#### 30.1817.1.1 var structrank\_\_info

**Initial value:**

```
=
[
 ["antgr", "structrank__info.html#ae804395fa428f28f7697ce7a5eefef0c", null],
 ["antrem", "structrank__info.html#a44eefd512ab6c68a6ed053e6e8b7c3ab", null],
 ["groups", "structrank__info.html#a73fee02c95e70e58861325236285ad37", null],
 ["prio", "structrank__info.html#a6f5b3419ea5bddaa4ffe212f4d794110", null]
]
```

Definition at line 1 of file structrank\_\_info.js.

## 30.1818 doc/html/submat\_8cpp.js File Reference

### Variables

- var [submat\\_8cpp](#)

### 30.1818.1 Variable Documentation

#### 30.1818.1.1 var submat\_8cpp

##### Initial value:

```
=
[
 ["REPORT", "submat_8cpp.html#a787099914a94ed31fa544b985b55752f", null]
]
```

Definition at line 1 of file submat\_8cpp.js.

## 30.1819 doc/html/svd\_8cpp.js File Reference

### Variables

- var [svd\\_8cpp](#)

### 30.1819.1 Variable Documentation

#### 30.1819.1.1 var svd\_8cpp

##### Initial value:

```
=
[
 ["REPORT", "svd_8cpp.html#a787099914a94ed31fa544b985b55752f", null],
 ["WANT_MATH", "svd_8cpp.html#a8335b327d416f961a5dbaa0e058cc515", null],
 ["SVD", "svd_8cpp.html#a5baabc28604c2542162289fb30558361", null],
 ["SVD", "svd_8cpp.html#a27a3125ab1b71b9eb6cc3960ec00c432", null]
]
```

Definition at line 1 of file svd\_8cpp.js.

## 30.1820 doc/html/test\_exc\_8cpp.js File Reference

### Variables

- var [test\\_exc\\_8cpp](#)



### 30.1820.1 Variable Documentation

#### 30.1820.1.1 var test\_\_exc\_8cpp

**Initial value:**

```
=
[
 ["WANT_STREAM", "test__exc_8cpp.html#a6ed6c2d6e68d8f0e7900326b4e30850c", null],
 ["main", "test__exc_8cpp.html#ae66f6b31b5ad750f1fe042a706a4e3d4", null]
]
```

Definition at line 1 of file test\_\_exc\_8cpp.js.

## 30.1821 doc/html/test\_suite\_8h.js File Reference

### Variables

- var [test\\_suite\\_8h](#)

### 30.1821.1 Variable Documentation

#### 30.1821.1.1 var test\_suite\_8h

Definition at line 1 of file test\_suite\_8h.js.

## 30.1822 doc/html/timeutils\_8h.js File Reference

### Variables

- var [timeutils\\_8h](#)

### 30.1822.1 Variable Documentation

#### 30.1822.1.1 var timeutils\_8h

**Initial value:**

```
=
[
 ["getCurrentTime", "timeutils_8h.html#ac0e253cc2c29edaee2bb57317c739afa", null],
 ["systemSleep", "timeutils_8h.html#ac6e7d9797bd3b2a08a074906884a330e", null]
]
```

Definition at line 1 of file timeutils\_8h.js.

## 30.1823 doc/html/tmt1\_8cpp.js File Reference

### Variables

- var [tmt1\\_8cpp](#)

### 30.1823.1 Variable Documentation

#### 30.1823.1.1 var tmt1\_8cpp

##### Initial value:

```
=
[
 ["WANT_STREAM", "tmt1_8cpp.html#a6ed6c2d6e68d8f0e7900326b4e30850c", null],
 ["trymat1", "tmt1_8cpp.html#adf8cd83287d3b4c581af88fd3d9a493f", null]
]
```

Definition at line 1 of file tmt1\_8cpp.js.

## 30.1824 doc/html/tmt2\_8cpp.js File Reference

### Variables

- var [tmt2\\_8cpp](#)

### 30.1824.1 Variable Documentation

#### 30.1824.1.1 var tmt2\_8cpp

##### Initial value:

```
=
[
 ["trymat2", "tmt2_8cpp.html#a9b12f26defc10f30396b41bdbf94d0e1", null]
]
```

Definition at line 1 of file tmt2\_8cpp.js.

## 30.1825 doc/html/tmt3\_8cpp.js File Reference

### Variables

- var [tmt3\\_8cpp](#)

### 30.1825.1 Variable Documentation

#### 30.1825.1.1 var tmt3\_8cpp

**Initial value:**

```
=
[
 ["trymat3", "tmt3_8cpp.html#a61d94bf0a4996b713fb0aa2147037bb3", null]
]
```

Definition at line 1 of file tmt3\_8cpp.js.

## 30.1826 doc/html/tmt4\_8cpp.js File Reference

### Variables

- var [tmt4\\_8cpp](#)

### 30.1826.1 Variable Documentation

#### 30.1826.1.1 var tmt4\_8cpp

**Initial value:**

```
=
[
 ["trymat4", "tmt4_8cpp.html#a3a464a95b2670f8d61a182891b355cfa", null]
]
```

Definition at line 1 of file tmt4\_8cpp.js.

## 30.1827 doc/html/tmt5\_8cpp.js File Reference

### Variables

- var [tmt5\\_8cpp](#)

### 30.1827.1 Variable Documentation

#### 30.1827.1.1 var tmt5\_8cpp

**Initial value:**

```
=
[
 ["Returner0", "tmt5_8cpp.html#a728c253b13aa555c3afc312db4b9b98d", null],
 ["Returner1", "tmt5_8cpp.html#adb2395bf88f189f4e628c7374b89fa21", null],
 ["Returner2", "tmt5_8cpp.html#a8674ddf6f28ee3eb04393f37fe1ced1", null],
 ["Returner3", "tmt5_8cpp.html#a42214b07dd6dc58736d2d339efccc977", null],
 ["Returner4", "tmt5_8cpp.html#a1894eea68488f515763e2edb1446d560", null],
 ["Returner5", "tmt5_8cpp.html#aa734f4973c9b6c6d77d79692e7400216", null],
 ["Returner6", "tmt5_8cpp.html#a70ddb0157f6cd2d8c57db7dd296fdef6", null],
 ["Returner7", "tmt5_8cpp.html#ac6f157c59309471a39cc087c34f3c2b6", null],
 ["trymat5", "tmt5_8cpp.html#a8e00dadf2342ac51cd1b008907a6c7eb", null]
]
```

Definition at line 1 of file tmt5\_8cpp.js.

## 30.1828 doc/html/tmt6\_8cpp.js File Reference

### Variables

- var [tmt6\\_8cpp](#)

### 30.1828.1 Variable Documentation

#### 30.1828.1.1 var tmt6\_8cpp

##### Initial value:

```
=
[
 ["WANT_MATH", "tmt6_8cpp.html#a8335b327d416f961a5dbaa0e058cc515", null],
 ["trymat6", "tmt6_8cpp.html#a46dabf6b2a26e7b353bc6480ae9d2806", null]
]
```

Definition at line 1 of file tmt6\_8cpp.js.

## 30.1829 doc/html/tmt7\_8cpp.js File Reference

### Variables

- var [tmt7\\_8cpp](#)

### 30.1829.1 Variable Documentation

#### 30.1829.1.1 var tmt7\_8cpp

##### Initial value:

```
=
[
 ["trymat7", "tmt7_8cpp.html#a921f1befd9a59c98bc34d2faa1fc1bb5", null]
]
```

Definition at line 1 of file tmt7\_8cpp.js.

## 30.1830 doc/html/tmt8\_8cpp.js File Reference

### Variables

- var [tmt8\\_8cpp](#)

### 30.1830.1 Variable Documentation

#### 30.1830.1.1 var tmt8\_8cpp

**Initial value:**

```
=
[
 ["TestReturn", "tmt8_8cpp.html#abb55317727e34fc56a15f10807764969", null],
 ["Transposer", "tmt8_8cpp.html#a0aa2d8f5c8c1623f60c6f9cd", null],
 ["trymat8", "tmt8_8cpp.html#a4a53bff873b626a6896efaf51e6ffaf5", null]
]
```

Definition at line 1 of file tmt8\_8cpp.js.

## 30.1831 doc/html/tmt9\_8cpp.js File Reference

### Variables

- var [tmt9\\_8cpp](#)

### 30.1831.1 Variable Documentation

#### 30.1831.1.1 var tmt9\_8cpp

**Initial value:**

```
=
[
 ["trymat9", "tmt9_8cpp.html#a0a16e0bed3edd004ec64a20c5baffdaf", null]
]
```

Definition at line 1 of file tmt9\_8cpp.js.

## 30.1832 doc/html/tmt\_8cpp.js File Reference

### Variables

- var [tmt\\_8cpp](#)

### 30.1832.1 Variable Documentation

#### 30.1832.1.1 var tmt\_8cpp

##### Initial value:

```
=
[
 ["PrintCounter", "class_print_counter.html", "class_print_counter"],
 ["WANT_STREAM", "tmt_8cpp.html#a6ed6c2d6e68d8f0e7900326b4e30850c", null],
 ["Clean", "tmt_8cpp.html#a135986a1dbc5462716a7b1534f64465b", null],
 ["Clean", "tmt_8cpp.html#acad9423ada3cd6a3b00d6bb66946d2fc", null],
 ["main", "tmt_8cpp.html#ae66f6b31b5ad750f1fe042a706a4e3d4", null],
 ["PCN", "tmt_8cpp.html#ad8c568418026515ef02d20569113e572", null],
 ["PCZ", "tmt_8cpp.html#a226ffa61ef1b7e208afdf08826bbd3f1", null],
 ["PentiumCheck", "tmt_8cpp.html#a445726f1bc5080fe3cc6e348c1b4f448", null],
 ["Print", "tmt_8cpp.html#a3eb604f5656885c261da566404dd868b", null],
 ["Print", "tmt_8cpp.html#a7bac0dc5d051b772aa72b58127ef92c0", null],
 ["Print", "tmt_8cpp.html#a881d96da4a037f72701c4bd0e39dbc72", null],
 ["Print", "tmt_8cpp.html#a4c79bdbb0b38ca2e6ca10d7667b54f75", null],
 ["Print", "tmt_8cpp.html#aa768a8e9313d7e15627a7d29fd4b28bd", null],
 ["TestTypeAdd", "tmt_8cpp.html#af220b6319631325a16185681a839293e", null],
 ["TestTypeConcat", "tmt_8cpp.html#a7aba478bb251659fde686f24bc42b638", null],
 ["TestTypeKP", "tmt_8cpp.html#a983461223a6bc2fbbd34734521804782", null],
 ["TestTypeMult", "tmt_8cpp.html#a41bf9d5b9037a3740ac96899d524950a", null],
 ["TestTypeOrder", "tmt_8cpp.html#a95508cael1746ce1acc83acbbe5db868a", null],
 ["TestTypeSP", "tmt_8cpp.html#aa3ec56ce85d47861fff5992303183986", null]
]
```

Definition at line 1 of file tmt\_8cpp.js.

### 30.1833 doc/html/tmt\_8h.js File Reference

#### Variables

- var [tmt\\_8h](#)

#### 30.1833.1 Variable Documentation

##### 30.1833.1.1 var tmt\_8h

Definition at line 1 of file tmt\_8h.js.

### 30.1834 doc/html/tmta\_8cpp.js File Reference

#### Variables

- var [tmta\\_8cpp](#)

### 30.1834.1 Variable Documentation

#### 30.1834.1.1 var tmta\_8cpp

**Initial value:**

```
=
[
 ["trymata", "tmta_8cpp.html#a11d5d10e6616b327fb308c8d5a258caa", null]
]
```

Definition at line 1 of file tmta\_8cpp.js.

## 30.1835 doc/html/tmtb\_8cpp.js File Reference

### Variables

- var [tmtb\\_8cpp](#)

### 30.1835.1 Variable Documentation

#### 30.1835.1.1 var tmtb\_8cpp

**Initial value:**

```
=
[
 ["TestClass", "class_test_class.html", "class_test_class"],
 ["trymatb", "tmtb_8cpp.html#a01ad919bd48316e6e589d57066d09a83", null]
]
```

Definition at line 1 of file tmtb\_8cpp.js.

## 30.1836 doc/html/tmtc\_8cpp.js File Reference

### Variables

- var [tmtc\\_8cpp](#)

### 30.1836.1 Variable Documentation

#### 30.1836.1.1 var tmtc\_8cpp

**Initial value:**

```
=
[
 ["trymatc", "tmtc_8cpp.html#ad842b5ce09c4e47733029b1365b4d0fe", null]
]
```

Definition at line 1 of file tmtc\_8cpp.js.

## 30.1837 doc/html/tmtd\_8cpp.js File Reference

### Variables

- var [tmtd\\_8cpp](#)

### 30.1837.1 Variable Documentation

#### 30.1837.1.1 var tmtd\_8cpp

##### Initial value:

```
=
[
 ["Inverter", "tmtd_8cpp.html#ab9af5da7bc1cef2aae2b9dc1920f3e7c", null],
 ["trymatd", "tmtd_8cpp.html#a17794c42f43b5d17dce86ee4966c0670", null]
]
```

Definition at line 1 of file tmtd\_8cpp.js.

## 30.1838 doc/html/tmte\_8cpp.js File Reference

### Variables

- var [tmte\\_8cpp](#)

### 30.1838.1 Variable Documentation

#### 30.1838.1.1 var tmte\_8cpp

##### Initial value:

```
=
[
 ["WANT_MATH", "tmte_8cpp.html#a8335b327d416f961a5dbaa0e058cc515", null],
 ["CheckIsSorted", "tmte_8cpp.html#a8ee61016bcc6a456f1b8adfad9cc0235", null],
 ["trymate", "tmte_8cpp.html#a9551e964abb474a2727342b57ae81f54", null]
]
```

Definition at line 1 of file tmte\_8cpp.js.

## 30.1839 doc/html/tmtf\_8cpp.js File Reference

### Variables

- var [tmtf\\_8cpp](#)



### 30.1839.1 Variable Documentation

#### 30.1839.1.1 var tmtf\_8cpp

**Initial value:**

```
=
[
 ["WANT_MATH", "tmtf_8cpp.html#a8335b327d416f961a5dbaa0e058cc515", null],
 ["trymatf", "tmtf_8cpp.html#a1caf44b5d784f8f179eced8d999509c5", null]
]
```

Definition at line 1 of file tmtf\_8cpp.js.

## 30.1840 doc/html/tmtg\_8cpp.js File Reference

### Variables

- var [tmtg\\_8cpp](#)

### 30.1840.1 Variable Documentation

#### 30.1840.1.1 var tmtg\_8cpp

**Initial value:**

```
=
[
 ["WANT_MATH", "tmtg_8cpp.html#a8335b327d416f961a5dbaa0e058cc515", null],
 ["trymatg", "tmtg_8cpp.html#aa7a5cd73a99176b30f1db1902c42da87", null]
]
```

Definition at line 1 of file tmtg\_8cpp.js.

## 30.1841 doc/html/tmth\_8cpp.js File Reference

### Variables

- var [tmth\\_8cpp](#)

### 30.1841.1 Variable Documentation

#### 30.1841.1.1 var tmth\_8cpp

**Initial value:**

```
=
[
 ["BandFunctions", "tmth_8cpp.html#aclf6e4e77610e4d49a2d8ed653c1fc66", null],
 ["LowerBandFunctions", "tmth_8cpp.html#a4b890058b46519a5ce24e41a758abd0c", null],
 ["SymmetricBandFunctions", "tmth_8cpp.html#aa920dbb90a50a0f408fbc753dcf283d4", null],
 ["trymath", "tmth_8cpp.html#ad0ee11e103cd5efc92d8c7c5a66663e9", null],
 ["UpperBandFunctions", "tmth_8cpp.html#ac9dcaddbf88c61b8431275fbd9240764", null]
]
```

Definition at line 1 of file tmth\_8cpp.js.

## 30.1842 doc/html/tmti\_8cpp.js File Reference

### Variables

- var [tmti\\_8cpp](#)

### 30.1842.1 Variable Documentation

#### 30.1842.1.1 var tmti\_8cpp

##### Initial value:

```
=
[
 ["ReSizeMatrix", "tmti_8cpp.html#a9e44a4958cc08f32780b5349d5c63608", null],
 ["trymati", "tmti_8cpp.html#a060685b2ce0f15fcf2c674ede9d89fda", null],
 ["WillNotConverge", "tmti_8cpp.html#a0f921978f9992fd411af3b20cca52c04", null]
]
```

Definition at line 1 of file tmti\_8cpp.js.

## 30.1843 doc/html/tmtj\_8cpp.js File Reference

### Variables

- var [tmtj\\_8cpp](#)

### 30.1843.1 Variable Documentation

#### 30.1843.1.1 var tmtj\_8cpp

##### Initial value:

```
=
[
 ["trymatj", "tmtj_8cpp.html#a1ea028cc8894408521cb89ab94198275", null]
]
```

Definition at line 1 of file tmtj\_8cpp.js.

## 30.1844 doc/html/tmtk\_8cpp.js File Reference

### Variables

- var [tmtk\\_8cpp](#)

### 30.1844.1 Variable Documentation

#### 30.1844.1.1 var tmtk\_8cpp

**Initial value:**

```
=
[
 ["WANT_STREAM", "tmtk_8cpp.html#a6ed6c2d6e68d8f0e7900326b4e30850c", null],
 ["trymatk", "tmtk_8cpp.html#a2106b6c4deb1061e6812db8b3950aefa", null]
]
```

Definition at line 1 of file tmtk\_8cpp.js.

## 30.1845 doc/html/tmtl\_8cpp.js File Reference

### Variables

- var [tmtl\\_8cpp](#)

### 30.1845.1 Variable Documentation

#### 30.1845.1.1 var tmtl\_8cpp

**Initial value:**

```
=
[
 ["WANT_MATH", "tmtl_8cpp.html#a8335b327d416f961a5dbaa0e058cc515", null],
 ["WANT_STREAM", "tmtl_8cpp.html#a6ed6c2d6e68d8f0e7900326b4e30850c", null],
 ["TestMax", "tmtl_8cpp.html#a84d95be44bbe612e692698848948b3bc", null],
 ["trymatl", "tmtl_8cpp.html#a529c68f2004396c7f4c17b742dba3dbb", null]
]
```

Definition at line 1 of file tmtl\_8cpp.js.

## 30.1846 doc/html/tmtm\_8cpp.js File Reference

### Variables

- var [tmtm\\_8cpp](#)

### 30.1846.1 Variable Documentation

#### 30.1846.1.1 var tmtm\_8cpp

##### Initial value:

```
=
[
 ["WANT_MATH", "tmtm_8cpp.html#a8335b327d416f961a5dbaa0e058cc515", null],
 ["WANT_STREAM", "tmtm_8cpp.html#a6ed6c2d6e68d8f0e7900326b4e30850c", null],
 ["trymatm", "tmtm_8cpp.html#af1aad2210a853ae6eb7ad8a3772f5313", null]
]
```

Definition at line 1 of file tmtm\_8cpp.js.

### 30.1847 doc/html/tp\_joint\_type\_8h.js File Reference

#### Variables

- var [tp\\_joint\\_type\\_8h](#)

### 30.1847.1 Variable Documentation

#### 30.1847.1.1 var tp\_joint\_type\_8h

##### Initial value:

```
=
[
 ["tpJointType", "tp_joint_type_8h.html#a1ff99d2b83bb317e5db42679477c31c2", [
 ["JOINT_G1", "
tp_joint_type_8h.html#a1ff99d2b83bb317e5db42679477c31c2ada229f3349f036f74128f22b91778c5b", null],
 ["JOINT_KINK", "
tp_joint_type_8h.html#a1ff99d2b83bb317e5db42679477c31c2ab918bde030f6bc020b44ec2422b39e25", null],
 ["JOINT_G0", "
tp_joint_type_8h.html#a1ff99d2b83bb317e5db42679477c31c2a7f5a6b29283bce2a3de4770df35da7d9", null],
 ["JOINT_GAP", "
tp_joint_type_8h.html#a1ff99d2b83bb317e5db42679477c31c2a7a64be06e4c446d0ae5d82ef5dce4949", null],
 ["JOINT_DISC", "
tp_joint_type_8h.html#a1ff99d2b83bb317e5db42679477c31c2a70a0cebf99fd902cf435fac73306b6e3", null],
 ["JOINT_NONE", "
tp_joint_type_8h.html#a1ff99d2b83bb317e5db42679477c31c2a45992da186ea931a1bf00b446280b433", null]
]]
]
```

Definition at line 1 of file tp\_joint\_type\_8h.js.

### 30.1848 doc/html/tp\_utils\_8h.js File Reference

#### Variables

- var [tp\\_utils\\_8h](#)

### 30.1848.1 Variable Documentation

#### 30.1848.1.1 var tp\_utils\_8h

##### Initial value:

```
=
[
 ["adjacentEdges", "tp_utils_8h.html#abfcb55015ae56b2f06c8b3df521bfe10", null],
 ["checkContinuity", "tp_utils_8h.html#a58146fa0d1b27a17933f61b9189280de", null]
]
```

Definition at line 1 of file tp\_utils\_8h.js.

## 30.1849 doc/html/transfutils\_8cpp.js File Reference

### Variables

- var [transfutils\\_8cpp](#)

### 30.1849.1 Variable Documentation

#### 30.1849.1.1 var transfutils\_8cpp

##### Initial value:

```
=
[
 ["rotate", "transfutils_8cpp.html#a4bf712e2384f81588420c38cdb216cc3", null],
 ["scale", "transfutils_8cpp.html#ac27f31adada4333fd9e5b1d683cc2dc3", null],
 ["translate", "transfutils_8cpp.html#a56405fd2b22f51d08b371fb188374a40", null],
 ["xrot", "transfutils_8cpp.html#a5175ae8b353bb511616aec812502269d", null],
 ["xrot_eps", "transfutils_8cpp.html#a7b5f2e7bb8d2b8248ec297c6adb57401", null],
 ["xscale", "transfutils_8cpp.html#ac9ealdf8d926e2a6ed51a93a56ea0eea", null],
 ["xtrans", "transfutils_8cpp.html#a42422815616c3bc39a325d5dac156628", null],
 ["yrot", "transfutils_8cpp.html#aa796e5a13fbc734d62adecbelc549960", null],
 ["yrot_eps", "transfutils_8cpp.html#a9437eac6121846f2a2db653a8b6bba3a", null],
 ["yscale", "transfutils_8cpp.html#a058a8b80939f5cad3ee95eeb4ae9d1d", null],
 ["ytrans", "transfutils_8cpp.html#a83f92fc3677e5d1f3c9b863ff919cd94", null],
 ["zrot", "transfutils_8cpp.html#a55c591292392271334a0b01046a92a98", null],
 ["zrot_eps", "transfutils_8cpp.html#ab05217d9b07d153246937544f2e52d94", null],
 ["zscale", "transfutils_8cpp.html#a3ab9a848f6176d9bb377edf88e828b07", null],
 ["ztrans", "transfutils_8cpp.html#ac2f5216f194125aac9bf823b3be05648", null]
]
```

Definition at line 1 of file transfutils\_8cpp.js.

## 30.1850 doc/html/transfutils\_8h.js File Reference

### Variables

- var [transfutils\\_8h](#)

## 30.1850.1 Variable Documentation

### 30.1850.1.1 var transfutils\_8h

#### Initial value:

```
=
[
 ["TRANSFUTILS_H_INCLUDED", "transfutils_8h.html#ac536c5c0243e9fa908f4bc1a905dd6af", null],
 ["rotate", "transfutils_8h.html#a4bf712e2384f81588420c38cdb216cc3", null],
 ["scale", "transfutils_8h.html#ac27f31adada4333fd9e5b1d683cc2dc3", null],
 ["translate", "transfutils_8h.html#a56405fd2b22f51d08b371fb188374a40", null],
 ["xrot", "transfutils_8h.html#a5175ae8b353bb511616aec812502269d", null],
 ["xrot_eps", "transfutils_8h.html#a7b5f2e7bb8d2b8248ec297c6adb57401", null],
 ["xscale", "transfutils_8h.html#ac9ealdf8d926e2a6ed51a93a56ea0eea", null],
 ["xtrans", "transfutils_8h.html#a42422815616c3bc39a325d5dac156628", null],
 ["yrot", "transfutils_8h.html#aa796e5a13fbe734d62adecbe1c549960", null],
 ["yrot_eps", "transfutils_8h.html#a9437eac6121846f2a2db653a8b6bba3a", null],
 ["yscale", "transfutils_8h.html#a058a8b80939f5cadc3ee95eeb4ae9d1d", null],
 ["ytrans", "transfutils_8h.html#a83f92fc3677e5d1f3c9b863ff919cd94", null],
 ["zrot", "transfutils_8h.html#a55c591292392271334a0b01046a92a98", null],
 ["zrot_eps", "transfutils_8h.html#ab05217d9b07d153246937544f2e52d94", null],
 ["zscale", "transfutils_8h.html#a3ab9a848f6176d9bb377edf88e828b07", null],
 ["ztrans", "transfutils_8h.html#ac2f5216f194125aac9bf823b3be05648", null]
]
```

Definition at line 1 of file transfutils\_8h.js.

## 30.1851 doc/html/tstcyclknt\_8c.js File Reference

### Variables

- var [tstcyclknt\\_8c](#)

### 30.1851.1 Variable Documentation

#### 30.1851.1.1 var tstcyclknt\_8c

#### Initial value:

```
=
[
 ["TEST_CYCLIC_KNOTS", "tstcyclknt_8c.html#a77b5bb7a941d583a9aee41d2be40d163", null],
 ["test_cyclic_knots", "tstcyclknt_8c.html#a7c828f6b4278bd340a0fa2be5069b51f", null]
]
```

Definition at line 1 of file tstcyclknt\_8c.js.

## 30.1852 doc/html/ttl\_8h.js File Reference

### Variables

- var [ttl\\_8h](#)

### 30.1852.1 Variable Documentation

#### 30.1852.1.1 var ttl\_8h

Definition at line 1 of file ttl\_8h.js.

## 30.1853 doc/html/ttl\_\_constr\_8h.js File Reference

### Variables

- var [ttl\\_\\_constr\\_8h](#)

### 30.1853.1 Variable Documentation

#### 30.1853.1.1 var ttl\_\_constr\_8h

##### Initial value:

```
=
[
 ["crossesConstraint", "ttl__constr_8h.html#ac06117985e114e1d5012f9b1d3645b4e", null],
 ["findCrossingEdges", "ttl__constr_8h.html#a628efc55cb8a02046cb368360c8945cc", null],
 ["getAtSmallestAngle", "ttl__constr_8h.html#a8abc808bf7d32aa443e8584acea5cf12", null],
 ["insertConstraint", "ttl__constr_8h.html#a6edf6e6b045b5a1b1d0b2ab541b71c7e", null],
 ["isTheConstraint", "ttl__constr_8h.html#a7a284e8f4dbc2ad284fe3c2230846fde", null],
 ["transformToConstraint", "ttl__constr_8h.html#ae09853b3319d1206958eb1f44a4cb7d0", null]
]
```

Definition at line 1 of file ttl\_\_constr\_8h.js.

## 30.1854 doc/html/ttl\_\_util\_8h.js File Reference

### Variables

- var [ttl\\_\\_util\\_8h](#)

### 30.1854.1 Variable Documentation

#### 30.1854.1.1 var ttl\_\_util\_8h

##### Initial value:

```
=
[
 ["createRandomData", "ttl__util_8h.html#a1b19556dbec504e33c7afab9bead26d7", null],
 ["crossProduct2d", "ttl__util_8h.html#a1ad213497a328b60ba326994d7d7501b", null],
 ["orient2dfast", "ttl__util_8h.html#a7f11adbe6f58eb0e3cac2b05ec1f01da", null],
 ["scalarProduct2d", "ttl__util_8h.html#ae91fc49a144917bec417b6b9acfd4c5", null]
]
```

Definition at line 1 of file ttl\_\_util\_8h.js.

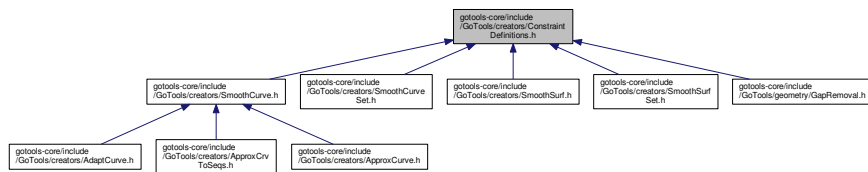








This graph shows which files directly or indirectly include this file:



## Classes

- struct [Go::sideConstraint](#)
- struct [Go::sideConstraintSet](#)

## Namespaces

- [Go](#)

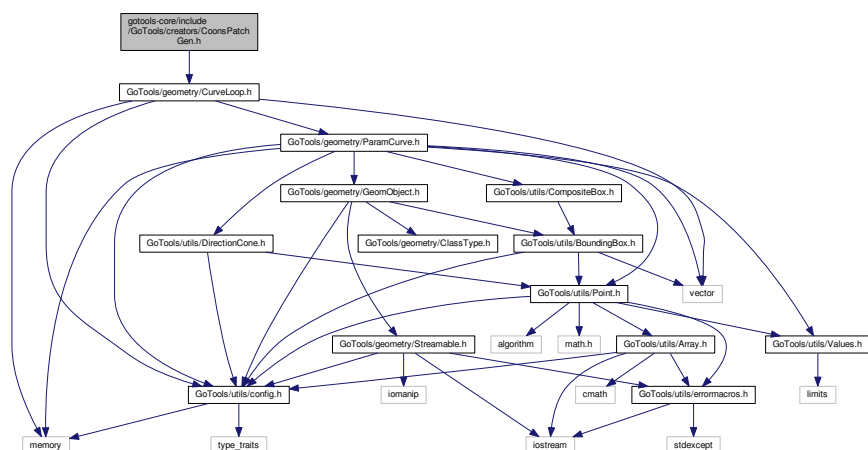
## Typedefs

- typedef struct [Go::sideConstraint](#) [Go::sideConstraint](#)
- typedef struct [Go::sideConstraintSet](#) [Go::sideConstraintSet](#)

## 30.1860 gtools-core/include/GoTools/creators/CoonsPatchGen.h File Reference

```
#include "GoTools/geometry/CurveLoop.h"
```

Include dependency graph for CoonsPatchGen.h:



## Classes

- class [Go::CoonsPatchGen::UnKnownError](#)

*Exception class.*

## Namespaces

- [Go](#)
- [Go::CoonsPatchGen](#)

## Functions

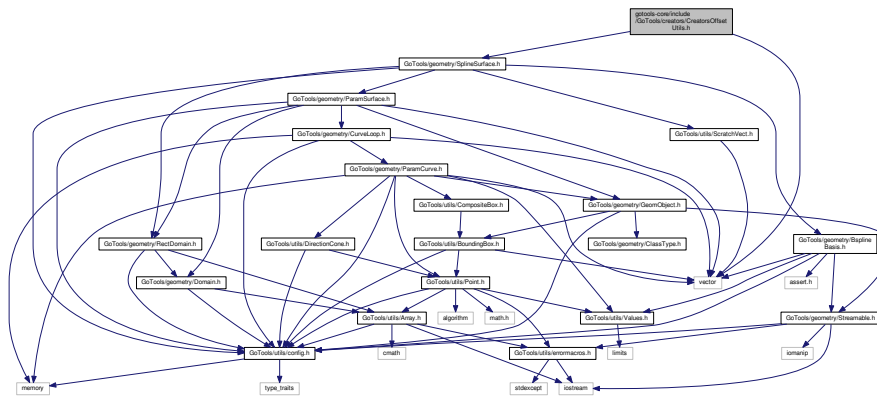
- SplineSurface \* [Go::CoonsPatchGen::createCoonsPatch](#) (const CurveLoop &boundary)
- SplineSurface \* [Go::CoonsPatchGen::createCoonsPatch](#) (std::vector< shared\_ptr< ParamCurve > > &bd\_curves, std::vector< shared\_ptr< ParamCurve > > &cross\_curves, double epsge, double kink\_tol)
- SplineSurface \* [Go::CoonsPatchGen::createCoonsPatch](#) (std::vector< shared\_ptr< SplineCurve > > &bd\_←\_curves, std::vector< shared\_ptr< SplineCurve > > &cross\_curves)
- SplineSurface \* [Go::CoonsPatchGen::createGordonSurface](#) (std::vector< shared\_ptr< SplineCurve > > &mesh\_curves, std::vector< double > &params, int &nmb\_u\_crvs, bool use\_param\_values)
- SplineSurface \* [Go::CoonsPatchGen::createGordonSurface](#) (std::vector< shared\_ptr< SplineCurve > > &mesh\_curves, std::vector< double > &params, int &nmb\_u\_crvs, std::vector< shared\_ptr< SplineCurve > > &cross\_curves, std::vector< int > &cross\_index, bool use\_param\_values=true)
- SplineSurface \* [Go::CoonsPatchGen::doCreateSurface](#) (std::vector< shared\_ptr< SplineCurve > > &mesh\_curves, std::vector< double > &params, int &nmb\_u\_crvs, std::vector< shared\_ptr< SplineCurve > > &cross\_curves, std::vector< int > &cross\_index)
- SplineSurface \* [Go::CoonsPatchGen::loftSurface](#) (std::vector< shared\_ptr< SplineCurve > >::iterator first\_curve, int nmb\_crvs)
- SplineSurface \* [Go::CoonsPatchGen::loftSurface](#) (std::vector< shared\_ptr< SplineCurve > >::iterator first\_curve, std::vector< double >::iterator first\_param, int nmb\_crvs)
- SplineSurface \* [Go::CoonsPatchGen::loftSurface](#) (std::vector< shared\_ptr< SplineCurve > >::iterator first\_curve, std::vector< double >::iterator first\_param, int nmb\_crvs, std::vector< shared\_ptr< SplineCurve > >::iterator first\_cross\_curve, std::vector< int > &cross\_index)
- SplineSurface \* [Go::CoonsPatchGen::tpSurface](#) (const std::vector< shared\_ptr< SplineCurve > > &mesh\_←\_curves, std::vector< double > params, int nmb\_u\_crvs, const std::vector< shared\_ptr< SplineCurve > > &cross\_curves, std::vector< int > &cross\_index)
- void [Go::CoonsPatchGen::splitMeshCurves](#) (std::vector< shared\_ptr< SplineCurve > > &mesh\_curves, std::vector< double > &params, int &nmb\_u\_crvs, std::vector< int > &cross\_index, double epsgeo)
- void [Go::CoonsPatchGen::sortMeshCurves](#) (std::vector< shared\_ptr< SplineCurve > > &mesh\_curves, std::vector< double > &params, int nmb\_u\_crvs, std::vector< int > &cross\_index)
- void [Go::CoonsPatchGen::getCrossTangs](#) (const std::vector< shared\_ptr< SplineCurve > > &curves, std\_←::vector< shared\_ptr< SplineCurve > > &mod\_cross\_curves, double tol1, double tol2)
- void [Go::CoonsPatchGen::addMissingCrossCurves](#) (const std::vector< shared\_ptr< SplineCurve > > &bnd\_←\_curves, std::vector< shared\_ptr< SplineCurve > > &cross\_crvs)
- void [Go::CoonsPatchGen::getTangBlends](#) (std::vector< shared\_ptr< SplineCurve > > &curves, int iedge, std::vector< shared\_ptr< SplineCurve > > &blend\_functions)
- void [Go::CoonsPatchGen::blendcoef](#) (double evecu[], double evecv[], double etang[], int idim, int isign, double \*coef1, double \*coef2)
- void [Go::CoonsPatchGen::hermit](#) (double econd[], int icond, bool hasder1, double astart, double aend, int idim)
- void [Go::CoonsPatchGen::fixCrossEndPts](#) (const std::vector< shared\_ptr< SplineCurve > > &bd\_curves, const std::vector< shared\_ptr< SplineCurve > > &cross\_curves)
- void [Go::CoonsPatchGen::makeLoftParams](#) (std::vector< shared\_ptr< SplineCurve > >::const\_iterator first\_curve, int nmb\_crvs, double param\_length, std::vector< double > &params)
- void [Go::CoonsPatchGen::reparamBoundaryCurve](#) (std::vector< shared\_ptr< SplineCurve > > &curves, double aconst)

## 30.1861 gotools-core/include/GoTools/creators/CreatorsOffsetUtils.h File Reference

```
#include "GoTools/geometry/SplineSurface.h"
```

```
#include <vector>
```

Include dependency graph for CreatorsOffsetUtils.h:



### Namespaces

- [Go](#)
- [Go::OffsetUtils](#)

*Related to the generation of cross tangent curves.*

### Macros

- `#define _OFFSETUTILS_H`

### Functions

- void [Go::OffsetUtils::blend\\_s1421](#) (`const SplineSurface *ps`, `double aoffset`, `int ider`, `const Point &epar`, `int &ifls`, `int &ilft`, `std::vector< Point > &coeffpnt`, `std::vector< Point > &epnt`, `int *jstat`)

### 30.1861.1 Macro Definition Documentation

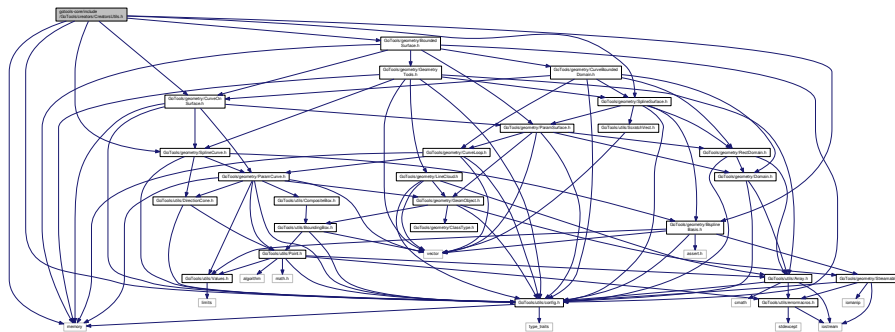
#### 30.1861.1.1 `#define _OFFSETUTILS_H`

Definition at line 85 of file `CreatorsOffsetUtils.h`.

## 30.1862 gotools-core/include/GoTools/creators/CreatorsUtils.h File Reference

```
#include "GoTools/geometry/SplineCurve.h"
#include "GoTools/geometry/SplineSurface.h"
#include "GoTools/geometry/CurveOnSurface.h"
#include "GoTools/geometry/BsplineBasis.h"
#include "GoTools/geometry/BoundedSurface.h"
#include <memory>
#include "GoTools/utils/config.h"
```

Include dependency graph for CreatorsUtils.h:



## Namespaces

- [Go](#)
- [Go::CreatorsUtils](#)

*Related to the generation of cross tangent curves.*

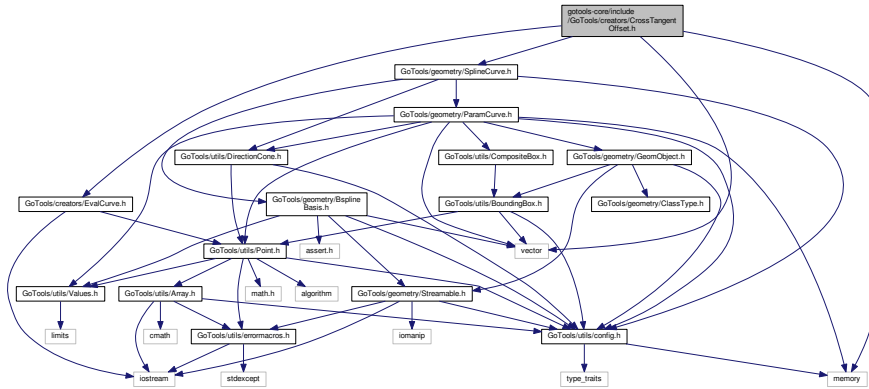
## Functions

- `SplineCurve` `GO_API` \* `Go::CreatorsUtils::getParametricCurve` (`const` `std::vector`< `shared_ptr`< `const` `CurveOnSurface` > > &cv)
- `shared_ptr`< `Go::SplineCurve` > `GO_API` `Go::CreatorsUtils::createCrossTangent` (`const` `Go::CurveOnSurface` &cv)
- `shared_ptr`< `Go::SplineCurve` > `GO_API` `Go::CreatorsUtils::createCrossTangent` (`const` `Go::CurveOnSurface` &cv, `shared_ptr`< `Go::SplineCurve` > `basis_space_cv`, `const` `Go::SplineCurve` \*`cross_cv_ref`, `bool` `appr_offset_cv=true`)
- `std::vector`< `Go::Point` > `GO_API` `Go::CreatorsUtils::projectPoint` (`const` `Go::ParamSurface` \*sf, `bool` `closed_dir_u`, `bool` `closed_dir_v`, `const` `Go::Point` &space\_pt, `double` `epsgeo=1e-04`)
- `shared_ptr`< `Go::Point` > `Go::CreatorsUtils::projectCurvePoint` (`const` `ParamSurface` \*sf, `bool` `closed_dir_u`, `bool` `closed_dir_v`, `const` `Go::ParamCurve` \*space\_cv, `double` `cv_par`, `double` `epsgeo=1e-04`)
- `shared_ptr`< `Go::Point` > `Go::CreatorsUtils::projectCurvePoint` (`const` `SplineSurface` &sf, `bool` `closed_dir_u`, `bool` `closed_dir_v`, `const` `Go::ParamCurve` \*space\_cv, `double` `cv_par`, `double` `epsgeo=1e-04`)
- `void` `GO_API` `Go::CreatorsUtils::fixSeemCurves` (`shared_ptr`< `BoundedSurface` > `bd_sf`, `std::vector`< `shared_ptr`< `CurveOnSurface` > > &loop\_cvs, `bool` `closed_dir_u`, `bool` `closed_dir_v`, `double` `tol`)
- `void` `GO_API` `Go::CreatorsUtils::fixTrimCurves` (`shared_ptr`< `Go::BoundedSurface` > `bd_sf`, `double` `epsgeo`, `double` `_frac=1.0`, `double` `tol=1.0e-3`, `double` `tol2=1.0e-2`, `double` `ang_tol=1.0e-2`)

### 30.1863 gotools-core/include/GoTools/creators/CrossTangentOffset.h File Reference

```
#include "GoTools/creators/EvalCurve.h"
#include "GoTools/geometry/SplineCurve.h"
#include <vector>
#include <memory>
```

Include dependency graph for CrossTangentOffset.h:



#### Classes

- class [Go::CrossTangentOffset](#)

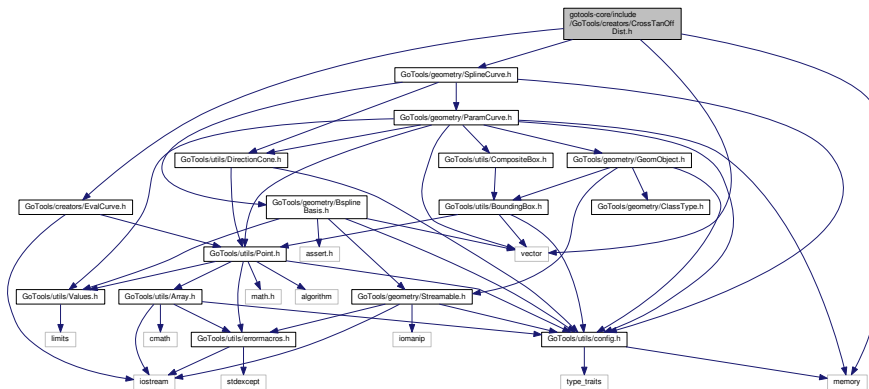
#### Namespaces

- [Go](#)

### 30.1864 gotools-core/include/GoTools/creators/CrossTanOffDist.h File Reference

```
#include "GoTools/creators/EvalCurve.h"
#include "GoTools/geometry/SplineCurve.h"
#include <vector>
#include <memory>
```

Include dependency graph for CrossTanOffDist.h:



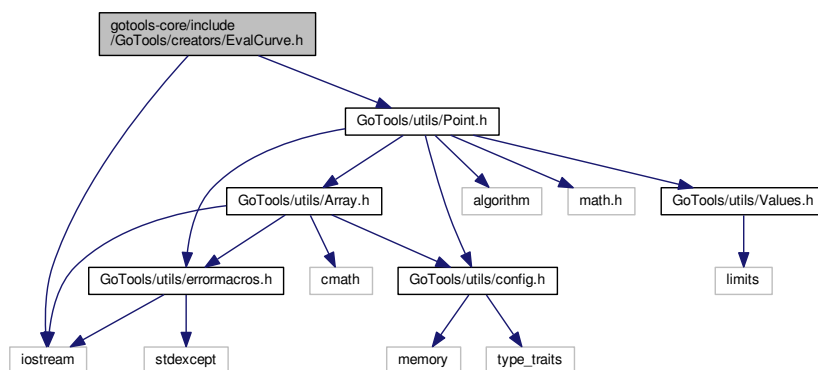




- SplineCurve [GO\\_API](#) \* [Go::CurveCreators::projectSpaceCurve](#) (shared\_ptr< ParamCurve > &space\_cv, shared\_ptr< ParamSurface > &surf, shared\_ptr< Point > &start\_par\_pt, shared\_ptr< Point > &end\_par\_pt, [double epsge](#), [const RectDomain](#) \*domain\_of\_interest=NULL)
- SplineCurve [GO\\_API](#) \* [Go::CurveCreators::liftParameterCurve](#) (shared\_ptr< ParamCurve > &parameter\_cv, shared\_ptr< ParamSurface > &surf, [double epsge](#))  
*Lift the parameter\_cv onto surf.*
- SplineCurve [GO\\_API](#) \* [Go::CurveCreators::createCircle](#) ([Point](#) center, [Point](#) axis, [Point](#) normal, [double](#) radius)  
*Create a circle.*
- shared\_ptr< [Go::SplineCurve](#) > [GO\\_API](#) [Go::CurveCreators::insertParamDomain](#) ([const Go::SplineCurve](#) &cv\_1d, [double](#) knot\_tol=1e-08)  
*Given input of 1-dimensional curve, return the 2-dimensional visualization (x, f(x)).*
- SplineCurve [GO\\_API](#) \* [Go::CurveCreators::offsetCurve](#) ([const SplineCurve](#) &cv, [Point](#) offset\_val)  
*Return the spline curve given by cv(t) + offset\_val.*

## 30.1866 gotools-core/include/GoTools/creators/EvalCurve.h File Reference

```
#include "GoTools/utils/Point.h"
#include <iostream>
Include dependency graph for EvalCurve.h:
```



This graph shows which files directly or indirectly include this file:



### Classes

- class [Go::EvalCurve](#)

### Namespaces

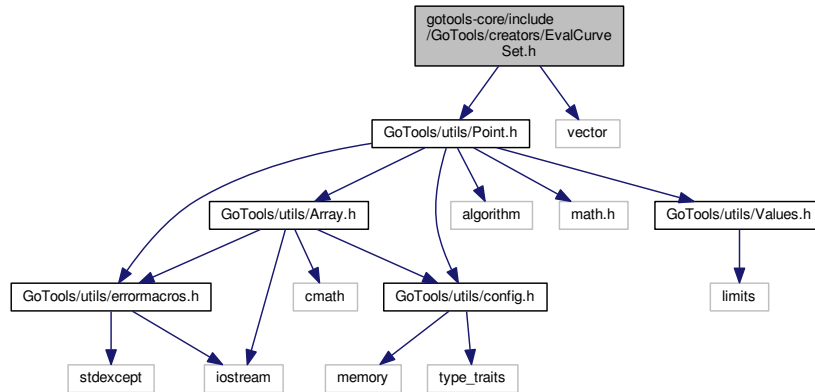
- [Go](#)

### 30.1867 gotools-core/include/GoTools/creators/EvalCurveSet.h File Reference

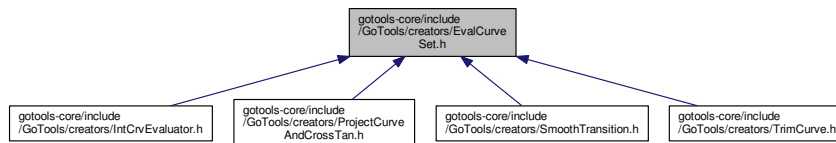
```
#include "GoTools/utils/Point.h"
```

```
#include <vector>
```

Include dependency graph for EvalCurveSet.h:



This graph shows which files directly or indirectly include this file:



#### Classes

- class [Go::EvalCurveSet](#)

#### Namespaces

- [Go](#)

### 30.1868 gotools-core/include/GoTools/creators/EvalParamCurve.h File Reference

```
#include "GoTools/utils/Point.h"
```

```
#include "GoTools/creators/EvalCurve.h"
```

```
#include "GoTools/geometry/SplineCurve.h"
```

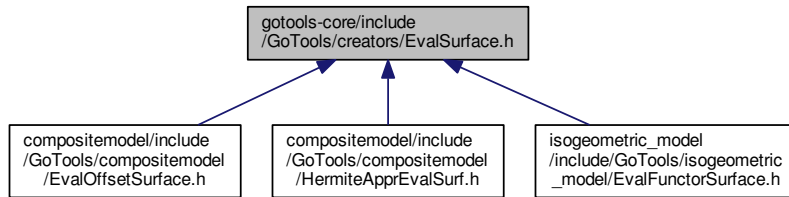
```
#include "GoTools/geometry/SplineSurface.h"
```

```
#include "GoTools/utils/config.h"
```

```
#include <memory>
```



This graph shows which files directly or indirectly include this file:



## Classes

- class [Go::EvalSurface](#)

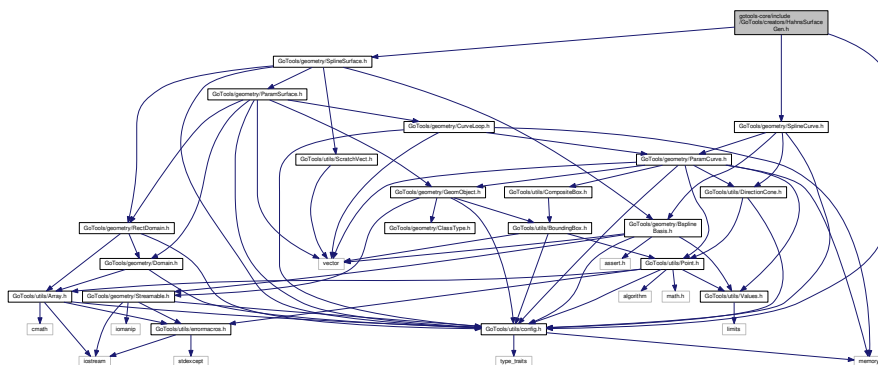
## Namespaces

- [Go](#)

## 30.1870 gotools-core/include/GoTools/creators/HahnsSurfaceGen.h File Reference

```
#include "GoTools/geometry/SplineSurface.h"
#include "GoTools/geometry/SplineCurve.h"
#include "GoTools/utils/config.h"
```

Include dependency graph for HahnsSurfaceGen.h:



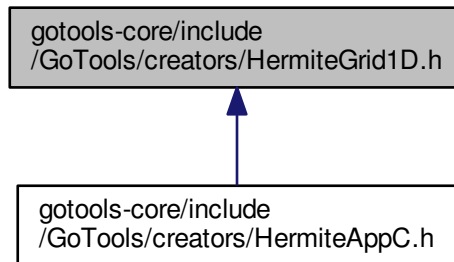
## Namespaces

- [Go](#)
- [Go::HahnsSurfaceGen](#)





This graph shows which files directly or indirectly include this file:



## Classes

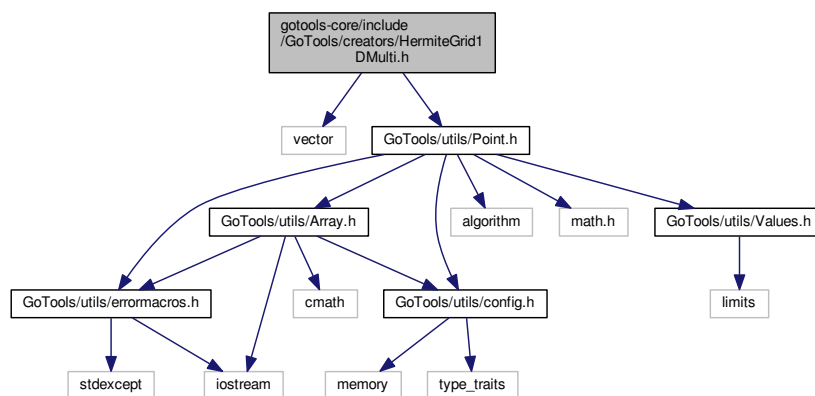
- class [Go::HermiteGrid1D](#)

## Namespaces

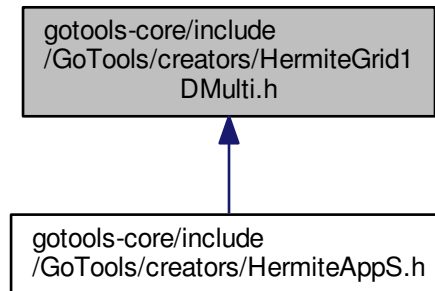
- [Go](#)

## 30.1874 gotools-core/include/GoTools/creators/HermiteGrid1DMulti.h File Reference

```
#include <vector>
#include "GoTools/utils/Point.h"
Include dependency graph for HermiteGrid1DMulti.h:
```



This graph shows which files directly or indirectly include this file:



## Classes

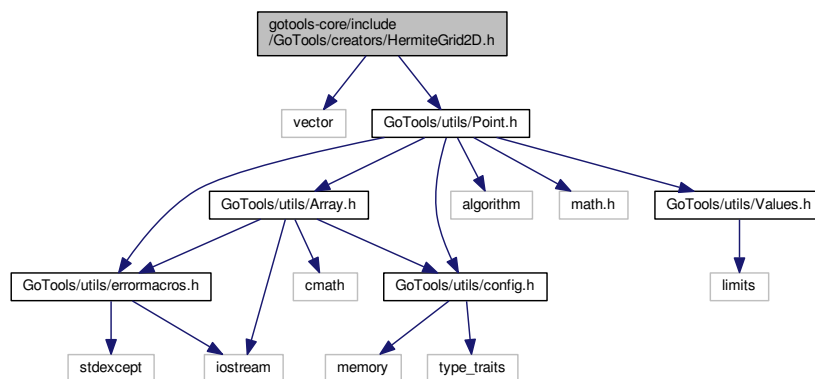
- class [Go::HermiteGrid1DMulti](#)

## Namespaces

- [Go](#)

## 30.1875 gtools-core/include/GoTools/creators/HermiteGrid2D.h File Reference

```
#include <vector>
#include "GoTools/utils/Point.h"
Include dependency graph for HermiteGrid2D.h:
```





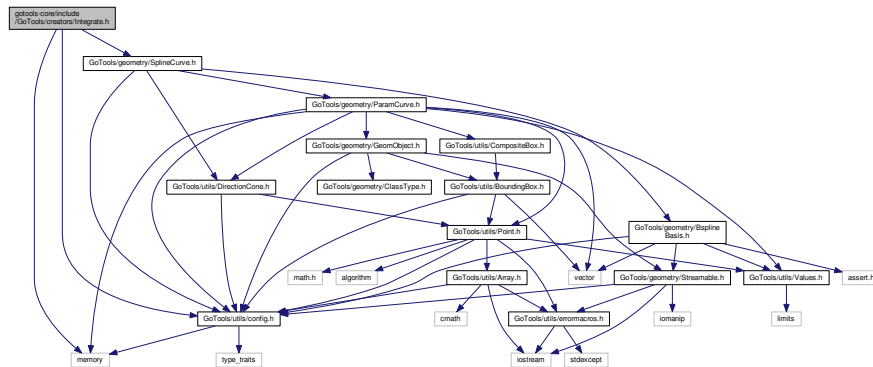


## Namespaces

- [Go](#)

## 30.1877 gotools-core/include/GoTools/creators/Integrate.h File Reference

```
#include <memory>
#include "GoTools/geometry/SplineCurve.h"
#include "GoTools/utils/config.h"
Include dependency graph for Integrate.h:
```



## Namespaces

- [Go](#)

## Functions

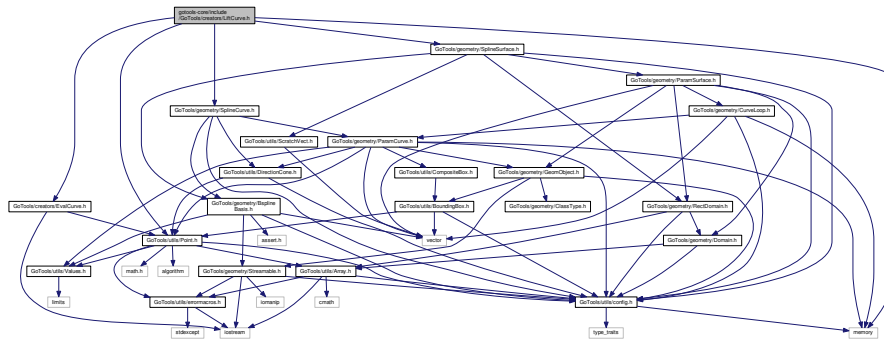
- void [Go::GaussQuadValues](#) (const BsplineBasis &basis, std::vector< double > &parameters, std::vector< double > &par\_weights)
 

*Functions used to compute integrals of inner products of B-splines.*
- void [Go::GaussQuadInner](#) (const BsplineBasis &basis, int ider, double lim1, double lim2, double \*\*\*integral)
- void [Go::GaussQuadInnerFlat](#) (const BsplineBasis &basis, int derivs, int start\_der, int gap, double lim1, double lim2, std::vector< double > &integral)
- void [Go::GaussQuadInner2](#) (const BsplineBasis &basis, int ider, double lim1, double lim2, double \*\*\*integral)
- void [Go::GaussQuadInnerRational](#) (const BsplineBasis &basis, int ider, double lim1, double lim2, shared\_ptr< SplineCurve > bspline\_curve, double \*\*\*integral)

## 30.1878 gotools-core/include/GoTools/creators/LiftCurve.h File Reference

```
#include "GoTools/utils/Point.h"
#include "GoTools/creators/EvalCurve.h"
#include "GoTools/geometry/SplineCurve.h"
#include "GoTools/geometry/SplineSurface.h"
#include <memory>
```

Include dependency graph for LiftCurve.h:



## Classes

- class [Go::LiftCurve](#)

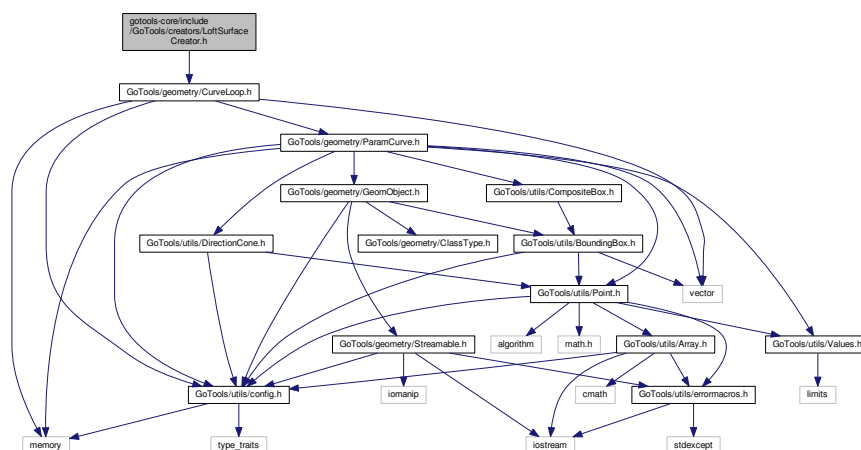
## Namespaces

- [Go](#)

## 30.1879 gotools-core/include/GoTools/creators/LoftSurfaceCreator.h File Reference

```
#include "GoTools/geometry/CurveLoop.h"
```

Include dependency graph for LoftSurfaceCreator.h:



















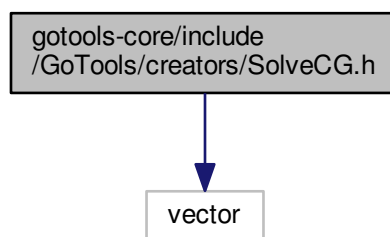
## Namespaces

- [Go](#)

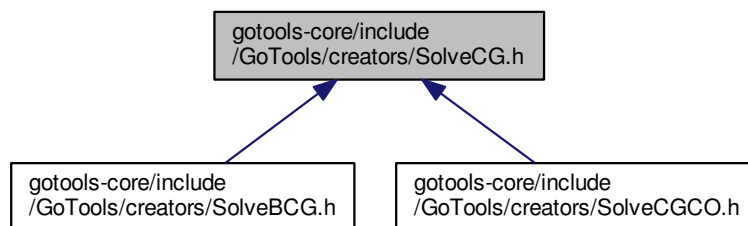
## 30.1890 gotools-core/include/GoTools/creators/SolveCG.h File Reference

```
#include <vector>
```

Include dependency graph for SolveCG.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [Go::SolveCG](#)

## Namespaces

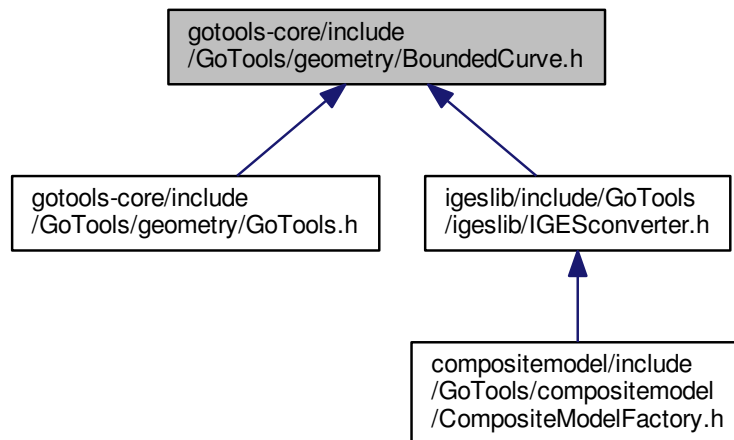
- [Go](#)







This graph shows which files directly or indirectly include this file:



## Classes

- class [Go::BoundedCurve](#)

*A bounded curve. Both parameter values and end points may be given to define the boundaries. Assuming that both points prefer parameter, or both points prefer points. Typically used to bound infinite curves, for instance lines.*

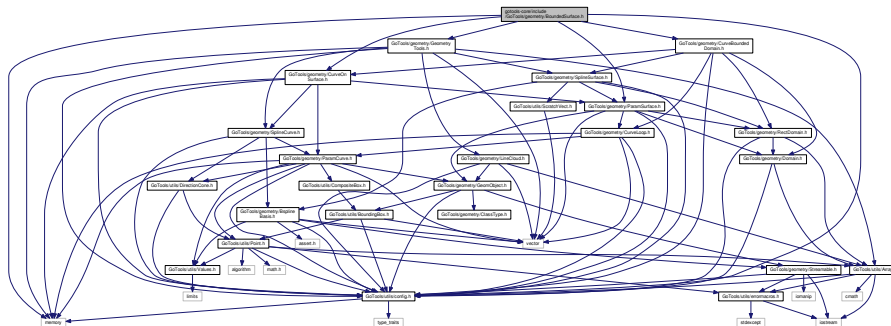
## Namespaces

- [Go](#)

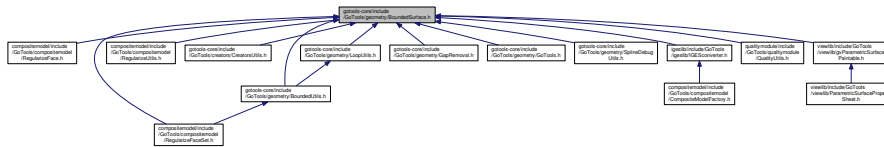
## 30.1896 gotools-core/include/GoTools/geometry/BoundedSurface.h File Reference

```
#include <memory>
#include "GoTools/geometry/ParamSurface.h"
#include "GoTools/geometry/CurveOnSurface.h"
#include "GoTools/geometry/CurveBoundedDomain.h"
#include "GoTools/geometry/GeometryTools.h"
#include "GoTools/utils/config.h"
```

Include dependency graph for `BoundedSurface.h`:



This graph shows which files directly or indirectly include this file:



## Classes

- class [Go::BoundedSurface](#)

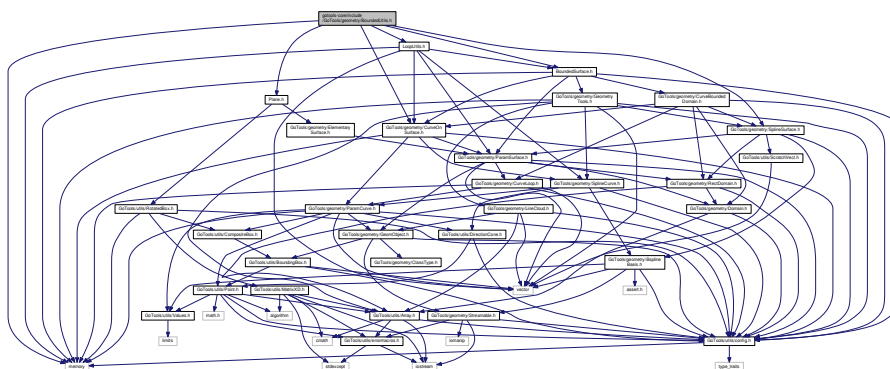
## Namespaces

- [Go](#)

## 30.1897 gotools-core/include/GoTools/geometry/BoundedUtils.h File Reference

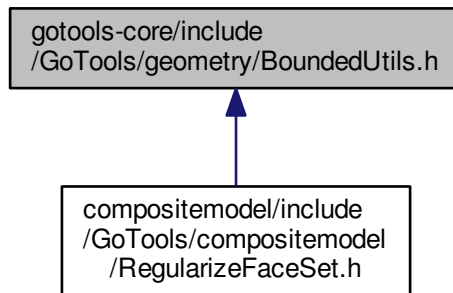
```
#include "BoundedSurface.h"
#include "SplineSurface.h"
#include "CurveOnSurface.h"
#include "LoopUtils.h"
#include "Plane.h"
#include <memory>
```

Include dependency graph for BoundedUtils.h:





This graph shows which files directly or indirectly include this file:



## Namespaces

- [Go](#)
- [Go::BoundedUtils](#)

## Functions

- `std::vector< shared_ptr< CurveOnSurface > >` [Go::BoundedUtils::intersectWithSurface](#) (`CurveOnSurface &curve`, `BoundedSurface &bounded_surf`, `double epsge`)
- `void` [Go::BoundedUtils::intersectWithSurfaces](#) (`std::vector< shared_ptr< CurveOnSurface > > &curves1`, `shared_ptr< BoundedSurface > &bd_sf1`, `std::vector< shared_ptr< CurveOnSurface > > &curves2`, `shared_ptr< BoundedSurface > &bd_sf2`, `double epsge`)
- `std::vector< shared_ptr< CurveOnSurface > >` [Go::BoundedUtils::getPlaneIntersections](#) (`const shared_ptr< ParamSurface > &surf`, `Point point`, `Point normal`, `double epsge`, `shared_ptr< BoundedSurface > &bounded_sf`)
- `std::vector< shared_ptr< CurveOnSurface > >` [Go::BoundedUtils::getCylinderIntersections](#) (`const shared_ptr< ParamSurface > &surf`, `Point point`, `Point axis`, `double radius`, `double epsge`, `shared_ptr< BoundedSurface > &bounded_sf`)
- `void` [Go::BoundedUtils::getSurfaceIntersections](#) (`const shared_ptr< ParamSurface > &surf1`, `const shared_ptr< ParamSurface > &surf2`, `double epsge`, `std::vector< shared_ptr< CurveOnSurface > > &int_cv1`, `shared_ptr< BoundedSurface > &bounded_sf1`, `std::vector< shared_ptr< CurveOnSurface > > &int_cv2`, `shared_ptr< BoundedSurface > &bounded_sf2`)
- `std::vector< shared_ptr< BoundedSurface > >` [Go::BoundedUtils::splitWithPlane](#) (`const shared_ptr< ParamSurface > &surf`, `Point point`, `Point normal`, `double epsge`)
- `std::vector< shared_ptr< BoundedSurface > >` [Go::BoundedUtils::splitBetweenParams](#) (`const shared_ptr< ParamSurface > &surf`, `Point parval1`, `Point parval2`, `double epsge`)
 

*Split a parametric surface between specified parameter values.*
- `std::vector< shared_ptr< BoundedSurface > >` [Go::BoundedUtils::splitBetweenParPairs](#) (`const shared_ptr< ParamSurface > &surf`, `std::vector< std::pair< Point, Point > > parvals`, `double epsge`)
- `std::vector< shared_ptr< CurveOnSurface > >` [Go::BoundedUtils::getTrimCrvsParam](#) (`const shared_ptr< ParamSurface > &surf`, `Point parval1`, `Point parval2`, `double epsge`, `shared_ptr< BoundedSurface > &bounded_sf`)
 

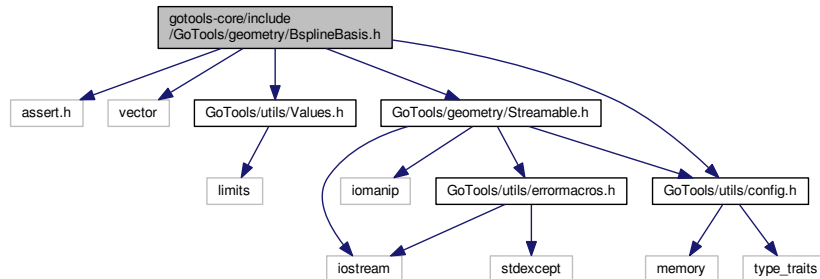
*Get the split curves between specified parameter values.*

- `std::vector< shared_ptr< CurveOnSurface > >` [Go::BoundedUtils::getTrimCrvsPcrv](#) (`const shared_ptr< ParamSurface > &surf`, `shared_ptr< ParamCurve > &pcurve`, `double epsge`, `shared_ptr< BoundedSurface > &bounded_sf`)
- `std::vector< shared_ptr< BoundedSurface > >` [Go::BoundedUtils::trimWithPlane](#) (`const shared_ptr< ParamSurface > &surf`, `Point point`, `Point normal`, `double epsge`)
- `std::vector< shared_ptr< BoundedSurface > >` [Go::BoundedUtils::trimSurfWithSurf](#) (`const shared_ptr< ParamSurface > &sf1`, `const shared_ptr< ParamSurface > &sf2`, `double epsge`)
- `std::vector< std::vector< shared_ptr< BoundedSurface > > >` [Go::BoundedUtils::trimSurfsWithSurfs](#) (`const std::vector< shared_ptr< ParamSurface > > &sfs1`, `const std::vector< shared_ptr< ParamSurface > > &sfs2`, `double epsge`)
- `BoundedSurface *` [Go::BoundedUtils::convertToBoundedSurface](#) (`const SplineSurface &surf`, `double space←_epsilon`)
- `std::vector< std::vector< shared_ptr< CurveOnSurface > > >` [Go::BoundedUtils::getBoundaryLoops](#) (`const BoundedSurface &sf`, `std::vector< shared_ptr< CurveOnSurface > > &part_bnd_cvs`, `double eps`, `int last←_split=-1`)
- `int` [Go::BoundedUtils::checkCurveCoincidence](#) (`shared_ptr< CurveOnSurface > cv1`, `shared_ptr< Curve←OnSurface > cv2`, `double tol`, `bool same_orient`)  
*Help function for getBoundaryLoops.*
- `std::vector< shared_ptr< BoundedSurface > >` [Go::BoundedUtils::createTrimmedSurfs](#) (`std::vector< std←::vector< shared_ptr< CurveOnSurface > > > &loops`, `shared_ptr< ParamSurface > under_sf`, `double epsgeo`)
- `std::vector< shared_ptr< BoundedSurface > >` [Go::BoundedUtils::splitWithTrimSegments](#) (`shared_ptr< BoundedSurface > surf`, `std::vector< shared_ptr< CurveOnSurface > > &bnd_cvs`, `double eps`)
- `std::vector< shared_ptr< BoundedSurface > >` [Go::BoundedUtils::subtractSfPart](#) (`shared_ptr< Bounded←Surface > surf`, `std::vector< shared_ptr< CurveOnSurface > > &bnd_cvs`, `double eps`)
- `std::vector< shared_ptr< CurveOnSurface > >` [Go::BoundedUtils::intersectWithPlane](#) (`shared_ptr< ParamSurface > &surf`, `Point pnt`, `Point normal`, `double geom_tol`)
- `std::vector< shared_ptr< CurveOnSurface > >` [Go::BoundedUtils::intersectWithCylinder](#) (`shared_ptr< ParamSurface > &surf`, `Point pnt`, `Point vec`, `double radius`, `double geom_tol`)
- `void` [Go::BoundedUtils::getIntersectionCurve](#) (`shared_ptr< ParamSurface > &sf1`, `shared_ptr< Param←Surface > &sf2`, `std::vector< shared_ptr< CurveOnSurface > > &int_segments1`, `std::vector< shared_ptr< CurveOnSurface > > &int_segments2`, `double epsge`)
- `void` [Go::BoundedUtils::translateBoundedSurf](#) (`Point trans_vec`, `BoundedSurface &bd_sf`, `double deg_eps`)
- `void` [Go::BoundedUtils::rotateBoundedSurf](#) (`Point rot_axis`, `double alpha`, `BoundedSurface &bf_sf`, `double deg_eps`)
- `void` [Go::BoundedUtils::trimSurfaceKinks](#) (`const BoundedSurface &sf`, `double max_normal_angle`, `std←::vector< double > &g1_disc_u`, `std::vector< double > &g1_disc_v`, `bool compute_g1_disc=true`)
- `int` [Go::BoundedUtils::checkAndFixLoopOrientation](#) (`shared_ptr< BoundedSurface > surf`)
- `shared_ptr< Go::SplineSurface >` [Go::BoundedUtils::makeTrimmedPlane](#) (`shared_ptr< Go::Plane > &plane`, `std::vector< shared_ptr< Go::ParamCurve > > &space_crvs`)
- `void` [Go::BoundedUtils::translatePlaneToCurves](#) (`shared_ptr< Go::Plane > &plane`, `std::vector< shared_←ptr< Go::ParamCurve > > &space_crvs`)
- `void` [Go::BoundedUtils::fixInvalidBoundedSurface](#) (`shared_ptr< Go::BoundedSurface > &bd_sf`, `double max_tol_mult=1.0`)  
*Fix the boundary loops of a surface in case of inconsistencies.*
- `bool` [Go::BoundedUtils::loopsDegenerate](#) (`std::vector< shared_ptr< CurveOnSurface > > &loop`, `double epsgeo`)
- `bool` [Go::BoundedUtils::createMissingParCvs](#) (`Go::BoundedSurface &bd_sf`)
- `bool` [Go::BoundedUtils::createMissingParCvs](#) (`std::vector< Go::CurveLoop > &bd_loops`)

## 30.1898 gotools-core/include/GoTools/geometry/BsplineBasis.h File Reference

```
#include <assert.h>
```

```
#include <vector>
#include "GoTools/utils/Values.h"
#include "GoTools/geometry/Streamable.h"
#include "GoTools/utils/config.h"
Include dependency graph for BsplineBasis.h:
```



This graph shows which files directly or indirectly include this file:



## Classes

- class [Go::BsplineBasis](#)

## Namespaces

- [Go](#)

## Macros

- #define [CHECK](#)(object) (object)->check()

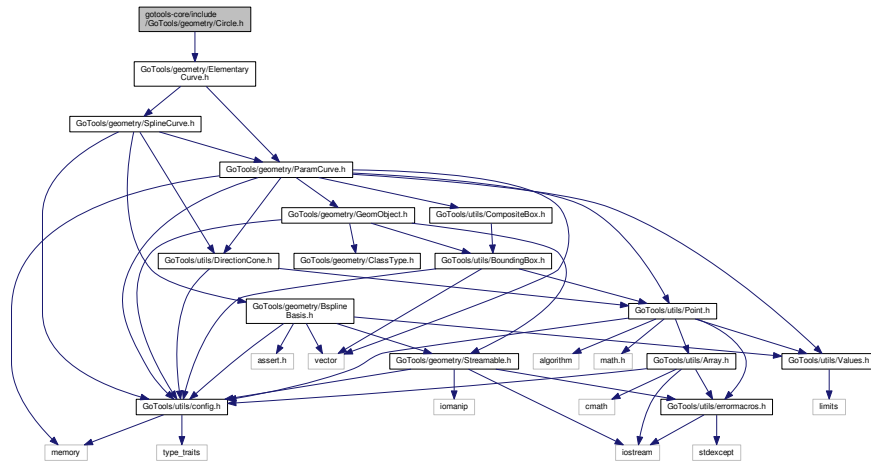
### 30.1898.1 Macro Definition Documentation

#### 30.1898.1.1 #define CHECK( *object* ) (object)->check()

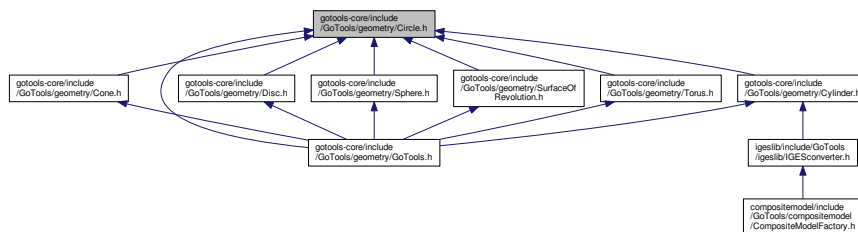
Definition at line 52 of file BsplineBasis.h.

## 30.1899 gotools-core/include/GoTools/geometry/Circle.h File Reference

```
#include "GoTools/geometry/ElementaryCurve.h"
Include dependency graph for Circle.h:
```



This graph shows which files directly or indirectly include this file:



## Classes

- class [Go::Circle](#)

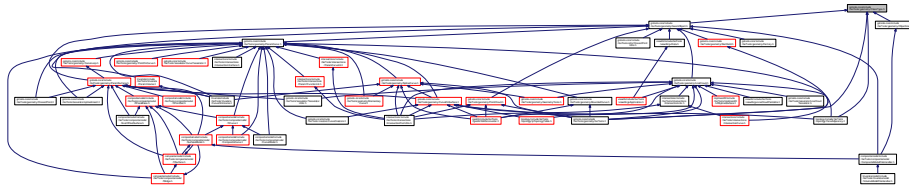
*Class that represents a circle. It is a subclass of [ElementaryCurve](#) and thus has a parametrization.*

## Namespaces

- [Go](#)

## 30.1900 gotools-core/include/GoTools/geometry/ClassType.h File Reference

This graph shows which files directly or indirectly include this file:



### Namespaces

- [Go](#)

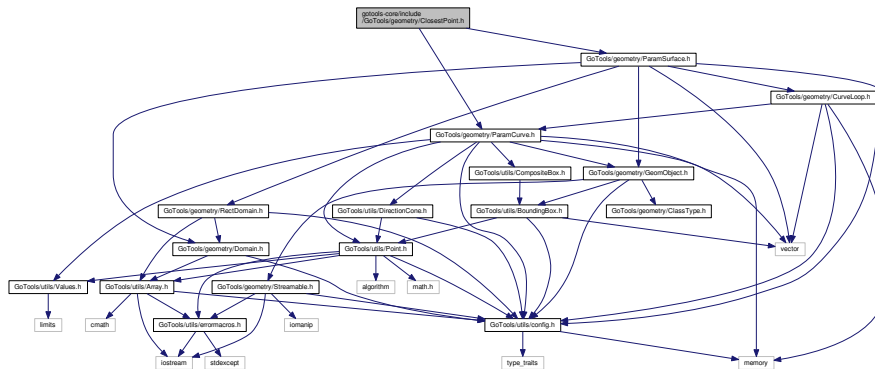
### Enumerations

- `enum Go::ClassType {`  
`Go::Class_Unknown = 0, Go::Class_SplineCurve = 100, Go::Class_CurveOnSurface = 110, Go::Class_↔`  
`CurveOnVolume = 111,`  
`Go::Class_Line = 120, Go::Class_Circle = 130, Go::Class_Ellipse = 140, Go::Class_BoundedCurve = 150,`  
`Go::Class_Hyperbola = 160, Go::Class_Parabola = 170, Go::Class_SplineSurface = 200, Go::Class_↔`  
`BoundedSurface = 210,`  
`Go::Class_SurfaceOnVolume = 211, Go::Class_GoBaryPolSurface = 220, Go::Class_GoHBSplineParam↔`  
`Surface = 230, Go::Class_CompositeSurface = 240,`  
`Go::Class_Plane = 250, Go::Class_Cylinder = 260, Go::Class_SurfaceOfLinearExtrusion = 261, Go::Class_↔`  
`_Sphere = 270,`  
`Go::Class_Cone = 280, Go::Class_Torus = 290, Go::Class_SurfaceOfRevolution = 291, Go::Class_Disc =`  
`292,`  
`Go::Class_LRSplineSurface = 293, Go::Class_TSplineSurface = 294, Go::Class_Go3dsObject = 300, Go::↔`  
`Class_GoHeTriang = 310,`  
`Go::Class_GoSdTriang = 320, Go::Class_GoQuadMesh = 330, Go::Class_GoHybridMesh = 340, Go::↔`  
`Class_ParamTriang = 350,`  
`Go::Class_GoVrmlGeometry = 360, Go::Class_PointCloud = 400, Go::Class_LineCloud = 410, Go::Class_↔`  
`_GoTriangleSets = 500,`  
`Go::Class_RectGrid = 510, Go::Class_SplineVolume = 700, Go::Class_BoundedVolume = 710, Go::Class_↔`  
`_Parallelepiped = 720,`  
`Go::Class_SphereVolume = 721, Go::Class_CylinderVolume = 722, Go::Class_ConeVolume = 723, Go::↔`  
`Class_TorusVolume = 724,`  
`Go::Class_LRSplineVolume = 793 }`

## 30.1901 gotools-core/include/GoTools/geometry/ClosestPoint.h File Reference

```
#include "GoTools/geometry/ParamCurve.h"
#include "GoTools/geometry/ParamSurface.h"
```

Include dependency graph for `ClosestPoint.h`:



## Namespaces

- [Go](#)
- [Go::ClosestPoint](#)

*Namespace for computing closest points.*

## Enumerations

- enum [Go::AlgorithmChoice](#) { [Go::GEOMETRICAL](#), [Go::FUNCTIONAL](#) }

## Functions

- void [Go::ClosestPoint::closestPtCurves2D](#) (ParamCurve \*cv1, ParamCurve \*cv2, double aepsge, double tmin1, double tmax1, double tmin2, double tmax2, double seed1, double seed2, int method, bool quick, double &par1, double &par2, double &dist, Point &ptc1, Point &ptc2, int &istat)
- void [Go::ClosestPoint::closestPtCurves](#) (const ParamCurve \*cv1, const ParamCurve \*cv2, double &par1, double &par2, double &dist, Point &ptc1, Point &ptc2)
- void [Go::ClosestPoint::closestPtCurves](#) (const ParamCurve \*cv1, const ParamCurve \*cv2, double tmin1, double tmax1, double tmin2, double tmax2, double seed1, double seed2, double &par1, double &par2, double &dist, Point &ptc1, Point &ptc2)
- void [Go::ClosestPoint::closestPtCurveSurf](#) (ParamCurve \*pcurve, ParamSurface \*psurf, double aepsge, double astart1, double aend1, RectDomain \*domain, double anext1, double enext2[], double &cpos1, double gpos2[], double &dist, Point &pt\_cv, Point &pt\_su, bool second\_order=false)
- void [Go::ClosestPoint::closestPtCurveSurf](#) (ParamCurve \*pcurve, ParamSurface \*psurf, double aepsge, double astart1, double estart2[], double aend1, double eend2[], double anext1, double enext2[], double &cpos1, double gpos2[], double &dist, Point &pt\_cv, Point &pt\_su, int &istat, bool second\_order=false)
- void [Go::ClosestPoint::closestPtSurfSurfPlane](#) (const std::vector< Point > &epoint, const std::vector< Point > &epnt1, const std::vector< Point > &epnt2, const Point &epar1, const Point &epar2, const ParamSurface \*psurf1, const ParamSurface \*psurf2, double aepsge, std::vector< Point > &gpnt1, std::vector< Point > &gpnt2, Point &gpar1, Point &gpar2, int &jstat, AlgorithmChoice algo=FUNCTIONAL)
- void [Go::ClosestPoint::closestPtSurfSurfPlaneGeometrical](#) (const std::vector< Point > &epoint, const std::vector< Point > &epnt1, const std::vector< Point > &epnt2, const Point &epar1, const Point &epar2, const ParamSurface \*psurf1, const ParamSurface \*psurf2, double aepsge, std::vector< Point > &gpnt1, std::vector< Point > &gpnt2, Point &gpar1, Point &gpar2, int &jstat)
- void [Go::ClosestPoint::closestPtSurfSurfPlaneFunctional](#) (const std::vector< Point > &epoint, const std::vector< Point > &epnt1, const std::vector< Point > &epnt2, const Point &epar1, const Point &epar2, const ParamSurface \*psurf1, const ParamSurface \*psurf2, double aepsge, std::vector< Point > &gpnt1, std::vector< Point > &gpnt2, Point &gpar1, Point &gpar2, int &jstat)



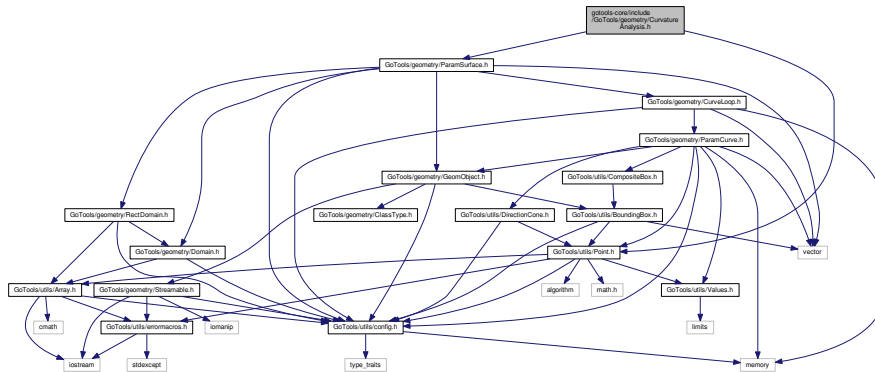






## 30.1907 gotools-core/include/GoTools/geometry/CurvatureAnalysis.h File Reference

```
#include "GoTools/geometry/ParamSurface.h"
#include "GoTools/utils/Point.h"
Include dependency graph for CurvatureAnalysis.h:
```



### Namespaces

- [Go](#)
- [Go::CurvatureAnalysis](#)

### Functions

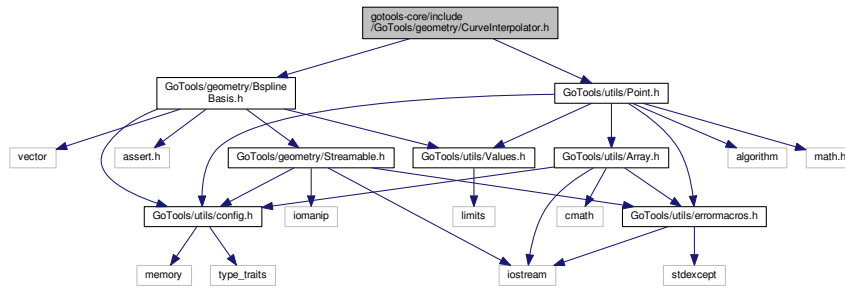
- void [Go::CurvatureAnalysis::computeFirstFundamentalForm](#) (const ParamSurface &sf, double u, double v, int derivs, std::vector< double > &form)
- void [Go::CurvatureAnalysis::computeSecondFundamentalForm](#) (const ParamSurface &sf, double u, double v, double form1[3], double form2[3])
- void [Go::CurvatureAnalysis::curvatures](#) (const ParamSurface &sf, double u, double v, double &K, double &H)
- void [Go::CurvatureAnalysis::principalCurvatures](#) (const ParamSurface &sf, double u, double v, double &k1, Point &d1, double &k2, Point &d2)
- void [Go::CurvatureAnalysis::minimalCurvatureRadius](#) (const ParamSurface &sf, double tolerance, double &mincurv, double &pos\_u, double &pos\_v, double degenerate\_eps, double curv\_tol=1.0e-3)
- void [Go::CurvatureAnalysis::evaluateMinCurvatureRadius](#) (const ParamSurface &sf, double start\_u, double end\_u, double start\_v, double end\_v, double tolerance, std::vector< double > &param\_u, std::vector< double > &param\_v, std::vector< std::vector< double > > &curvs, double &mincurv, double &minpos\_u, double &minpos\_v, bool initialize)

### 30.1907.1 Detailed Description

Curvature analysis related to surfaces. Functions computing fundamental forms and curvature.



Include dependency graph for CurveInterpolator.h:



## Namespaces

- [Go](#)
- [Go::CurveInterpolator](#)

Curve interpolation functionality not adapted to the *Interpolator* class.

## Functions

- SplineCurve \* [Go::CurveInterpolator::regularInterpolation](#) (const BsplineBasis &basis, std::vector< double > &par, std::vector< double > &points, int dimension, bool rational, std::vector< double > &weights)

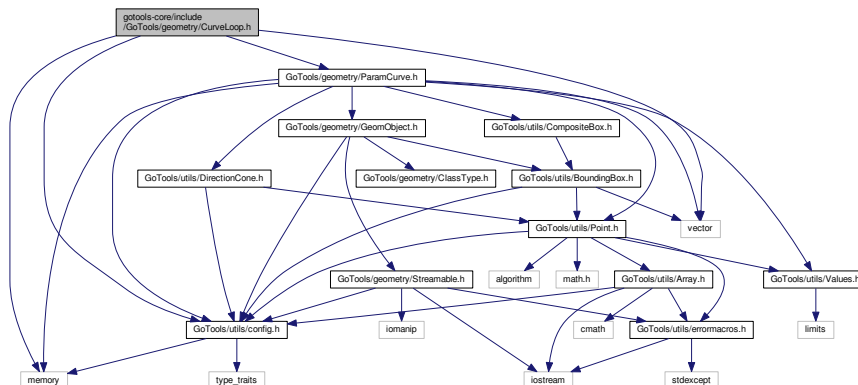
## 30.1910 gtools-core/include/GoTools/geometry/CurveLoop.h File Reference

```

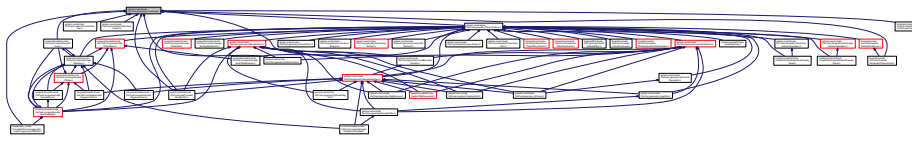
#include <memory>
#include <vector>
#include "GoTools/utills/config.h"
#include "GoTools/geometry/ParamCurve.h"

```

Include dependency graph for CurveLoop.h:



This graph shows which files directly or indirectly include this file:



**Classes**

- class [Go::CurveLoop](#)

**Namespaces**

- [Go](#)

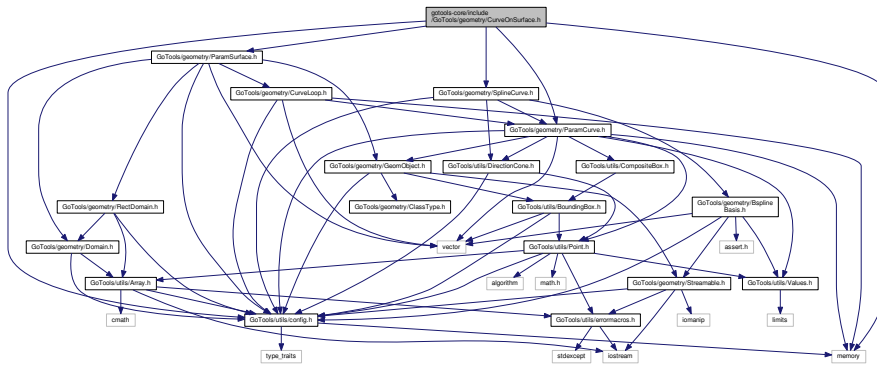
**Functions**

- `template<class PtrToCurveType >`  
`double Go::computeLoopGap (const std::vector< PtrToCurveType > &curves)`  
*Computes the largest gap in the loop specified by the vector of curves.*

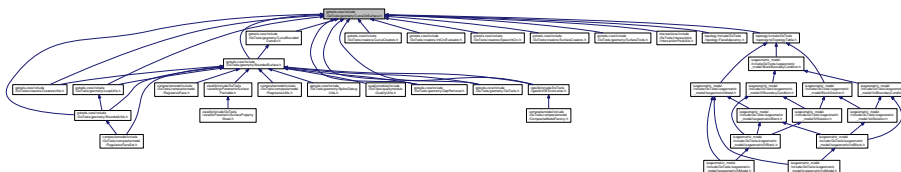
**30.1911 gotools-core/include/GoTools/geometry/CurveOnSurface.h File Reference**

```
#include "GoTools/geometry/ParamSurface.h"
#include <memory>
#include "GoTools/geometry/ParamCurve.h"
#include "GoTools/geometry/SplineCurve.h"
#include "GoTools/utils/config.h"
```

Include dependency graph for CurveOnSurface.h:



This graph shows which files directly or indirectly include this file:





## Classes

- class [Go::Cylinder](#)

Class that represents a cylinder. It is a subclass of [ElementarySurface](#), and thus has a parametrization and is non-selfintersecting.

## Namespaces

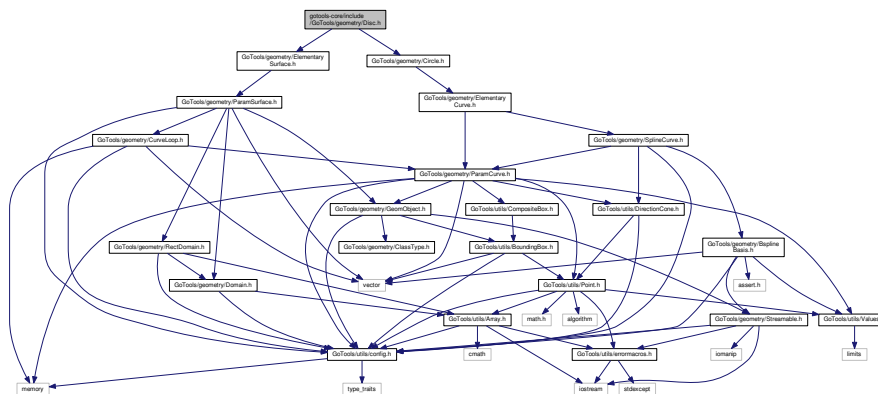
- [Go](#)

## 30.1913 gotools-core/include/GoTools/geometry/Disc.h File Reference

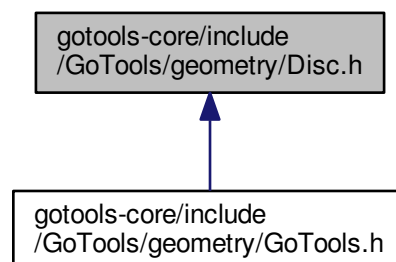
```
#include "GoTools/geometry/ElementarySurface.h"
```

```
#include "GoTools/geometry/Circle.h"
```

Include dependency graph for Disc.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [Go::Disc](#)

Class that represents a circular disc. It is a subclass of [ElementarySurface](#), and has a natural parametrization by polar coordinates in terms of a radius  $r$  and angle  $v$ :  $\mathbf{p}(r, v) = \mathbf{C} + r((\cos v) \mathbf{x} + (\sin v) \mathbf{y})$ , where  $\mathbf{C}$  is the centre position vector and  $\mathbf{x}$  and  $\mathbf{y}$  are the (local) axes. The parametrization is bounded by:  $0 \leq r \leq R$  and  $0 \leq v \leq 2\pi$ , where  $R$  is the disc radius. The dimension is 2 or 3.

## Namespaces

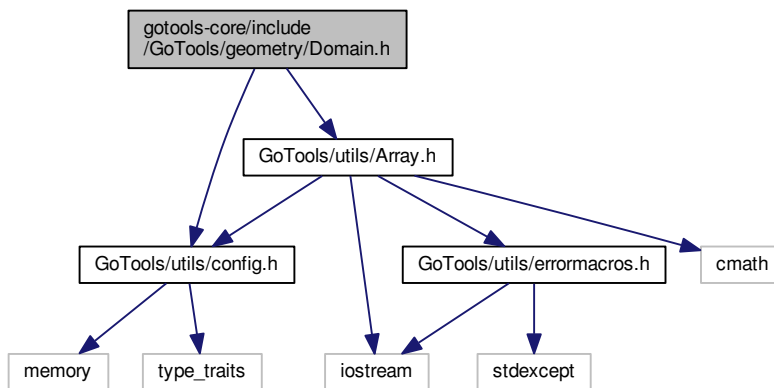
- [Go](#)

## 30.1914 gotools-core/include/GoTools/geometry/Domain.h File Reference

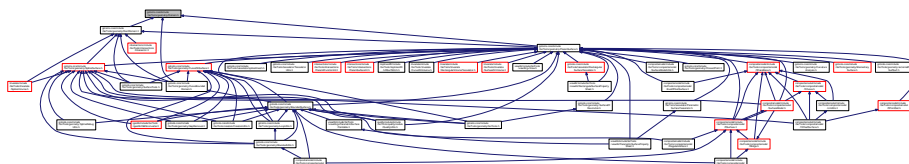
```
#include "GoTools/utils/config.h"
```

```
#include "GoTools/utils/Array.h"
```

Include dependency graph for Domain.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [Go::Domain](#)







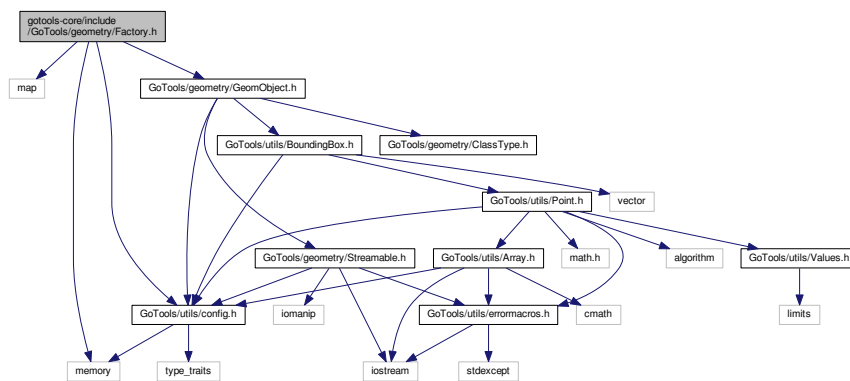




## 30.1921 gotools-core/include/GoTools/geometry/Factory.h File Reference

```
#include <map>
#include <memory>
#include "GoTools/geometry/GeomObject.h"
#include "GoTools/utils/config.h"
```

Include dependency graph for Factory.h:



## Classes

- class [Go::Creator](#)
- class [Go::ConcreteCreator< T >](#)
- class [Go::Factory](#)
- class [Go::Registrar< T >](#)

## Namespaces

- [Go](#)

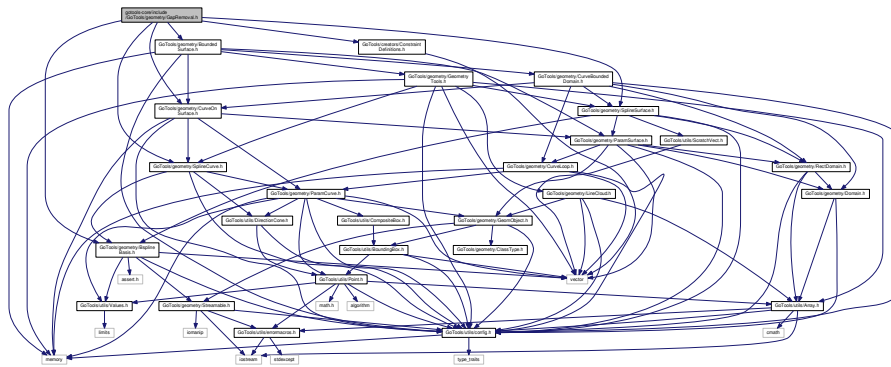
## Functions

- `template<class T >`  
void [Go::Register](#) ()

## 30.1922 gotools-core/include/GoTools/geometry/GapRemoval.h File Reference

```
#include "GoTools/geometry/SplineCurve.h"
#include "GoTools/geometry/SplineSurface.h"
#include "GoTools/geometry/CurveOnSurface.h"
#include "GoTools/geometry/BsplineBasis.h"
#include "GoTools/geometry/BoundedSurface.h"
#include "GoTools/creators/ConstraintDefinitions.h"
```

Include dependency graph for GapRemoval.h:



## Namespaces

- [Go](#)
- [Go::GapRemoval](#)

*Functionality for removal of gaps between two adjacent surfaces of various types.*

## Functions

- void [Go::GapRemoval::removeGapSpline](#) (shared\_ptr< SplineSurface > &srf1, shared\_ptr< CurveOnSurface > &bd\_cv1, double start1, double end1, shared\_ptr< SplineSurface > &srf2, shared\_ptr< CurveOnSurface > &bd\_cv2, double start2, double end2, Point vertex1, Point vertex2, double epsge, bool \*same\_orientation=NULL)
- double [Go::GapRemoval::removeGapTrim](#) (shared\_ptr< CurveOnSurface > &bd\_cv1, double start1, double end1, shared\_ptr< CurveOnSurface > &bd\_cv2, double start2, double end2, Point vertex1, Point vertex2, double epsge)
- bool [Go::GapRemoval::removeGapSplineTrim](#) (shared\_ptr< SplineSurface > &srf1, std::vector< shared\_ptr< CurveOnSurface > > &bd\_cv1, std::vector< double > start1, std::vector< double > end1, std::vector< shared\_ptr< CurveOnSurface > > &bd\_cv2, std::vector< double > start2, std::vector< double > end2, Point vertex1, Point vertex2, double epsge)
- void [Go::GapRemoval::modifySplineSf](#) (shared\_ptr< ParamSurface > &srf1, std::vector< shared\_ptr< CurveOnSurface > > &bd\_cv1, std::vector< double > start1, std::vector< double > end1, shared\_ptr< ParamSurface > &srf2, std::vector< shared\_ptr< CurveOnSurface > > &bd\_cv2, std::vector< double > start2, std::vector< double > end2, double epsge)
- void [Go::GapRemoval::modifySplines](#) (shared\_ptr< ParamSurface > &srf1, std::vector< shared\_ptr< CurveOnSurface > > &bd\_cv1, std::vector< double > &start1, std::vector< double > &end1, shared\_ptr< ParamSurface > &srf2, std::vector< shared\_ptr< CurveOnSurface > > &bd\_cv2, std::vector< double > &start2, std::vector< double > &end2, std::vector< Point > &vertex, double epsge)
- void [Go::GapRemoval::removeGapSpline2](#) (std::vector< shared\_ptr< CurveOnSurface > > &bd\_cv1, std::vector< double > &start1, std::vector< double > &end1, std::vector< shared\_ptr< CurveOnSurface > > &bd\_cv2, std::vector< double > &start2, std::vector< double > &end2, std::vector< Point > &vertex, double epsge)
- std::vector< shared\_ptr< sideConstraintSet > > [Go::GapRemoval::getCoefConstraints](#) (shared\_ptr< SplineCurve > &crv1, int idx1, shared\_ptr< SplineCurve > &crv2, int idx2, double tol)
- shared\_ptr< SplineCurve > [Go::GapRemoval::replaceCurvePiece](#) (shared\_ptr< SplineCurve > crv, shared\_ptr< SplineCurve > sub\_crv, double par1, int cont1, double par2, int cont2)
- shared\_ptr< SplineSurface > [Go::GapRemoval::getSplineAndBd](#) (shared\_ptr< ParamSurface > psurf, std::vector< shared\_ptr< CurveOnSurface > > &bd\_crvs)



## Functions

- int `GO_API Go::GeometryTools::analyzePeriodicity` (`const` SplineCurve &cv, `double` knot\_tol=1e-12)
- int `GO_API Go::GeometryTools::analyzePeriodicity` (`const` SplineSurface &sf, int direction, `double` knot\_↔tol=1e-12)
- int `GO_API Go::GeometryTools::analyzePeriodicity` (`const` BsplineBasis &basis, `double` knot\_tol=1e-12)
- int `GO_API Go::GeometryTools::analyzePeriodicityDerivs` (`const` ParamCurve &cv, int max\_derivs, `double` tol=1e-14)
- int `GO_API Go::GeometryTools::analyzePeriodicityDerivs` (`const` SplineSurface &sf, int direction, int max\_↔derivs, `double` tol=1e-14)
- shared\_ptr< SplineCurve > `GO_API Go::GeometryTools::curveSum` (`const` SplineCurve &cv1, `double` fac1, `const` SplineCurve &cv2, `double` fac2, `double` num\_tol=1e-05)
- shared\_ptr< SplineSurface > `GO_API Go::GeometryTools::surfaceSum` (`const` SplineSurface &sf1, `double` fac1, `const` SplineSurface &sf2, `double` fac2, `double` num\_tol=1e-05)
- void `GO_API Go::GeometryTools::estimateSurfaceSize` (`const` ParamSurface &srf, `double` &length\_u, `double` &length\_v, `double` \*area=NULL)
- void `GO_API Go::GeometryTools::estimateIsoCurveLength` (`const` SplineSurface &srf, `bool` dir\_u, `double` par, `double` &length)
- `bool` `GO_API Go::GeometryTools::degenerateToCurve` (`const` SplineSurface &srf, `bool` dir\_u, `double` tol)
- void `GO_API Go::GeometryTools::makeBdDegenerate` (SplineSurface &srf, int bd\_idx)
 

*Make a specified surface boundary exactly degenerate.*
- void `GO_API Go::GeometryTools::makeUnionKnots` (std::vector< BsplineBasis > &bbasis, `double` tol, std↔::vector< `double` > &union\_knots)
 

*Compute the union of a set of knot vectors.*
- void `GO_API Go::GeometryTools::makeUnionKnots` (std::vector< std::vector< `double` > > &knots, `double` tol, std::vector< `double` > &union\_knots)
- `bool` `GO_API Go::GeometryTools::checkConstantCoef` (SplineCurve &cv, int idx, `double` val, `double` max\_dist, `double` tol)
- void `GO_API Go::GeometryTools::setSfBdCoefToConst` (SplineSurface &srf, int bd\_idx, int idx\_d, `double` val, `double` deg\_tol)
- void `GO_API Go::GeometryTools::findDominant` (`const` SplineSurface &surface, Vector3D &dominant\_u, Vector3D &dominant\_v)
- void `GO_API Go::GeometryTools::getGnJoints` (`const` ParamCurve &curve, `const` std::vector< `double` > &cont, std::vector< `double` > &gn\_joints)
- void `GO_API Go::GeometryTools::getGnJoints` (`const` CurveLoop &loop, `const` std::vector< `double` > &cont, std::vector< std::vector< `double` > > &gn\_joints)
- `bool` `GO_API Go::GeometryTools::isCoincident` (`const` ParamCurve &cv1, `const` ParamCurve &cv2, `double` epsge)
- `bool` `GO_API Go::GeometryTools::negativeProj` (`const` SplineSurface &surface, `const` Array< Vector3D, 2 > &refvector, `const` `double` eps=0.0)
- shared\_ptr< ParamCurve > `GO_API Go::GeometryTools::projectCurve` (shared\_ptr< ParamCurve > in↔curve, `const` Point &normal, `bool` planar)
- shared\_ptr< SplineCurve > `GO_API Go::GeometryTools::projectCurve` (`const` SplineCurve &incurve, `const` Point &normal, `bool` planar)
- shared\_ptr< SplineCurve > `GO_API Go::GeometryTools::representSurfaceAsCurve` (`const` SplineSurface &surface, int cv\_dir)
- shared\_ptr< SplineSurface > `GO_API Go::GeometryTools::representCurveAsSurface` (`const` SplineCurve &curve, int cv\_dir, `const` BsplineBasis &other\_bas, `bool` rational)
- std::vector< `double` > `GO_API Go::GeometryTools::getRotationMatrix` (`const` Point &unit\_axis\_dir, `double` alpha)
- void `GO_API Go::GeometryTools::rotateSplineSurf` (Point rot\_axis, `double` alpha, SplineSurface &sf)
- void `GO_API Go::GeometryTools::rotateSplineCurve` (Point rot\_axis, `double` alpha, SplineCurve &cv)
- void `GO_API Go::GeometryTools::rotateLineCloud` (Point rot\_axis, `double` alpha, LineCloud &lc)
- void `GO_API Go::GeometryTools::rotatePoint` (Point rot\_axis, `double` alpha, `double` \*space\_pt)
- void `GO_API Go::GeometryTools::rotatePoint` (Point rot\_axis, `double` alpha, Point &space\_pt)

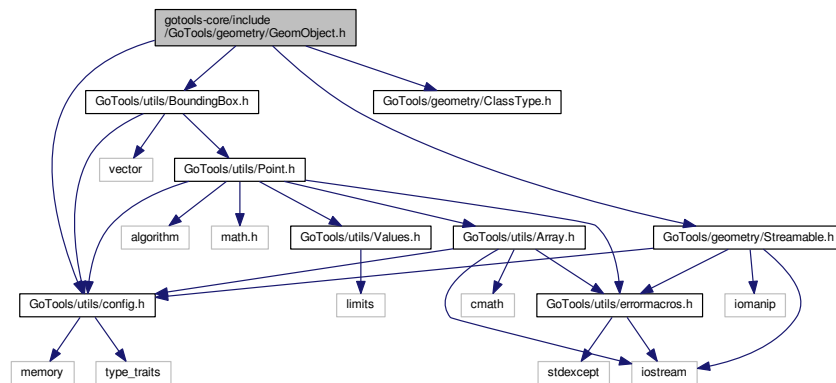


- void `GO_API Go::GeometryTools::splitCurveIntoSegments` (`const SplineCurve &cv`, `std::vector< SplineCurve > &seg`)  
*Split a spline curve into Bezier segments.*
- `std::vector< shared_ptr< SplineSurface > >` `GO_API Go::GeometryTools::splitInKinks` (`const SplineSurface &sf`, `const std::vector< double > &u_kinks`, `const std::vector< double > &v_kinks`)  
*Extract sub patches from the surface given by input parameters.*
- void `GO_API Go::GeometryTools::splitSurfaceIntoPatches` (`const SplineSurface &sf`, `std::vector< SplineSurface > &pat`)  
*Splits a spline surface into Bezier patches.*
- void `GO_API Go::GeometryTools::surfaceKinks` (`const SplineSurface &sf`, `double max_normal_angle`, `std::vector< double > &g1_disc_u`, `std::vector< double > &g1_disc_v`, `bool compute_g1_disc=true`)
- void `GO_API Go::GeometryTools::curveKinks` (`const SplineCurve &cv`, `double tol`, `double ang_tol`, `std::vector< double > &c1_disconts`, `std::vector< double > &g1_disconts`)  
*Find parameter values where a curve is G1- or C1-discontinuous.*
- void `GO_API Go::GeometryTools::translateSplineSurf` (`const Point &trans_vec`, `SplineSurface &sf`)  
*Translate the given SplineSurface by trans\_vec.*
- void `GO_API Go::GeometryTools::translateSplineCurve` (`const Point &trans_vec`, `SplineCurve &cv`)  
*Translate the given SplineCurve by trans\_vec.*
- void `GO_API Go::GeometryTools::translateLineCloud` (`const Point &trans_vec`, `LineCloud &lc`)  
*Translate the given LineCloud by trans\_vec.*
- void `GO_API Go::GeometryTools::averageBoundaryCoefs` (`shared_ptr< SplineSurface > &srf1`, `int bd1`, `bool keep_first`, `shared_ptr< SplineSurface > &srf2`, `int bd2`, `bool keep_second`, `bool found_corner1`, `Point corner1`, `bool found_corner2`, `Point corner2`, `bool opposite`)
- void `GO_API Go::GeometryTools::unifyCurveSplineSpace` (`std::vector< shared_ptr< SplineCurve > > &curves`, `double tol`)
- void `GO_API Go::GeometryTools::unifySurfaceSplineSpace` (`std::vector< shared_ptr< SplineSurface > > &surfaces`, `double tol`, `int dir=0`)
- void `GO_API Go::GeometryTools::unifySurfaceSplineSpaceOneDir` (`std::vector< shared_ptr< SplineSurface > > &surfaces`, `double tol`, `bool unify_u_dir`)
- `shared_ptr< SplineSurface >` `GO_API Go::GeometryTools::joinPatches` (`const std::vector< shared_ptr< SplineSurface > > &patches`, `const SplineSurface &spline_space`)
- void `GO_API Go::GeometryTools::insertKnotsEvenly` (`BsplineBasis &basis`, `int num_knots`)
- void `GO_API Go::GeometryTools::insertKnotsEvenly` (`BsplineBasis &basis`, `double tmin`, `double tmax`, `int num_knots`, `double knot_diff_tol=1e-05`)
- `double` `GO_API Go::GeometryTools::getKnotAtLargestInterval` (`const BsplineBasis &basis`)
- `std::pair< double, double >` `GO_API Go::GeometryTools::getLargestParameterInterval` (`const BsplineBasis &basis`)
- void `Go::GeometryTools::setParameterDomain` (`std::vector< shared_ptr< BoundedSurface > > &sfs`, `double u1`, `double u2`, `double v1`, `double v2`)  
*Reparameterize a set of bounded surfaces to the same domain.*

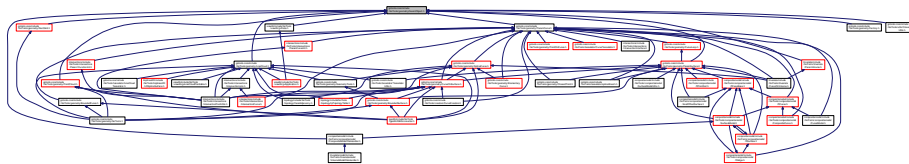
## 30.1925 gotools-core/include/GoTools/geometry/GeomObject.h File Reference

```
#include "GoTools/utils/BoundingBox.h"
#include "GoTools/geometry/Streamable.h"
#include "GoTools/geometry/ClassType.h"
#include "GoTools/utils/config.h"
```

Include dependency graph for `GeomObject.h`:



This graph shows which files directly or indirectly include this file:



## Classes

- class [Go::GeomObject](#)

## Namespaces

- [Go](#)

## Variables

- `const int` [Go::MAJOR\\_VERSION](#) = 1
- `const int` [Go::MINOR\\_VERSION](#) = 0

## 30.1926 gotools-core/include/GoTools/geometry/GoIntersections.h File Reference

```
#include "GoTools/geometry/SplineCurve.h"
#include "GoTools/geometry/SplineSurface.h"
#include <utility>
```



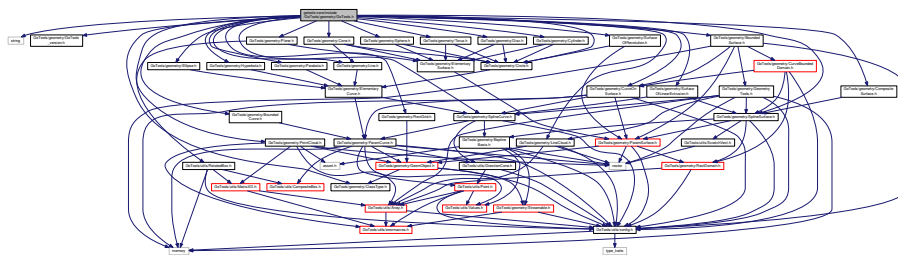
## 30.1927 gotools-core/include/GoTools/geometry/GoTools.h File Reference

```

#include <string>
#include "GoTools/geometry/GoTools_version.h"
#include "GoTools/geometry/ClassType.h"
#include <GoTools/geometry/SplineCurve.h>
#include <GoTools/geometry/CurveOnSurface.h>
#include <GoTools/geometry/Line.h>
#include <GoTools/geometry/Circle.h>
#include <GoTools/geometry/Ellipse.h>
#include <GoTools/geometry/BoundedCurve.h>
#include <GoTools/geometry/Hyperbola.h>
#include <GoTools/geometry/Parabola.h>
#include <GoTools/geometry/SplineSurface.h>
#include <GoTools/geometry/BoundedSurface.h>
#include <GoTools/geometry/CompositeSurface.h>
#include <GoTools/geometry/Plane.h>
#include <GoTools/geometry/Cylinder.h>
#include <GoTools/geometry/Sphere.h>
#include <GoTools/geometry/Cone.h>
#include <GoTools/geometry/Torus.h>
#include <GoTools/geometry/SurfaceOfRevolution.h>
#include <GoTools/geometry/SurfaceOfLinearExtrusion.h>
#include <GoTools/geometry/Disc.h>
#include <GoTools/geometry/PointCloud.h>
#include <GoTools/geometry/LineCloud.h>
#include <GoTools/geometry/RectGrid.h>

```

Include dependency graph for GoTools.h:



### Classes

- class [Go::GoTools](#)

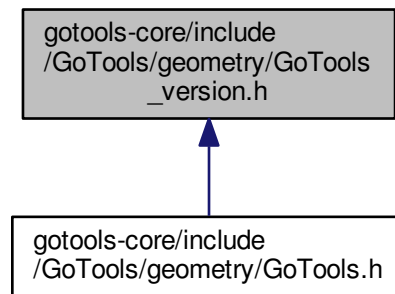
*Class containing some service functions for the [GoTools](#) "system".*

### Namespaces

- [Go](#)

## 30.1928 gotools-core/include/GoTools/geometry/GoTools\_version.h File Reference

This graph shows which files directly or indirectly include this file:



### Macros

- `#define GO_VERSION_MAJOR 4`
- `#define GO_VERSION_MINOR 3`
- `#define GO_VERSION_PATCH 0`

### 30.1928.1 Macro Definition Documentation

#### 30.1928.1.1 `#define GO_VERSION_MAJOR 4`

Definition at line 44 of file `GoTools_version.h`.

#### 30.1928.1.2 `#define GO_VERSION_MINOR 3`

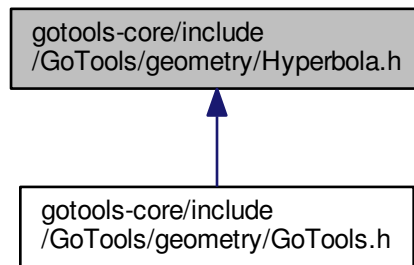
Definition at line 45 of file `GoTools_version.h`.

#### 30.1928.1.3 `#define GO_VERSION_PATCH 0`

Definition at line 46 of file `GoTools_version.h`.



This graph shows which files directly or indirectly include this file:



### Classes

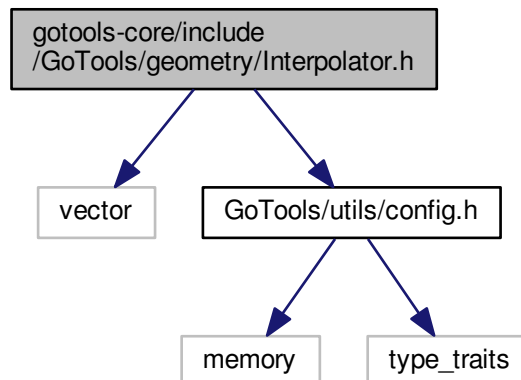
- class [Go::Hyperbola](#)  
*Class that represents a hyperbola. It is a subclass of [ElementaryCurve](#) and thus has a parametrization.*

### Namespaces

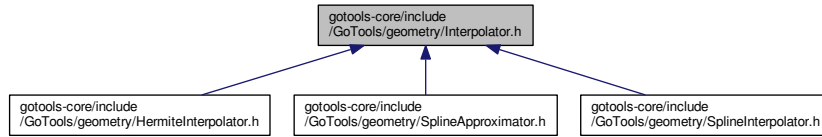
- [Go](#)

## 30.1931 gotools-core/include/GoTools/geometry/Interpolator.h File Reference

```
#include <vector>
#include "GoTools/utils/config.h"
Include dependency graph for Interpolator.h:
```



This graph shows which files directly or indirectly include this file:



### Classes

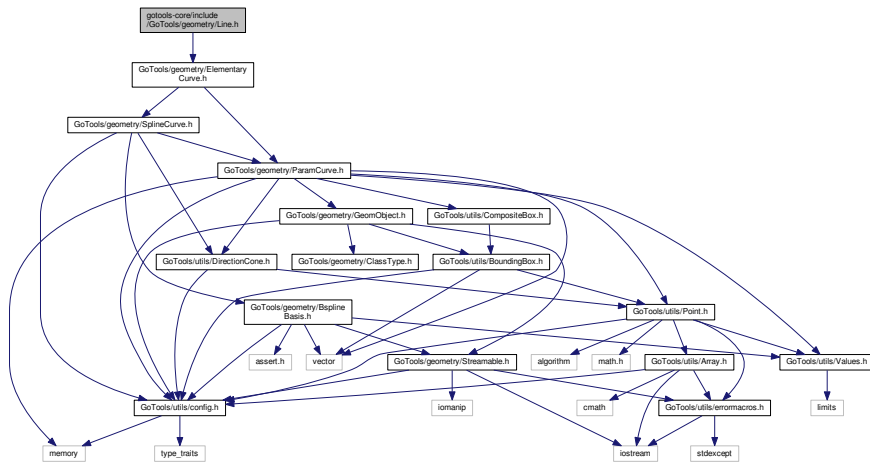
- class [Go::Interpolator](#)  
*Base class for spline interpolators or approximators.*

### Namespaces

- [Go](#)

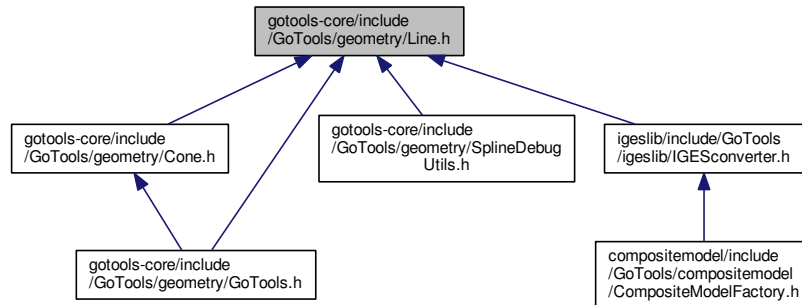
## 30.1932 gotools-core/include/GoTools/geometry/Line.h File Reference

#include "GoTools/geometry/ElementaryCurve.h"  
 Include dependency graph for Line.h:





This graph shows which files directly or indirectly include this file:



## Classes

- class [Go::Line](#)

Class that represents a line. It is a subclass of [ElementaryCurve](#) and thus has a parametrization.

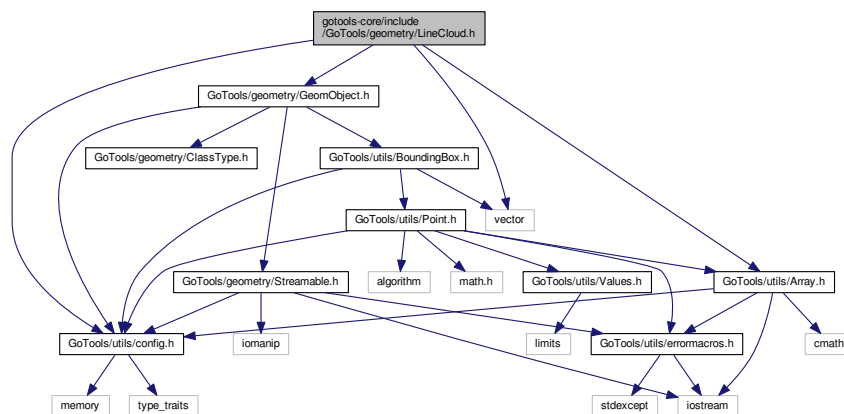
## Namespaces

- [Go](#)

## 30.1933 gotools-core/include/GoTools/geometry/LineCloud.h File Reference

```
#include "GoTools/geometry/GeomObject.h"
#include "GoTools/utils/Array.h"
#include <vector>
#include "GoTools/utils/config.h"
```

Include dependency graph for LineCloud.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [Go::LineCloud](#)

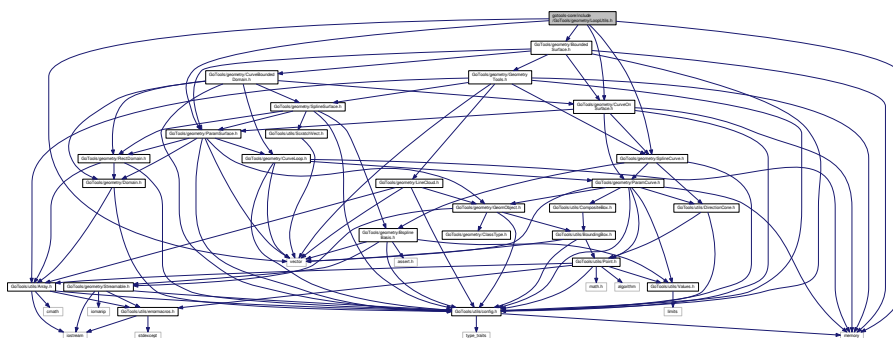
## Namespaces

- [Go](#)

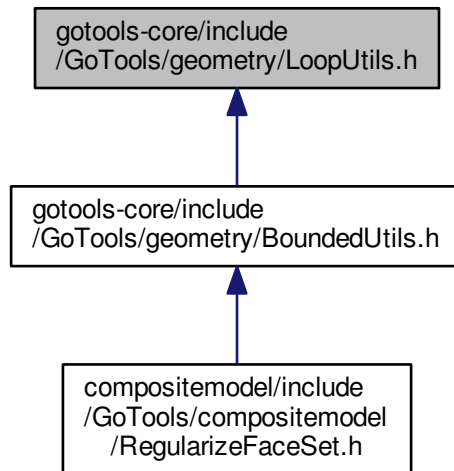
## 30.1934 gotools-core/include/GoTools/geometry/LoopUtils.h File Reference

```
#include "GoTools/geometry/SplineCurve.h"
#include "GoTools/geometry/CurveOnSurface.h"
#include "GoTools/geometry/ParamSurface.h"
#include "GoTools/geometry/BoundedSurface.h"
#include <vector>
#include <memory>
```

Include dependency graph for LoopUtils.h:



This graph shows which files directly or indirectly include this file:



## Namespaces

- [Go](#)
- [Go::LoopUtils](#)

## Functions

- void [Go::LoopUtils::representAsSurfaceCurves](#) (std::vector< shared\_ptr< ParamCurve > > &curves, shared\_ptr< BoundedSurface > surf, std::vector< shared\_ptr< CurveOnSurface > > &cvs\_on\_sf)
- bool [Go::LoopUtils::loopsCCW](#) (const std::vector< shared\_ptr< [Go::SplineCurve](#) > > &simple\_par\_loop, double space\_epsilon, double int\_tol)
- bool [Go::LoopUtils::paramsCCW](#) (const std::vector< shared\_ptr< [Go::CurveOnSurface](#) > > &loop, double space\_epsilon, double int\_tol)
- bool [Go::LoopUtils::loopsCCW](#) (const CurveLoop &loop, double int\_tol)
- bool [Go::LoopUtils::firstLoopInsideSecond](#) (const std::vector< shared\_ptr< [Go::CurveOnSurface](#) > > &first\_loop, const std::vector< shared\_ptr< [Go::CurveOnSurface](#) > > &second\_loop, double loop\_tol, double int\_tol)
 

*Loops expected to be disjoint, except possibly share part of boundary.*
- bool [Go::LoopUtils::makeLoopCCW](#) (std::vector< shared\_ptr< ParamCurve > > &loop\_cvs, double tol)

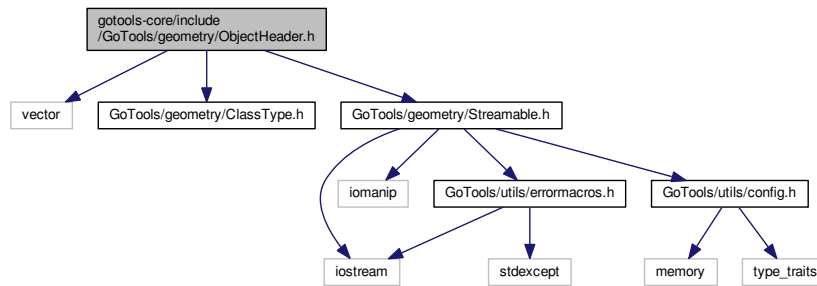
## 30.1935 gotools-core/include/GoTools/geometry/ObjectHeader.h File Reference

```

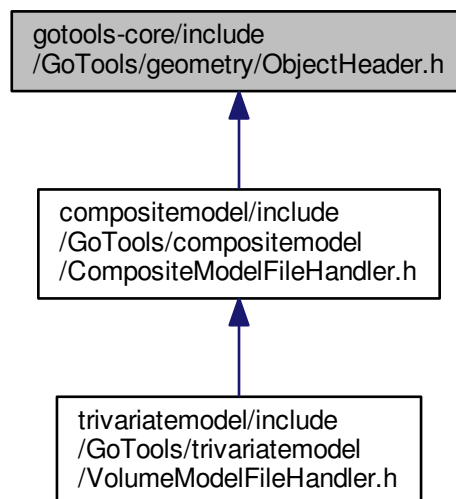
#include <vector>
#include "GoTools/geometry/ClassType.h"
#include "GoTools/geometry/Streamable.h"

```

Include dependency graph for ObjectHeader.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [Go::ObjectHeader](#)

## Namespaces

- [Go](#)

## 30.1936 gotools-core/include/GoTools/geometry/orientCurves.h File Reference

## Namespaces

- [Go](#)
- [Go::orientCurves](#)

*Sorts and orients a set of curves.*

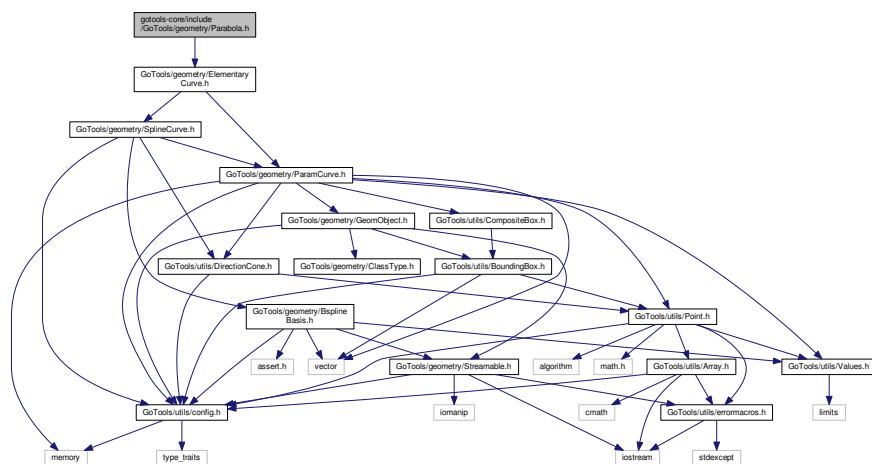
## Functions

- `template<class PtrToCurveType >`  
`void Go::orientCurves::orientCurves (const std::vector< PtrToCurveType > &curves, std::vector< int > &permutation, std::vector< bool > &reversed, double neighbour_tol, bool assume_manifold=true)`

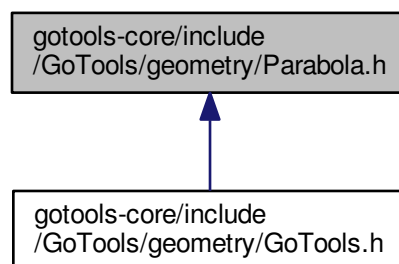
## 30.1937 gotools-core/include/GoTools/geometry/Parabola.h File Reference

```
#include "GoTools/geometry/ElementaryCurve.h"
```

Include dependency graph for Parabola.h:



This graph shows which files directly or indirectly include this file:













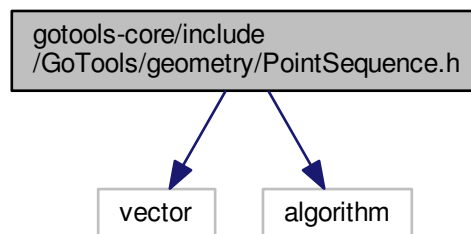


## Namespaces

- [Go](#)

## 30.1943 gotools-core/include/GoTools/geometry/PointSequence.h File Reference

```
#include <vector>
#include <algorithm>
Include dependency graph for PointSequence.h:
```



## Classes

- class [Go::PointSequence](#)

## Namespaces

- [Go](#)

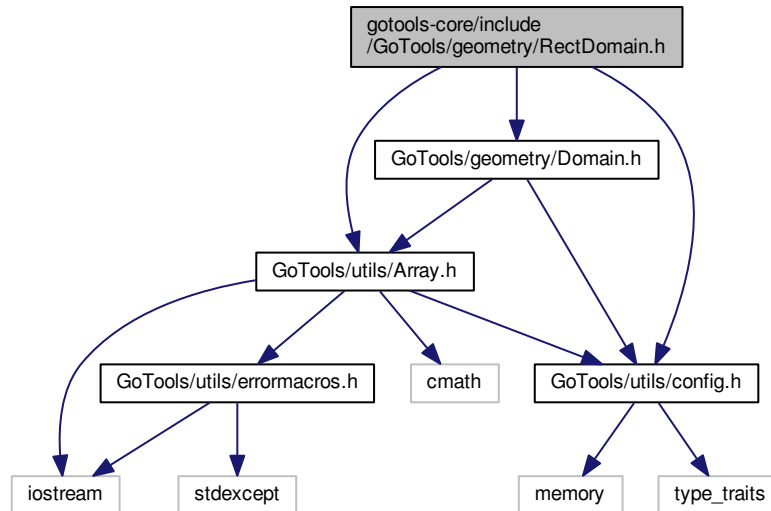
## Enumerations

- enum [Go::PointSequenceType](#) { [Go::PSTPoint](#), [Go::PSTPointTangent](#), [Go::PSTScattered](#) }
- Type of point.*

## 30.1944 gotools-core/include/GoTools/geometry/RectDomain.h File Reference

```
#include "GoTools/utils/Array.h"
#include "GoTools/geometry/Domain.h"
#include "GoTools/utils/config.h"
```

Include dependency graph for RectDomain.h:



This graph shows which files directly or indirectly include this file:



### Classes

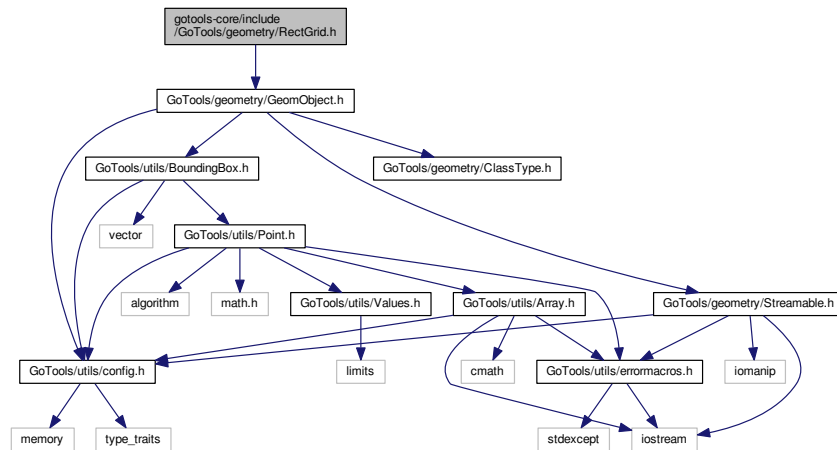
- class [Go::RectDomain](#)

### Namespaces

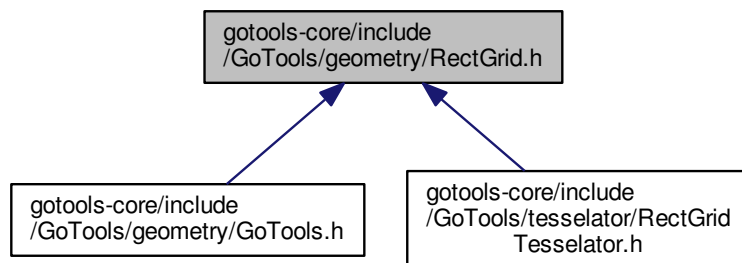
- [Go](#)

## 30.1945 gotools-core/include/GoTools/geometry/RectGrid.h File Reference

```
#include "GoTools/geometry/GeomObject.h"
Include dependency graph for RectGrid.h:
```



This graph shows which files directly or indirectly include this file:



## Classes

- class [Go::RectGrid](#)

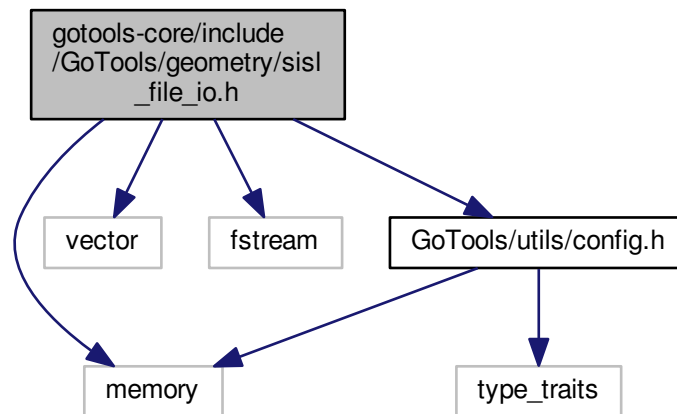
## Namespaces

- [Go](#)

### 30.1946 gotools-core/include/GoTools/geometry/SISL\_code.h File Reference

### 30.1947 gotools-core/include/GoTools/geometry/sisl\_file\_io.h File Reference

```
#include <memory>
#include <vector>
#include <fstream>
#include "GoTools/utils/config.h"
Include dependency graph for sisl_file_io.h:
```



#### Macros

- `#define` [STD\\_FILE](#) FILE

#### Functions

- void [read\\_non\\_comment](#) (STD\_FILE \*fp, char \*string)
- void [file\\_to\\_obj](#) (STD\_FILE \*fp, SISLObject \*\*wo, int \*jstat)
- void [curve\\_to\\_file](#) (STD\_FILE \*f, struct SISLCurve \*c1)
- void [surface\\_to\\_file](#) (STD\_FILE \*f, struct SISLSurf \*surf)
- int [get\\_next\\_surface](#) (STD\_FILE \*fp, SISLSurf \*\*qc)
- int [get\\_sisl\\_surfaces](#) (STD\_FILE \*fp, std::vector< shared\_ptr< SISLSurf > > &sisl\_sfs)

#### 30.1947.1 Macro Definition Documentation

##### 30.1947.1.1 `#define` STD\_FILE FILE

Definition at line 55 of file `sisl_file_io.h`.

30.1947.2 Function Documentation

30.1947.2.1 void curve\_to\_file ( STD\_FILE \* f, struct SISLCurve \* c1 )

30.1947.2.2 void file\_to\_obj ( STD\_FILE \* fp, SISLObject \*\* wo, int \* jstat )

30.1947.2.3 int get\_next\_surface ( STD\_FILE \* fp, SISLSurf \*\* qc )

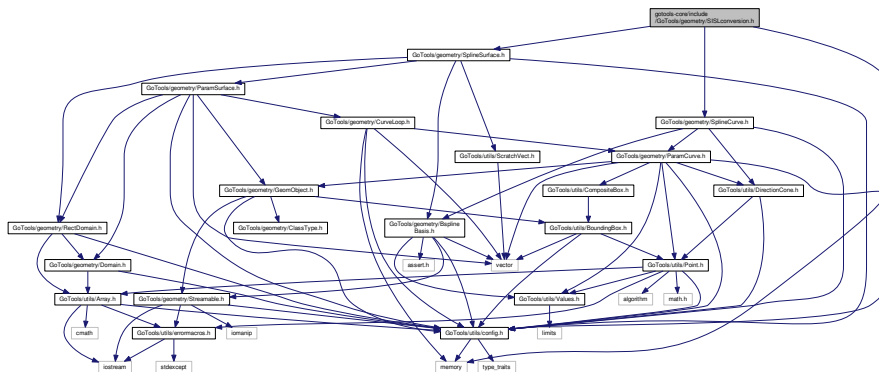
30.1947.2.4 int get\_sisl\_surfaces ( STD\_FILE \* fp, std::vector< shared\_ptr< SISLSurf > > & sisl\_sfs )

30.1947.2.5 void read\_non\_comment ( STD\_FILE \* fp, char \* string )

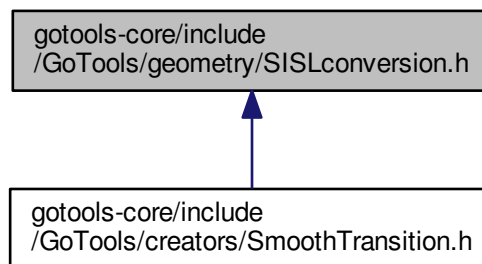
30.1947.2.6 void surface\_to\_file ( STD\_FILE \* f, struct SISLSurf \* surf )

30.1948 gotools-core/include/GoTools/geometry/SISLconversion.h File Reference

```
#include "GoTools/geometry/SplineCurve.h"
#include "GoTools/geometry/SplineSurface.h"
#include "GoTools/utils/config.h"
Include dependency graph for SISLconversion.h:
```



This graph shows which files directly or indirectly include this file:







## Classes

- class [Go::Sphere](#)

Class that represents a sphere. It is a subclass of [ElementarySurface](#), and thus has a parametrization and is non-selfintersecting.

## Namespaces

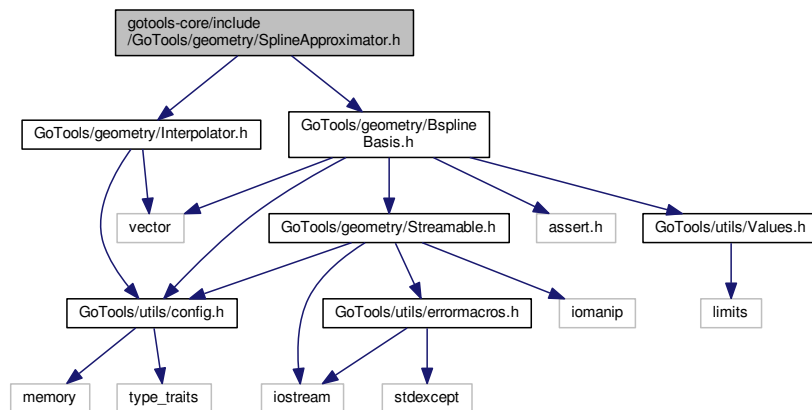
- [Go](#)

## 30.1950 gotools-core/include/GoTools/geometry/SplineApproximator.h File Reference

```
#include "GoTools/geometry/Interpolator.h"
```

```
#include "GoTools/geometry/BsplineBasis.h"
```

Include dependency graph for SplineApproximator.h:



## Classes

- class [Go::SplineApproximator](#)

## Namespaces

- [Go](#)



## Namespaces

- [Go](#)
- [Go::SplineDebugUtils](#)

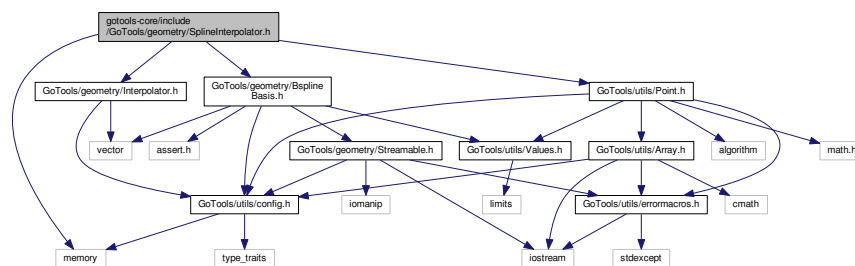
## Functions

- void [GO\\_API Go::SplineDebugUtils::writeSpaceParamCurve](#) ([const](#) SplineCurve &pcurve, [std::ostream](#) &os, [double](#) z=0.0)
- void [GO\\_API Go::SplineDebugUtils::writeSpaceParamCurve](#) ([const](#) Line &pline, [std::ostream](#) &os, [double](#) z=0.0)
- void [GO\\_API Go::SplineDebugUtils::writeTrimmedInfo](#) (BoundedSurface &bd\_sf, [std::ostream](#) &os, [double](#) z=0.0)
- void [GO\\_API Go::SplineDebugUtils::writeOuterBoundaryLoop](#) (ParamSurface &sf, [std::ostream](#) &os)
- void [GO\\_API Go::SplineDebugUtils::objToFile](#) (GeomObject \*geom\_obj, [char](#) \*to\_file)
- void [GO\\_API Go::SplineDebugUtils::objsToFile](#) ([std::vector](#)< [shared\\_ptr](#)< GeomObject > > &geom\_objs, [char](#) \*to\_file)
- void [GO\\_API Go::SplineDebugUtils::writeSISLFormat](#) ([const](#) SplineCurve &spline\_cv, [std::ostream](#) &os)
- void [GO\\_API Go::SplineDebugUtils::writeCvsOnSf](#) ([const](#) [std::vector](#)< [shared\\_ptr](#)< [Go::ParamCurve](#) > > &loop\_cvs, [double](#) epsgeo, [std::ofstream](#) &fileout)
- void [GO\\_API Go::SplineDebugUtils::writeCvsOnSf](#) ([const](#) [std::vector](#)< [shared\\_ptr](#)< [Go::CurveOnSurface](#) > > &loop\_cvs, [double](#) epsgeo, [std::ofstream](#) &fileout)

## 30.1953 gotools-core/include/GoTools/geometry/SplineInterpolator.h File Reference

```
#include "GoTools/geometry/Interpolator.h"
#include "GoTools/geometry/BsplineBasis.h"
#include "GoTools/utils/Point.h"
#include <memory>
```

Include dependency graph for SplineInterpolator.h:



## Classes

- class [Go::SplineInterpolator](#)

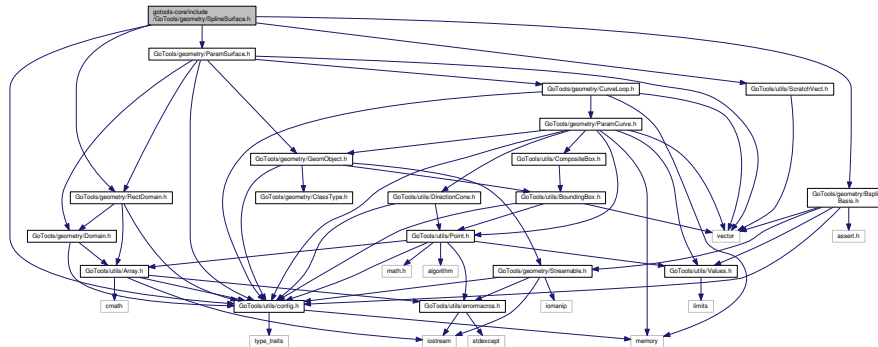
## Namespaces

- [Go](#)

### 30.1954 gotools-core/include/GoTools/geometry/SplineSurface.h File Reference

```
#include "GoTools/geometry/ParamSurface.h"
#include "GoTools/geometry/BsplineBasis.h"
#include "GoTools/geometry/RectDomain.h"
#include "GoTools/utils/ScratchVect.h"
#include "GoTools/utils/config.h"
```

Include dependency graph for SplineSurface.h:



#### Classes

- struct [Go::BasisPtsSf](#)
- struct [Go::BasisDerivsSf](#)
- struct [Go::BasisDerivsSf2](#)
- class [Go::SplineSurface](#)

*SplineSurface* provides methodes for storing, reading and manipulating rational and non-rational B-spline surfaces.

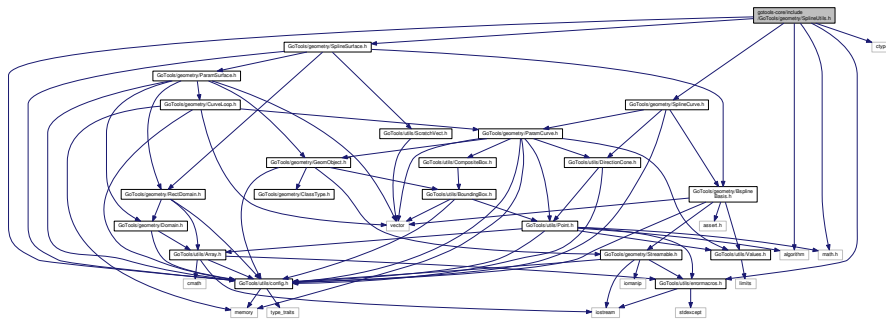
#### Namespaces

- [Go](#)

### 30.1955 gotools-core/include/GoTools/geometry/SplineUtils.h File Reference

```
#include "GoTools/geometry/SplineSurface.h"
#include "GoTools/geometry/SplineCurve.h"
#include "GoTools/utils/errormacros.h"
#include <math.h>
#include <algorithm>
#include <ctype.h>
#include "GoTools/utils/config.h"
```

Include dependency graph for SplineUtils.h:



## Namespaces

- [Go](#)
- [Go::SplineUtils](#)

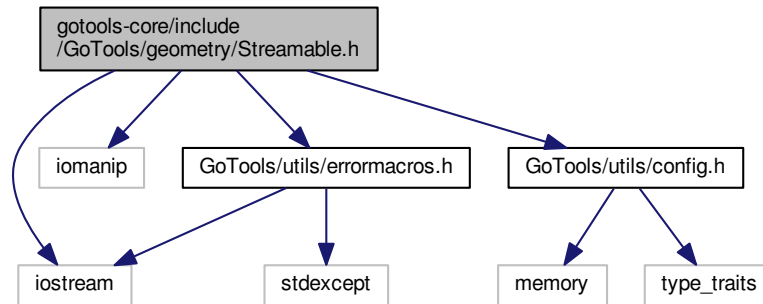
## Functions

- `void GO_API Go::SplineUtils::transpose_array (int dim, int m, int n, double *array_start)`
- `int GO_API Go::SplineUtils::closest_in_array (const double *pt, const double *array, int n, int dim)`
- `Vector3D GO_API Go::SplineUtils::closest_on_triangle (const Vector3D &pt, const Vector3D tri[3], double &clo_dist2)`
- `double GO_API Go::SplineUtils::closest_on_line_segment (const Vector3D &pt, const Vector3D &beg, const Vector3D &end)`
- `void GO_API Go::SplineUtils::closest_on_rectgrid (const double *pt, const double *array, int m, int n, double &clo_u, double &clo_v)`
- `void GO_API Go::SplineUtils::closest_on_rectgrid (const double *pt, const double *array, int u_min, int u_max, int v_min, int v_max, int nmb_coefs_u, double &clo_u, double &clo_v)`
- `void GO_API Go::SplineUtils::make_coef_array_from_rational_coefs (const double *rationals, double *coefs, int num_coefs, int dim)`
- `void GO_API Go::SplineUtils::curve_ratder (double const eder[], int idim, int ider, double gder[])`
- `void GO_API Go::SplineUtils::surface_ratder (double const eder[], int idim, int ider, double gder[])`
- `void GO_API Go::SplineUtils::osloalg (int ij, int imy, int ik, int in, int *jpl, int *jfi, int *jla, const double *et, const double *etau, double *galfa)`
- `void GO_API Go::SplineUtils::refmatrix (const double *et, int im, int ik, const double *etau, int in, double *ea, int *nfirst, int *nlast)`
- `void GO_API Go::SplineUtils::splineToBezierTransfMat (const double *knots, std::vector< double > &transf_mat)`
- `void GO_API Go::SplineUtils::extractBezierCoefs (const double *coefs, const int num_coefs_u, const int num_coefs_v, const int ind_u_min, const int ind_v_min, const std::vector< double > &transf_mat_u, const std::vector< double > &transf_mat_v, std::vector< double > &bezier_coefs)`
- `void GO_API Go::SplineUtils::refinedBezierCoefsCubic (Go::SplineSurface &spline_sf, int ind_u_min, int ind_v_min, std::vector< double > &bez_coefs)`  
*Method expecting bi-cubic input.*
- `shared_ptr< SplineSurface > GO_API Go::SplineUtils::refineToBezier (const Go::SplineSurface &spline_sf)`
- `shared_ptr< SplineSurface > GO_API Go::SplineUtils::insertKnots (const Go::SplineSurface &spline_sf, const std::vector< double > new_knots_u, const std::vector< double > new_knots_v)`

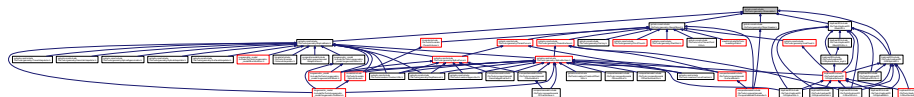
## 30.1956 gotools-core/include/GoTools/geometry/Streamable.h File Reference

```
#include <iostream>
#include <iomanip>
#include "GoTools/utils/errormacros.h"
#include "GoTools/utils/config.h"
```

Include dependency graph for Streamable.h:



This graph shows which files directly or indirectly include this file:



### Classes

- class [Go::Streamable](#)
- class [Go::Streamable::EofException](#)

### Namespaces

- [Go](#)

### Functions

- `std::istream & Go::operator>> (std::istream &is, Go::Streamable &obj)`
- `std::ostream & Go::operator<< (std::ostream &os, const Go::Streamable &obj)`

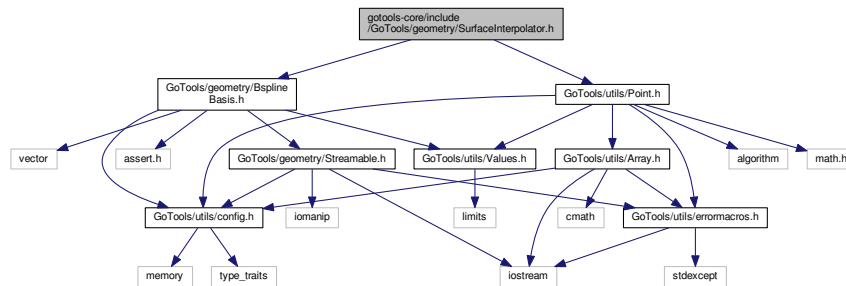
## 30.1957 gotools-core/include/GoTools/geometry/streamable\_doxygen.h File Reference

## 30.1958 gotools-core/include/GoTools/geometry/SurfaceInterpolator.h File Reference

```
#include "GoTools/geometry/BsplineBasis.h"
```

```
#include "GoTools/utils/Point.h"
```

Include dependency graph for SurfaceInterpolator.h:



## Namespaces

- [Go](#)
- [Go::SurfaceInterpolator](#)

*Functionality for creating interpolating surfaces.*

## Functions

- SplineSurface \* [Go::SurfaceInterpolator::regularInterpolation](#) (const BsplineBasis &basis\_u, const BsplineBasis &basis\_v, std::vector< double > &par\_u, std::vector< double > &par\_v, std::vector< double > &points, int dimension, bool rational, std::vector< double > &weights)

## 30.1959 gotools-core/include/GoTools/geometry/SurfaceOfLinearExtrusion.h File Reference

```
#include "GoTools/geometry/SplineCurve.h"
```

```
#include "GoTools/geometry/SplineSurface.h"
```



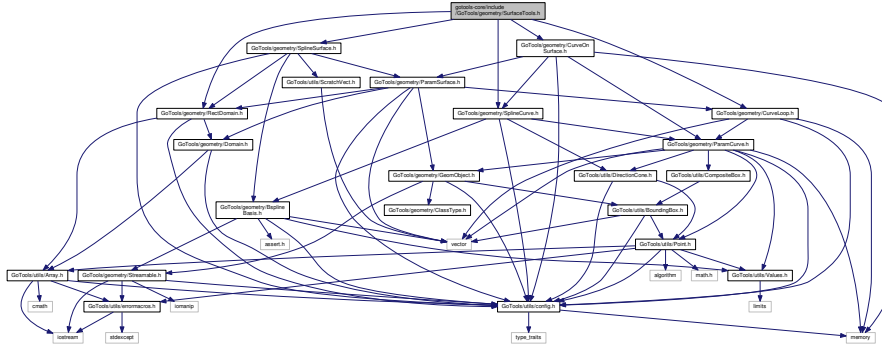




## 30.1961 gotools-core/include/GoTools/geometry/SurfaceTools.h File Reference

```
#include "GoTools/geometry/SplineSurface.h"
#include "GoTools/geometry/SplineCurve.h"
#include "GoTools/geometry/CurveOnSurface.h"
#include "GoTools/geometry/CurveLoop.h"
#include "GoTools/geometry/RectDomain.h"
```

Include dependency graph for SurfaceTools.h:



### Namespaces

- [Go](#)
- [Go::SurfaceTools](#)

*Free functions operating on parametric surfaces.*

### Functions

- `CurveLoop` [Go::SurfaceTools::outerBoundarySfLoop](#) (`shared_ptr< ParamSurface > surf`, `double degenerate_epsilon`)  
*Fetch the outer boundary loop of a parametric surface.*
- `std::vector< CurveLoop >` [Go::SurfaceTools::allBoundarySfLoops](#) (`shared_ptr< ParamSurface > surf`, `double degenerate_epsilon`)  
*Fetch all boundary loops of a parametric surface.*
- `std::vector< CurveLoop >` [Go::SurfaceTools::absolutelyAllBoundarySfLoops](#) (`shared_ptr< ParamSurface > surf`, `double degenerate_epsilon`)
- `void` [Go::SurfaceTools::iterateCornerPos](#) (`Point &vertex`, `std::vector< std::pair< shared_ptr< ParamSurface >, Point >> sfs`, `double tol`)
- `bool` [Go::SurfaceTools::cornerToCornerSfs](#) (`shared_ptr< ParamSurface > sf1`, `shared_ptr< CurveOnSurface > sf_cv1`, `shared_ptr< ParamSurface > sf2`, `shared_ptr< CurveOnSurface > sf_cv2`, `double tol`)
- `bool` [Go::SurfaceTools::getSfAdjacencyInfo](#) (`shared_ptr< ParamSurface > sf1`, `shared_ptr< CurveOnSurface > sf_cv1`, `shared_ptr< ParamSurface > sf2`, `shared_ptr< CurveOnSurface > sf_cv2`, `double tol`, `int &bd1`, `int &bd2`, `bool &same_orient`)
- `bool` [Go::SurfaceTools::getCorrCoefEnum](#) (`shared_ptr< SplineSurface > sf1`, `shared_ptr< SplineSurface > sf2`, `int bd1`, `int bd2`, `bool same_orient`, `std::vector< std::pair< int, int >> &enumeration`)
- `bool` [Go::SurfaceTools::getCoefEnumeration](#) (`shared_ptr< SplineSurface > sf`, `int bd`, `std::vector< int > &enumeration`)
- `bool` [Go::SurfaceTools::getCoefEnumeration](#) (`shared_ptr< SplineSurface > sf`, `int bd`, `std::vector< int > &enumeration_bd`, `std::vector< int > &enumeration_bd2`)







*normalize* makes the length of a vector 1.0

- `template<typename ForwardIterator >`  
`go_iterator_traits< ForwardIterator >::value_type` `Go::Utils::inner` (ForwardIterator first, ForwardIterator last, ForwardIterator second)

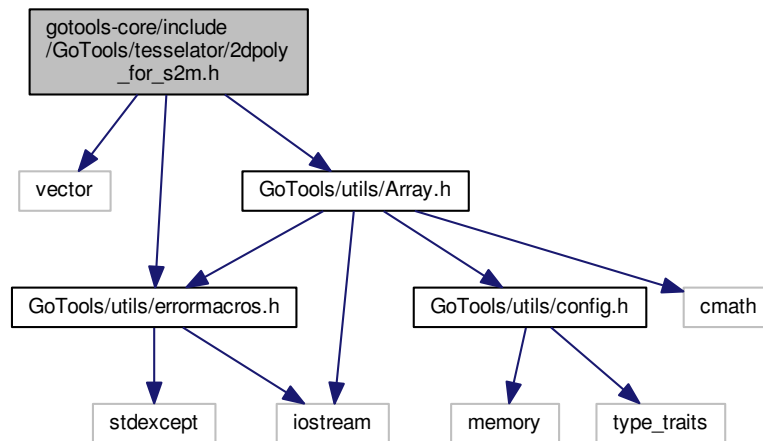
*inner product*

- `template<typename InputStream >`  
`InputStream & Go::Utils::eatwhite` (InputStream &is)

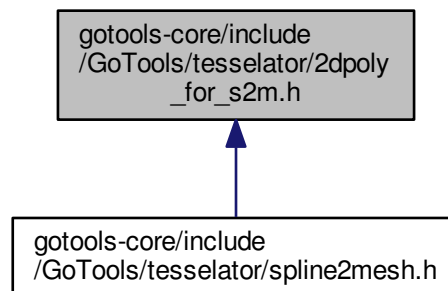
*eat white space*

### 30.1965 gotools-core/include/GoTools/tesselator/2dpoly\_for\_s2m.h File Reference

```
#include <vector>
#include "GoTools/Utils/Array.h"
#include "GoTools/Utils/errormacros.h"
Include dependency graph for 2dpoly_for_s2m.h:
```



This graph shows which files directly or indirectly include this file:



## Namespaces

- [Go](#)

## Typedefs

- typedef std::pair< short, std::vector< short > \* > [short\\_list](#)
- typedef std::pair< short, std::vector< [short\\_list](#) > \* > [short\\_list\\_short\\_list](#)

## Functions

- [bool Go::point\\_inside\\_contour](#) (const double x0, const double y0, const double \*const vertices, const std::vector< int > &contour)
- [bool Go::segment\\_contour\\_intersection\\_for\\_s2m](#) (const double x0, const double y0, const double x1, const double y1, const double \*const vertices, const std::vector< int > &contour, double &x, double &y, double &s, const bool snap\_ends=false)
- std::vector< [short\\_list\\_short\\_list](#) > [Go::sort\\_2dpoly\\_segments](#) (const double \*const vertices, const std::vector< int > &contour, const bool transposed=false)
- int [Go::is\\_inside](#) (const std::vector< [Go::Vector3D](#) > &trim\_curve\_p, const std::vector< int > &contour, const double u, const double v)
- [bool Go::is\\_on\\_corner](#) (const std::vector< [Go::Vector3D](#) > &trim\_curve\_p, const std::vector< int > &contour, const double u, const double v)
- int [Go::is\\_on\\_contour](#) (const std::vector< [Go::Vector3D](#) > &trim\_curve\_p, const std::vector< int > &contour, const double u, const double v)
- [bool Go::degenerate\\_triangle](#) (const [Vector2D](#) &c1, const [Vector2D](#) &c2, const [Vector2D](#) &c3)

### 30.1965.1 Typedef Documentation

#### 30.1965.1.1 typedef std::pair<short, std::vector<short>\*> short\_list

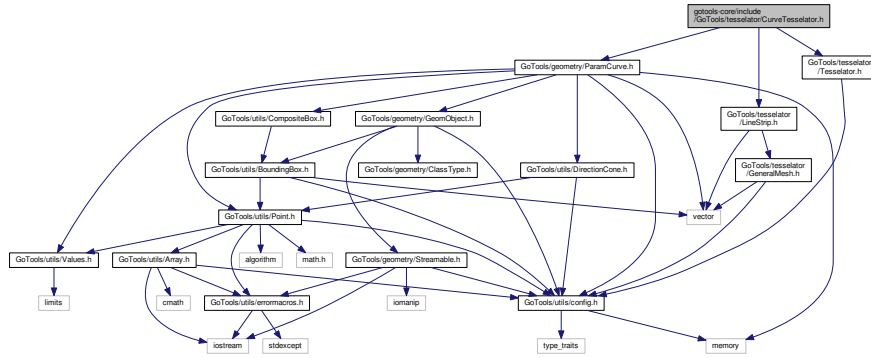
Definition at line 104 of file 2dpoly\_for\_s2m.h.

#### 30.1965.1.2 typedef std::pair<short, std::vector<short\_list>\*> short\_list\_short\_list

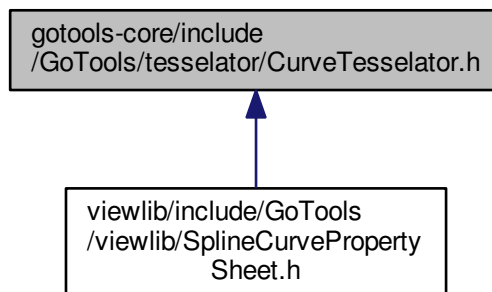
Definition at line 105 of file 2dpoly\_for\_s2m.h.

## 30.1966 gotools-core/include/GoTools/tesselator/CurveTesselator.h File Reference

```
#include "GoTools/tesselator/Tesselator.h"
#include "GoTools/tesselator/LineStrip.h"
#include "GoTools/geometry/ParamCurve.h"
Include dependency graph for CurveTesselator.h:
```



This graph shows which files directly or indirectly include this file:



### Classes

- class [Go::CurveTesselator](#)

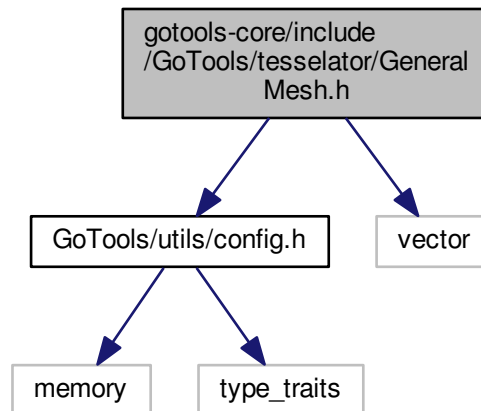
### Namespaces

- [Go](#)

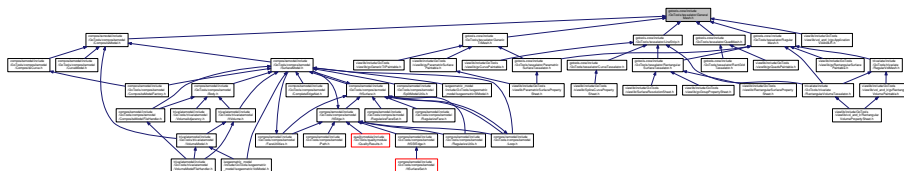


## 30.1967 gotools-core/include/GoTools/tesselator/GeneralMesh.h File Reference

```
#include "GoTools/utils/config.h"
#include <vector>
Include dependency graph for GeneralMesh.h:
```



This graph shows which files directly or indirectly include this file:



## Classes

- class [Go::GeneralMesh](#)

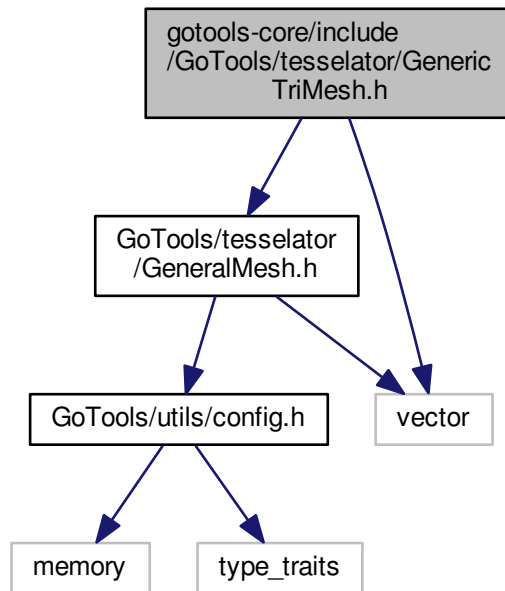
## Namespaces

- [Go](#)

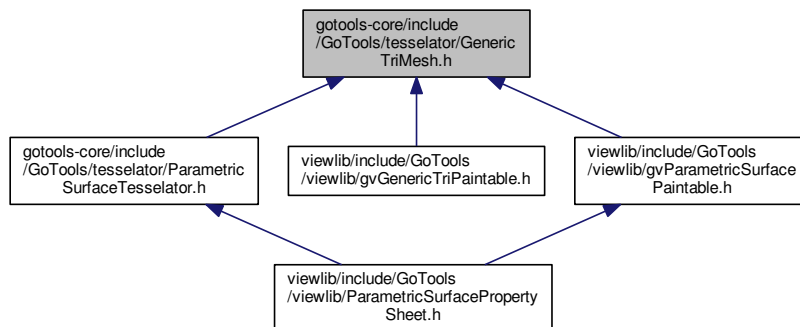
## 30.1968 gotools-core/include/GoTools/tessellator/GenericTriMesh.h File Reference

```
#include "GoTools/tessellator/GeneralMesh.h"
#include <vector>
```

Include dependency graph for GenericTriMesh.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [Go::GenericTriMesh](#)

## Namespaces

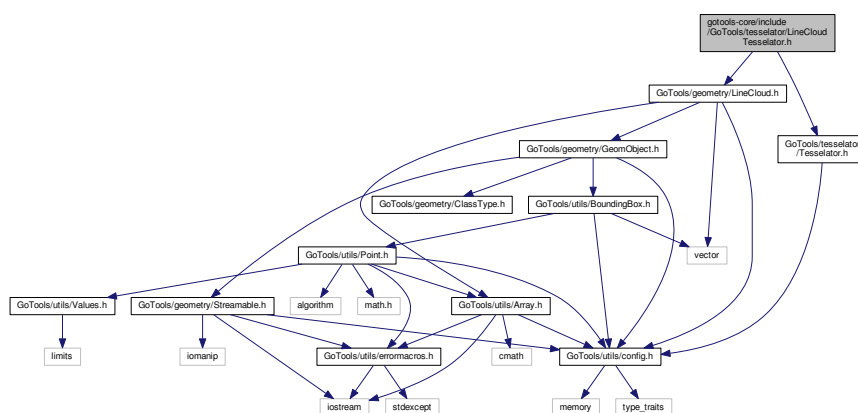
- [Go](#)

## 30.1969 gotools-core/include/GoTools/tesselator/LineCloudTesselator.h File Reference

```
#include "GoTools/tesselator/Tesselator.h"
```

```
#include "GoTools/geometry/LineCloud.h"
```

Include dependency graph for LineCloudTesselator.h:



## Classes

- class [Go::LineCloudTesselator](#)

## Namespaces

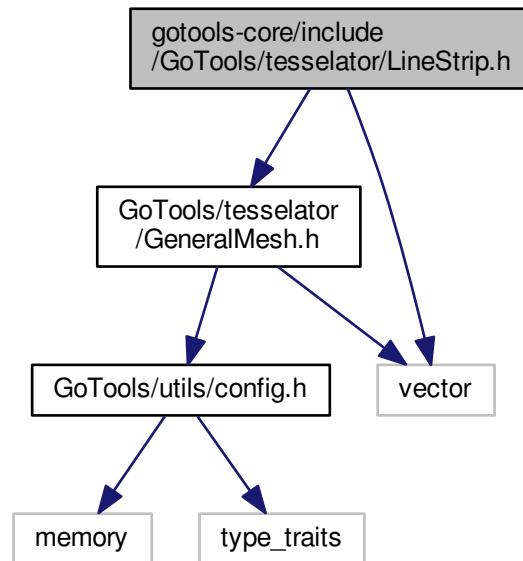
- [Go](#)

## 30.1970 gotools-core/include/GoTools/tesselator/LineStrip.h File Reference

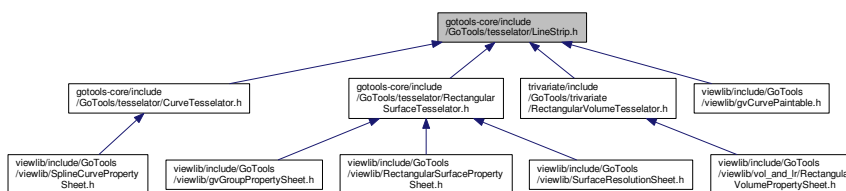
```
#include "GoTools/tesselator/GeneralMesh.h"
```

```
#include <vector>
```

Include dependency graph for LineStrip.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [Go::LineStrip](#)

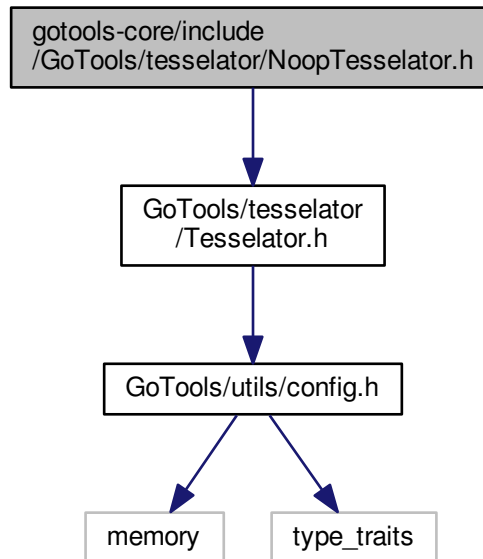
## Namespaces

- [Go](#)

## 30.1971 gotools-core/include/GoTools/tesselator/NoopTesselator.h File Reference

```
#include "GoTools/tesselator/Tesselator.h"
```

Include dependency graph for NoopTesselator.h:



### Classes

- class [Go::NoopTesselator](#)

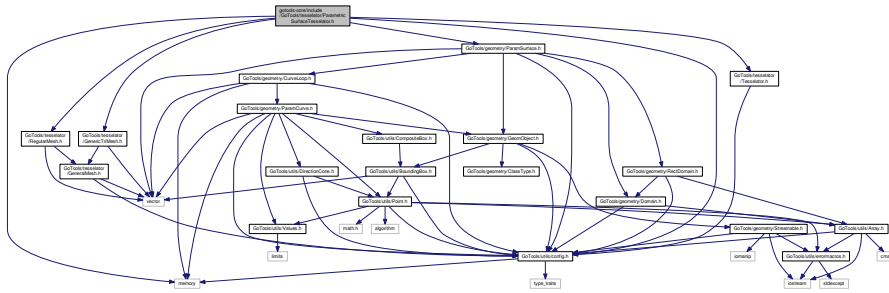
### Namespaces

- [Go](#)

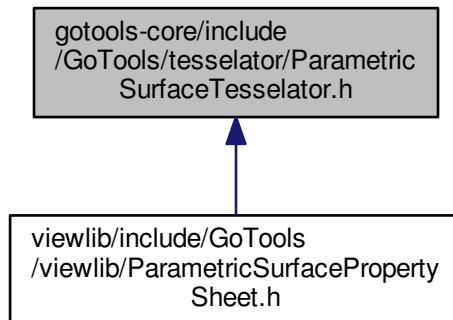
## 30.1972 gotools-core/include/GoTools/tesselator/ParametricSurfaceTesselator.h File Reference

```
#include "GoTools/tesselator/Tesselator.h"
#include "GoTools/tesselator/RegularMesh.h"
#include "GoTools/geometry/ParamSurface.h"
#include "GoTools/tesselator/GenericTriMesh.h"
#include <memory>
#include "GoTools/utils/config.h"
```

Include dependency graph for ParametricSurfaceTesselator.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [Go::ParametricSurfaceTesselator](#)

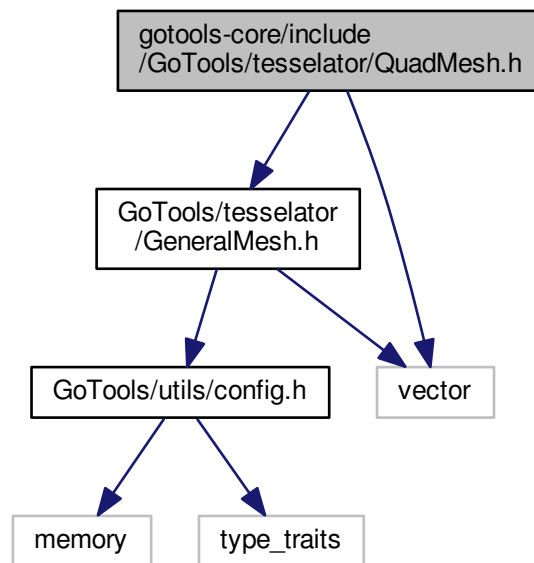
## Namespaces

- [Go](#)

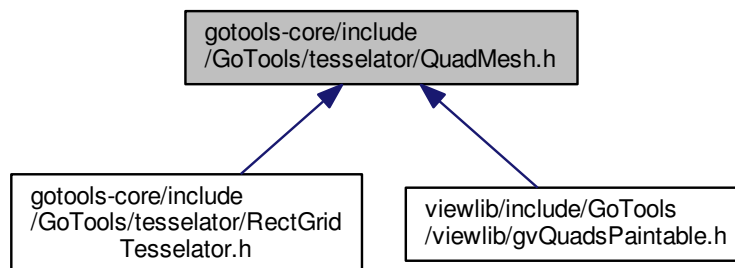
## 30.1973 gtools-core/include/GoTools/tesselator/QuadMesh.h File Reference

```
#include "GoTools/tesselator/GeneralMesh.h"
#include <vector>
```

Include dependency graph for QuadMesh.h:



This graph shows which files directly or indirectly include this file:



## Classes

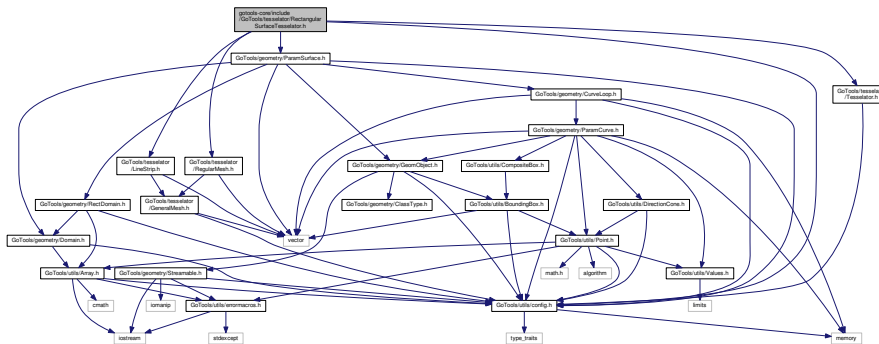
- class [Go::QuadMesh](#)

## Namespaces

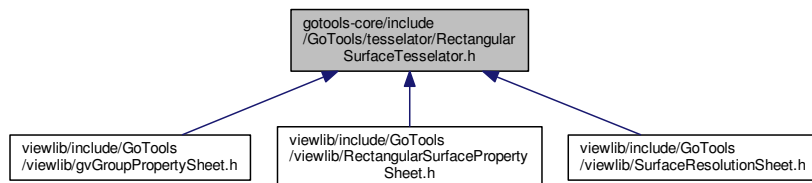
- [Go](#)

## 30.1974 gotools-core/include/GoTools/tessalator/RectangularSurfaceTessalator.h File Reference

```
#include "GoTools/tessalator/Tessalator.h"
#include "GoTools/tessalator/RegularMesh.h"
#include "GoTools/tessalator/LineStrip.h"
#include "GoTools/geometry/ParamSurface.h"
#include "GoTools/utils/config.h"
Include dependency graph for RectangularSurfaceTessalator.h:
```



This graph shows which files directly or indirectly include this file:



### Classes

- class [Go::RectangularSurfaceTessalator](#)

### Namespaces

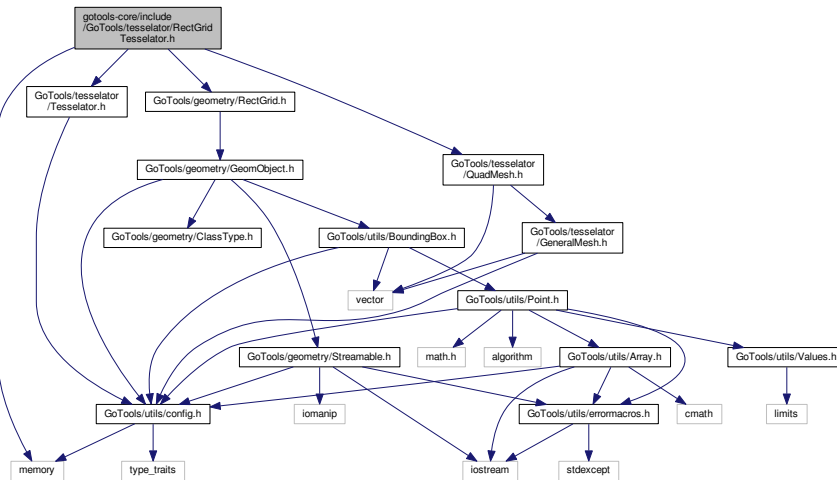
- [Go](#)



## 30.1975 gotools-core/include/GoTools/tessellator/RectGridTessellator.h File Reference

```
#include "GoTools/tessellator/Tessellator.h"
#include "GoTools/geometry/RectGrid.h"
#include "GoTools/tessellator/QuadMesh.h"
#include <memory>
```

Include dependency graph for RectGridTessellator.h:



## Classes

- class [Go::RectGridTessellator](#)

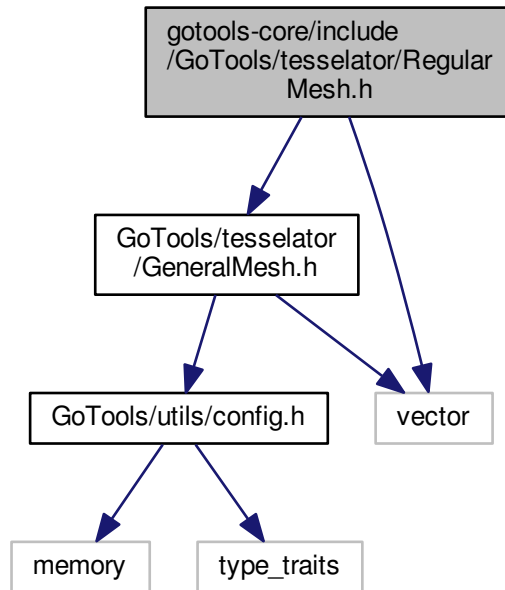
## Namespaces

- [Go](#)

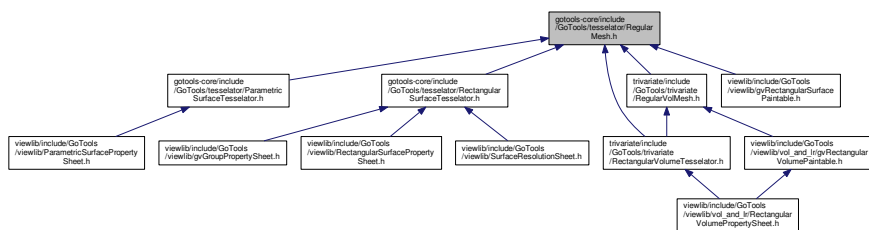
## 30.1976 gotools-core/include/GoTools/tessellator/RegularMesh.h File Reference

```
#include "GoTools/tessellator/GeneralMesh.h"
#include <vector>
```

Include dependency graph for RegularMesh.h:



This graph shows which files directly or indirectly include this file:



## Classes

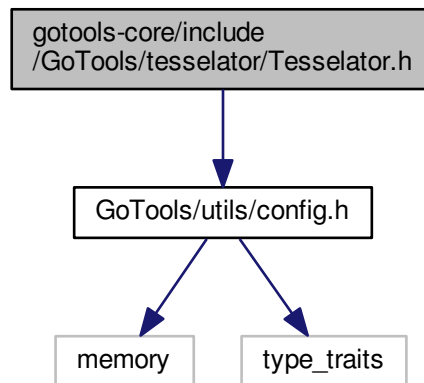
- class [Go::Triangle](#)
- class [Go::RegularMesh](#)

## Namespaces

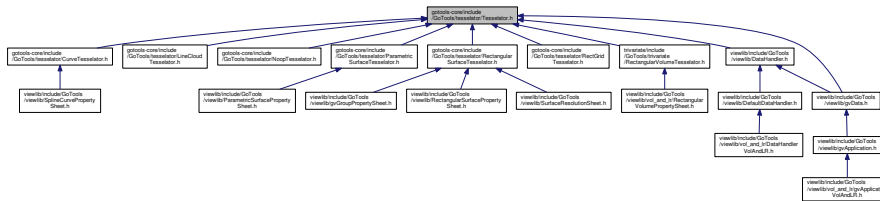
- [Go](#)



Include dependency graph for Tesselator.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [Go::Tesselator](#)

## Namespaces

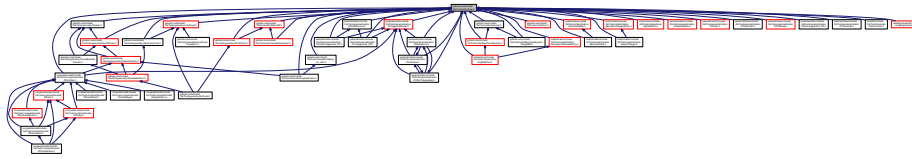
- [Go](#)

## 30.1979 gtools-core/include/GoTools/tesselator/TesselatorUtils.h File Reference

```
#include "GoTools/geometry/ParamSurface.h"
#include "GoTools/geometry/LineCloud.h"
```



This graph shows which files directly or indirectly include this file:



## Classes

- class [Go::Array< T, Dim >](#)

## Namespaces

- [Go](#)

## Typedefs

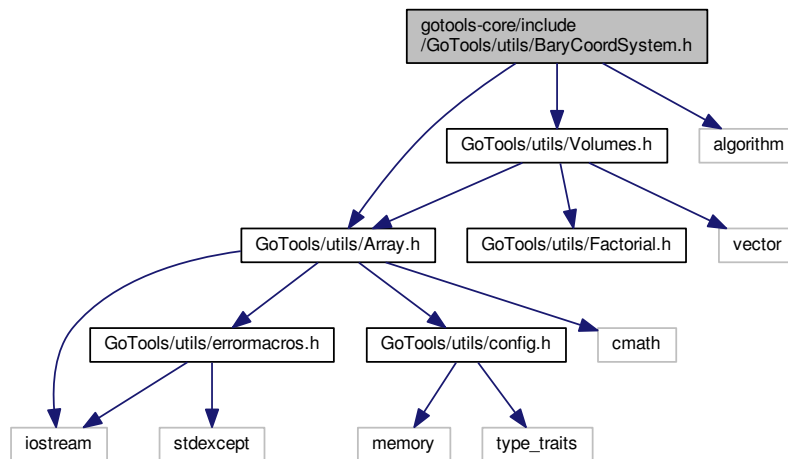
- typedef [Array< double, 2 >](#) [Go::Vector2D](#)  
*Typedef for ease of use in frequently used case.*
- typedef [Array< double, 3 >](#) [Go::Vector3D](#)  
*Typedef for ease of use in frequently used case.*
- typedef [Array< double, 4 >](#) [Go::Vector4D](#)  
*Typedef for ease of use in frequently used case.*

## Functions

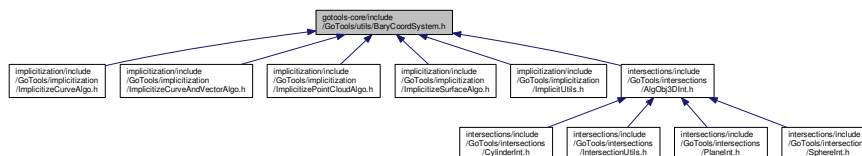
- template<typename T , int Dim>  
[Go::Array< T, Dim >](#) [Go::operator\\*](#) (T d, [const Go::Array< T, Dim >](#) &v)  
*The product of a vector and a scalar.*
- template<typename T , int Dim>  
[std::istream &](#) [Go::operator>>](#) ([std::istream &](#)is, [Go::Array< T, Dim >](#) &v)  
*Stream extraction for [Array](#).*
- template<typename T , int Dim>  
[std::ostream &](#) [Go::operator<<](#) ([std::ostream &](#)os, [const Go::Array< T, Dim >](#) &v)  
*Stream insertion for [Array](#).*

## 30.1981 gotools-core/include/GoTools/Utils/BaryCoordSystem.h File Reference

```
#include "GoTools/Utils/Array.h"
#include "GoTools/Utils/Volumes.h"
#include <algorithm>
Include dependency graph for BaryCoordSystem.h:
```



This graph shows which files directly or indirectly include this file:



## Classes

- class [Go::BaryCoordSystem< Dim >](#)

## Namespaces

- [Go](#)
- [std](#)

## Typedefs

- typedef BaryCoordSystem< 2 > [Go::BaryCoordSystem2D](#)
- typedef BaryCoordSystem< 3 > [Go::BaryCoordSystem3D](#)

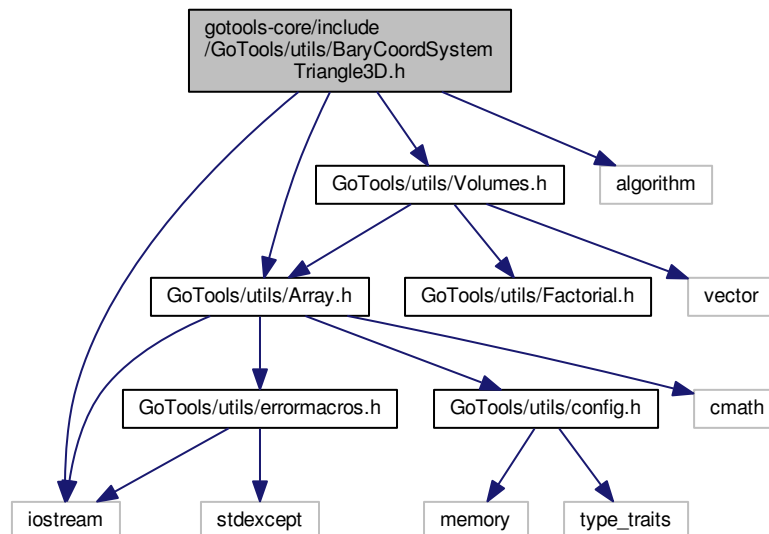
## Functions

- `template<class T , int Dim>`  
`Array< T, Dim > Go::operator* (const Array< double, Dim > &a, const T b)`
- `template<int Dim>`  
`std::istream & std::operator>> (std::istream &is, Go::BaryCoordSystem< Dim > &bc)`  
*Read BaryCoordSystem from input stream.*
- `template<int Dim>`  
`std::ostream & std::operator<< (std::ostream &os, const Go::BaryCoordSystem< Dim > &bc)`  
*Write BaryCoordSystem to output stream.*

## 30.1982 gotools-core/include/GoTools/utills/BaryCoordSystemTriangle3D.h File Reference

```
#include "GoTools/utills/Array.h"
#include "GoTools/utills/Volumes.h"
#include <iostream>
#include <algorithm>
```

Include dependency graph for BaryCoordSystemTriangle3D.h:



## Classes

- class [Go::BaryCoordSystemTriangle3D](#)

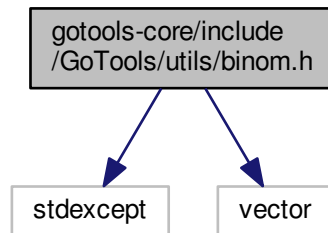
## Namespaces

- [Go](#)



## 30.1983 gotools-core/include/GoTools/utils/binom.h File Reference

```
#include <stdexcept>
#include <vector>
Include dependency graph for binom.h:
```



### Namespaces

- [Go](#)

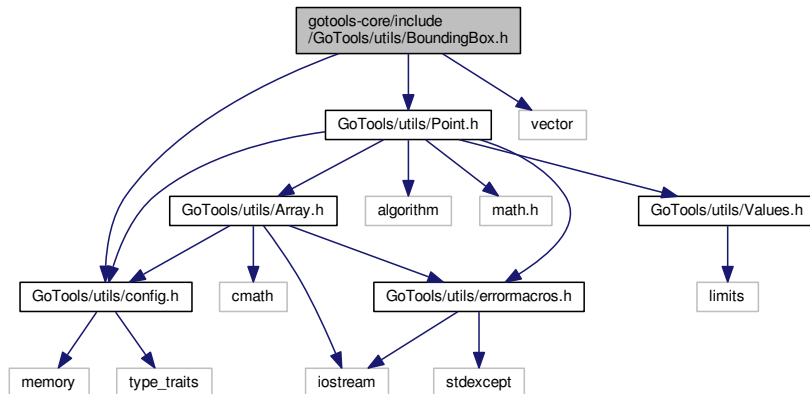
### Functions

- [double Go::binom](#) (int n, int i)  
*Computes the binomial coefficient:  $n! / (i! (n-i)!)$*
- [double Go::factorial](#) (int n)  
*computes  $n!$  ( $n$  factorial)*
- [double Go::trinomial](#) (int n, int i, int j)  
*computes the trinomial coefficient:  $n! / (i! j! (n-i-j)!)$*
- [double Go::quadrinomial](#) (int n, int i, int j, int k)  
*computes the quadrinomial coefficient:  $n! / (i! j! k! (n-i-j-k)!)$*

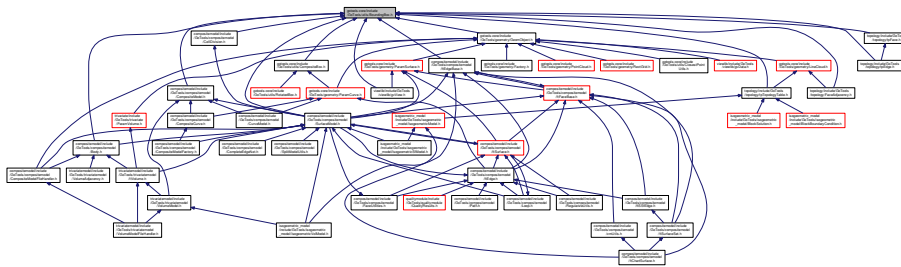
## 30.1984 gotools-core/include/GoTools/utils/BoundingBox.h File Reference

```
#include "GoTools/utils/Point.h"
#include <vector>
#include "GoTools/utils/config.h"
```

Include dependency graph for BoundingBox.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [Go::BoundingBox](#)

## Namespaces

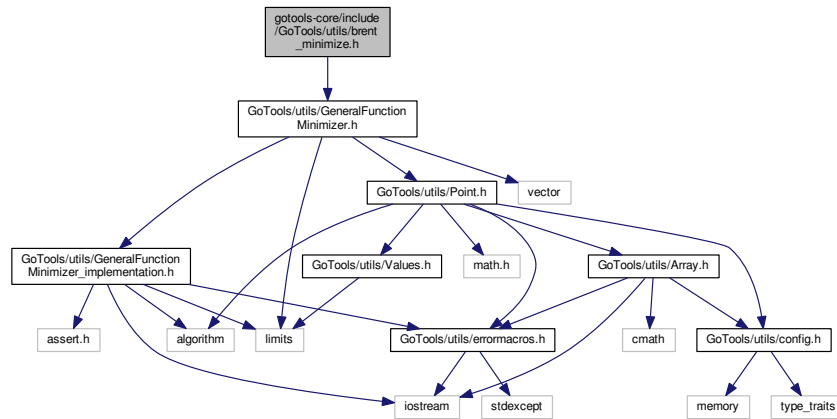
- [Go](#)
- [std](#)

## Functions

- `std::istream & std::operator>> (std::istream &is, Go::BoundingBox &bbox)`  
*Read BoundingBox from input stream.*
- `std::ostream & std::operator<< (std::ostream &os, const Go::BoundingBox &bbox)`  
*Write BoundingBox to output stream.*

## 30.1985 gotools-core/include/GoTools/utills/brent\_minimize.h File Reference

```
#include "GoTools/utills/GeneralFunctionMinimizer.h"
Include dependency graph for brent_minimize.h:
```



## Classes

- class [Go::Fun2Fun< Functor >](#)

## Namespaces

- [Go](#)

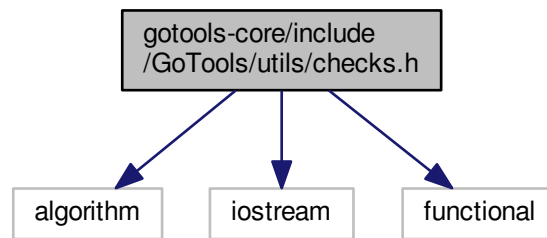
## Functions

- `template<class Functor >`  
`double Go::brent_minimize (const Functor &f, double a, double b, double c, double &parmin, const double rel_tolerance=std::sqrt(std::numeric_limits< double >::epsilon()))`

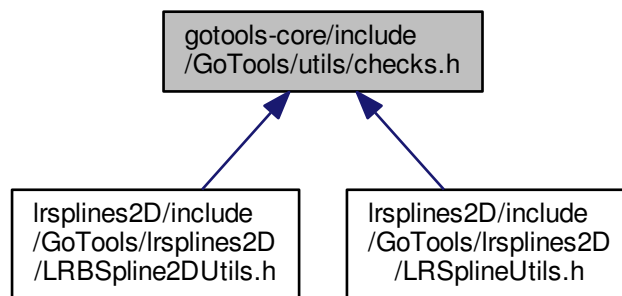
## 30.1986 gotools-core/include/GoTools/utills/checks.h File Reference

```
#include <algorithm>
#include <iostream>
#include <functional>
```

Include dependency graph for checks.h:



This graph shows which files directly or indirectly include this file:



## Namespaces

- [Go](#)

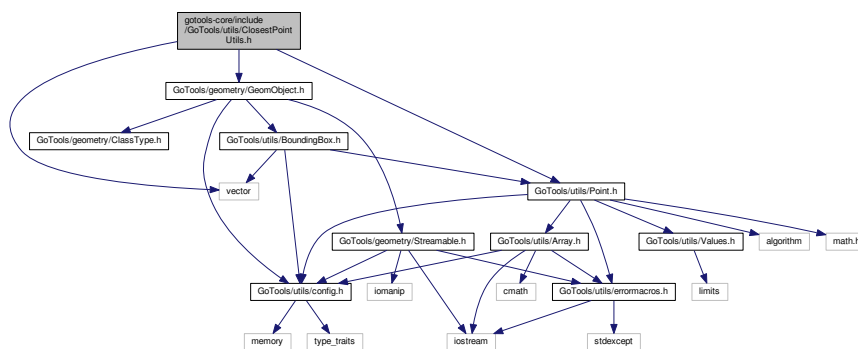
## Functions

- `template<typename Iterator >`  
`bool Go::strictly_increasing` (Iterator begin, Iterator end)
- `template<typename Array >`  
`bool Go::strictly_increasing` (const Array &A)
- `template<typename Iterator >`  
`bool Go::weakly_increasing` (Iterator begin, Iterator end)
- `template<typename Array >`  
`bool Go::weakly_increasing` (const Array &A)
- `template<typename Iterator >`  
`bool Go::strictly_decreasing` (Iterator begin, Iterator end)

- `template<typename Array >`  
`bool Go::strictly_decreasing (const Array &A)`
- `template<typename Iterator >`  
`bool Go::weakly_decreasing (Iterator begin, Iterator end)`
- `template<typename Array >`  
`bool Go::weakly_decreasing (const Array &A)`
- `template<typename ValueType >`  
`bool Go::nondecreasing (ValueType a, ValueType b, ValueType c)`
- `template<typename ValueType >`  
`bool Go::nonincreasing (ValueType a, ValueType b, ValueType c)`
- `template<typename ValueType >`  
`bool Go::strictly_decreasing (ValueType a, ValueType b, ValueType c)`
- `template<typename ValueType >`  
`bool Go::strictly_increasing (ValueType a, ValueType b, ValueType c)`
- `template<typename ValueType >`  
`bool Go::interval_overlap (ValueType front1, ValueType back1, ValueType front2, ValueType back2)`
- `template<typename Iterator >`  
`int Go::compare_seq (Iterator begin_1, Iterator end_1, Iterator begin_2, Iterator end_2)`

## 30.1987 gotools-core/include/GoTools/Utils/ClosestPointUtils.h File Reference

```
#include <vector>
#include "GoTools/Utils/Point.h"
#include "GoTools/geometry/GeomObject.h"
Include dependency graph for ClosestPointUtils.h:
```



### Classes

- class `Go::boxStructuring::SurfaceData`
- class `Go::boxStructuring::SubSurfaceBoundingBox`
- class `Go::boxStructuring::BoundingBoxStructure`

### Namespaces

- `Go`
- `Go::boxStructuring`

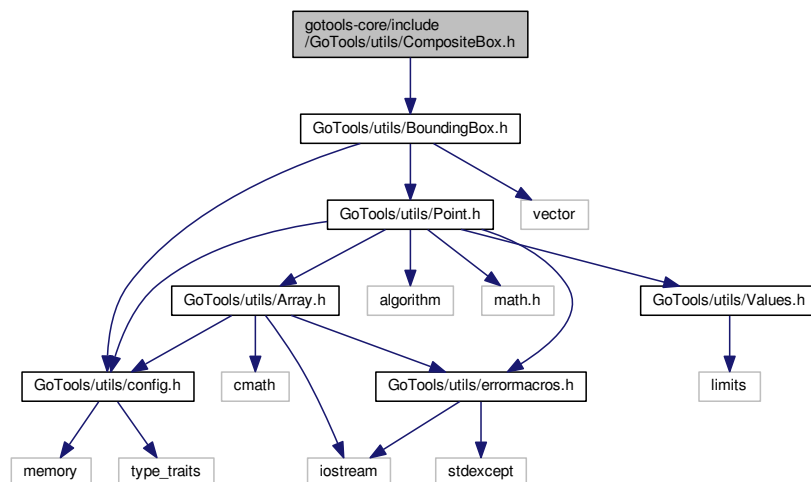
## Functions

- `shared_ptr< boxStructuring::BoundingBoxStructure > Go::preProcessClosestVectors (const std::vector< shared_ptr< GeomObject > > &surfaces, double par_len_el)`
- `void Go::closestPointSingleCalculation (int pt_idx, int start_idx, int skip, const std::vector< float > &inPoints, const std::vector< std::vector< double > > &rotationMatrix, const Point &translation, const shared_ptr< boxStructuring::BoundingBoxStructure > &boxStructure, std::vector< float > &result, std::vector< std::vector< int > > &lastBoxCall, int return_type, int search_extend)`
- `std::vector< float > Go::closestPointCalculations (const std::vector< float > &pts, const shared_ptr< boxStructuring::BoundingBoxStructure > &structure, const std::vector< std::vector< double > > &rotationMatrix, const Point &translation, int return_type, int start_idx, int skip, int max_idx, int search_extend=3, bool m_core=true)`
- `std::vector< float > Go::closestVectorsOld (const std::vector< float > &inPoints, const shared_ptr< boxStructuring::BoundingBoxStructure > &boxStructure, const std::vector< std::vector< double > > &rotationMatrix, const Point &translation, int return_type, int start_idx, int skip, int max_idx, int search_extend=3)`  
*Calculates the closest points of a point cloud to a surface model, by not using the inside polygons in closestVectors()*
- `std::vector< float > Go::closestPointCalculations (const std::vector< float > &pts, const shared_ptr< boxStructuring::BoundingBoxStructure > &structure, const std::vector< std::vector< double > > &rotationMatrix, const Point &translation, int return_type)`
- `std::vector< float > Go::closestDistances (const std::vector< float > &pts, const shared_ptr< boxStructuring::BoundingBoxStructure > &structure, const std::vector< std::vector< double > > &rotationMatrix, const Point &translation)`
- `std::vector< float > Go::closestSignedDistances (const std::vector< float > &pts, const shared_ptr< boxStructuring::BoundingBoxStructure > &structure, const std::vector< std::vector< double > > &rotationMatrix, const Point &translation)`
- `std::vector< float > Go::closestPoints (const std::vector< float > &pts, const shared_ptr< boxStructuring::BoundingBoxStructure > &structure, const std::vector< std::vector< double > > &rotationMatrix, const Point &translation)`

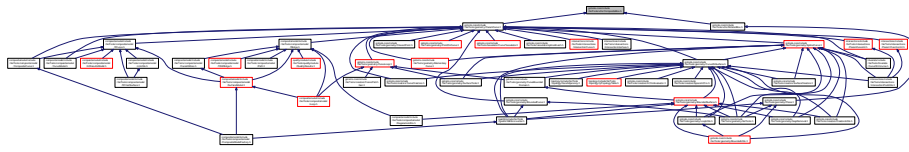
## 30.1988 gotools-core/include/GoTools/utis/CompositeBox.h File Reference

```
#include "GoTools/utis/BoundingBox.h"
```

Include dependency graph for CompositeBox.h:



This graph shows which files directly or indirectly include this file:



## Classes

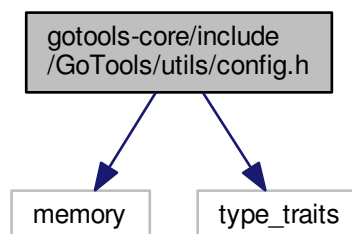
- class [Go::CompositeBox](#)

## Namespaces

- [Go](#)

## 30.1989 gotools-core/include/GoTools/utils/config.h File Reference

```
#include <memory>
#include <type_traits>
Include dependency graph for config.h:
```



## Macros

- #define [GO\\_API](#)

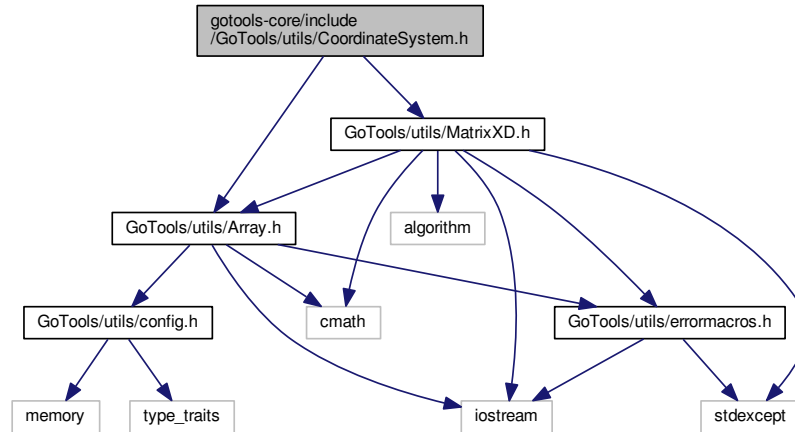
### 30.1989.1 Macro Definition Documentation

#### 30.1989.1.1 #define GO\_API

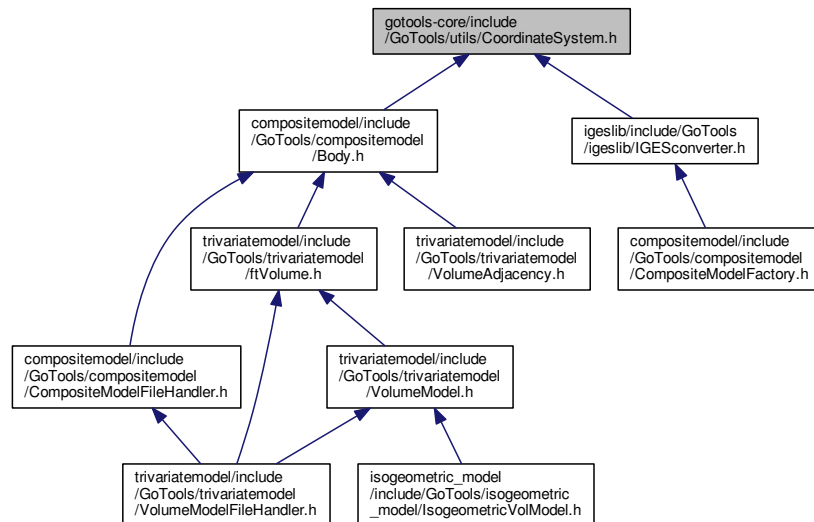
Definition at line 52 of file config.h.

## 30.1990 gotools-core/include/GoTools/utis/CoordinateSystem.h File Reference

```
#include "GoTools/utis/Array.h"
#include "GoTools/utis/MatrixXD.h"
Include dependency graph for CoordinateSystem.h:
```



This graph shows which files directly or indirectly include this file:



## Classes

- class [Go::CoordinateSystem< Dim >](#)

*Defines a Cartesian coordinate system.*

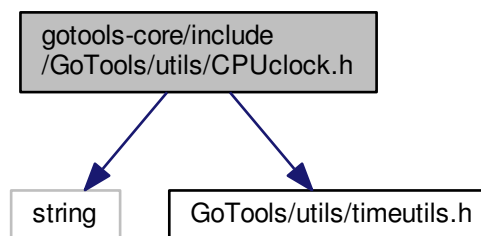


## Namespaces

- [Go](#)

## 30.1991 gotools-core/include/GoTools/Utils/CPUclock.h File Reference

```
#include <string>
#include "GoTools/Utils/timeutils.h"
Include dependency graph for CPUclock.h:
```



## Classes

- class [Go::CPUclock](#)

*A class for measuring CPU time in programs.*

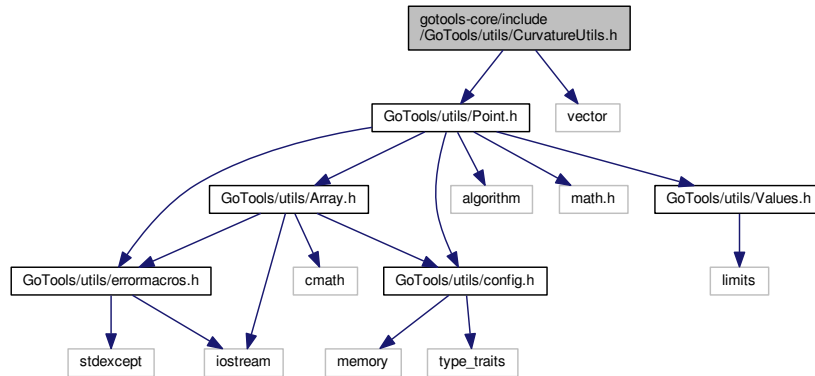
## Namespaces

- [Go](#)

## 30.1992 gotools-core/include/GoTools/Utils/CurvatureUtils.h File Reference

```
#include "GoTools/Utils/Point.h"
#include <vector>
```

Include dependency graph for CurvatureUtils.h:



## Namespaces

- [Go](#)

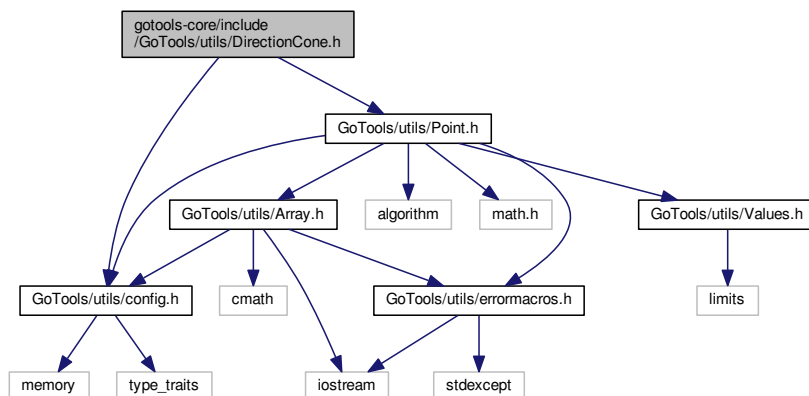
## Functions

- `double Go::curvatureRadius (const std::vector< Point > &der, std::vector< Point > &unitder)`
- `double Go::stepLenFromRadius (double radius, double aepsge)`
- `double Go::tanLenFromRadius (double radius, double angle)`
- `void Go::getHermiteData (const std::vector< Point > &der1, const std::vector< Point > &der2, double &parint, double &len1, double &len2)`

## 30.1993 gotools-core/include/GoTools/utis/DirectionCone.h File Reference

```
#include "GoTools/utis/Point.h"
#include "GoTools/utis/config.h"
```

Include dependency graph for DirectionCone.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [Go::DirectionCone](#)

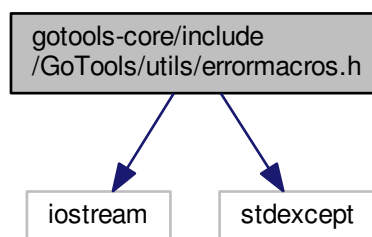
## Namespaces

- [Go](#)

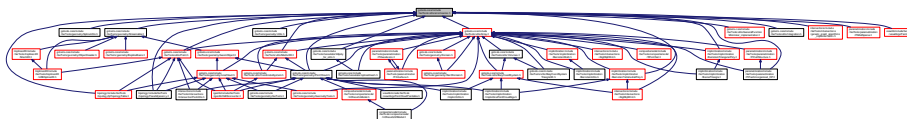
## 30.1994 gotools-core/include/GoTools/utils/errormacros.h File Reference

```
#include <iostream>
#include <stdexcept>
```

Include dependency graph for errormacros.h:



This graph shows which files directly or indirectly include this file:



## Macros

- #define [REPORT](#) std::cerr << "\nIn file " << \_\_FILE\_\_ << ", line " << \_\_LINE\_\_ << std::endl
- #define [MESSAGE](#)(x) std::cerr << "\nIn file " << \_\_FILE\_\_ << ", line " << \_\_LINE\_\_ << ": " << x << std::endl
- #define [MESSAGE\\_IF](#)(cond, m) do {if(cond) [MESSAGE](#)(m);} while(0)
- #define [THROW](#)(x) [MESSAGE](#)(x), throw std::exception()
- #define [GO\\_NO\\_CHECKS](#)
- #define [ALWAYS\\_ERROR\\_IF](#)(condition, message) do {if(condition){ [THROW](#)(message);} } while(0)
- #define [ASSERT](#)(cond) if (!(cond)) [THROW](#)("Assertion \"' #cond \"' failed.")
- #define [ASSERT2](#)(cond, x) do { if (!(cond)) [THROW](#)(x);} while(0)
- #define [DEBUG\\_ERROR\\_IF](#)(cond, x) do { if (cond) [THROW](#)(x); } while(0)

### 30.1994.1 Macro Definition Documentation

30.1994.1.1 `#define ALWAYS_ERROR_IF( condition, message ) do {if(condition){ THROW(message);}} while(0)`

Definition at line 83 of file `errormacros.h`.

30.1994.1.2 `#define ASSERT( cond ) if (!(cond)) THROW("Assertion \"' #cond \"' failed.")`

Usage: `ASSERT(condition)` Usage: `ASSERT2(condition, "Error message string.")` Usage: `DEBUG_ERROR_IF(condition, "Error message string.");`

Definition at line 100 of file `errormacros.h`.

30.1994.1.3 `#define ASSERT2( cond, x ) do { if (!(cond)) THROW(x);} while(0)`

Definition at line 103 of file `errormacros.h`.

30.1994.1.4 `#define DEBUG_ERROR_IF( cond, x ) do { if (cond) THROW(x); } while(0)`

Definition at line 107 of file `errormacros.h`.

30.1994.1.5 `#define GO_NO_CHECKS`

Definition at line 79 of file `errormacros.h`.

30.1994.1.6 `#define MESSAGE( x ) std::cerr << "\nIn file " << __FILE__ << ", line " << __LINE__ << ": " << x << std::endl`

Definition at line 68 of file `errormacros.h`.

30.1994.1.7 `#define MESSAGE_IF( cond, m ) do {if(cond) MESSAGE(m);} while(0)`

Definition at line 71 of file `errormacros.h`.

30.1994.1.8 `#define REPORT std::cerr << "\nIn file " << __FILE__ << ", line " << __LINE__ << std::endl`

Usage: `REPORT`; Usage: `MESSAGE("Message string.");` Usage: `THROW("Error message string.");`

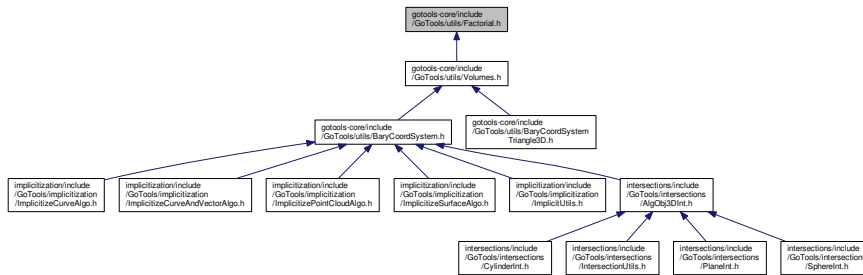
Definition at line 65 of file `errormacros.h`.

30.1994.1.9 `#define THROW( x ) MESSAGE(x), throw std::exception()`

Definition at line 74 of file `errormacros.h`.

### 30.1995 gotools-core/include/GoTools/utis/Factorial.h File Reference

This graph shows which files directly or indirectly include this file:



#### Classes

- struct [Go::Factorial< N >](#)  
*Compile-time factorial calculations.*
- struct [Go::Factorial< 1 >](#)
- struct [Go::InverseFactorial< T, N >](#)

#### Namespaces

- [Go](#)

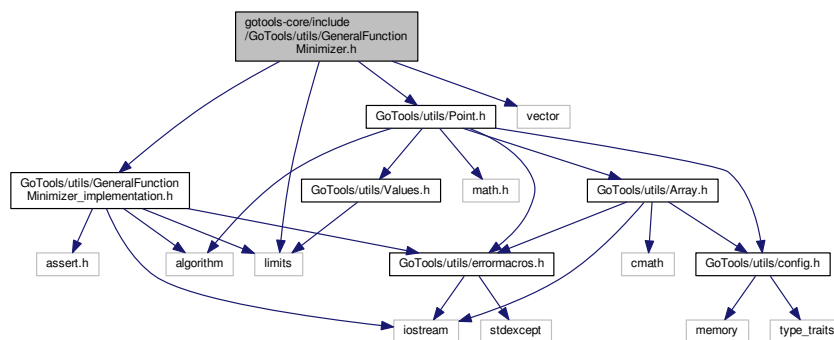
### 30.1996 gotools-core/include/GoTools/utis/GeneralFunctionMinimizer.h File Reference

```

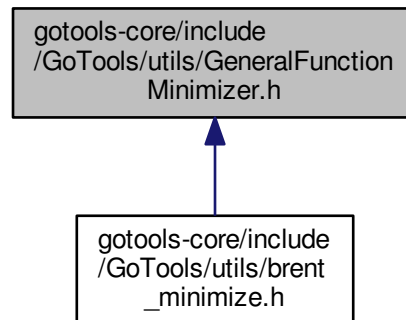
#include "GoTools/utis/Point.h"
#include <vector>
#include <limits>
#include "GoTools/utis/GeneralFunctionMinimizer_implementation.h"

```

Include dependency graph for GeneralFunctionMinimizer.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [Go::FunctionMinimizer< Functor >](#)
- class [Go::FunctionMinimizer< Functor >](#)

## Namespaces

- [Go](#)

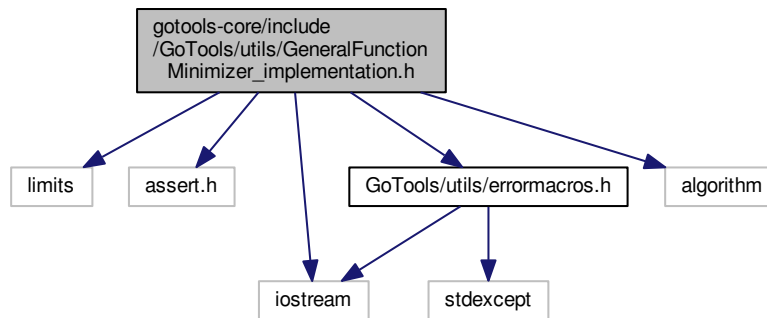
## Functions

- [template<class Functor >](#)  
[void Go::minimise\\_conjugated\\_gradient](#) (FunctionMinimizer< Functor > &dfmin)

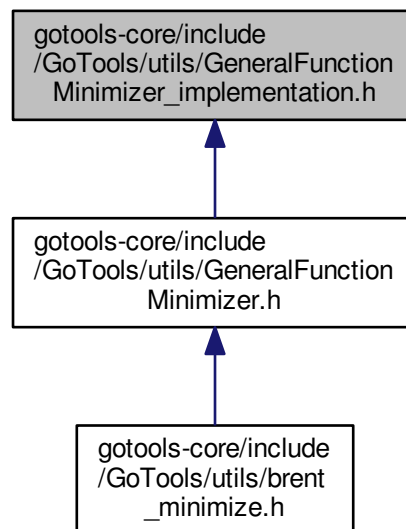
## 30.1997 gtools-core/include/GoTools/Utils/GeneralFunctionMinimizer\_implementation.h File Reference

```
#include <limits>
#include <assert.h>
#include "GoTools/Utils/errormacros.h"
#include <algorithm>
#include <iostream>
```

Include dependency graph for GeneralFunctionMinimizer\_implementation.h:



This graph shows which files directly or indirectly include this file:



## Namespaces

- [Go](#)

## Functions

- `template<class Functor >`  
void [Go::minimise\\_conjugated\\_gradient](#) (FunctionMinimizer< Functor > &dfmin)

## 30.1998 gtools-core/include/GoTools/utils/gtools-doxymain.h File Reference

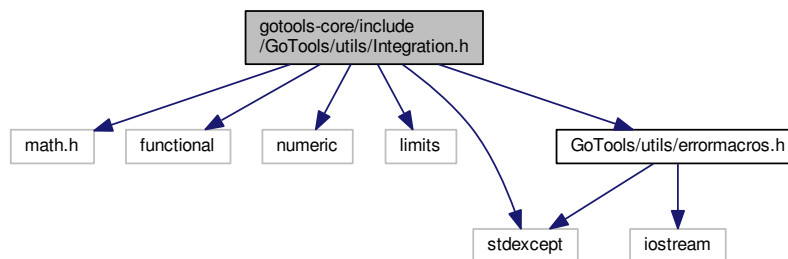
### Namespaces

- [Go](#)

## 30.1999 gtools-core/include/GoTools/utils/Integration.h File Reference

```
#include <math.h>
#include <functional>
#include <numeric>
#include <limits>
#include <stdexcept>
#include "GoTools/utils/errormacros.h"
```

Include dependency graph for Integration.h:



### Namespaces

- [Go](#)

### Functions

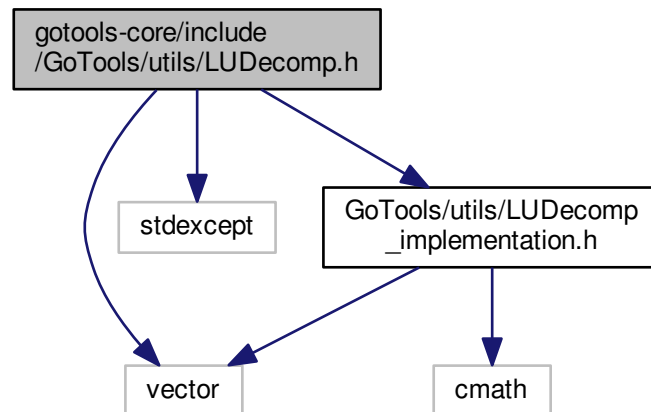
- `template<typename Functor >`  
void `Go::trapezoidal` (Functor &f, double a, double b, double &s, int n)
- `template<typename Functor >`  
double `Go::simpsons_rule` (Functor &f, double a, double b, const double eps=1.0e-6, const int max\_iter=20)
- `template<typename Functor >`  
double `Go::gaussian_quadrature` (Functor &f, double a, double b)
- `template<typename Functor2D >`  
double `Go::simpsons_rule2D` (Functor2D &f, double ax, double bx, double ay, double by, const double eps=1.0e-6, const int max\_iter=20)
- `template<typename Functor2D >`  
double `Go::gaussian_quadrature2D` (Functor2D &f, double ax, double bx, double ay, double by)



## 30.2000 gotools-core/include/GoTools/Utils/LUDecomp.h File Reference

```
#include <vector>
#include <stdexcept>
#include "GoTools/Utils/LUDecomp_implementation.h"
```

Include dependency graph for LUDecomp.h:



## Namespaces

- [Go](#)

## Functions

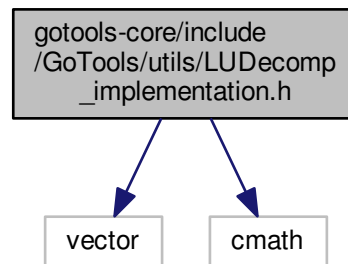
- `template<typename SquareMatrix >`  
void [Go::LUDecomp](#) (SquareMatrix &mat, int num\_rows, int \*perm, bool &parity)
- `template<typename SquareMatrix , typename T >`  
void [Go::LUSolveSystem](#) (SquareMatrix &A, int num\_unknowns, T \*vec)
- `template<typename SquareMatrix , typename T >`  
void [Go::forwardSubstitution](#) (const SquareMatrix &L, T \*x, int num\_unknowns)
- `template<typename SquareMatrix , typename T >`  
void [Go::backwardSubstitution](#) (const SquareMatrix &U, T \*x, int num\_unknowns)
- `template<typename SquareMatrix >`  
void [Go::forwardSubstitution](#) (const SquareMatrix &L, std::vector< double > \*x, int num\_unknowns)
- `template<typename SquareMatrix >`  
void [Go::backwardSubstitution](#) (const SquareMatrix &U, std::vector< double > \*x, int num\_unknowns)

## 30.2001 gotools-core/include/GoTools/utils/LUDecomp\_implementation.h File Reference

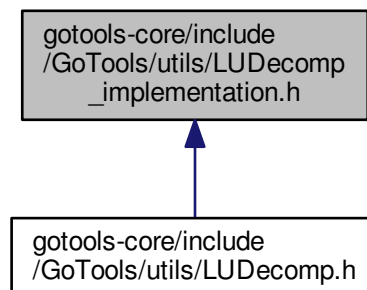
```
#include <vector>
```

```
#include <cmath>
```

Include dependency graph for LUDecomp\_implementation.h:



This graph shows which files directly or indirectly include this file:



### Namespaces

- [Go](#)

### Functions

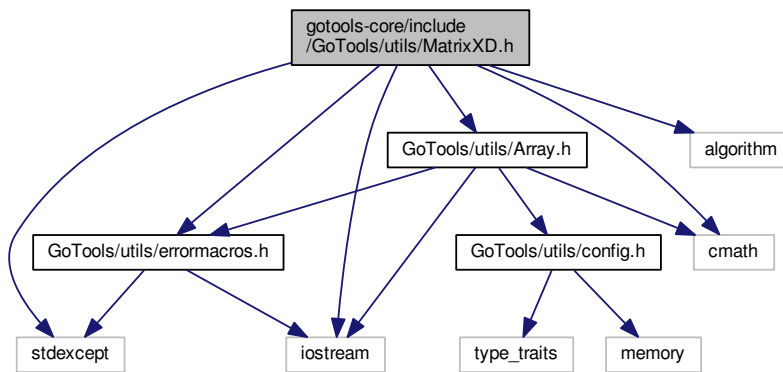
- `template<typename SquareMatrix >`  
void [Go::LUDecomp](#) (SquareMatrix &mat, int num\_rows, int \*perm, bool &parity)
- `template<typename SquareMatrix , typename T >`  
void [Go::LUsolveSystem](#) (SquareMatrix &A, int num\_unknowns, T \*vec)

- template<typename SquareMatrix , typename T >  
void `Go::forwardSubstitution` (const SquareMatrix &L, T \*x, int num\_unknowns)
- template<typename SquareMatrix >  
void `Go::forwardSubstitution` (const SquareMatrix &L, std::vector< double > \*x, int num\_unknowns)
- template<typename SquareMatrix , typename T >  
void `Go::backwardSubstitution` (const SquareMatrix &U, T \*x, int num\_unknowns)
- template<typename SquareMatrix >  
void `Go::backwardSubstitution` (const SquareMatrix &U, std::vector< double > \*x, int num\_unknowns)

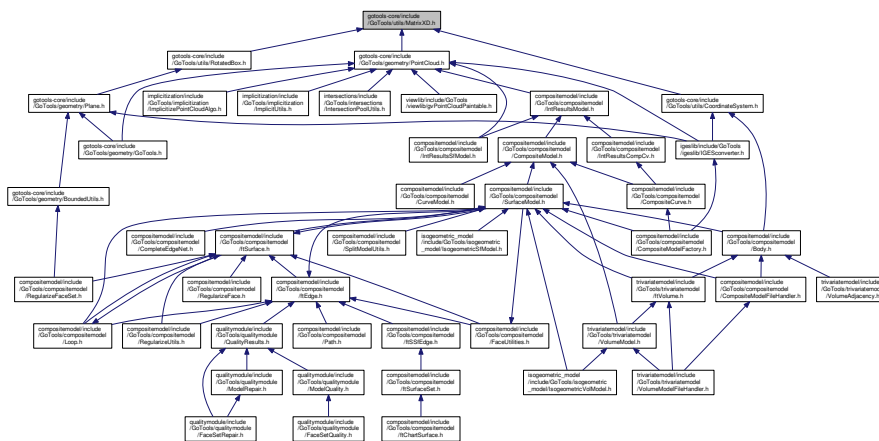
### 30.2002 gotools-core/include/GoTools/utills/MatrixXD.h File Reference

```
#include "GoTools/utills/errormacros.h"
#include "GoTools/utills/Array.h"
#include <algorithm>
#include <iostream>
#include <cmath>
#include <stdexcept>
```

Include dependency graph for MatrixXD.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [Go::MatrixXD< T, Dim >](#)

## Namespaces

- [Go](#)

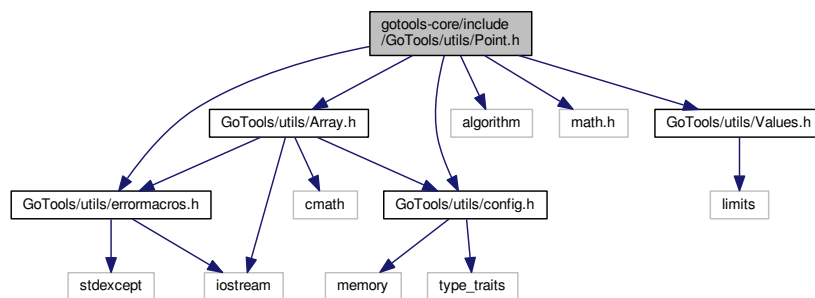
## Functions

- `template<typename T , int Dim>`  
`std::ostream & Go::operator<< (std::ostream &os, const MatrixXD< T, Dim > &m)`  
*output operator*

## 30.2003 gotools-core/include/GoTools/Utils/Point.h File Reference

```
#include "GoTools/Utils/Array.h"
#include "GoTools/Utils/errormacros.h"
#include <algorithm>
#include <math.h>
#include "GoTools/Utils/config.h"
#include "GoTools/Utils/Values.h"
```

Include dependency graph for Point.h:



## Classes

- class [Go::Point](#)

## Namespaces

- [Go](#)

## Functions

- [Point Go::operator\\*](#) ([double](#) d, [const Point &p](#))  
*The product of a vector and a scalar.*
- [std::istream & Go::operator>>](#) ([std::istream &is](#), [Go::Point &v](#))  
*Stream extraction for [Point](#).*
- [std::ostream & Go::operator<<](#) ([std::ostream &os](#), [const Go::Point &v](#))  
*Stream insertion for [Point](#).*
- [bool Go::operator<](#) ([const Point &p1](#), [const Point &p2](#))  
*Less than operator.*
- [bool Go::operator==](#) ([const Point &p1](#), [const Point &p2](#))  
*Equal operator.*

## 30.2004 gotools-core/include/GoTools/utils/randomnoise.h File Reference

### Namespaces

- [Go](#)

### Functions

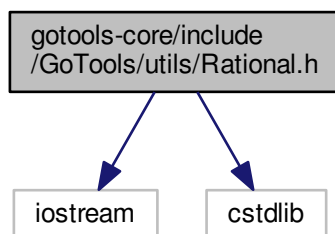
- void [Go::normalNoise](#) ([double \\*res](#), [double mean\\_err](#), [int num\\_samples](#))
- void [Go::uniformNoise](#) ([double \\*res](#), [double lval](#), [double uval](#), [int num\\_samples](#))

## 30.2005 gotools-core/include/GoTools/utils/Rational.h File Reference

```
#include <iostream>
```

```
#include <cstdlib>
```

Include dependency graph for Rational.h:



### Classes

- class [Go::Rational](#)

## Namespaces

- [Go](#)

## Functions

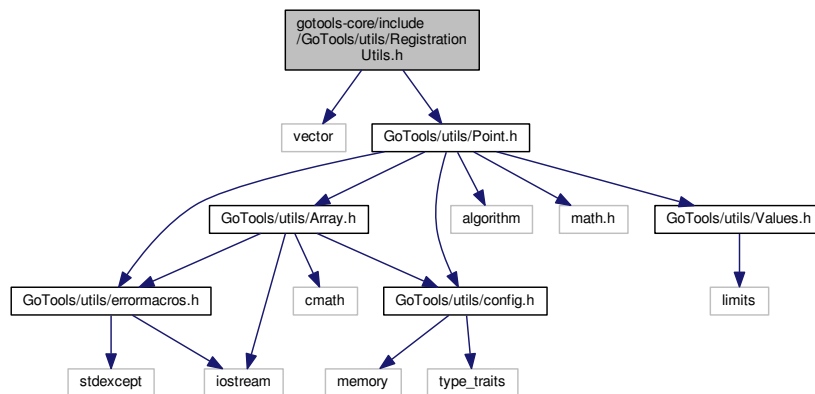
- Rational [Go::operator+](#) (`const Rational &r1`, `const Rational r2`)
- Rational [Go::operator-](#) (`const Rational &r1`, `const Rational r2`)
- Rational [Go::operator\\*](#) (`const Rational &r1`, `const Rational r2`)
- Rational [Go::operator/](#) (`const Rational &r1`, `const Rational r2`)
- `std::ostream &` [Go::operator<<](#) (`std::ostream &os`, `const Rational &p`)

## 30.2006 gotools-core/include/GoTools/Utils/RegistrationUtils.h File Reference

```
#include <vector>
```

```
#include "GoTools/Utils/Point.h"
```

Include dependency graph for RegistrationUtils.h:



## Classes

- struct [Go::RegistrationInput](#)  
*Struct for input to registration, either raw, fine or combined.*
- struct [Go::RegistrationResult](#)  
*Struct for result from registration process, either raw, fine or combined.*

## Namespaces

- [Go](#)

## Enumerations

- enum [Go::RegistrationReturnType](#) {  
[Go::RegistrationOK](#), [Go::TooFewPoints](#), [Go::PointSetSizeDiff](#), [Go::AreaTooSmall](#),  
[Go::SolveFailed](#) }

## Functions

- RegistrationResult `Go::rawRegistration` (`const std::vector< Point > &points_fixed`, `const std::vector< Point > &points_transform`, `bool allow_rescaling`, RegistrationInput params)
- void `Go::addToLinearSystem` (`int pt_idx`, `const std::vector< Point > &points_fixed`, `const std::vector< Point > &points_transform`, `bool allow_rescaling`, `const std::vector< std::vector< double > > &id`, `const Point &fine_R`, `const Point &fine_T`, `double fine_s`, `const std::vector< std::vector< double > > &m_rot_R`, `double s2`, `double R2`, `bool zero_R`, `const std::vector< std::vector< std::vector< double > > > &lhs_matrix`, `const std::vector< std::vector< double > > &rhs_matrix`)

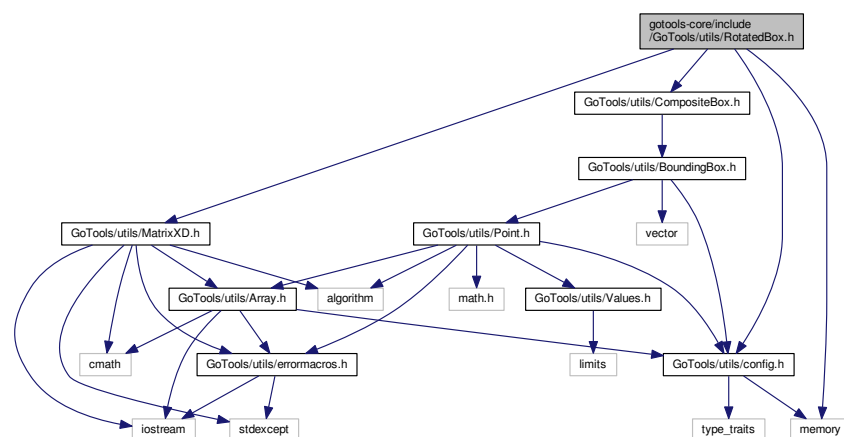
*Add the contribution from one single point to the linear system coefficients used during fine registration.*

- RegistrationResult `Go::fineRegistration` (`const std::vector< Point > &points_fixed`, `const std::vector< Point > &points_transform`, `bool allow_rescaling`, RegistrationInput params)
- RegistrationResult `Go::registration` (`const std::vector< Point > &points_fixed`, `const std::vector< Point > &points_transform`, `bool allow_rescaling`, RegistrationInput params)

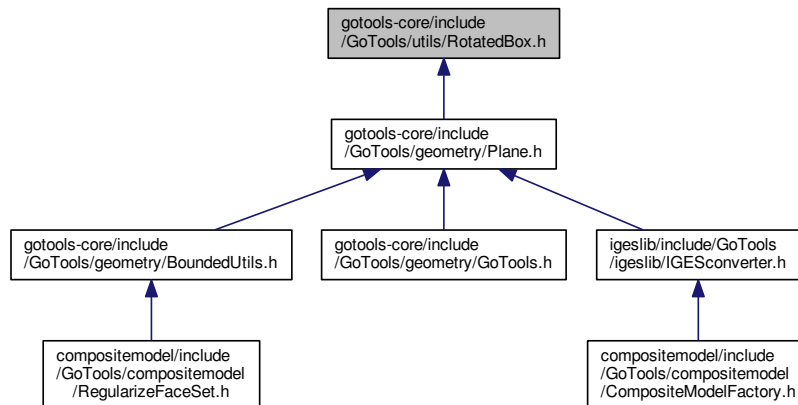
## 30.2007 gotools-core/include/GoTools/Utils/RotatedBox.h File Reference

```
#include "GoTools/Utils/CompositeBox.h"
#include "GoTools/Utils/MatrixXD.h"
#include "GoTools/Utils/config.h"
#include <memory>
```

Include dependency graph for RotatedBox.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [Go::RotatedBox](#)

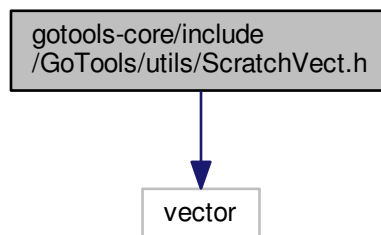
## Namespaces

- [Go](#)

## 30.2008 gotools-core/include/GoTools/Utils/ScratchVect.h File Reference

```
#include <vector>
```

Include dependency graph for ScratchVect.h:



This graph shows which files directly or indirectly include this file:





## Classes

- class [Go::ScratchVect< T, N >](#)

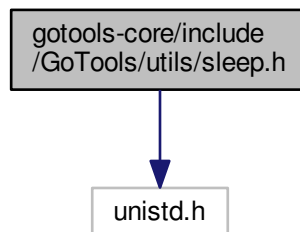
## Namespaces

- [Go](#)

## 30.2009 gotools-core/include/GoTools/utils/sleep.h File Reference

```
#include <unistd.h>
```

Include dependency graph for sleep.h:



## Macros

- #define [msleep](#)(t) `usleep((t) * 1000)`

### 30.2009.1 Macro Definition Documentation

#### 30.2009.1.1 #define `msleep( t ) usleep((t) * 1000)`

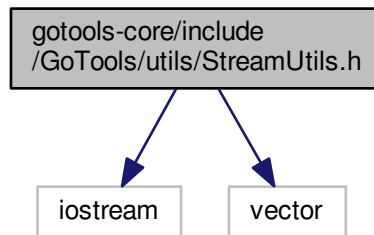
Definition at line 57 of file `sleep.h`.

## 30.2010 gotools-core/include/GoTools/Utils/StreamUtils.h File Reference

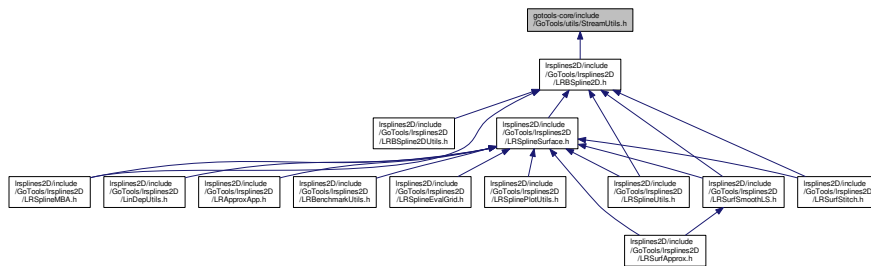
```
#include <iostream>
```

```
#include <vector>
```

Include dependency graph for StreamUtils.h:



This graph shows which files directly or indirectly include this file:



## Functions

- `template<typename T >`  
void `object_to_stream` (std::ostream &os, const T &obj)
- `template<typename T >`  
void `object_to_stream` (std::wostream &os, const T &obj)
- `template<typename T >`  
void `object_from_stream` (std::istream &is, T &obj)
- `template<typename T >`  
void `object_from_stream` (std::wistream &is, T &obj)
- `template<typename T >`  
void `object_to_stream` (std::ostream &os, const std::vector< T > &v)
- `template<typename T >`  
void `object_from_stream` (std::istream &is, std::vector< T > &v)
- `template<typename T >`  
void `object_to_stream` (std::wostream &os, const std::vector< T > &v)
- `template<typename T >`  
void `object_from_stream` (std::wistream &is, std::vector< T > &v)

## 30.2010.1 Function Documentation

30.2010.1.1 `template<typename T> void object_from_stream ( std::istream & is, T & obj )`

Definition at line 59 of file StreamUtils.h.

30.2010.1.2 `template<typename T> void object_from_stream ( std::wistream & is, T & obj )`

Definition at line 60 of file StreamUtils.h.

30.2010.1.3 `template<typename T> void object_from_stream ( std::istream & is, std::vector< T > & v )`

Definition at line 80 of file StreamUtils.h.

30.2010.1.4 `template<typename T> void object_from_stream ( std::wistream & is, std::vector< T > & v )`

Definition at line 103 of file StreamUtils.h.

30.2010.1.5 `template<typename T> void object_to_stream ( std::ostream & os, const T & obj )`

Definition at line 55 of file StreamUtils.h.

30.2010.1.6 `template<typename T> void object_to_stream ( std::wostream & os, const T & obj )`

Definition at line 56 of file StreamUtils.h.

30.2010.1.7 `template<typename T> void object_to_stream ( std::ostream & os, const std::vector< T > & v )`

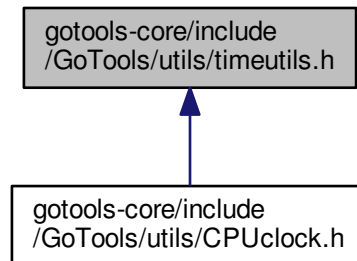
Definition at line 69 of file StreamUtils.h.

30.2010.1.8 `template<typename T> void object_to_stream ( std::wostream & os, const std::vector< T > & v )`

Definition at line 92 of file StreamUtils.h.

## 30.2011 gotools-core/include/GoTools/utls/timeutils.h File Reference

This graph shows which files directly or indirectly include this file:



### Namespaces

- [Go](#)

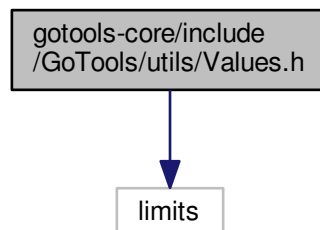
### Functions

- [double Go::getCurrentTime \(\)](#)  
*Number of seconds since some (probably system-dependent) epoch.*
- [void Go::systemSleep \(double sleep\\_time\)](#)  
*Sleep for sleep\_time seconds.*

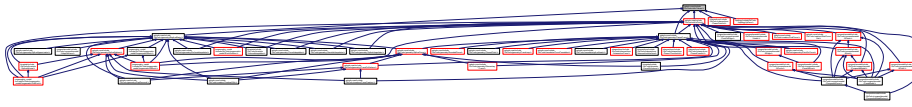
## 30.2012 gotools-core/include/GoTools/utls/Values.h File Reference

```
#include <limits>
```

Include dependency graph for Values.h:



This graph shows which files directly or indirectly include this file:



## Macros

- `#define` [DEFAULT\\_SPACE\\_EPSILON](#) 1e-10
- `#define` [DEFAULT\\_PARAMETER\\_EPSILON](#) 1e-10

## Variables

- `const int` [MAXINT](#) = `std::numeric_limits<int>::max()`
- `const double` [MAXDOUBLE](#) = `std::numeric_limits<double>::max()`
- `const double` [M\\_PI](#) = 3.14159265358979323846

### 30.2012.1 Detailed Description

Defines the constants `MAXINT`, `MAXDOUBLE` and `M_PI` if they are not defined by system.

### 30.2012.2 Macro Definition Documentation

#### 30.2012.2.1 `#define` [DEFAULT\\_PARAMETER\\_EPSILON](#) 1e-10

Definition at line 71 of file `Values.h`.

#### 30.2012.2.2 `#define` [DEFAULT\\_SPACE\\_EPSILON](#) 1e-10

Definition at line 66 of file `Values.h`.

### 30.2012.3 Variable Documentation

#### 30.2012.3.1 `const double` [M\\_PI](#) = 3.14159265358979323846

Definition at line 62 of file `Values.h`.

#### 30.2012.3.2 `const double` [MAXDOUBLE](#) = `std::numeric_limits<double>::max()`

Definition at line 58 of file `Values.h`.

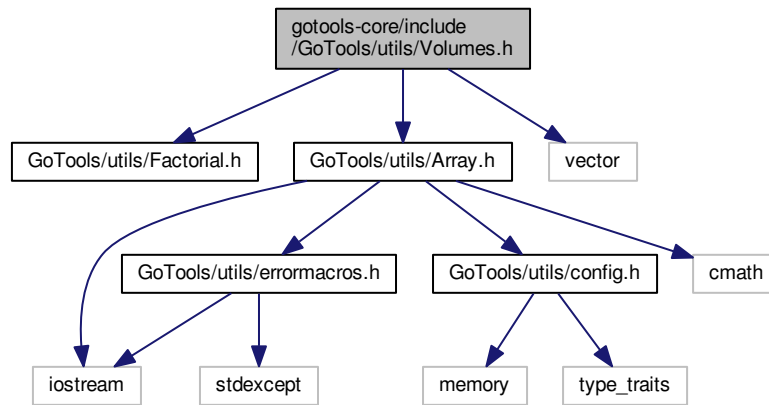
30.2012.3.3 `const int MAXINT = std::numeric_limits<int>::max()`

Definition at line 53 of file Values.h.

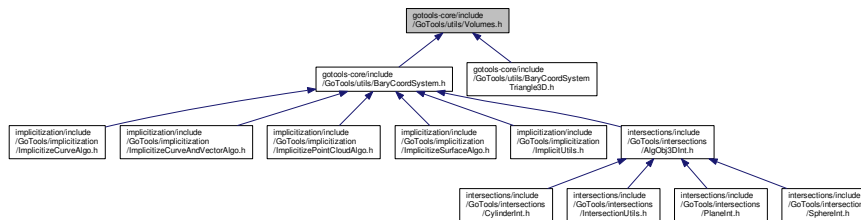
### 30.2013 gotools-core/include/GoTools/utills/Volumes.h File Reference

```
#include "GoTools/utills/Factorial.h"
#include "GoTools/utills/Array.h"
#include <vector>
```

Include dependency graph for Volumes.h:



This graph shows which files directly or indirectly include this file:



### Namespaces

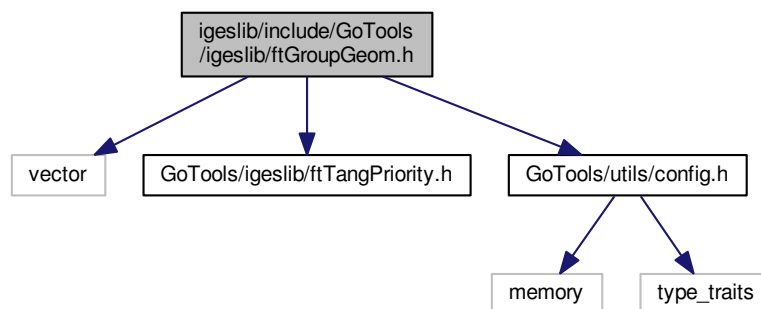
- [Go](#)

## Functions

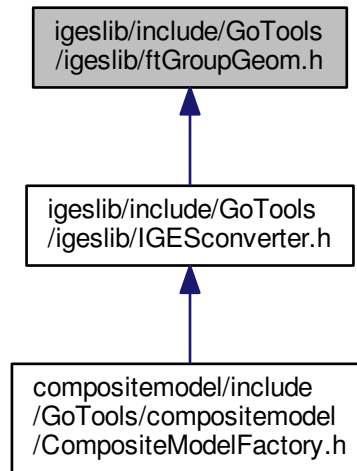
- `template<typename T >`  
`T Go::determinantOf (const Array< T, 2 > *a)`
- `template<typename T >`  
`T Go::determinantOf (const Array< T, 3 > *a)`
- `template<typename T, int Dim>`  
`T Go::simplex_volume (const Array< T, Dim > *a)`
- `template<typename T >`  
`T Go::area (const Array< T, 2 > *c)`
- `template<typename T >`  
`T Go::area (const Array< T, 3 > *c)`
- `template<typename T >`  
`T Go::volume (const Array< T, 3 > *c)`  
*Computes the volume of a 3D simplex (embedded i 3D space).*
- `template<typename T >`  
`T Go::signed_area (const Array< T, 3 > *c, const Array< T, 3 > &normal)`

## 30.2014 igeslib/include/GoTools/igeslib/ftGroupGeom.h File Reference

```
#include <vector>
#include "GoTools/igeslib/ftTangPriority.h"
#include "GoTools/utils/config.h"
Include dependency graph for ftGroupGeom.h:
```



This graph shows which files directly or indirectly include this file:



## Classes

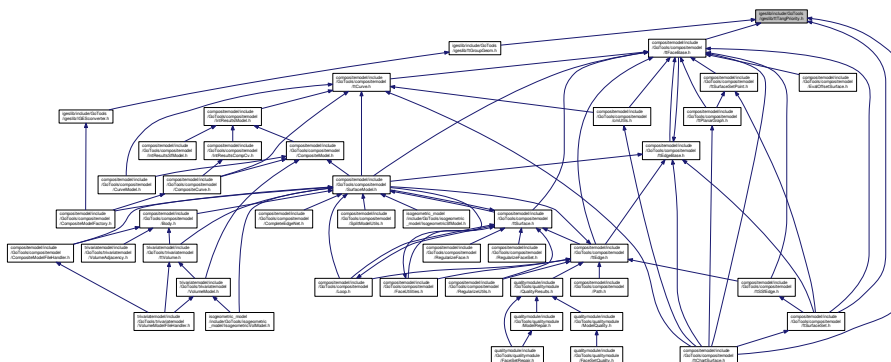
- class [Go::ftGroupGeom](#)  
*A group of geometrical objects.*

## Namespaces

- [Go](#)

## 30.2015 igeslib/include/GoTools/igeslib/ftTangPriority.h File Reference

This graph shows which files directly or indirectly include this file:





## Namespaces

- [Go](#)

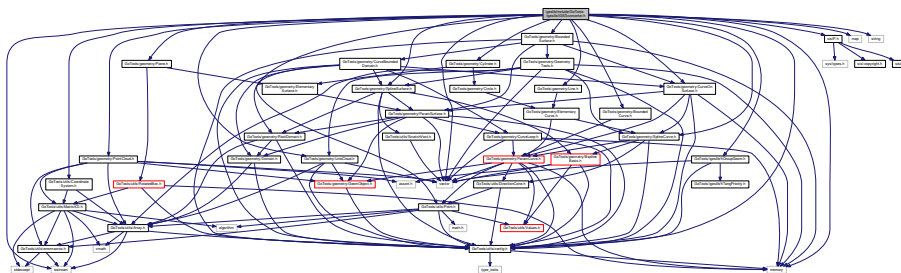
## Enumerations

- enum [Go::ftTangPriority](#) { [Go::ftNoType](#) = 0, [Go::ftMaster](#), [Go::ftSlave](#) }

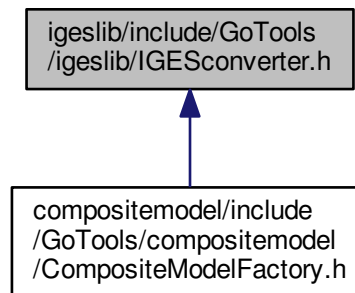
## 30.2016 igeslib/include/GoTools/igeslib/IGESconverter.h File Reference

```
#include "GoTools/utils/CoordinateSystem.h"
#include "GoTools/geometry/SplineSurface.h"
#include "GoTools/geometry/SplineCurve.h"
#include "GoTools/geometry/LineCloud.h"
#include "GoTools/geometry/Line.h"
#include "GoTools/geometry/BoundedSurface.h"
#include "GoTools/geometry/CurveOnSurface.h"
#include "GoTools/geometry/BoundedCurve.h"
#include "GoTools/geometry/Cylinder.h"
#include "GoTools/utils/Point.h"
#include "GoTools/igeslib/ftGroupGeom.h"
#include "GoTools/geometry/PointCloud.h"
#include "GoTools/geometry/Plane.h"
#include "GoTools/utils/config.h"
#include "sislP.h"
#include <memory>
#include <map>
#include <vector>
#include <string>
#include <iostream>
```

Include dependency graph for IGESconverter.h:



This graph shows which files directly or indirectly include this file:



## Classes

- struct [Go::IGESheader](#)  
*Storage of all data contained in the IGES header.*
- struct [Go::EntityList](#)  
*The entity number of all supported IGES entites.*
- struct [Go::IGESdirentry](#)  
*Storage of all data contained in an IGES directory entity.*
- class [Go::IGESconverter](#)

## Namespaces

- [Go](#)

## Enumerations

- enum [Go::FileFormat](#) { [Go::go](#), [Go::disp](#), [Go::IGES](#) }
- enum [Go::IGESSection](#) { [Go::S](#), [Go::G](#), [Go::D](#), [Go::P](#), [Go::T](#), [Go::E](#) }

## 30.2017 igeslib/include/GoTools/igeslib/igeslib\_doxyman.h File Reference

### Macros

- `#define \_IGESLIB -DOXYMAIN_H`

## 30.2017.1 Macro Definition Documentation

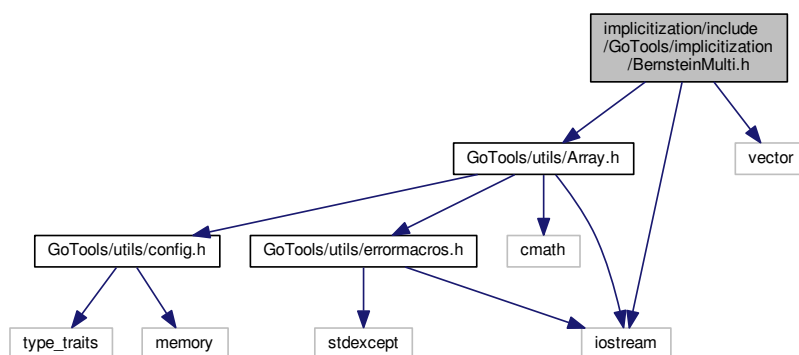
## 30.2017.1.1 #define \_IGESLIB -DOXYMAIN\_H

Definition at line 41 of file igeslib\_doxymain.h.

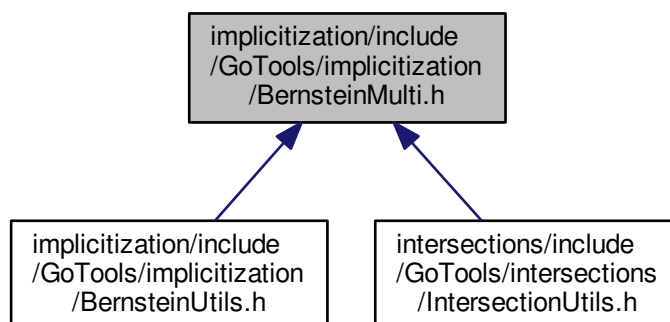
## 30.2018 implicitization/include/GoTools/implicitization/BernsteinMulti.h File Reference

```
#include "GoTools/utils/Array.h"
#include <vector>
#include <iostream>
```

Include dependency graph for BernsteinMulti.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [Go::BernsteinMulti](#)

## Namespaces

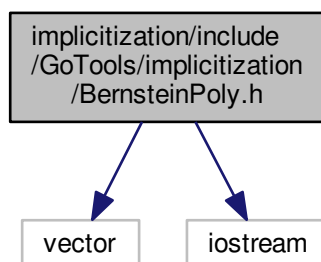
- [Go](#)
- [std](#)

## Functions

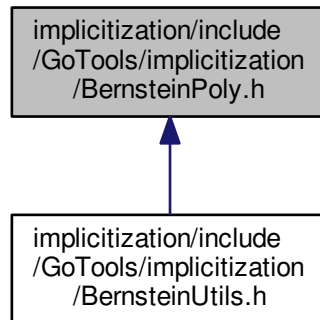
- BernsteinMulti [Go::operator\\*](#) ([const](#) BernsteinMulti &m1, [const](#) BernsteinMulti &m2)  
*Multiplication of two polynomials.*
- BernsteinMulti [Go::operator\\*](#) ([const](#) BernsteinMulti &m1, [double](#) c)  
*Multiplication of a polynomial with a scalar.*
- BernsteinMulti [Go::operator\\*](#) ([double](#) c, [const](#) BernsteinMulti &m1)  
*Multiplication of a scalar with a polynomial.*
- BernsteinMulti [Go::operator+](#) ([const](#) BernsteinMulti &m1, [const](#) BernsteinMulti &m2)  
*Addition of two polynomials.*
- BernsteinMulti [Go::operator+](#) ([const](#) BernsteinMulti &m, [double](#) c)  
*Addition of a polynomial with a scalar.*
- BernsteinMulti [Go::operator+](#) ([double](#) c, [const](#) BernsteinMulti &m)  
*Addition of a scalar with a polynomial.*
- BernsteinMulti [Go::operator-](#) ([const](#) BernsteinMulti &m1, [const](#) BernsteinMulti &m2)  
*Subtraction of two polynomials.*
- BernsteinMulti [Go::operator-](#) ([const](#) BernsteinMulti &m, [double](#) c)  
*Subtraction of a scalar from a polynomial.*
- BernsteinMulti [Go::operator-](#) ([double](#) c, [const](#) BernsteinMulti &m)  
*Subtraction of a polynomial from a scalar.*
- BernsteinMulti [Go::operator/](#) ([const](#) BernsteinMulti &m, [double](#) c)  
*Division of a polynomial with a scalar.*
- [std::istream & std::operator>>](#) ([std::istream &is](#), [Go::BernsteinMulti &m](#))  
*Read BernsteinMulti from input stream.*
- [std::ostream & std::operator<<](#) ([std::ostream &os](#), [const Go::BernsteinMulti &m](#))  
*Write BernsteinMulti to output stream.*

## 30.2019 implicitization/include/GoTools/implicitization/BernsteinPoly.h File Reference

```
#include <vector>
#include <iostream>
Include dependency graph for BernsteinPoly.h:
```



This graph shows which files directly or indirectly include this file:



## Classes

- class [Go::BernsteinPoly](#)

## Namespaces

- [Go](#)
- [std](#)

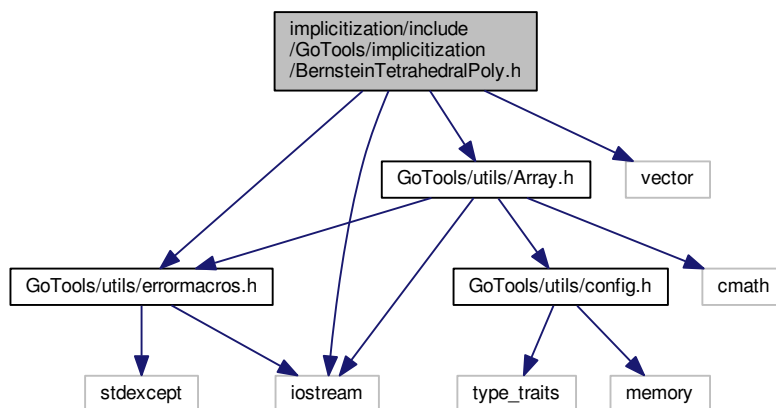
## Functions

- BernsteinPoly [Go::operator\\*](#) ([const](#) BernsteinPoly &p1, [const](#) BernsteinPoly &p2)  
*Multiplication of two polynomials.*
- BernsteinPoly [Go::operator\\*](#) ([const](#) BernsteinPoly &p, [double](#) c)  
*Multiplication of a polynomial with a scalar.*
- BernsteinPoly [Go::operator\\*](#) ([double](#) c, [const](#) BernsteinPoly &p)  
*Multiplication of a scalar with a polynomial.*
- BernsteinPoly [Go::operator+](#) ([const](#) BernsteinPoly &p1, [const](#) BernsteinPoly &p2)  
*Addition of two polynomials.*
- BernsteinPoly [Go::operator+](#) ([double](#) c, [const](#) BernsteinPoly &p)  
*Addition of a scalar with a polynomial.*
- BernsteinPoly [Go::operator+](#) ([const](#) BernsteinPoly &p, [double](#) c)  
*Addition of a polynomial with a scalar.*
- BernsteinPoly [Go::operator-](#) ([const](#) BernsteinPoly &p1, [const](#) BernsteinPoly &p2)  
*Subtraction of two polynomials.*
- BernsteinPoly [Go::operator-](#) ([double](#) c, [const](#) BernsteinPoly &p)  
*Subtraction of a polynomial from a scalar.*
- BernsteinPoly [Go::operator-](#) ([const](#) BernsteinPoly &p, [double](#) c)  
*Subtraction of a scalar from a polynomial.*
- BernsteinPoly [Go::operator/](#) ([const](#) BernsteinPoly &p, [double](#) c)  
*Division of a polynomial with a scalar.*
- [std::istream](#) & [std::operator>>](#) ([std::istream](#) &is, [Go::BernsteinPoly](#) &p)  
*Read BernsteinPoly from input stream.*
- [std::ostream](#) & [std::operator<<](#) ([std::ostream](#) &os, [const](#) [Go::BernsteinPoly](#) &p)  
*Write BernsteinPoly to output stream.*

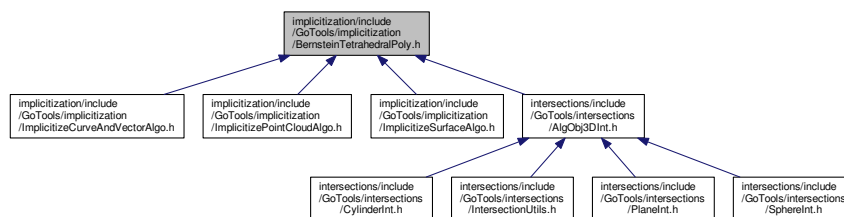
## 30.2020 implicitization/include/GoTools/implicitization/BernsteinTetrahedralPoly.h File Reference

```
#include "GoTools/utils/Array.h"
#include "GoTools/utils/errormacros.h"
#include <vector>
#include <iostream>
```

Include dependency graph for BernsteinTetrahedralPoly.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [Go::BernsteinTetrahedralPoly](#)

## Namespaces

- [Go](#)

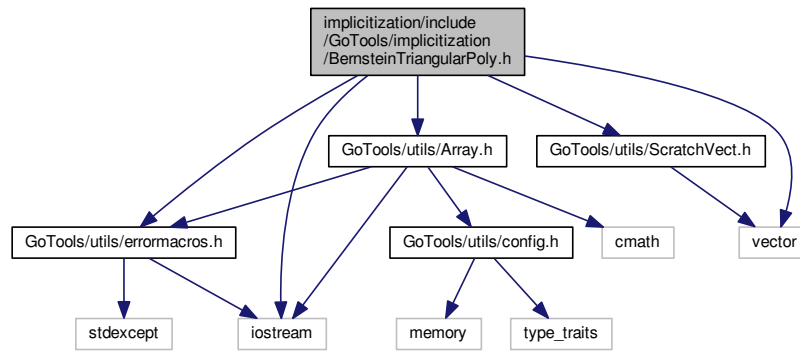
## Functions

- BernsteinTetrahedralPoly [Go::operator\\*](#) (const BernsteinTetrahedralPoly &p1, const BernsteinTetrahedralPoly &p2)  
*Multiplication of two polynomials.*
- BernsteinTetrahedralPoly [Go::operator\\*](#) (const BernsteinTetrahedralPoly &p, double c)  
*Multiplication of a polynomial with a scalar.*
- BernsteinTetrahedralPoly [Go::operator\\*](#) (double c, const BernsteinTetrahedralPoly &p)  
*Multiplication of a scalar with a polynomial.*
- BernsteinTetrahedralPoly [Go::operator+](#) (const BernsteinTetrahedralPoly &p1, const BernsteinTetrahedralPoly &p2)  
*Addition of two polynomials.*
- BernsteinTetrahedralPoly [Go::operator+](#) (double c, const BernsteinTetrahedralPoly &p)  
*Addition of a scalar with a polynomial.*
- BernsteinTetrahedralPoly [Go::operator+](#) (const BernsteinTetrahedralPoly &p, double c)  
*Addition of a polynomial with a scalar.*
- BernsteinTetrahedralPoly [Go::operator-](#) (const BernsteinTetrahedralPoly &p1, const BernsteinTetrahedralPoly &p2)  
*Subtraction of two polynomials.*
- BernsteinTetrahedralPoly [Go::operator-](#) (double c, const BernsteinTetrahedralPoly &p)  
*Subtraction of a polynomial from a scalar.*
- BernsteinTetrahedralPoly [Go::operator-](#) (const BernsteinTetrahedralPoly &p, double c)  
*Subtraction of a scalar from a polynomial.*
- BernsteinTetrahedralPoly [Go::operator/](#) (const BernsteinTetrahedralPoly &p, double c)  
*Division of a polynomial with a scalar.*
- std::istream & [Go::operator>>](#) (std::istream &is, Go::BernsteinTetrahedralPoly &p)  
*Read [BernsteinTetrahedralPoly](#) from input stream.*
- std::ostream & [Go::operator<<](#) (std::ostream &os, const Go::BernsteinTetrahedralPoly &p)  
*Write [BernsteinTetrahedralPoly](#) to output stream.*

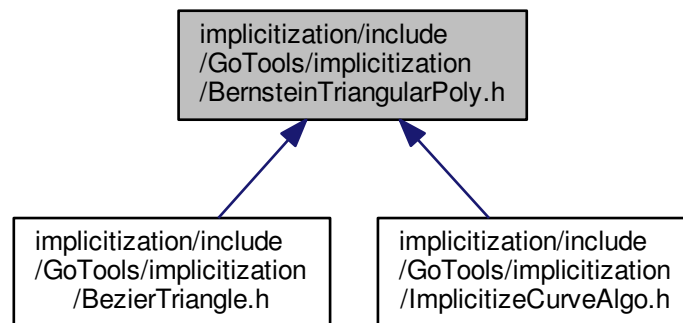
## 30.2021 implicitization/include/GoTools/implicitization/BernsteinTriangularPoly.h File Reference

```
#include "GoTools/utils/Array.h"
#include "GoTools/utils/ScratchVect.h"
#include "GoTools/utils/errormacros.h"
#include <vector>
#include <iostream>
```

Include dependency graph for BernsteinTriangularPoly.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [Go::BernsteinTriangularPoly](#)

## Namespaces

- [Go](#)

## Functions

- BernsteinTriangularPoly [Go::operator\\*](#) (`const` BernsteinTriangularPoly &p1, `const` BernsteinTriangularPoly &p2)  
*Multiplication of two polynomials.*

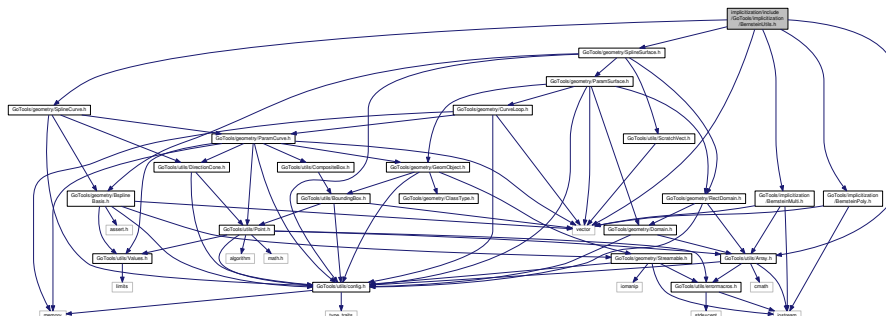


- BernsteinTriangularPoly [Go::operator\\*](#) (const BernsteinTriangularPoly &p, double c)  
*Multiplication of a polynomial with a scalar.*
- BernsteinTriangularPoly [Go::operator\\*](#) (double c, const BernsteinTriangularPoly &p)  
*Multiplication of a scalar with a polynomial.*
- BernsteinTriangularPoly [Go::operator+](#) (const BernsteinTriangularPoly &p1, const BernsteinTriangularPoly &p2)  
*Addition of two polynomials.*
- BernsteinTriangularPoly [Go::operator+](#) (double c, const BernsteinTriangularPoly &p)  
*Addition of a scalar with a polynomial.*
- BernsteinTriangularPoly [Go::operator+](#) (const BernsteinTriangularPoly &p, double c)  
*Addition of a polynomial with a scalar.*
- BernsteinTriangularPoly [Go::operator-](#) (const BernsteinTriangularPoly &p1, const BernsteinTriangularPoly &p2)  
*Subtraction of two polynomials.*
- BernsteinTriangularPoly [Go::operator-](#) (double c, const BernsteinTriangularPoly &p)  
*Subtraction of a polynomial from a scalar.*
- BernsteinTriangularPoly [Go::operator-](#) (const BernsteinTriangularPoly &p, double c)  
*Subtraction of a scalar from a polynomial.*
- BernsteinTriangularPoly [Go::operator/](#) (const BernsteinTriangularPoly &p, double c)  
*Division of a polynomial with a scalar.*
- std::istream & [Go::operator>>](#) (std::istream &is, Go::BernsteinTriangularPoly &p)  
*Read [BernsteinTriangularPoly](#) from input stream.*
- std::ostream & [Go::operator<<](#) (std::ostream &os, const Go::BernsteinTriangularPoly &p)  
*Write [BernsteinTriangularPoly](#) to output stream.*

## 30.2022 implicitization/include/GoTools/implicitization/BernsteinUtils.h File Reference

```
#include "GoTools/implicitization/BernsteinPoly.h"
#include "GoTools/implicitization/BernsteinMulti.h"
#include "GoTools/Utils/Array.h"
#include "GoTools/geometry/SplineCurve.h"
#include "GoTools/geometry/SplineSurface.h"
#include <vector>
```

Include dependency graph for BernsteinUtils.h:



## Namespaces

- [Go](#)

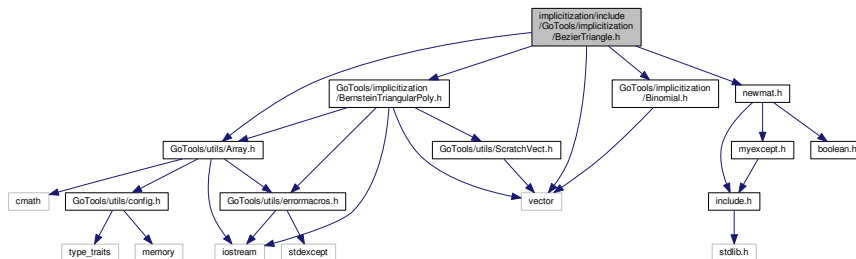
## Functions

- void `Go::spline_to_bernstein` (`const` SplineCurve &seg, int dd, BernsteinPoly &bp)
- void `Go::spline_to_bernstein` (`const` SplineSurface &pat, int dd, BernsteinMulti &bm)
- void `Go::spline_to_bernstein` (`const` SplineCurve &seg, `std::vector`< BernsteinPoly > &seg\_bp)
- void `Go::spline_to_bernstein` (`const` SplineSurface &pat, `std::vector`< BernsteinMulti > &pat\_bm)
- `template`<int Ndim>  
void `Go::splineToBernstein` (`const` SplineCurve &segment, `Array`< BernsteinPoly, Ndim > &curve\_bp)
- `template`<int Ndim>  
void `Go::splineToBernstein` (`const` SplineSurface &patch, `Array`< BernsteinMulti, Ndim > &surface\_bm)
- `template`<int Ndim>  
void `Go::bernsteinToSpline` (`const` `Array`< BernsteinPoly, Ndim > &curve\_bp, `bool` rational, SplineCurve &segment)
- `template`<int Ndim>  
void `Go::bernsteinToSpline` (`const` `Array`< BernsteinMulti, Ndim > &surface\_bm, `bool` rational, SplineSurface &patch)

## 30.2023 implicitization/include/GoTools/implicitization/BezierTriangle.h File Reference

```
#include "GoTools/Utils/Array.h"
#include "GoTools/implicitization/BernsteinTriangularPoly.h"
#include "newmat.h"
#include "GoTools/implicitization/Binomial.h"
#include <vector>
```

Include dependency graph for BezierTriangle.h:



## Classes

- class `Go::BezierTriangle`< N >  
*Not documented.*

## Namespaces

- `std`
- `Go`

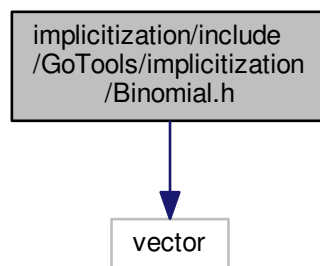
## Functions

- `template<class _Tp >`  
`_Tp std::identity_element (plus< _Tp >)`
- `template<class _Tp >`  
`_Tp std::identity_element (multiplies< _Tp >)`
- `template<class _Tp, class _Integer, class _MonoidOperation >`  
`_Tp std::__power (_Tp __x, _Integer __n, _MonoidOperation __oper)`
- `template<class _Tp, class _Integer >`  
`_Tp std::__power (_Tp __x, _Integer __n)`
- `template<class _Tp, class _Integer, class _MonoidOperation >`  
`_Tp std::power (_Tp __x, _Integer __n, _MonoidOperation __oper)`
- `template<class _Tp, class _Integer >`  
`_Tp std::power (_Tp __x, _Integer __n)`

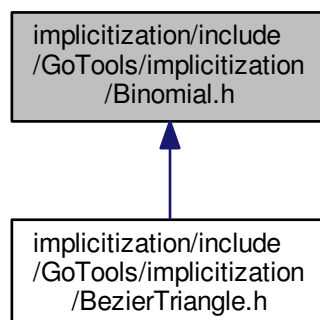
## 30.2024 implicitization/include/GoTools/implicitization/Binomial.h File Reference

```
#include <vector>
```

Include dependency graph for Binomial.h:



This graph shows which files directly or indirectly include this file:



## Classes

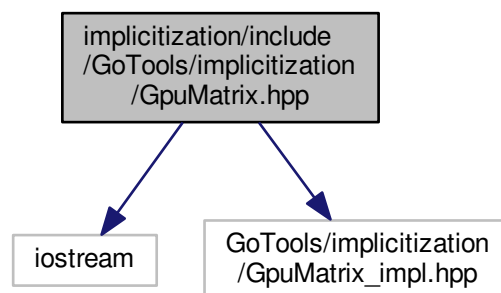
- class [Go::Binomial](#)

## Namespaces

- [Go](#)

## 30.2025 implicitization/include/GoTools/implicitization/GpuMatrix.hpp File Reference

```
#include <iostream>
#include "GoTools/implicitization/GpuMatrix_impl.hpp"
Include dependency graph for GpuMatrix.hpp:
```



## Classes

- class [Go::GpuMatrix< T >](#)

## Namespaces

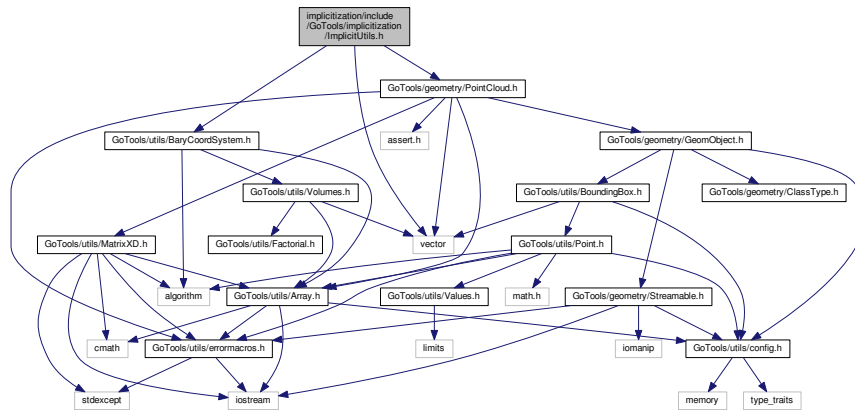
- [Go](#)







Include dependency graph for ImplicitUtils.h:



## Namespaces

- [Go](#)

## Functions

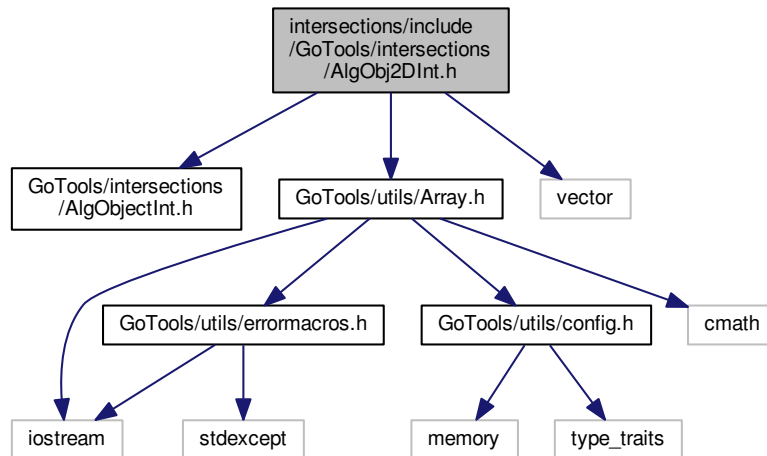
- void [Go::create\\_bary\\_coord\\_system2D](#) (const SplineCurve &curve, BaryCoordSystem2D &bc)  
*Creates a barycentric coordinate system from a given spline curve.*
- void [Go::create\\_bary\\_coord\\_system3D](#) (const SplineCurve &curve, BaryCoordSystem3D &bc)  
*Creates a barycentric coordinate system from a given 3D spline curve.*
- void [Go::create\\_bary\\_coord\\_system3D](#) (const SplineSurface &surface, BaryCoordSystem3D &bc)  
*Creates a barycentric coordinate system from a given spline surface.*
- void [Go::create\\_bary\\_coord\\_system3D](#) (const PointCloud3D &cloud, BaryCoordSystem3D &bc)  
*Creates a barycentric coordinate system from a point cloud.*
- void [Go::create\\_bary\\_coord\\_system3D](#) (const BoundingBox &box, BaryCoordSystem3D &bc)  
*Creates a barycentric coordinate system from a bounding box.*
- void [Go::cart\\_to\\_bary](#) (const SplineCurve &cv, const BaryCoordSystem2D &bc, SplineCurve &cv\_bc)
- void [Go::cart\\_to\\_bary](#) (const SplineCurve &cv, const BaryCoordSystem3D &bc, SplineCurve &cv\_bc)
- void [Go::cart\\_to\\_bary](#) (const SplineSurface &sf, const BaryCoordSystem3D &bc, SplineSurface &sf\_bc)
- void [Go::cart\\_to\\_bary](#) (const PointCloud3D &cloud, const BaryCoordSystem3D &bc, PointCloud4D &cloud\_←\_bc)
- void [Go::make\\_matrix](#) (const SplineCurve &curve, int deg, std::vector< std::vector< double > > &mat)  
*Make the matrix D.*
- void [Go::make\\_matrix](#) (const SplineSurface &surf, int deg, std::vector< std::vector< double > > &mat)  
*Make the matrix D.*
- void [Go::make\\_matrix](#) (const PointCloud4D &cloud, int deg, std::vector< std::vector< double > > &mat)  
*Make the matrix D.*
- void [Go::make\\_implicit\\_svd](#) (std::vector< std::vector< double > > &mat, std::vector< double > &b, double &sigma\_min)
- void [Go::make\\_implicit\\_gauss](#) (std::vector< std::vector< double > > &mat, std::vector< double > &b)



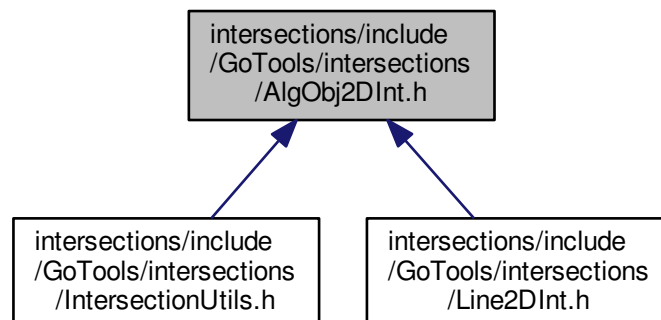
## 30.2032 intersections/include/GoTools/intersections/AlgObj2DInt.h File Reference

```
#include "GoTools/intersections/AlgObjectInt.h"
#include "GoTools/utils/Array.h"
#include <vector>
```

Include dependency graph for AlgObj2DInt.h:



This graph shows which files directly or indirectly include this file:



## Classes

- struct [Go::Alg2DElem](#)
- class [Go::AlgObj2DInt](#)

*Class for 2-dimensional algebraic intersection objects.*

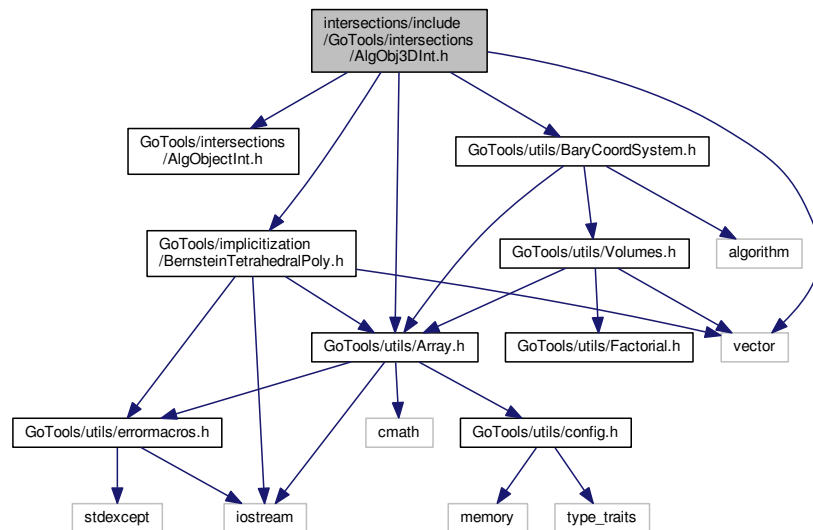
## Namespaces

- [Go](#)

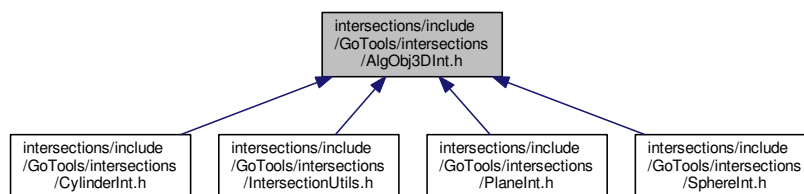
### 30.2033 intersections/include/GoTools/intersections/AlgObj3DInt.h File Reference

```
#include "GoTools/intersections/AlgObjectInt.h"
#include "GoTools/utils/Array.h"
#include "GoTools/implicitization/BernsteinTetrahedralPoly.h"
#include "GoTools/utils/BaryCoordSystem.h"
#include <vector>
```

Include dependency graph for AlgObj3DInt.h:



This graph shows which files directly or indirectly include this file:



## Classes

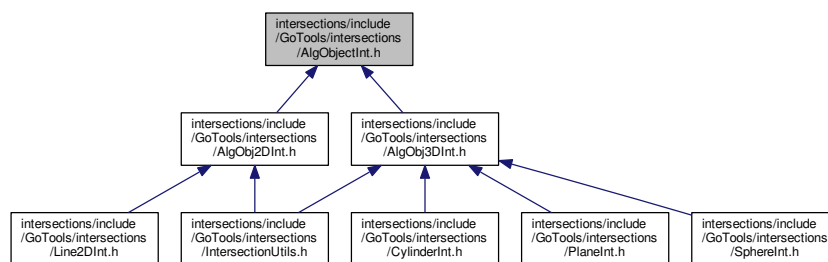
- struct [Go::Alg3DElem](#)
- class [Go::AlgObj3DInt](#)

## Namespaces

- [Go](#)

## 30.2034 intersections/include/GoTools/intersections/AlgObjectInt.h File Reference

This graph shows which files directly or indirectly include this file:



## Classes

- class [Go::AlgObjectInt](#)

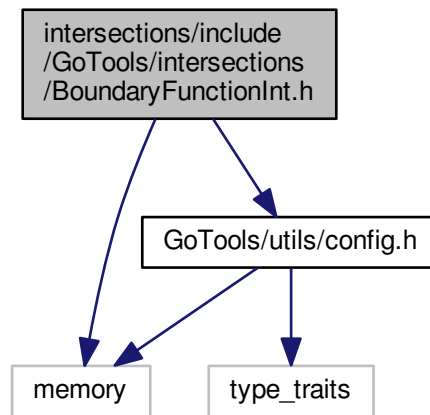
## Namespaces

- [Go](#)

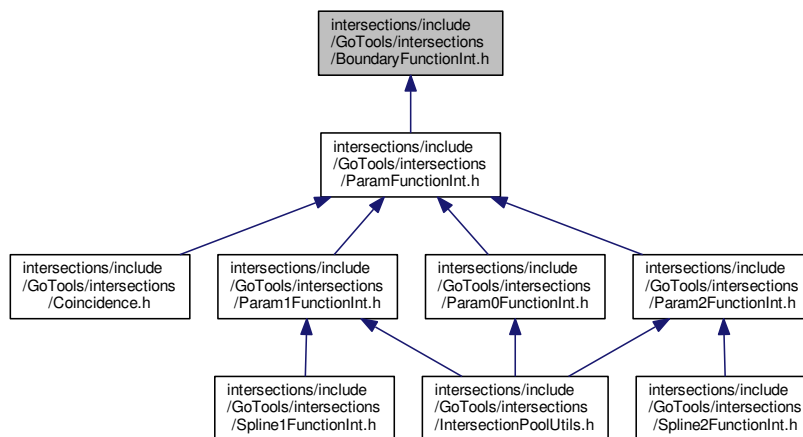
## 30.2035 intersections/include/GoTools/intersections/BoundaryFunctionInt.h File Reference

```
#include "GoTools/utils/config.h"
#include <memory>
```

Include dependency graph for `BoundaryFunctionInt.h`:



This graph shows which files directly or indirectly include this file:



## Classes

- struct [Go::BoundaryFunctionInt](#)

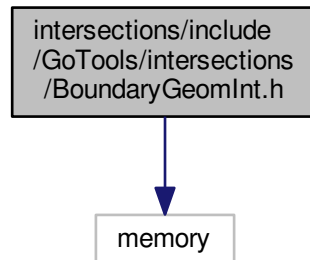
## Namespaces

- [Go](#)

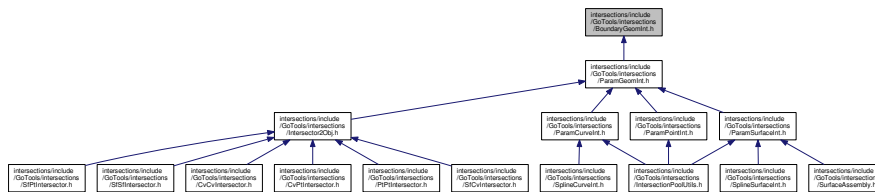
## 30.2036 intersections/include/GoTools/intersections/BoundaryGeomInt.h File Reference

```
#include <memory>
```

Include dependency graph for BoundaryGeomInt.h:



This graph shows which files directly or indirectly include this file:



## Classes

- struct [Go::BoundaryGeomInt](#)

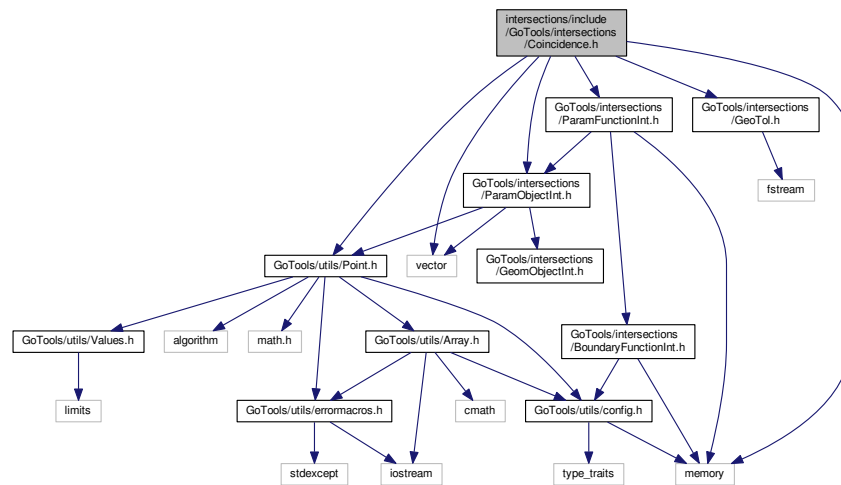
## Namespaces

- [Go](#)

## 30.2037 intersections/include/GoTools/intersections/Coincidence.h File Reference

```
#include <memory>
#include "GoTools/utils/Point.h"
#include "GoTools/intersections/ParamObjectInt.h"
#include "GoTools/intersections/GeoTol.h"
#include "GoTools/intersections/ParamFunctionInt.h"
#include <vector>
```

Include dependency graph for Coincidence.h:



## Namespaces

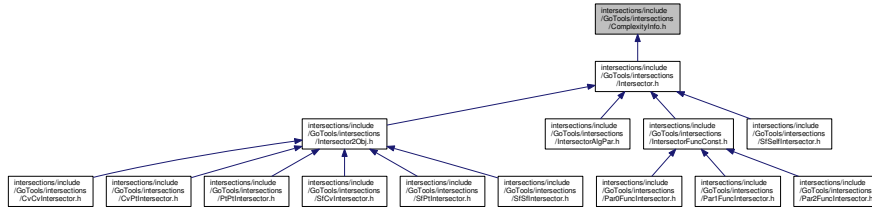
- [Go](#)

## Functions

- `int Go::checkCoincide (ParamCurveInt *curve, double start, double end, shared_ptr< GeoTol > tol, ParamCurveInt *other, double other_start, double other_end)`
- `int Go::checkCoincide (ParamCurveInt *curve, double start, double end, ParamSurfaceInt *surf, Point su_start, Point su_end, shared_ptr< GeoTol > tol)`
- `int Go::checkCoincide (ParamCurveInt *curve, double start, double end, SplineSurfaceInt *surf, const Point &su_start, const Point &su_end, shared_ptr< GeoTol > tol)`
- `int Go::checkCoincide (Param1FunctionInt *func1, double start, double end, Param0FunctionInt *C, shared_ptr< GeoTol > tol)`
- `int Go::checkCoincide (ParamSurfaceInt *surf1, ParamSurfaceInt *surf2, std::vector< double > &par_loop, const shared_ptr< GeoTol > tol)`  
*by the parameter values of the intersection points representing it*
- `int Go::stepLength (const std::vector< Point > &ft, const std::vector< Point > &gs, double delta, bool forward, double &delta_t)`
- `double Go::stepLength (const std::vector< Point > &ft, const std::vector< Point > &gs, bool forward, std::vector< Point > &cvder, double min_step, double max_step, double aepsge)`
- `int Go::evalProjectedCurve (const std::vector< Point > &ft, const std::vector< Point > &gs, std::vector< Point > &res)`
- `void Go::evalDistCurve (const std::vector< Point > &marching_curve, const std::vector< Point > &other_curve, std::vector< Point > &dist_curve)`
- `bool Go::determineCoincidenceRegion (const ParamObjectInt *obj1, const ParamObjectInt *obj2, shared_ptr< const GeoTol > tol, const double *current_params, int dir, bool forward, double &last_param_val_inside, double &first_param_val_outside)`
- `bool Go::determineCoincidenceRegion (const ParamFunctionInt *obj_1d, double C, shared_ptr< const GeoTol > tol, const double *current_params, int dir, bool forward, double &last_param_val_inside, double &first_param_val_outside)`

### 30.2038 intersections/include/GoTools/intersections/ComplexityInfo.h File Reference

This graph shows which files directly or indirectly include this file:



#### Classes

- class [Go::ComplexityInfo](#)

#### Namespaces

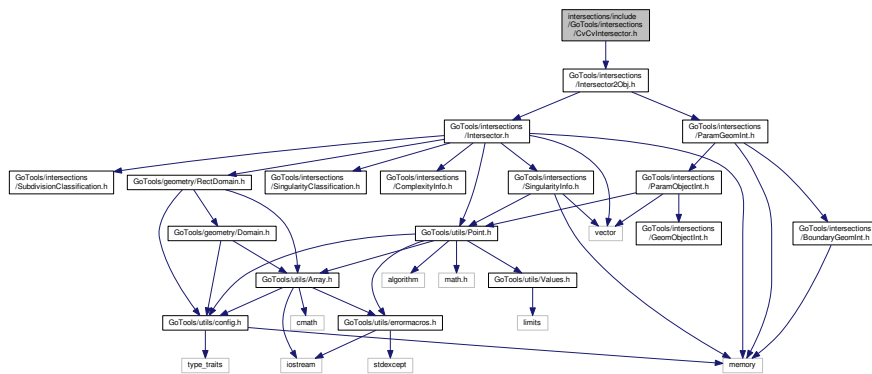
- [Go](#)

### 30.2039 intersections/include/GoTools/intersections/Conelnt.h File Reference

### 30.2040 intersections/include/GoTools/intersections/CvCvIntersector.h File Reference

```
#include "GoTools/intersections/Intersector2Obj.h"
```

Include dependency graph for CvCvIntersector.h:



#### Classes

- class [Go::CvCvIntersector](#)  
Class that performs intersection between two parametric curves.





## Classes

- class [Go::CylinderInt](#)

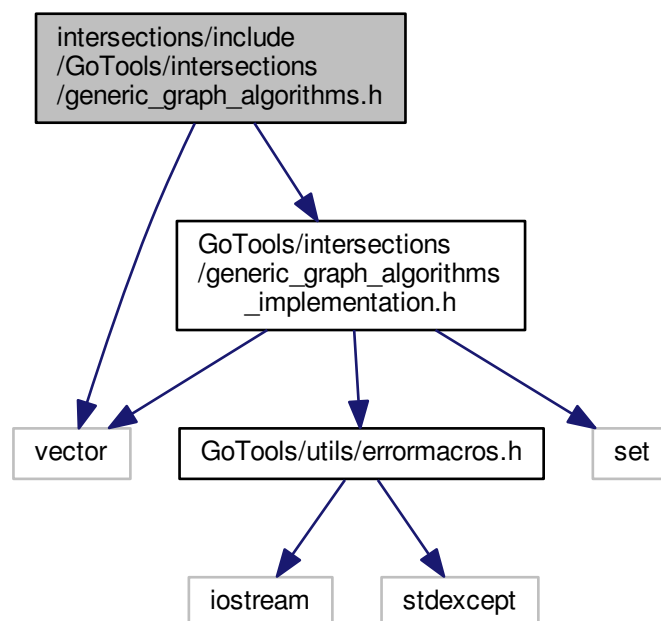
*Class for cylindrical algebraic intersection objects.*

## Namespaces

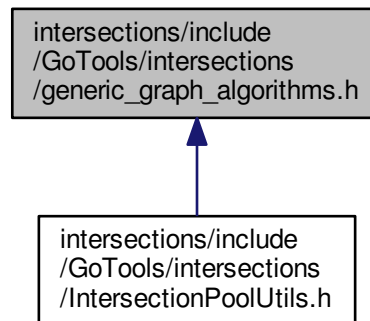
- [Go](#)

## 30.2043 intersections/include/GoTools/intersections/generic\_graph\_algorithms.h File Reference

```
#include <vector>
#include "GoTools/intersections/generic_graph_algorithms_implementation.h"
Include dependency graph for generic_graph_algorithms.h:
```



This graph shows which files directly or indirectly include this file:



## Functions

- `template<class FunctorConnectedTo >`  
`void get_fundamental_cycle_set (int num_nodes, FunctorConnectedTo connectedTo, std::vector< std::vector< int > > &result)`
- `void get_fundamental_cycle_set (int num_nodes, const std::vector< std::vector< int > > &table, std::vector< std::vector< int > > &result)`
- `template<class FunctorConnectedTo >`  
`bool is_path_connected (int node1_index, int node2_index, int num_nodes, FunctorConnectedTo connected_to)`
- `template<class FunctorConnectedTo >`  
`void get_individual_paths (int num_nodes, FunctorConnectedTo connectedTo, std::vector< std::vector< int > > &paths, std::vector< std::vector< int > > &cycles, std::vector< int > &isolated_nodes)`

### 30.2043.1 Function Documentation

**30.2043.1.1** `template<class FunctorConnectedTo > void get_fundamental_cycle_set ( int num_nodes, FunctorConnectedTo connectedTo, std::vector< std::vector< int > > & result )`

Definition at line 128 of file generic\_graph\_algorithms\_implementation.h.

**30.2043.1.2** `void get_fundamental_cycle_set ( int num_nodes, const std::vector< std::vector< int > > & table, std::vector< std::vector< int > > & result )`

Definition at line 139 of file generic\_graph\_algorithms\_implementation.h.

**30.2043.1.3** `template<class FunctorConnectedTo > void get_individual_paths ( int num_nodes, FunctorConnectedTo connectedTo, std::vector< std::vector< int > > & paths, std::vector< std::vector< int > > & cycles, std::vector< int > & isolated_nodes )`

Definition at line 183 of file generic\_graph\_algorithms\_implementation.h.

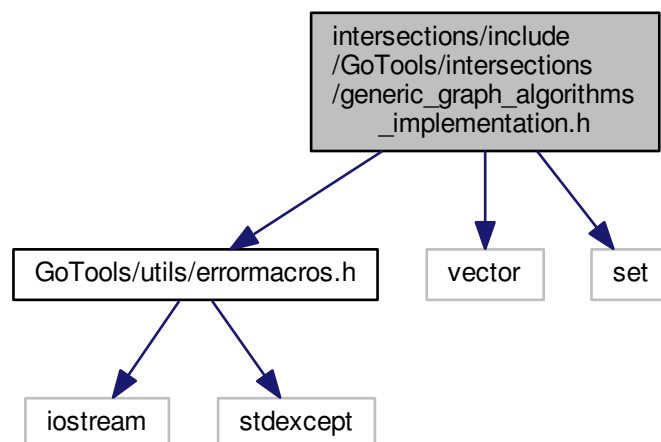
30.2043.1.4 `template<class FunctorConnectedTo > bool is_path_connected ( int node1_index, int node2_index, int num_nodes, FunctorConnectedTo connected_to )`

Definition at line 158 of file generic\_graph\_algorithms\_implementation.h.

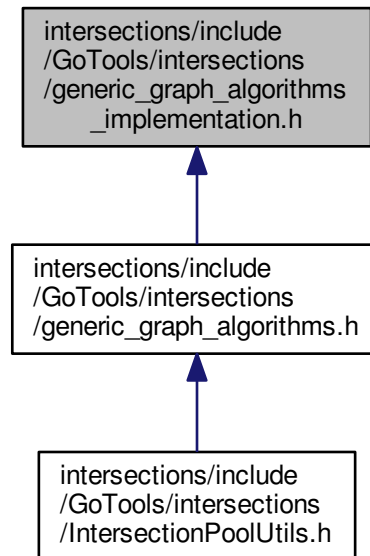
### 30.2044 intersections/include/GoTools/intersections/generic\_graph\_algorithms\_implementation.h File Reference

```
#include "GoTools/utils/errormacros.h"
#include <vector>
#include <set>
```

Include dependency graph for generic\_graph\_algorithms\_implementation.h:



This graph shows which files directly or indirectly include this file:



## Enumerations

- enum [NodeStatus](#)

## Functions

- template<class FunctorConnectedTo >  
void [get\\_fundamental\\_cycle\\_set](#) (int num\_nodes, FunctorConnectedTo connectedTo, std::vector< std::vector< int > > &result)
- void [get\\_fundamental\\_cycle\\_set](#) (int num\_nodes, const std::vector< std::vector< int > > &table, std::vector< std::vector< int > > &result)
- template<class FunctorConnectedTo >  
[bool is\\_path\\_connected](#) (int node1\_index, int node2\_index, int num\_nodes, FunctorConnectedTo connected\_to)
- template<class FunctorConnectedTo >  
void [get\\_individual\\_paths](#) (int num\_nodes, FunctorConnectedTo connectedTo, std::vector< std::vector< int > > &paths, std::vector< std::vector< int > > &cycles, std::vector< int > &isolated\_nodes)

### 30.2044.1 Enumeration Type Documentation

#### 30.2044.1.1 enum NodeStatus

Definition at line 52 of file generic\_graph\_algorithms\_implementation.h.

### 30.2044.2 Function Documentation

30.2044.2.1 `template<class FunctorConnectedTo > void get_fundamental_cycle_set ( int num_nodes, FunctorConnectedTo connectedTo, std::vector< std::vector< int > > & result )`

Definition at line 128 of file generic\_graph\_algorithms\_implementation.h.

30.2044.2.2 `void get_fundamental_cycle_set ( int num_nodes, const std::vector< std::vector< int > > & table, std::vector< std::vector< int > > & result )`

Definition at line 139 of file generic\_graph\_algorithms\_implementation.h.

30.2044.2.3 `template<class FunctorConnectedTo > void get_individual_paths ( int num_nodes, FunctorConnectedTo connectedTo, std::vector< std::vector< int > > & paths, std::vector< std::vector< int > > & cycles, std::vector< int > & isolated_nodes )`

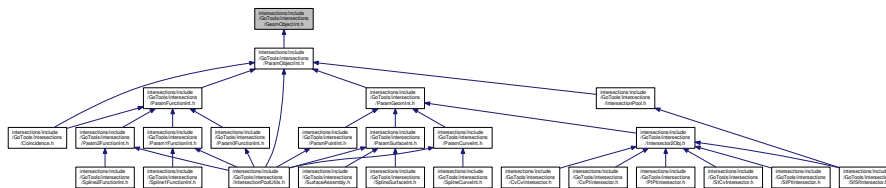
Definition at line 183 of file generic\_graph\_algorithms\_implementation.h.

30.2044.2.4 `template<class FunctorConnectedTo > bool is_path_connected ( int node1_index, int node2_index, int num_nodes, FunctorConnectedTo connected_to )`

Definition at line 158 of file generic\_graph\_algorithms\_implementation.h.

## 30.2045 intersections/include/GoTools/intersections/GeomObjectInt.h File Reference

This graph shows which files directly or indirectly include this file:



### Classes

- class [Go::GeomObjectInt](#)

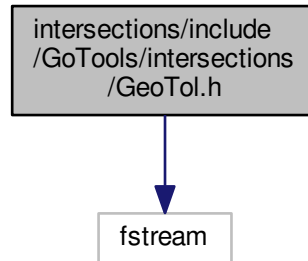
### Namespaces

- [Go](#)

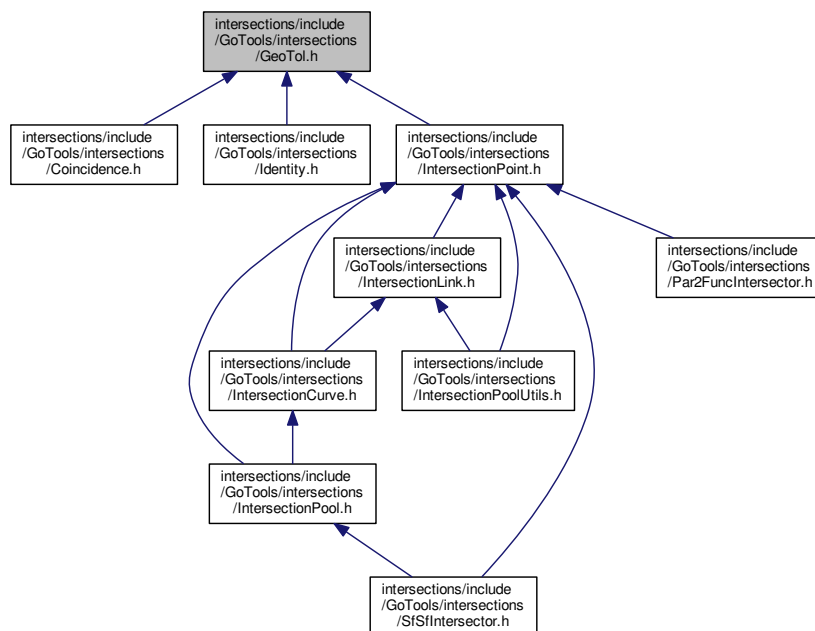
## 30.2046 intersections/include/GoTools/intersections/GeoTol.h File Reference

```
#include <fstream>
```

Include dependency graph for GeoTol.h:



This graph shows which files directly or indirectly include this file:



### Classes

- class [Go::GeoTol](#)

*Class handling various tolerances.*

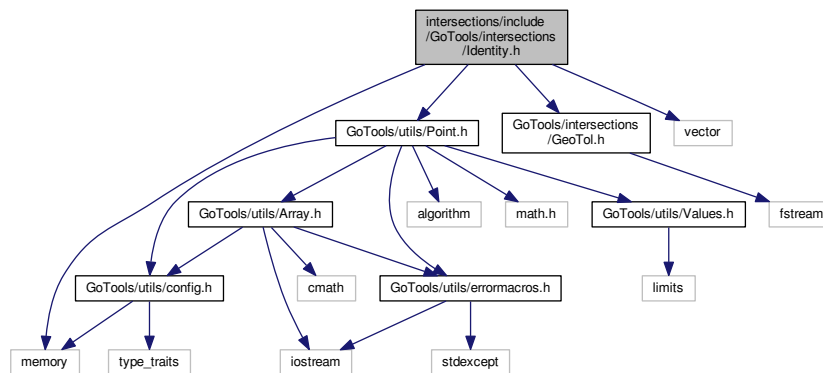
## Namespaces

- [Go](#)

## 30.2047 intersections/include/GoTools/intersections/Identity.h File Reference

```
#include <memory>
#include "GoTools/utils/Point.h"
#include "GoTools/intersections/GeoTol.h"
#include <vector>
```

Include dependency graph for Identity.h:



## Classes

- class [Go::Identity](#)  
*Check coincidence.*

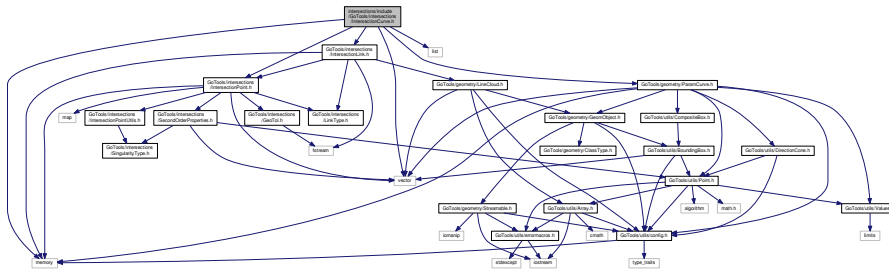
## Namespaces

- [Go](#)

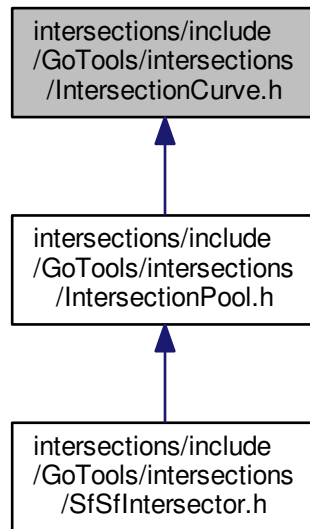
## 30.2048 intersections/include/GoTools/intersections/IntersectionCurve.h File Reference

```
#include "GoTools/intersections/IntersectionPoint.h"
#include "GoTools/intersections/IntersectionLink.h"
#include "GoTools/geometry/ParamCurve.h"
#include <memory>
#include <list>
#include <vector>
```

Include dependency graph for IntersectionCurve.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [Go::Zero\\_Parameter\\_Span\\_Error](#)  
*Error object used internally in [IntersectionCurve](#).*
- class [Go::IntersectionCurve](#)
- class [Go::DegeneratedIntersectionCurve](#)  
*[IntersectionCurve](#) that is degenerated into a single point.*
- class [Go::NonEvaluableIntersectionCurve](#)  
*[IntersectionCurve](#) that cannot be evaluated.*
- class [Go::IsoparametricIntersectionCurve](#)
- class [Go::InterpolatedIntersectionCurve](#)

## Namespaces

- [Go](#)



## Enumerations

- enum [Go::EvalKind](#) { [Go::SPACECURVE](#), [Go::PARAMCURVE\\_1](#), [Go::PARAMCURVE\\_2](#) }
- enum [Go::TangentDomain](#) { [Go::GEOM](#), [Go::PARAM1](#), [Go::PARAM2](#) }
- enum [Go::EstimateDirection](#) { [Go::FORWARDS](#), [Go::BACKWARDS](#) }

## Functions

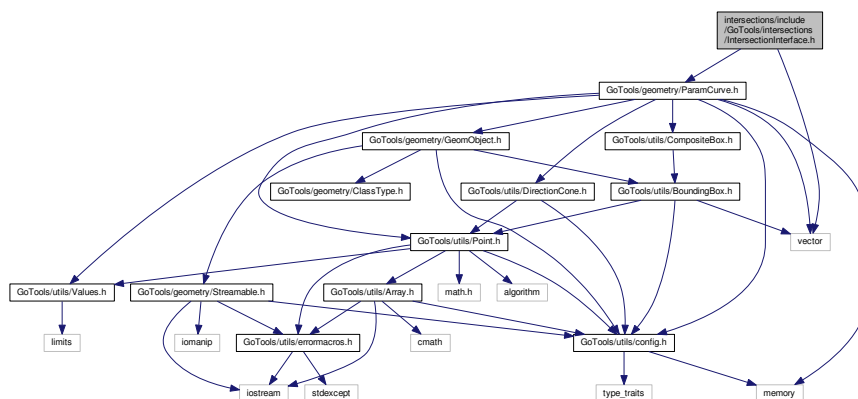
- template<class iterator >  
shared\_ptr< IntersectionCurve > [Go::constructIntersectionCurve](#) (const iterator begin, const iterator end)

## 30.2049 intersections/include/GoTools/intersections/IntersectionInterface.h File Reference

```
#include "GoTools/geometry/ParamCurve.h"
```

```
#include <vector>
```

Include dependency graph for IntersectionInterface.h:



## Namespaces

- [Go](#)

## Functions

- void [Go::intersectCurves](#) (shared\_ptr< ParamCurve > crv1, shared\_ptr< ParamCurve > crv2, double tol, std::vector< std::pair< double, double > > &intersection\_points)

*Intersection between two parametric curves.*



## Classes

- class [Go::IntersectionLink](#)

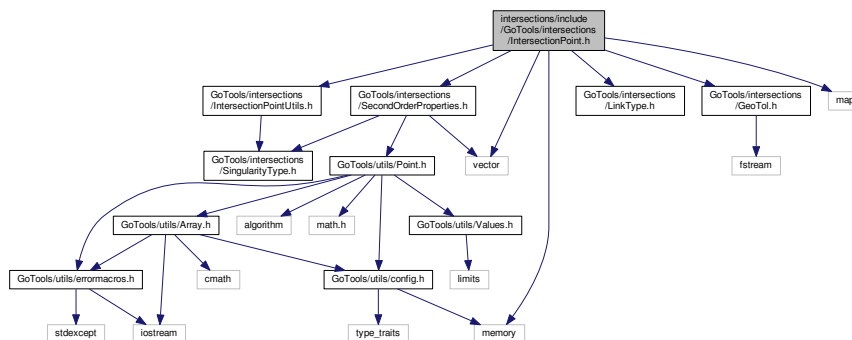
## Namespaces

- [Go](#)

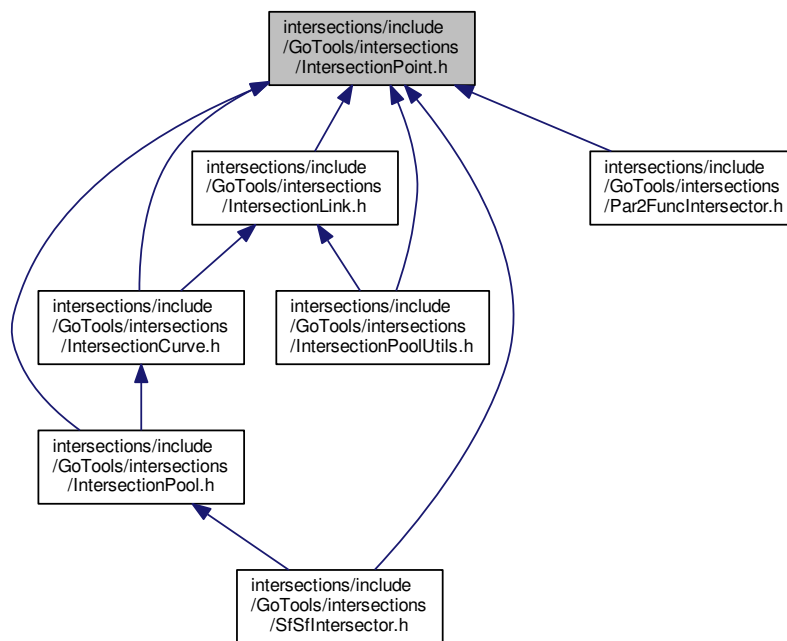
## 30.2051 intersections/include/GoTools/intersections/IntersectionPoint.h File Reference

```
#include "GoTools/intersections/IntersectionPointUtils.h"
#include "GoTools/intersections/LinkType.h"
#include "GoTools/intersections/GeoTol.h"
#include "GoTools/intersections/SecondOrderProperties.h"
#include <memory>
#include <vector>
#include <map>
```

Include dependency graph for IntersectionPoint.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [Go::IntersectionPoint](#)

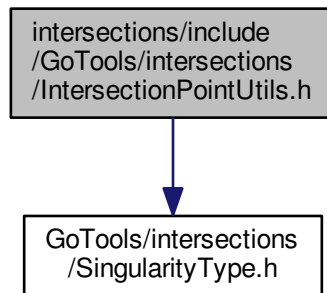
## Namespaces

- [Go](#)

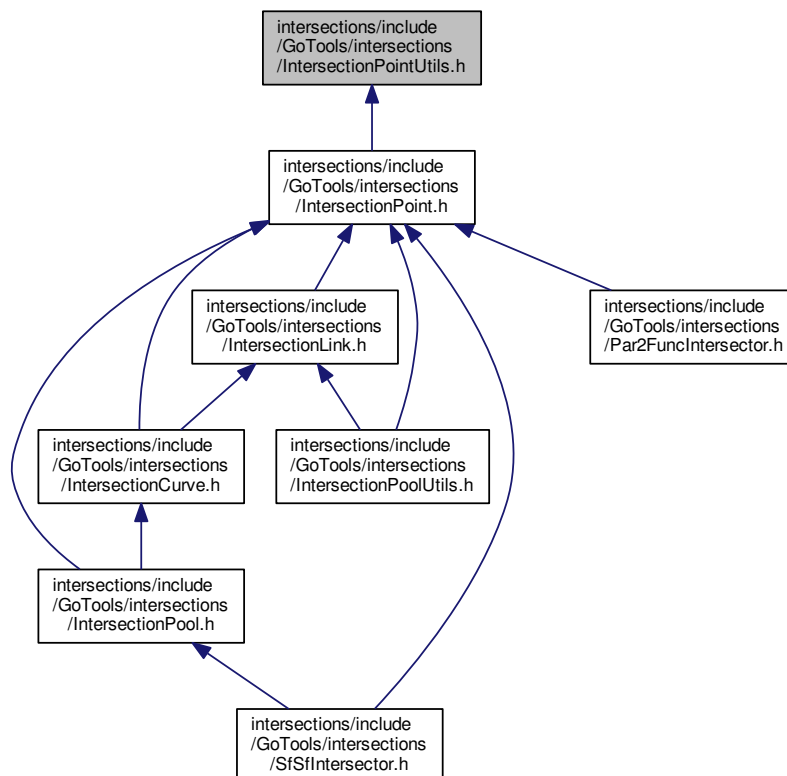
## 30.2052 intersections/include/GoTools/intersections/IntersectionPointUtils.h File Reference

```
#include "GoTools/intersections/SingularityType.h"
```

Include dependency graph for IntersectionPointUtils.h:



This graph shows which files directly or indirectly include this file:



## Classes

- struct [Go::CachedInterval](#)  
*Helper struct for saving bracketed bounds of influence areas.*
- struct [Go::IntPtInfo](#)

## Namespaces

- [Go](#)

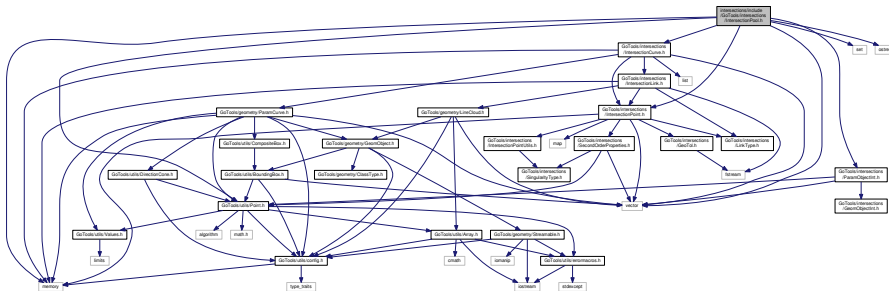
## Enumerations

- enum [Go::IntPtClassification](#) {  
[Go::DIR\\_UNDEF](#) = 0, [Go::DIR\\_IN](#), [Go::DIR\\_OUT](#), [Go::DIR\\_PARALLEL](#),  
[Go::DIR\\_PERPENDICULAR](#), [Go::DIR\\_HIGHLY\\_SINGULAR](#), [Go::DIR\\_TOUCH](#) }
- enum [Go::IntPtLocation](#) {  
[Go::LOC\\_OUTSIDE\\_BOTH](#) = 0, [Go::LOC\\_INSIDE\\_BOTH](#), [Go::LOC\\_EDGE\\_ONE](#), [Go::LOC\\_CORNER\\_↔](#)  
[ONE](#),  
[Go::LOC\\_CORNER\\_BOTH](#), [Go::LOC\\_EDGE\\_BOTH](#) }

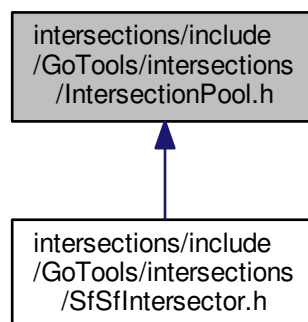
### 30.2053 intersections/include/GoTools/intersections/IntersectionPool.h File Reference

```
#include "GoTools/intersections/ParamObjectInt.h"
#include "GoTools/intersections/IntersectionPoint.h"
#include "GoTools/intersections/IntersectionCurve.h"
#include "GoTools/utils/Point.h"
#include <memory>
#include <vector>
#include <set>
#include <ostream>
```

Include dependency graph for IntersectionPool.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [Go::IntersectionPool](#)
- struct [Go::BoundaryIntersectionData](#)

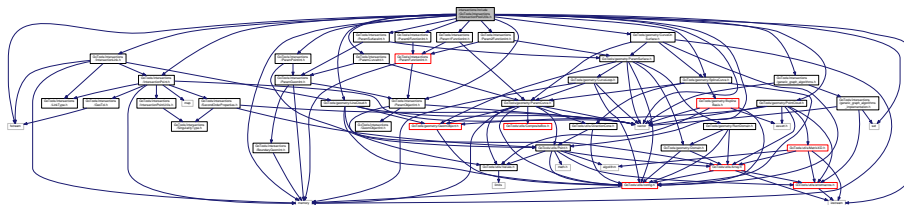
## Namespaces

- [Go](#)

## 30.2054 intersections/include/GoTools/intersections/IntersectionPoolUtils.h File Reference

```
#include "GoTools/intersections/IntersectionPoint.h"
#include "GoTools/intersections/IntersectionLink.h"
#include "GoTools/intersections/ParamObjectInt.h"
#include "GoTools/intersections/ParamPointInt.h"
#include "GoTools/intersections/ParamCurveInt.h"
#include "GoTools/intersections/ParamSurfaceInt.h"
#include "GoTools/intersections/Param0FunctionInt.h"
#include "GoTools/intersections/Param1FunctionInt.h"
#include "GoTools/intersections/Param2FunctionInt.h"
#include "GoTools/geometry/SplineCurve.h"
#include "GoTools/geometry/CurveOnSurface.h"
#include "GoTools/intersections/generic_graph_algorithms.h"
#include "GoTools/utils/Point.h"
#include "GoTools/geometry/LineCloud.h"
#include "GoTools/geometry/PointCloud.h"
#include <memory>
#include <vector>
#include <set>
#include <iostream>
#include <fstream>
```

Include dependency graph for IntersectionPoolUtils.h:



## Classes

- struct [Go::raw\\_pointer\\_comp< T >](#)
- class [Go::CrossesValue](#)  
*IntersectionPoolUtils. predicate for STL function.*
- class [Go::TestInDomain](#)  
*IntersectionPoolUtils. predicate for STL function.*
- class [Go::ClosestPointCalculator](#)
- class [Go::LockedDirDistFunc](#)
- class [Go::ConnectionFuncor](#)

## Namespaces

- [Go](#)

## Functions

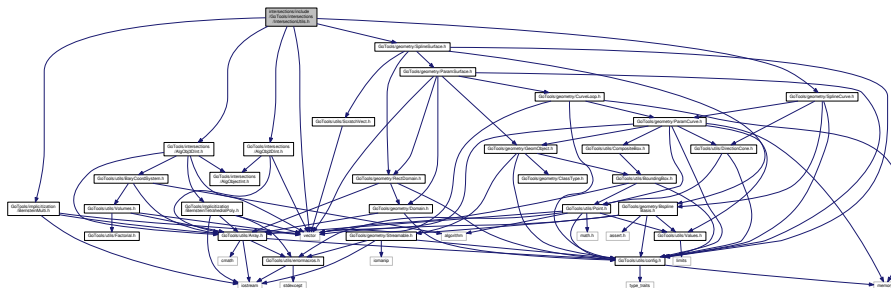
- void [Go::debug\\_write\\_line](#) (const Point &p1, const Point &p2, const char \*fname)
- void [Go::debug\\_write\\_point](#) (const Point &p1, const char \*fname)
- bool [Go::no\\_parent](#) (const shared\_ptr< IntersectionPoint > &p)
- IntersectionPoint \* [Go::flip\\_intersecting\\_objects](#) (IntersectionPoint \*p)
- bool [Go::link\\_is\\_iso\\_in](#) (shared\_ptr< IntersectionLink > link, int dir)
- bool [Go::link\\_is\\_iso](#) (shared\_ptr< IntersectionLink > link)
- bool [Go::link\\_is\\_iso\\_in\\_other\\_than](#) (shared\_ptr< IntersectionLink > link, int dir)
- template<class T1 , class T2 >  
bool [Go::compare\\_first](#) (const std::pair< T1, T2 > &x, const std::pair< T1, T2 > &y)
- void [Go::add\\_reachables\\_from](#) (const IntersectionPoint \*pt, const IntersectionPoint \*last\_pt, std::set< IntersectionPoint \* > &result)
- void [Go::estimate\\_seed\\_by\\_interpolation](#) (const IntersectionPoint \*p1, const IntersectionPoint \*p2, int fixed\_dir, double fixed\_value, double \*result)
- void [Go::determine\\_seed](#) (double \*par, int par\_start, int par\_end, const Point &pt, const IntersectionPoint \*const ip1, const IntersectionPoint \*const ip2)
- int [Go::find\\_point\\_in](#) (IntersectionPoint \*p, const std::vector< shared\_ptr< IntersectionPoint > > &vec)
- shared\_ptr< const CurveOnSurface > [Go::make\\_curve\\_on\\_surface](#) (shared\_ptr< const ParamSurface > surf, double u\_start, double v\_start, double u\_end, double v\_end, double p\_start, double p\_end)
- void [Go::extract\\_chains](#) (const std::vector< shared\_ptr< IntersectionPoint > > pts, std::vector< std::vector< shared\_ptr< IntersectionPoint > > > &chains)

### 30.2055 intersections/include/GoTools/intersections/intersections-doxymain.h File Reference

### 30.2056 intersections/include/GoTools/intersections/IntersectionUtils.h File Reference

```
#include "GoTools/geometry/SplineCurve.h"
#include "GoTools/intersections/AlgObj2DInt.h"
#include "GoTools/intersections/AlgObj3DInt.h"
#include "GoTools/geometry/SplineSurface.h"
#include "GoTools/implicitization/BernsteinMulti.h"
#include <vector>
```

Include dependency graph for IntersectionUtils.h:





## Namespaces

- [Go](#)
- [Go::IntersectionUtils](#)

*Various functions related to the intersection algorithms.*

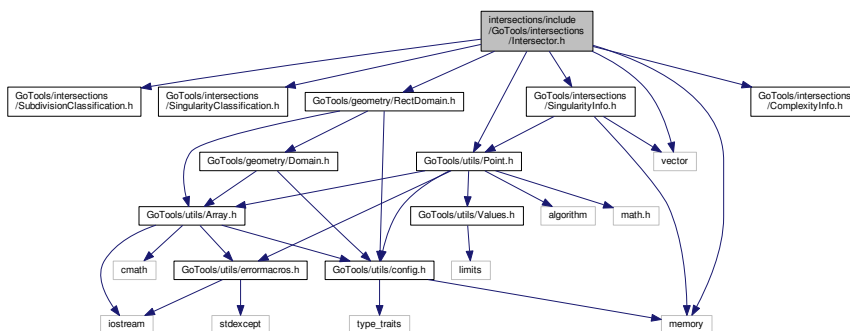
## Functions

- `shared_ptr< SplineCurve >` [Go::IntersectionUtils::create1DSplineCurve](#) (`const SplineCurve &cv`, `int dim_id`)
- `shared_ptr< SplineSurface >` [Go::IntersectionUtils::create1DSplineSurface](#) (`const SplineSurface &sf`, `int dim_id`)
- `shared_ptr< SplineCurve >` [Go::IntersectionUtils::splineCurveProduct](#) (`std::vector< shared_ptr< SplineCurve > > &cv`, `Alg2DElem term`)
- `shared_ptr< SplineSurface >` [Go::IntersectionUtils::splineSurfaceProduct](#) (`std::vector< shared_ptr< SplineSurface > > &sf`, `Alg3DElem term`)
- `shared_ptr< SplineCurve >` [Go::IntersectionUtils::insertCvInAlgcv](#) (`const SplineCurve &cv`, `AlgObj2DInt *alg_obj2d_int`)
- `shared_ptr< SplineSurface >` [Go::IntersectionUtils::insertSfInAlgsf](#) (`const SplineSurface &sf`, `AlgObj3DInt *alg_obj3d_int`)
- `shared_ptr< SplineSurface >` [Go::IntersectionUtils::insertSfInAlgsf2](#) (`const SplineSurface &sf`, `AlgObj3DInt *alg_obj3d_int`)
- `shared_ptr< SplineSurface >` [Go::IntersectionUtils::insertSfInImplObj](#) (`const SplineSurface &spline_sf`, `const BernsteinTetrahedralPoly &impl`, `const BaryCoordSystem3D &bc`)
- `double` [Go::IntersectionUtils::distImplRepresentationCompFunction](#) (`const SplineSurface &spline_sf`, `const BernsteinTetrahedralPoly &impl`, `const BaryCoordSystem3D &bc`, `const SplineSurface &comp_1d_sf`, `double upar`, `double vpar`)

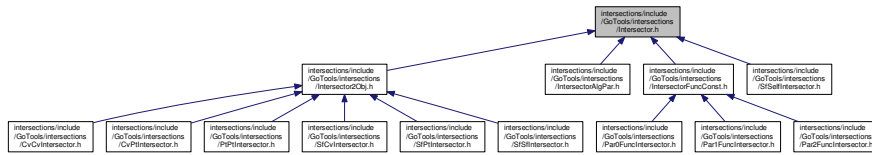
## 30.2057 intersections/include/GoTools/intersections/Intersector.h File Reference

```
#include "GoTools/intersections/SubdivisionClassification.h"
#include "GoTools/intersections/SingularityClassification.h"
#include "GoTools/intersections/SingularityInfo.h"
#include "GoTools/intersections/ComplexityInfo.h"
#include "GoTools/utils/Point.h"
#include "GoTools/geometry/RectDomain.h"
#include <vector>
#include <memory>
```

Include dependency graph for Intersector.h:



This graph shows which files directly or indirectly include this file:



Classes

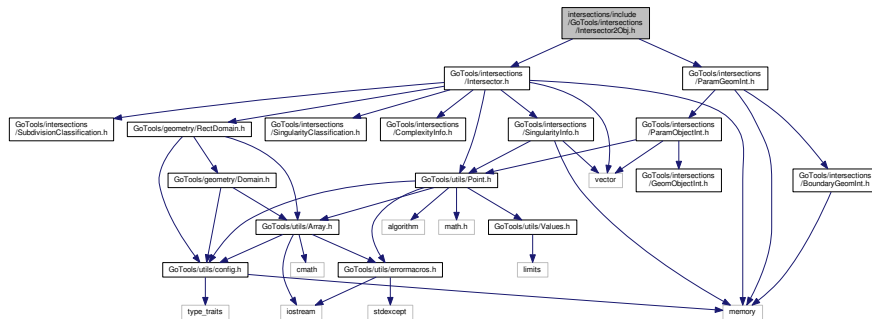
- class [Go::Intersector](#)

Namespaces

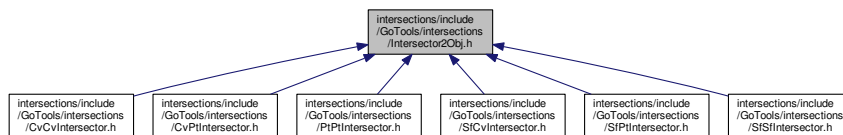
- [Go](#)

30.2058 intersections/include/GoTools/intersections/Intersector2Obj.h File Reference

```
#include "GoTools/intersections/Intersector.h"
#include "GoTools/intersections/ParamGeomInt.h"
Include dependency graph for Intersector2Obj.h:
```



This graph shows which files directly or indirectly include this file:

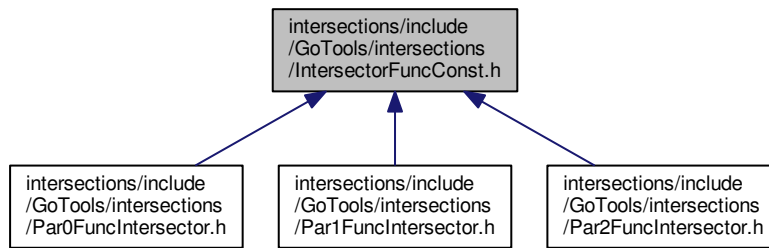


Classes

- class [Go::Intersector2Obj](#)



This graph shows which files directly or indirectly include this file:



## Classes

- class [Go::IntersectorFuncConst](#)

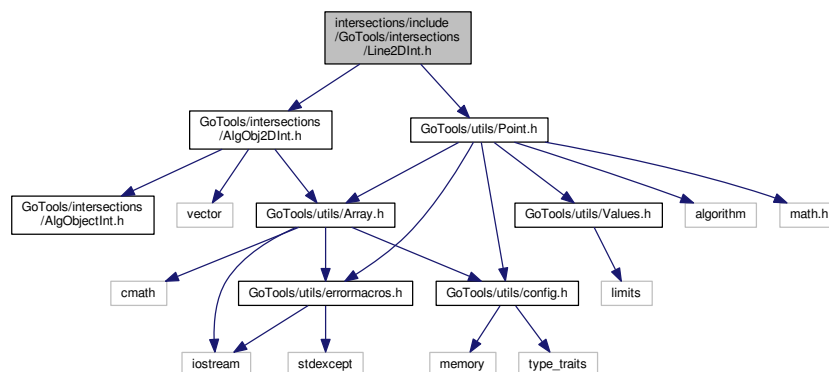
## Namespaces

- [Go](#)

## 30.2061 intersections/include/GoTools/intersections/Line2DInt.h File Reference

```
#include "GoTools/intersections/AlgObj2DInt.h"
#include "GoTools/utils/Point.h"
```

Include dependency graph for Line2DInt.h:



## Classes

- class [Go::Line2DInt](#)

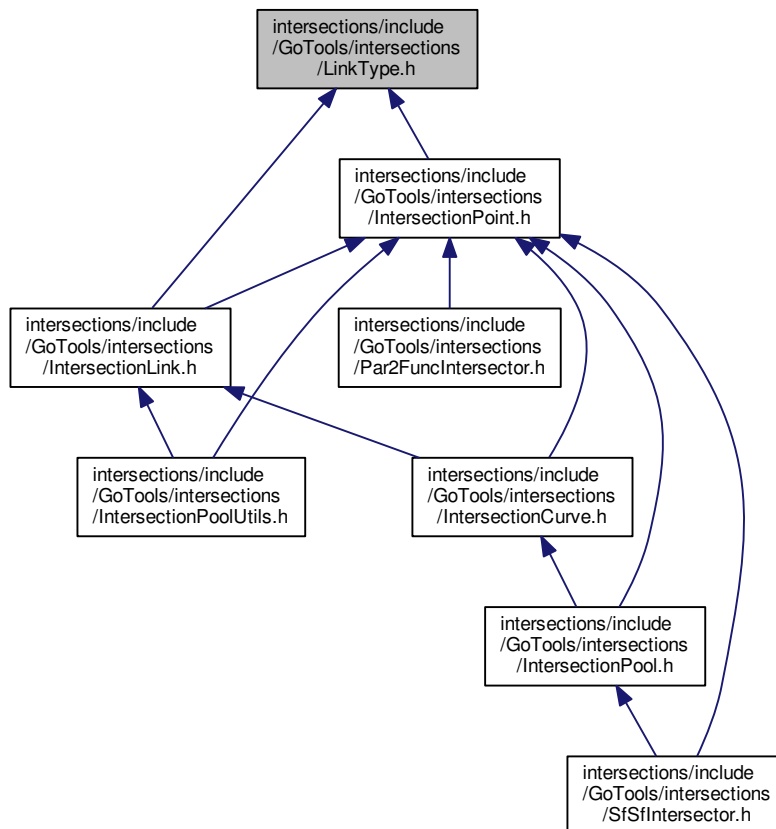
*Class representing an algebraic line in 2-dimensional space.*

## Namespaces

- [Go](#)

## 30.2062 intersections/include/GoTools/intersections/LinkType.h File Reference

This graph shows which files directly or indirectly include this file:



## Namespaces

- [Go](#)

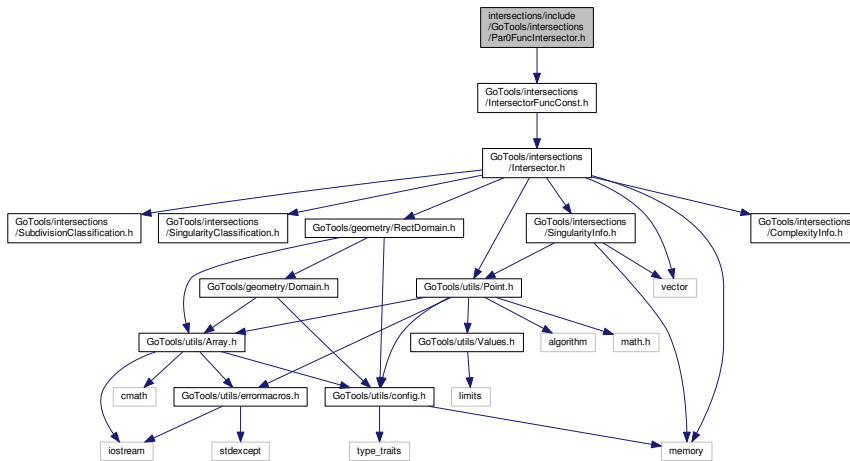
## Enumerations

- enum `Go::LinkType` {  
`Go::LINK_UNDEFINED = 0`, `Go::SIMPLE_CONE`, `Go::SIMPLE_MONOTONE`, `Go::SIMPLE_IMPLICIT`,  
`Go::SIMPLE_TWO_POINTS`, `Go::COINCIDENCE_CVPT`, `Go::COINCIDENCE_CVCV`, `Go::COINCIDENCE_SFCV`,  
`Go::COINCIDENCE_SFPT`, `Go::MICRO_CVCV`, `Go::MICRO_SFPT`, `Go::MICRO_SFCV`,  
`Go::MICRO_SFSF`, `Go::MICRO_PAR1FUNC`, `Go::MICRO_PAR2FUNC`, `Go::LINEAR_CVCV`,  
`Go::LINEAR_SFCV`, `Go::LINEAR_SFSF`, `Go::MERGED_UNDEFINED`, `Go::MERGED_SIMPLE_CONE`,  
`Go::MERGED_SIMPLE_MONOTONE`, `Go::MERGED_SIMPLE_CONE_MONOTONE`, `Go::MERGED_COINCIDENCE_SFCV_SFCV`,  
`Go::DEG_TRIANGLE`,  
`Go::POST_ITERATE`, `Go::BRANCH_CONNECTION`, `Go::COMPLEX_SFSF`, `Go::SPLIT_LINK`,  
`Go::REPAIRED_MISSING_LINK`, `Go::INSIDE_OUTSIDE_SINGULARITY_BOX` }

## 30.2063 intersections/include/GoTools/intersections/Par0FuncIntersector.h File Reference

```
#include "GoTools/intersections/IntersectorFuncConst.h"
```

Include dependency graph for Par0FuncIntersector.h:



### Classes

- class [Go::Par0FuncIntersector](#)

*This class is performing intersections between two constants.*

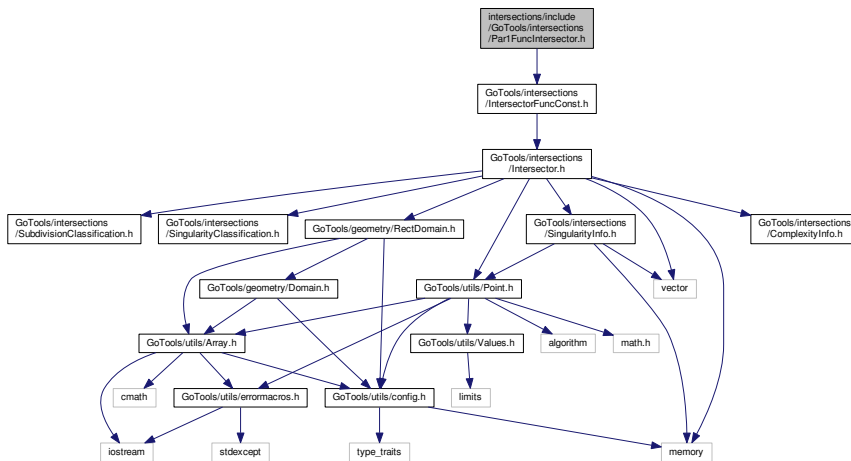
### Namespaces

- [Go](#)

## 30.2064 intersections/include/GoTools/intersections/Par1FuncIntersector.h File Reference

```
#include "GoTools/intersections/IntersectorFuncConst.h"
```

Include dependency graph for Par1FuncIntersector.h:



### Classes

- class [Go::Par1FuncIntersector](#)

### Namespaces

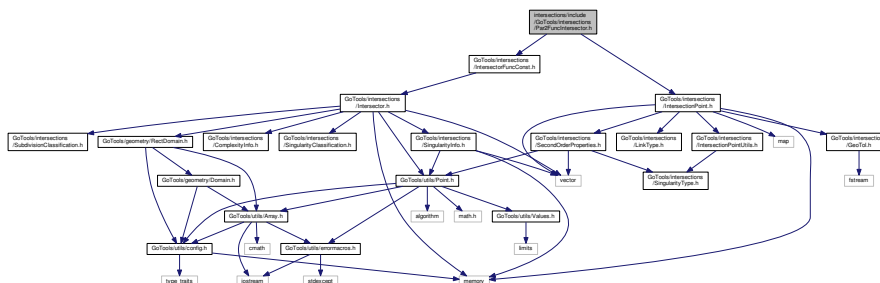
- [Go](#)

## 30.2065 intersections/include/GoTools/intersections/Par2FuncIntersector.h File Reference

```
#include "GoTools/intersections/IntersectorFuncConst.h"
```

```
#include "GoTools/intersections/IntersectionPoint.h"
```

Include dependency graph for Par2FuncIntersector.h:



### Classes

- class [Go::Par2FuncIntersector](#)

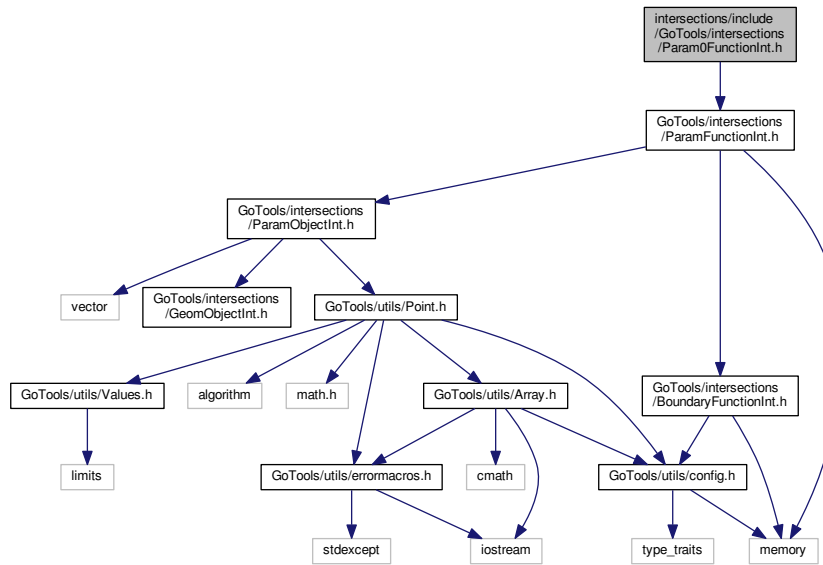
## Namespaces

- [Go](#)

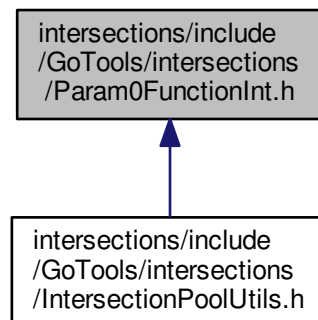
## 30.2066 intersections/include/GoTools/intersections/Param0FunctionInt.h File Reference

```
#include "GoTools/intersections/ParamFunctionInt.h"
```

Include dependency graph for Param0FunctionInt.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [Go::Param0FunctionInt](#)





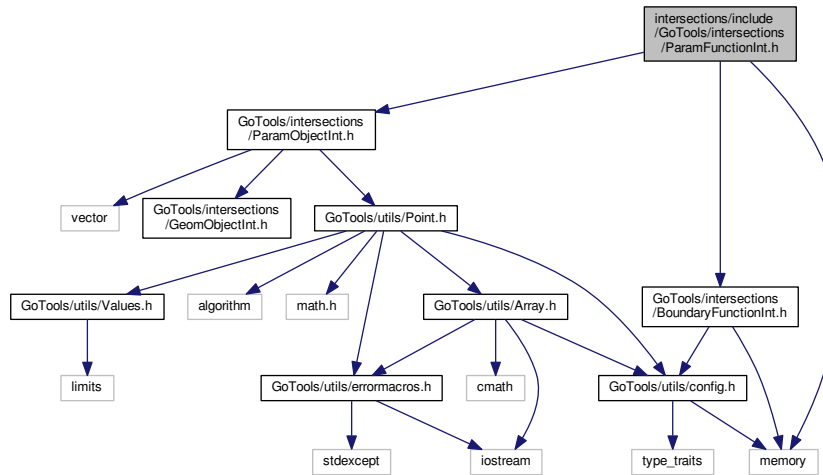




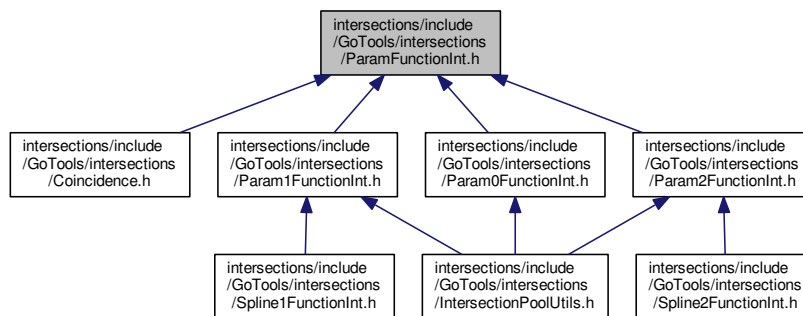
## 30.2070 intersections/include/GoTools/intersections/ParamFunctionInt.h File Reference

```
#include "GoTools/intersections/ParamObjectInt.h"
#include "GoTools/intersections/BoundaryFunctionInt.h"
#include <memory>
```

Include dependency graph for ParamFunctionInt.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [Go::ParamFunctionInt](#)

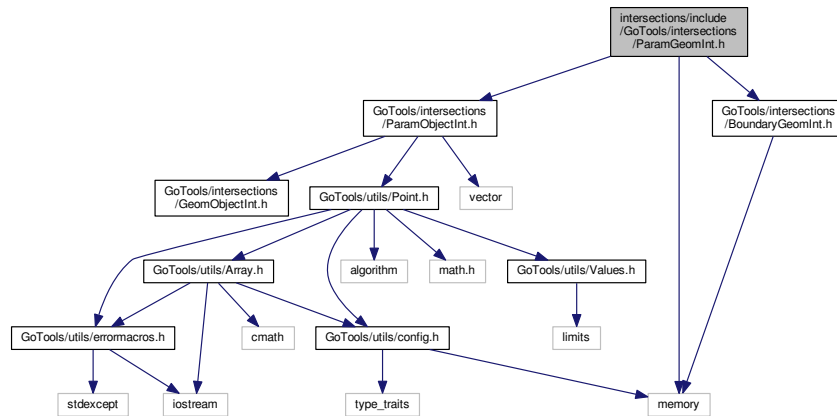
## Namespaces

- [Go](#)

## 30.2071 intersections/include/GoTools/intersections/ParamGeomInt.h File Reference

```
#include "GoTools/intersections/ParamObjectInt.h"
#include "GoTools/intersections/BoundaryGeomInt.h"
#include <memory>
```

Include dependency graph for ParamGeomInt.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [Go::ParamGeomInt](#)

## Namespaces

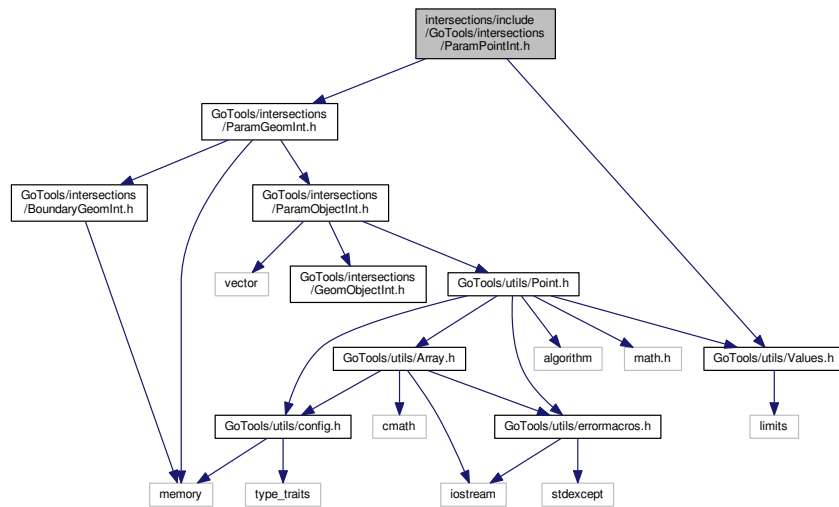
- [Go](#)

## 30.2072 intersections/include/GoTools/intersections/ParamObjectInt.h File Reference

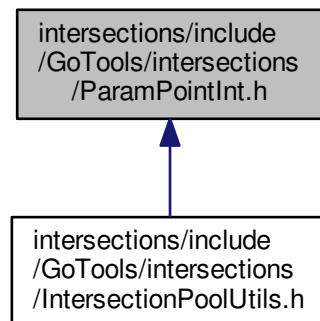
```
#include "GoTools/intersections/GeomObjectInt.h"
#include "GoTools/Utils/Point.h"
#include <vector>
```



Include dependency graph for ParamPointInt.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [Go::ParamPointInt](#)

## Namespaces

- [Go](#)





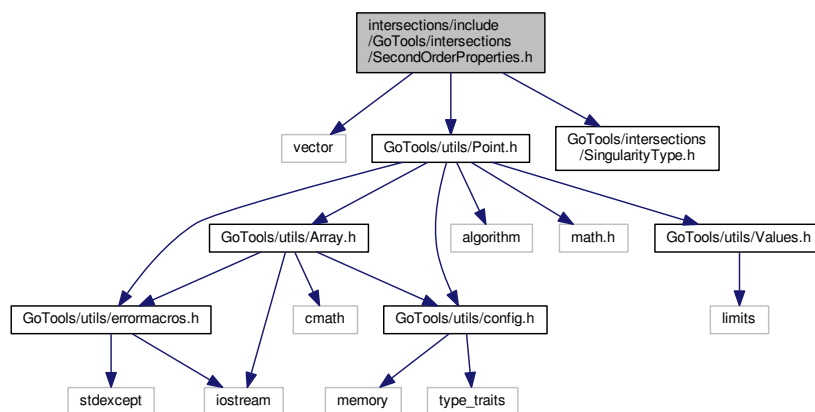


## Namespaces

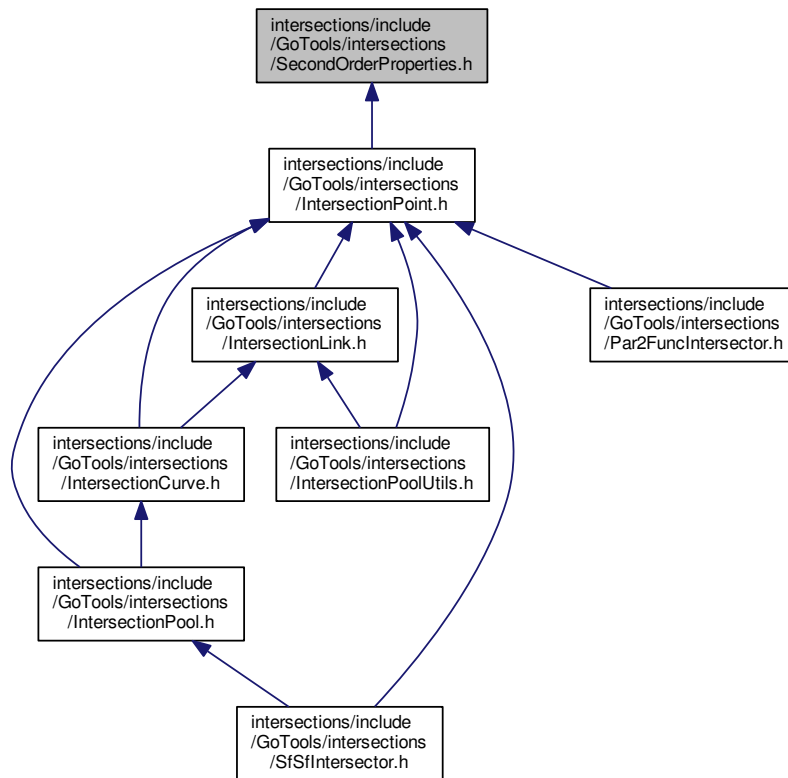
- [Go](#)

### 30.2077 intersections/include/GoTools/intersections/SecondOrderProperties.h File Reference

```
#include <vector>
#include "GoTools/utils/Point.h"
#include "GoTools/intersections/SingularityType.h"
Include dependency graph for SecondOrderProperties.h:
```



This graph shows which files directly or indirectly include this file:



## Classes

- struct [Go::SecondOrderProperties](#)

## Namespaces

- [Go](#)

## 30.2078 intersections/include/GoTools/intersections/SfCvIntersector.h File Reference

```
#include "GoTools/intersections/Intersector2Obj.h"
```





## Classes

- class [Go::SfSfIntersector](#)

*This class performs intersection between two parametric surfaces.*

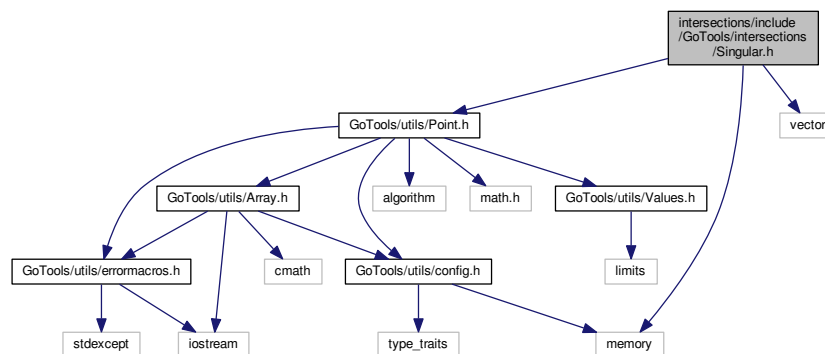
## Namespaces

- [Go](#)

## 30.2082 intersections/include/GoTools/intersections/Singular.h File Reference

```
#include "GoTools/utils/Point.h"
#include <vector>
#include <memory>
```

Include dependency graph for Singular.h:



## Namespaces

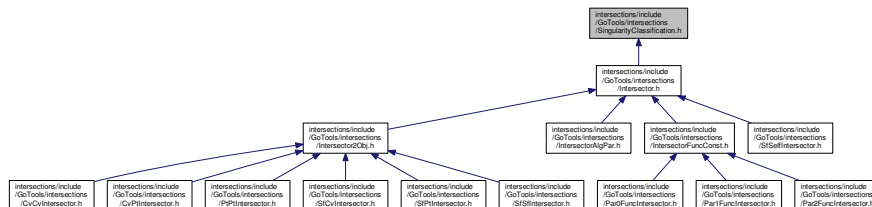
- [Go](#)
- [Go::Singular](#)

## Functions

- void [Go::Singular::vanishingNormal](#) (shared\_ptr< ParamSurface > srf, double tol, std::vector< Point > &singular\_pts, std::vector< std::vector< Point > > &singular\_sequences)
- void [Go::Singular::vanishingTangent](#) (shared\_ptr< ParamCurve > crv, double start, double end, double tol, std::vector< double > &singular\_pts, std::vector< std::vector< double > > &singular\_sequences)

## 30.2083 intersections/include/GoTools/intersections/SingularityClassification.h File Reference

This graph shows which files directly or indirectly include this file:



## Namespaces

- [Go](#)

## Enumerations

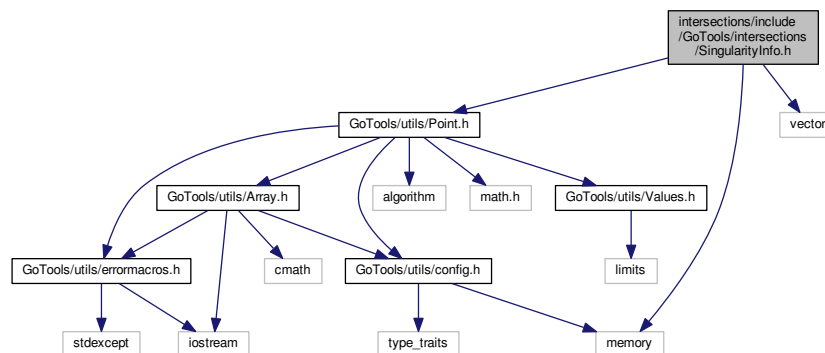
- enum [Go::SingularityClassification](#) { [Go::NO\\_SING](#) = 0, [Go::INIT\\_SING](#), [Go::DIVIDED\\_SING](#), [Go::KEEP\\_SING](#) }

*This enum classifies the type of singularity.*

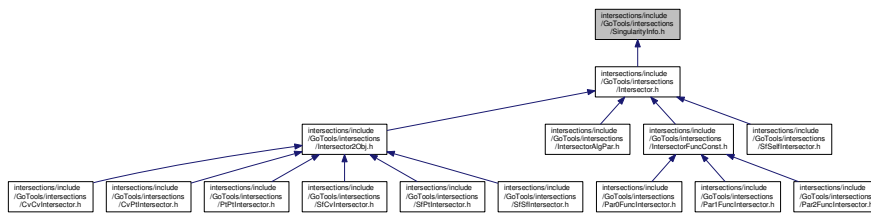
## 30.2084 intersections/include/GoTools/intersections/SingularityInfo.h File Reference

```
#include "GoTools/utils/Point.h"
#include <vector>
#include <memory>
```

Include dependency graph for SingularityInfo.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [Go::SingularityInfo](#)

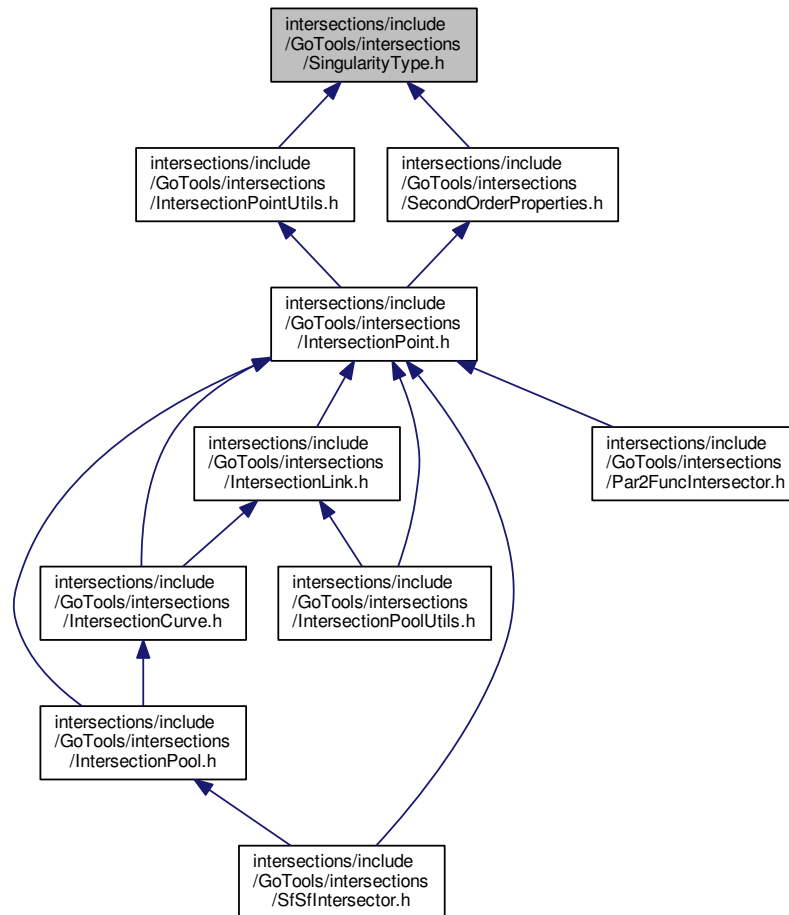
## Namespaces

- [Go](#)



## 30.2085 intersections/include/GoTools/intersections/SingularityType.h File Reference

This graph shows which files directly or indirectly include this file:



### Namespaces

- [Go](#)

### Enumerations

- `enum Go::SingularityType {  
    Go::ORDINARY\_POINT = 0, Go::TANGENTIAL\_POINT, Go::ISOLATED\_POINT, Go::BRANCH\_POINT,  
    Go::HIGHER\_ORDER\_POINT }`









## Classes

- class [Go::SurfaceAssembly](#)

## Namespaces

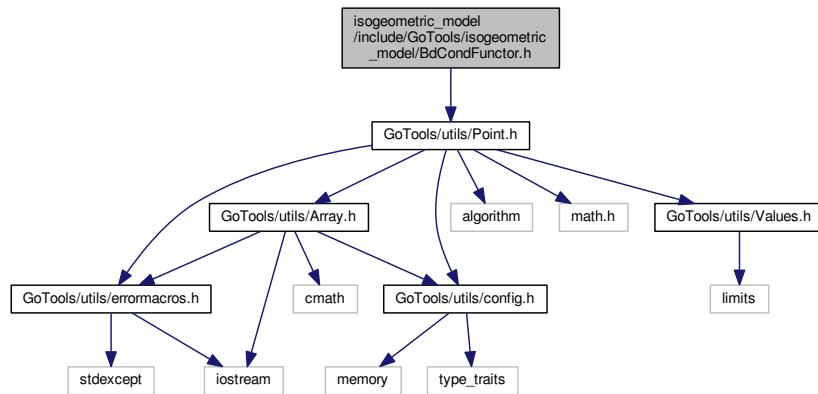
- [Go](#)

## 30.2093 intersections/include/GoTools/intersections/TorusInt.h File Reference

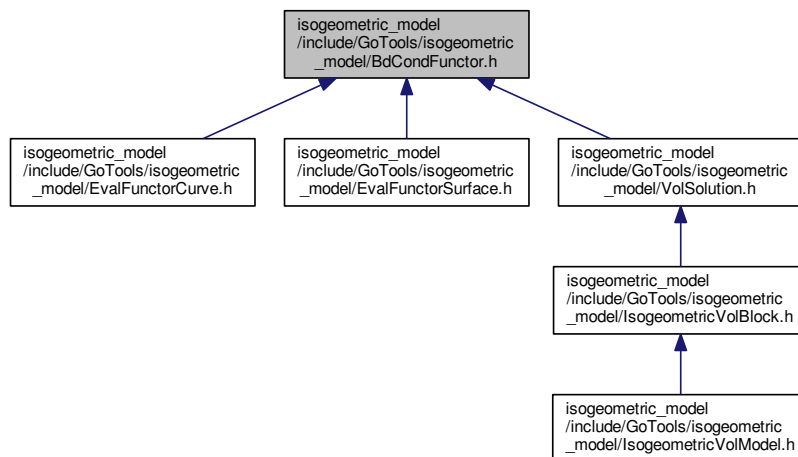
## 30.2094 isogeometric\_model/include/GoTools/isogeometric\_model/BdCondFuncor.h File Reference

```
#include "GoTools/utils/Point.h"
```

Include dependency graph for BdCondFuncor.h:



This graph shows which files directly or indirectly include this file:



## Classes

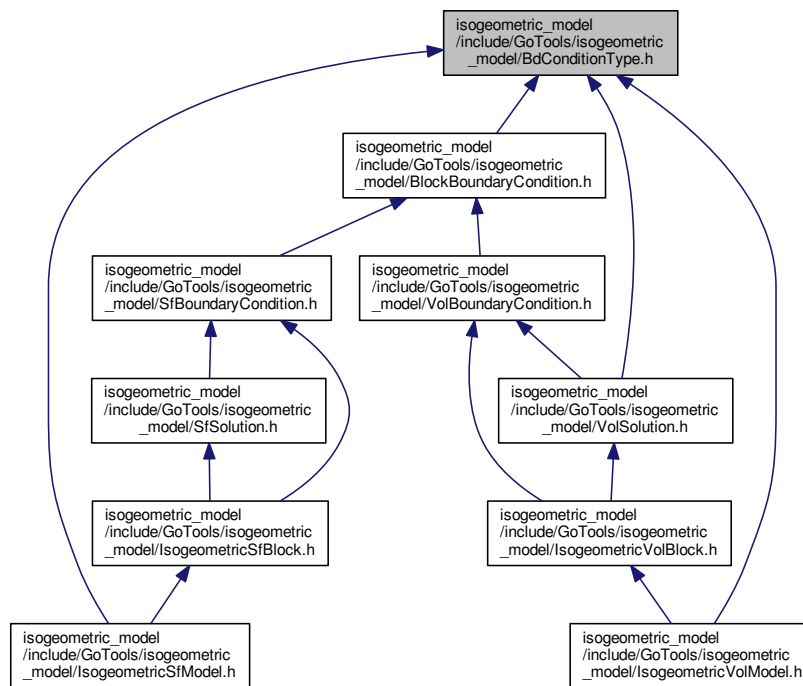
- class [Go::BdCondFuncor](#)

## Namespaces

- [Go](#)

## 30.2095 isogeometric\_model/include/GoTools/isogeometric\_model/BdConditionType.h File Reference

This graph shows which files directly or indirectly include this file:



## Namespaces

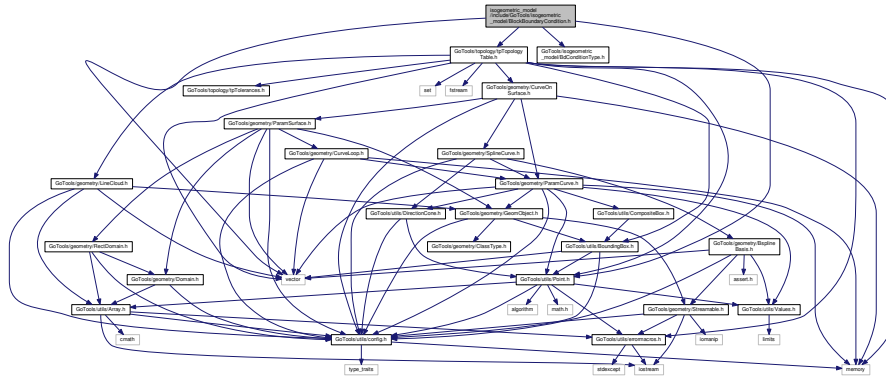
- [Go](#)

## Enumerations

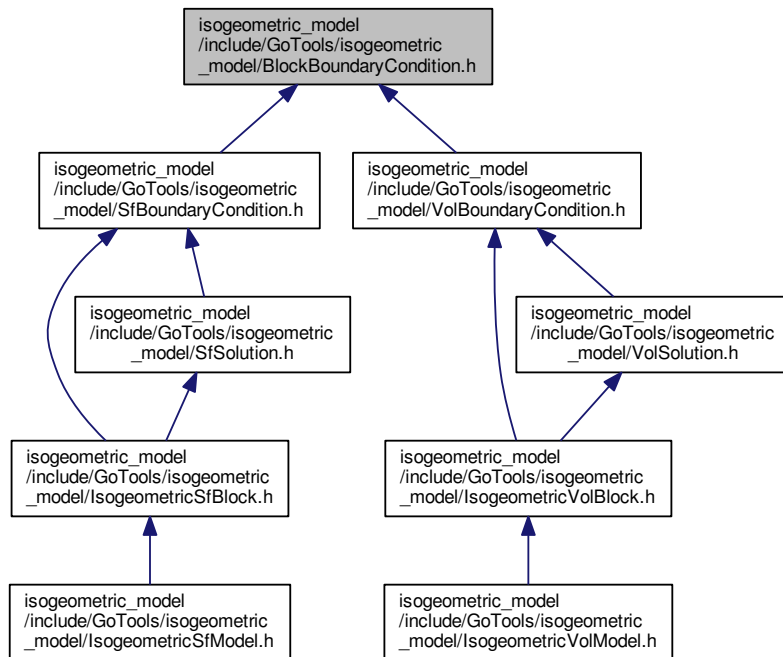
- enum [Go::BdConditionType](#) {  
[Go::UNKNOWN](#) = 0, [Go::ZERO\\_DIRICHLET](#), [Go::CONSTANT\\_DIRICHLET](#), [Go::DIRICHLET](#),  
[Go::ZERO\\_NEUMANN](#), [Go::NEUMANN](#), [Go::SYMMETRY](#) }

### 30.2096 isogeometric\_model/include/GoTools/isogeometric\_model/BlockBoundaryCondition.h File Reference

```
#include <vector>
#include "GoTools/Utils/Point.h"
#include "GoTools/isogeometric_model/BdConditionType.h"
#include "GoTools/topology/tpTopologyTable.h"
Include dependency graph for BlockBoundaryCondition.h:
```



This graph shows which files directly or indirectly include this file:



#### Classes

- class [Go::BlockBoundaryCondition](#)

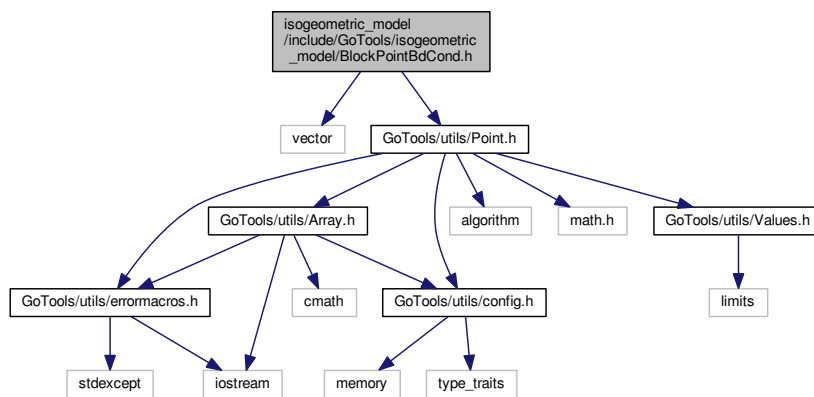


## Namespaces

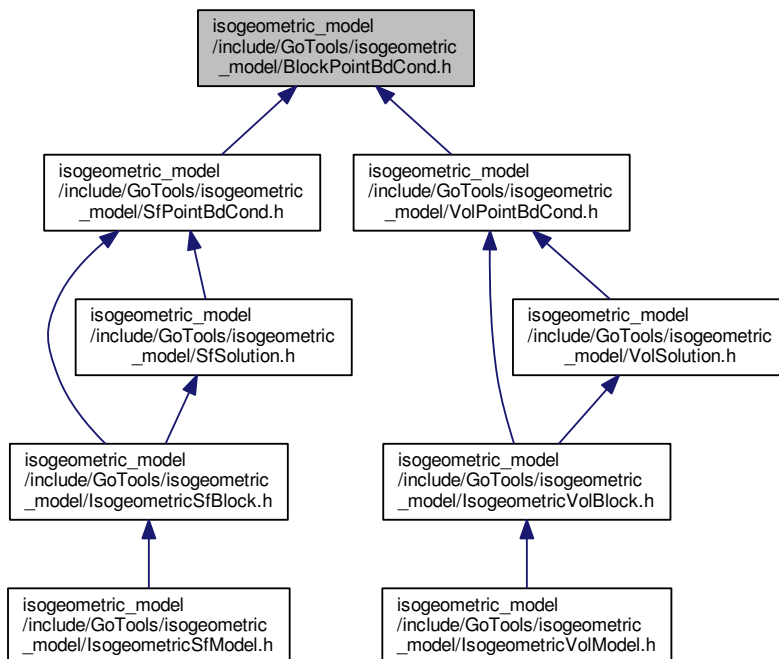
- [Go](#)

## 30.2097 isogeometric\_model/include/GoTools/isogeometric\_model/BlockPointBdCond.h File Reference

```
#include <vector>
#include "GoTools/utils/Point.h"
Include dependency graph for BlockPointBdCond.h:
```

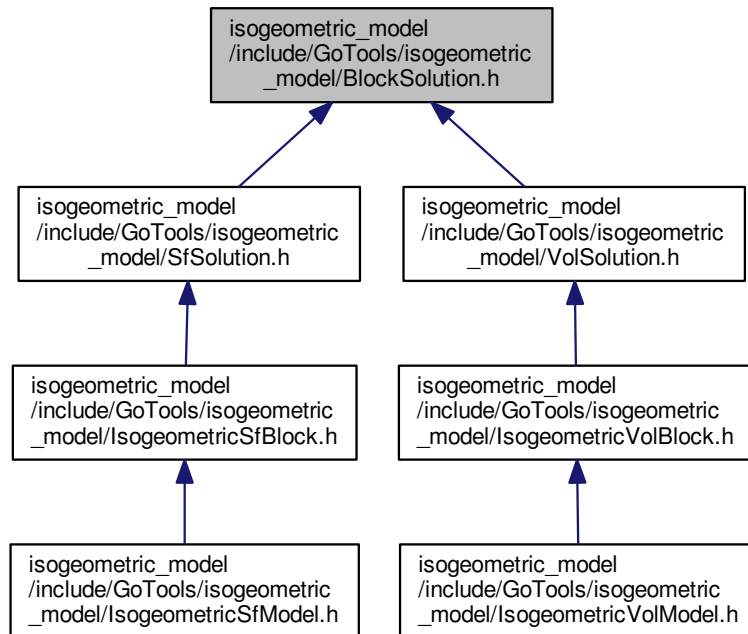


This graph shows which files directly or indirectly include this file:





This graph shows which files directly or indirectly include this file:



## Classes

- class [Go::BlockSolution](#)

## Namespaces

- [Go](#)

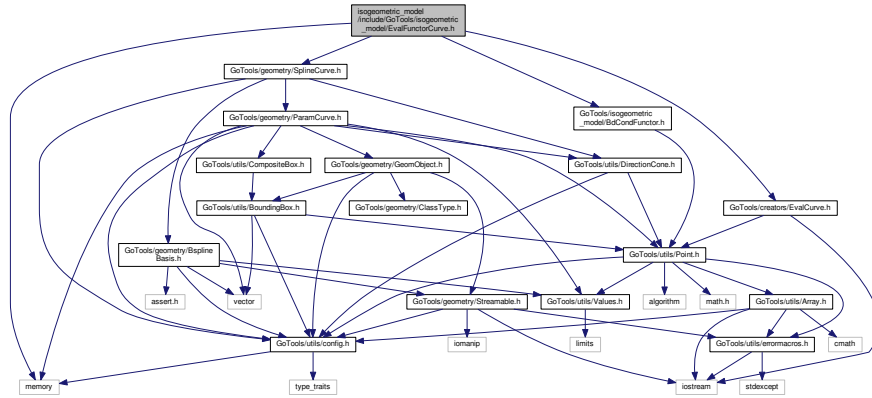
## 30.2099 isogeometric\_model/include/GoTools/isogeometric\_model/EvalFunctorCurve.h File Reference

```

#include <memory>
#include "GoTools/creators/EvalCurve.h"
#include "GoTools/geometry/SplineCurve.h"
#include "GoTools/isogeometric_model/BdCondFunctor.h"

```

Include dependency graph for EvalFunctorCurve.h:



Classes

- class [Go::EvalFunctorCurve](#)

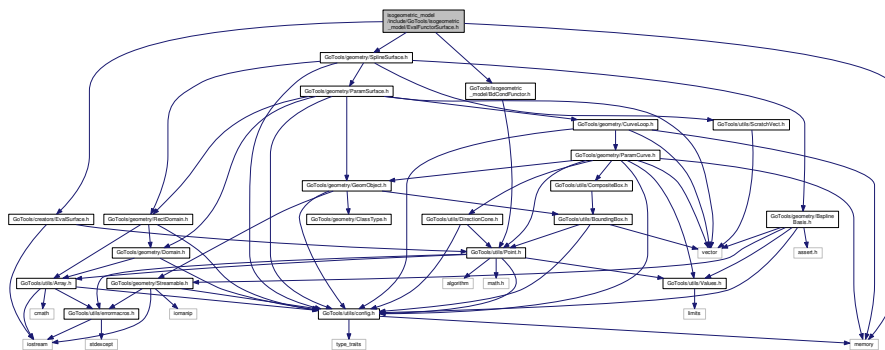
Namespaces

- [Go](#)

30.2100 isogeometric\_model/include/GoTools/isogeometric\_model/EvalFunctorSurface.h  
File Reference

```
#include <memory>
#include "GoTools/creators/EvalSurface.h"
#include "GoTools/geometry/SplineSurface.h"
#include "GoTools/isogeometric_model/BdCondFunctor.h"
```

Include dependency graph for EvalFunctorSurface.h:



Classes

- class [Go::EvalFunctorSurface](#)

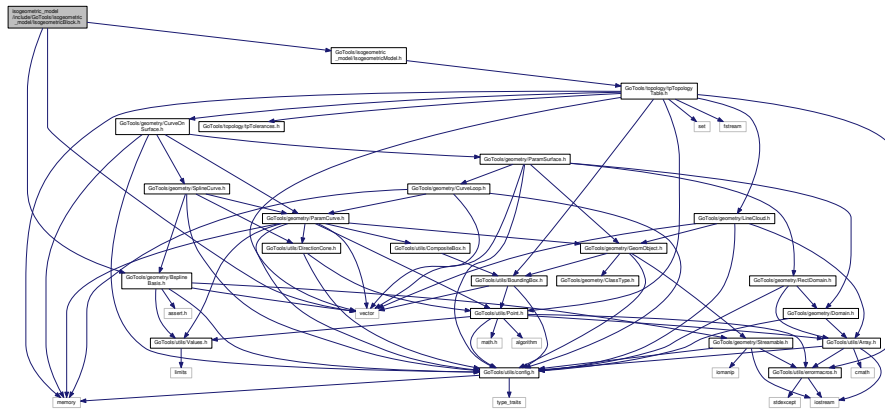
Namespaces

- [Go](#)

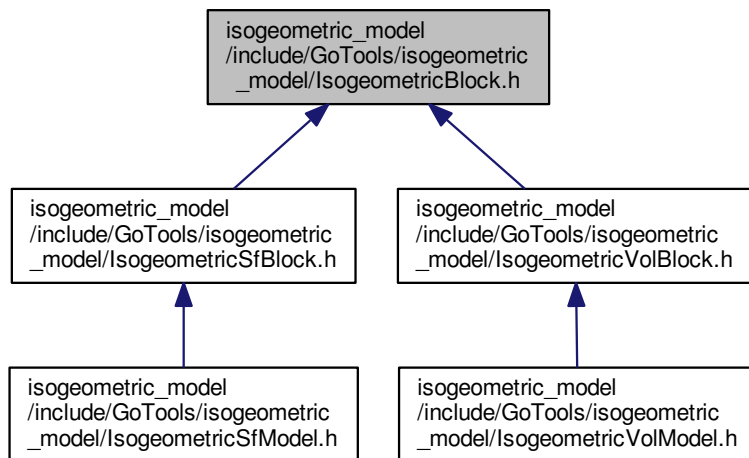
30.2101 isogeometric\_model/include/GoTools/isogeometric\_model/isogeometric\_model-doxymain.h File Reference

30.2102 isogeometric\_model/include/GoTools/isogeometric\_model/IsogeometricBlock.h File Reference

```
#include <vector>
#include "GoTools/geometry/BsplineBasis.h"
#include "GoTools/isogeometric_model/IsogeometricModel.h"
Include dependency graph for IsogeometricBlock.h:
```



This graph shows which files directly or indirectly include this file:



## Classes

- class [Go::IsogeometricBlock](#)

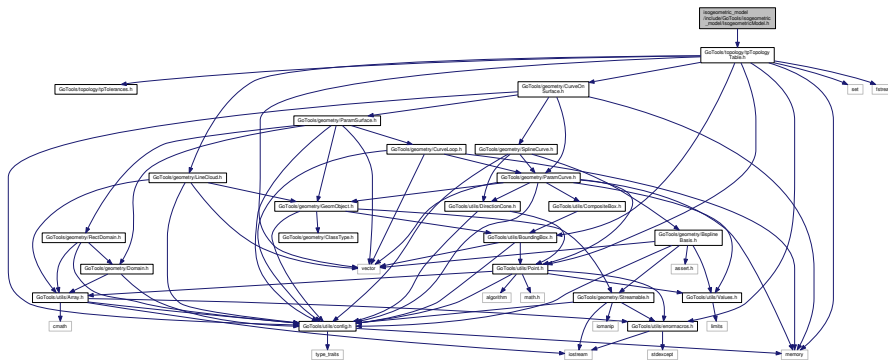
## Namespaces

- [Go](#)

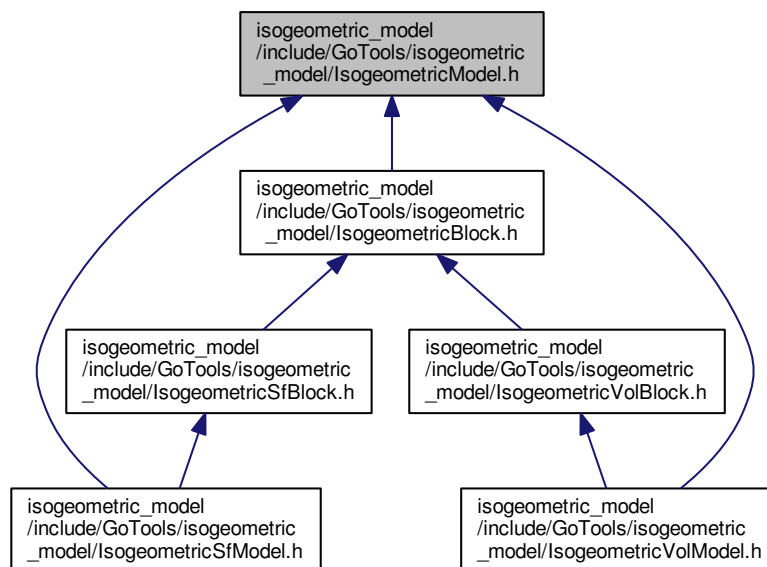
### 30.2103 isogeometric\_model/include/GoTools/isogeometric\_model/IsogeometricModel.h File Reference

```
#include "GoTools/topology/tpTopologyTable.h"
```

Include dependency graph for IsogeometricModel.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [Go::IsogeometricModel](#)

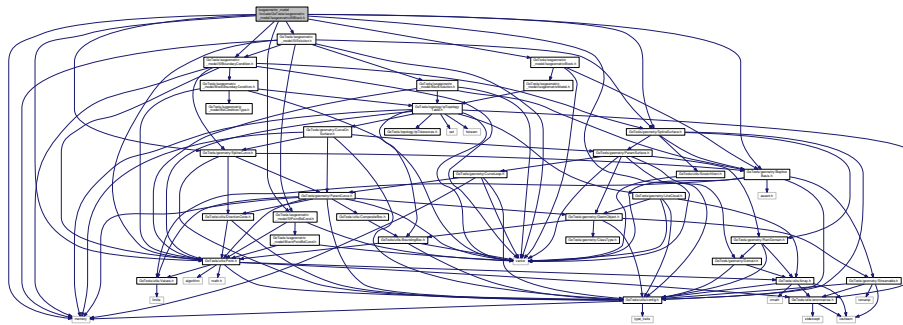
## Namespaces

- [Go](#)

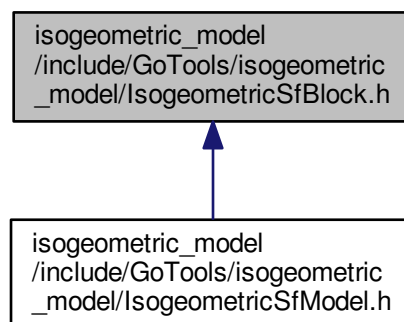
## 30.2104 isogeometric\_model/include/GoTools/isogeometric\_model/IsogeometricSfBlock.h File Reference

```
#include <vector>
#include <memory>
#include "GoTools/geometry/BsplineBasis.h"
#include "GoTools/geometry/SplineCurve.h"
#include "GoTools/geometry/SplineSurface.h"
#include "GoTools/utils/Point.h"
#include "GoTools/isogeometric_model/IsogeometricBlock.h"
#include "GoTools/isogeometric_model/SfSolution.h"
#include "GoTools/isogeometric_model/SfBoundaryCondition.h"
#include "GoTools/isogeometric_model/SfPointBdCond.h"
```

Include dependency graph for IsogeometricSfBlock.h:



This graph shows which files directly or indirectly include this file:



## Classes

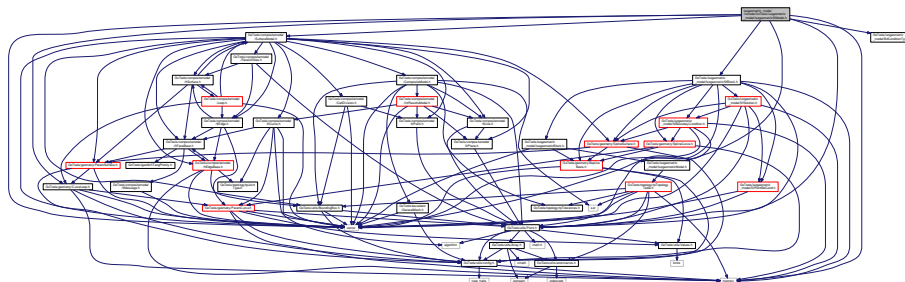
- class [Go::IsogeometricSfBlock](#)

## Namespaces

- [Go](#)

### 30.2105 isogeometric\_model/include/GoTools/isogeometric\_model/IsogeometricSfModel.h File Reference

```
#include <vector>
#include <memory>
#include "GoTools/utils/Point.h"
#include "GoTools/compositemodel/SurfaceModel.h"
#include "GoTools/geometry/CurveLoop.h"
#include "GoTools/isogeometric_model/IsogeometricModel.h"
#include "GoTools/isogeometric_model/BdConditionType.h"
#include "GoTools/isogeometric_model/IsogeometricSfBlock.h"
Include dependency graph for IsogeometricSfModel.h:
```



## Classes

- class [Go::IsogeometricSfModel](#)

## Namespaces

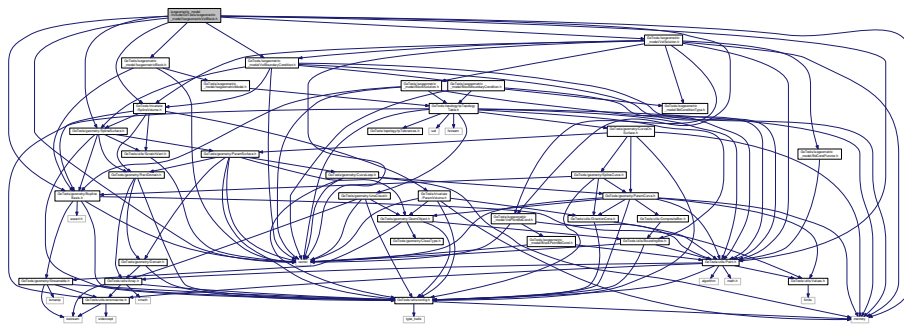
- [Go](#)



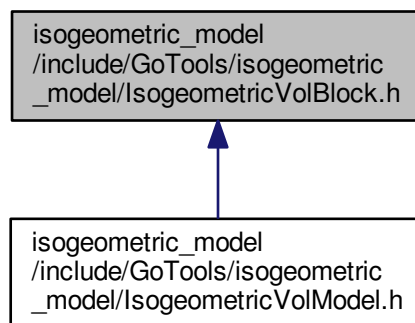
## 30.2106 isogeometric\_model/include/GoTools/isogeometric\_model/IsogeometricVolBlock.h File Reference

```
#include <vector>
#include <memory>
#include "GoTools/geometry/BsplineBasis.h"
#include "GoTools/geometry/SplineSurface.h"
#include "GoTools/utils/Point.h"
#include "GoTools/trivariate/SplineVolume.h"
#include "GoTools/isogeometric_model/IsogeometricBlock.h"
#include "GoTools/isogeometric_model/VolSolution.h"
#include "GoTools/isogeometric_model/VolBoundaryCondition.h"
#include "GoTools/isogeometric_model/VolPointBdCond.h"
```

Include dependency graph for IsogeometricVolBlock.h:



This graph shows which files directly or indirectly include this file:



### Classes

- class [Go::IsogeometricVolBlock](#)

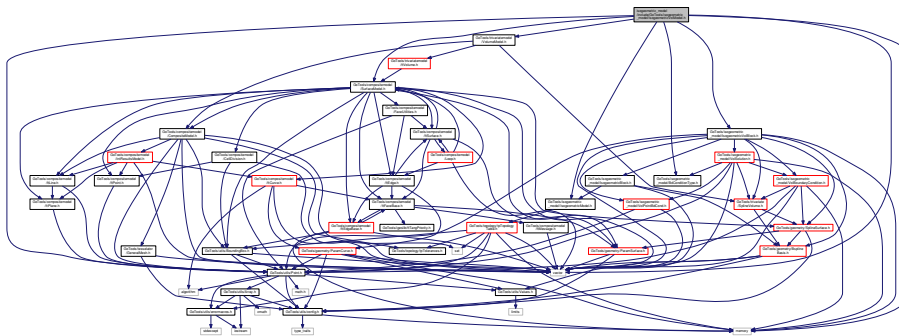
### Namespaces

- [Go](#)

## 30.2107 isogeometric\_model/include/GoTools/isogeometric\_model/IsogeometricVolModel.h File Reference

```
#include <vector>
#include <memory>
#include "GoTools/utils/Point.h"
#include "GoTools/compositemodel/SurfaceModel.h"
#include "GoTools/trivariatemodel/VolumeModel.h"
#include "GoTools/isogeometric_model/IsogeometricModel.h"
#include "GoTools/isogeometric_model/BdConditionType.h"
#include "GoTools/isogeometric_model/IsogeometricVolBlock.h"
```

Include dependency graph for IsogeometricVolModel.h:



### Classes

- class [Go::IsogeometricVolModel](#)

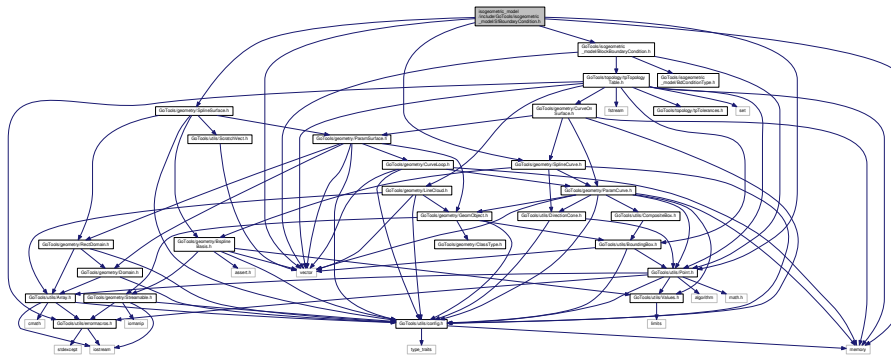
### Namespaces

- [Go](#)

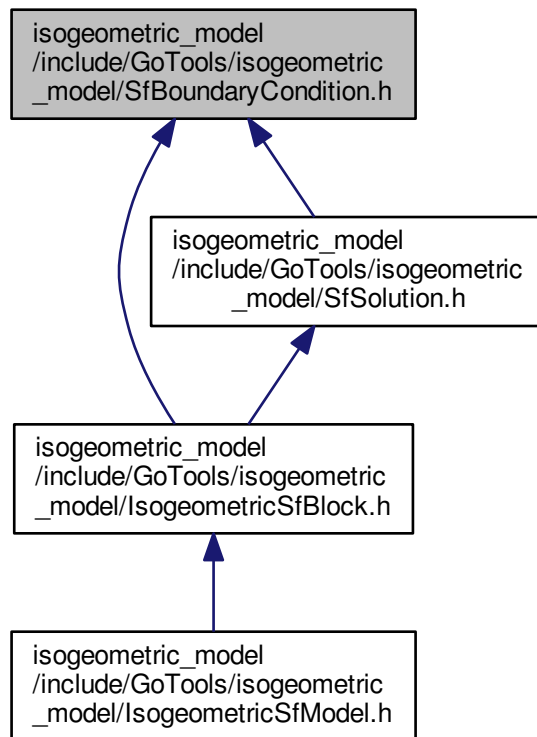
## 30.2108 isogeometric\_model/include/GoTools/isogeometric\_model/SfBoundaryCondition.h File Reference

```
#include <vector>
#include <memory>
#include "GoTools/geometry/SplineCurve.h"
#include "GoTools/geometry/SplineSurface.h"
#include "GoTools/utils/Point.h"
#include "GoTools/isogeometric_model/BlockBoundaryCondition.h"
```

Include dependency graph for SfBoundaryCondition.h:



This graph shows which files directly or indirectly include this file:



### Classes

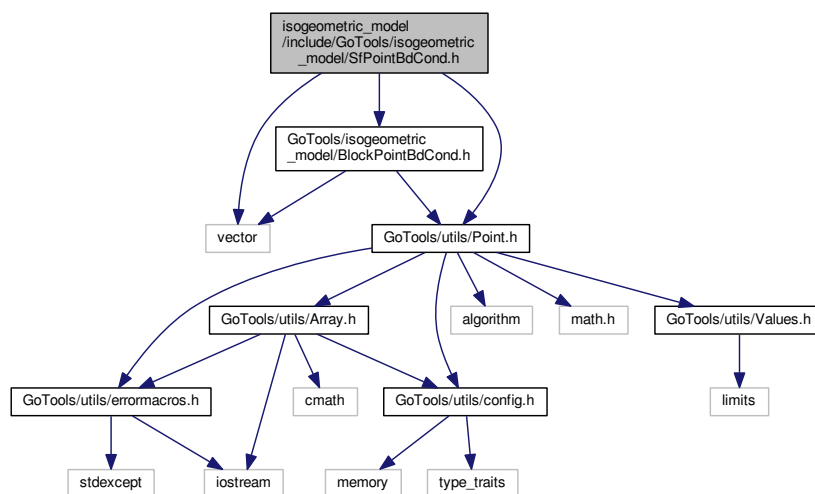
- class [Go::SfBoundaryCondition](#)

### Namespaces

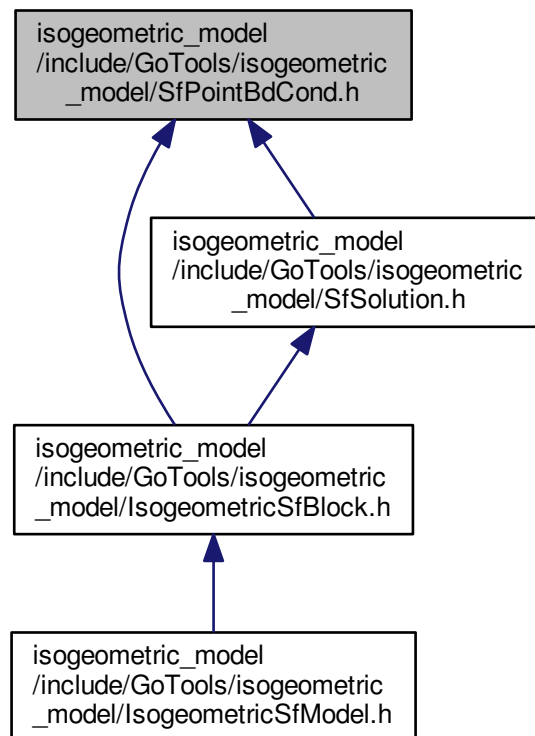
- [Go](#)

### 30.2109 isogeometric\_model/include/GoTools/isogeometric\_model/SfPointBdCond.h File Reference

```
#include <vector>
#include "GoTools/utils/Point.h"
#include "GoTools/isogeometric_model/BlockPointBdCond.h"
Include dependency graph for SfPointBdCond.h:
```



This graph shows which files directly or indirectly include this file:



## Classes

- class [Go::SfPointBdCond](#)

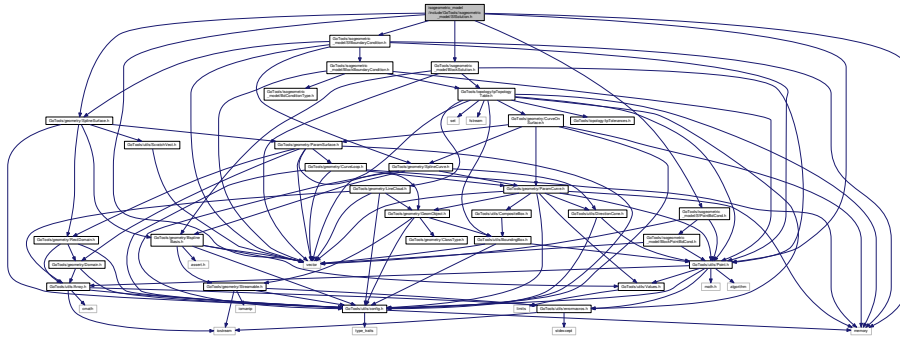
## Namespaces

- [Go](#)

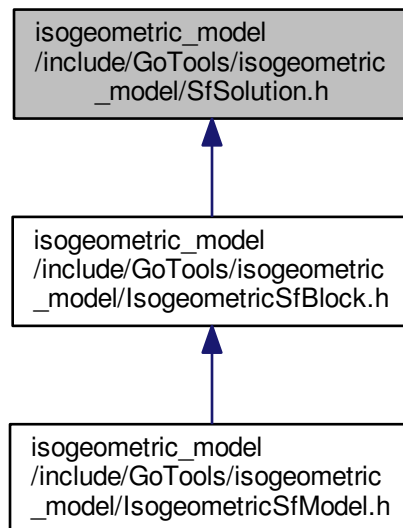
## 30.2110 isogeometric\_model/include/GoTools/isogeometric\_model/SfSolution.h File Reference

```
#include <vector>
#include <memory>
#include "GoTools/geometry/SplineSurface.h"
#include "GoTools/utils/Point.h"
#include "GoTools/isogeometric_model/BlockSolution.h"
#include "GoTools/isogeometric_model/SfBoundaryCondition.h"
#include "GoTools/isogeometric_model/SfPointBdCond.h"
```

Include dependency graph for SfSolution.h:



This graph shows which files directly or indirectly include this file:



## Classes

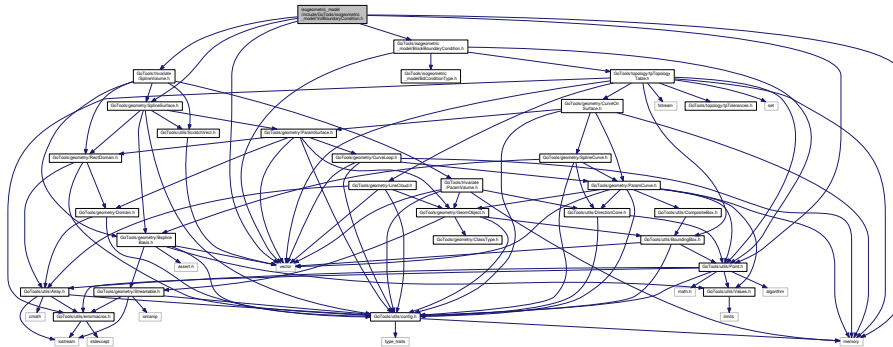
- struct [Go::preEvaluationSf](#)
- class [Go::SfSolution](#)

## Namespaces

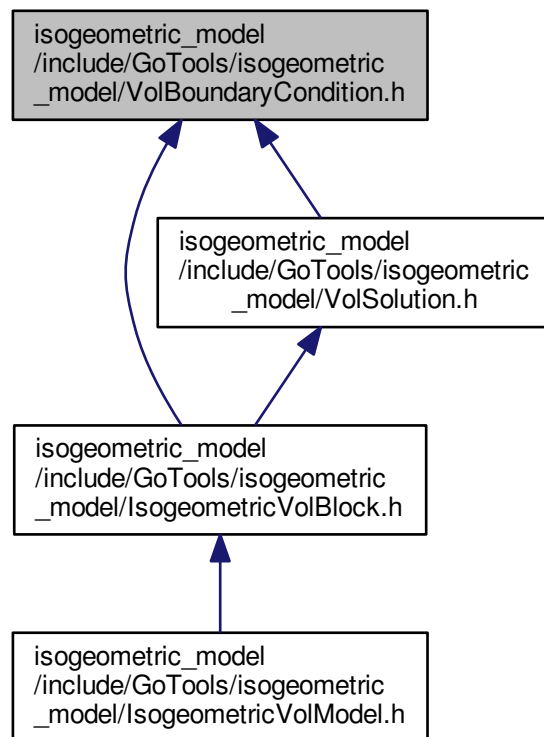
- [Go](#)

### 30.2111 isogeometric\_model/include/GoTools/isogeometric\_model/VolBoundaryCondition.h File Reference

```
#include <vector>
#include <memory>
#include "GoTools/geometry/SplineSurface.h"
#include "GoTools/utils/Point.h"
#include "GoTools/trivariate/SplineVolume.h"
#include "GoTools/isogeometric_model/BlockBoundaryCondition.h"
Include dependency graph for VolBoundaryCondition.h:
```



This graph shows which files directly or indirectly include this file:



## Classes

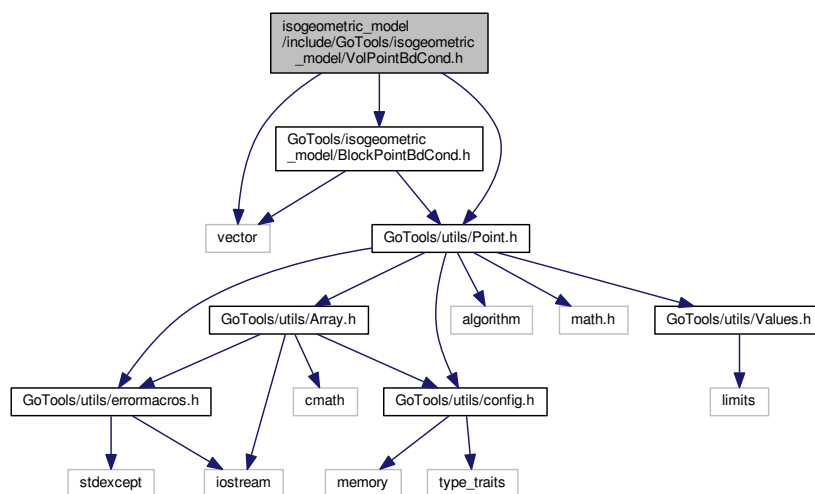
- class [Go::VolBoundaryCondition](#)

## Namespaces

- [Go](#)

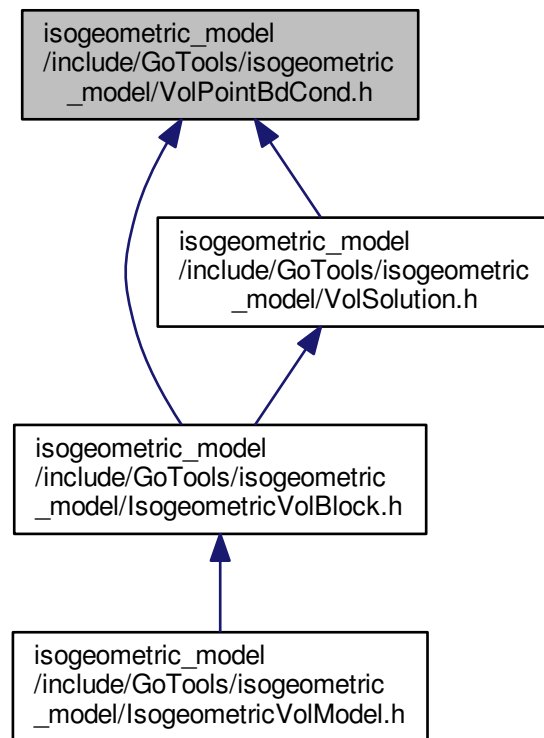
### 30.2112 isogeometric\_model/include/GoTools/isogeometric\_model/VolPointBdCond.h File Reference

```
#include <vector>
#include "GoTools/utils/Point.h"
#include "GoTools/isogeometric_model/BlockPointBdCond.h"
Include dependency graph for VolPointBdCond.h:
```





This graph shows which files directly or indirectly include this file:



## Classes

- class [Go::VolPointBdCond](#)

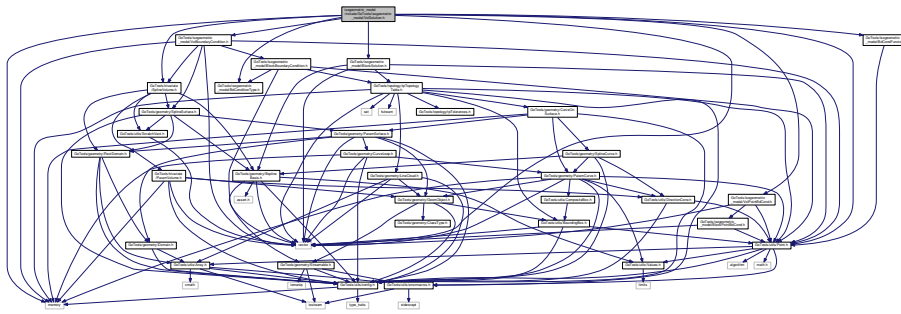
## Namespaces

- [Go](#)

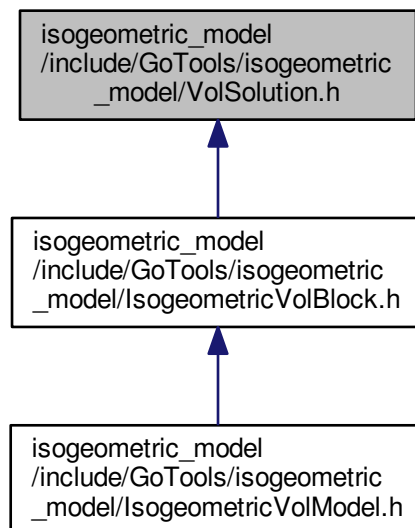
## 30.2113 isogeometric\_model/include/GoTools/isogeometric\_model/VolSolution.h File Reference

```
#include "GoTools/utils/Point.h"
#include "GoTools/trivariate/SplineVolume.h"
#include "GoTools/isogeometric_model/VolBoundaryCondition.h"
#include "GoTools/isogeometric_model/BlockSolution.h"
#include "GoTools/isogeometric_model/VolPointBdCond.h"
#include "GoTools/isogeometric_model/BdConditionType.h"
#include "GoTools/isogeometric_model/BdCondFunctor.h"
#include <vector>
#include <memory>
```

Include dependency graph for VolSolution.h:



This graph shows which files directly or indirectly include this file:



## Classes

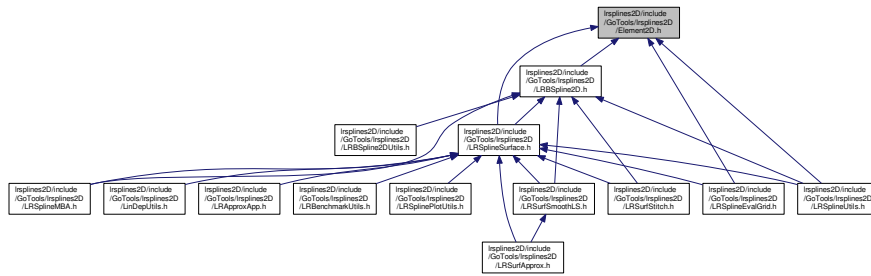
- struct [Go::preEvaluationVol](#)
- class [Go::VolSolution](#)

## Namespaces

- [Go](#)



This graph shows which files directly or indirectly include this file:



## Classes

- struct [Go::LSSmoothData](#)
- class [Go::Element2D](#)

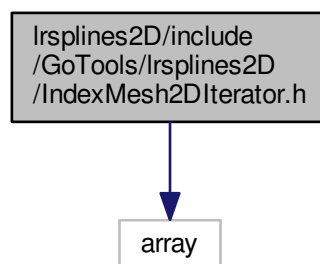
## Namespaces

- [Go](#)

## 30.2116 Irspines2D/include/GoTools/Irspines2D/IndexMesh2DIterator.h File Reference

```
#include <array>
```

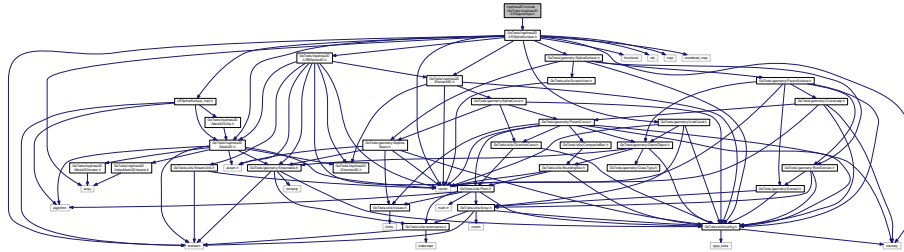
Include dependency graph for IndexMesh2DIterator.h:





## 30.2118 Irsplines2D/include/GoTools/Irsplines2D/LRAproxApp.h File Reference

```
#include "GoTools/Irsplines2D/LRSplineSurface.h"
Include dependency graph for LRAproxApp.h:
```



### Namespaces

- [Go](#)
- [Go::LRAproxApp](#)

### Functions

- void [Go::LRAproxApp::pointCloud2Spline](#) (std::vector< double > &points, int dim, double domain[], double reduced\_domain[], double eps, int max\_iter, shared\_ptr< LRSplineSurface > &surf, double &maxdist, double &avdist, double &avdist\_out, int &nmb\_out)
- void [Go::LRAproxApp::computeDistPointSpline](#) (std::vector< double > &points, shared\_ptr< LRSplineSurface > &surf, double &max\_above, double &max\_below, double &avdist, int &nmb\_points, std::vector< double > &pointsdist)
- void [Go::LRAproxApp::computeDistPointSpline\\_omp](#) (std::vector< double > &points, shared\_ptr< LRSplineSurface > &surf, double &max\_above, double &max\_below, double &avdist, int &nmb\_points, std::vector< double > &pointsdist)
- void [Go::LRAproxApp::classifyCloudFromDist](#) (std::vector< double > &points, shared\_ptr< LRSplineSurface > &surf, std::vector< double > &limits, double &max\_above, double &max\_below, double &avdist, int &nmb\_points, std::vector< std::vector< double > > &level\_points, std::vector< int > &nmb\_group)
- void [Go::LRAproxApp::classifyCloudFromDist\\_omp](#) (std::vector< double > &points, shared\_ptr< LRSplineSurface > &surf, std::vector< double > &limits, double &max\_above, double &max\_below, double &avdist, int &nmb\_points, std::vector< std::vector< double > > &level\_points, std::vector< int > &nmb\_group)
- void [Go::LRAproxApp::categorizeCloudFromDist](#) (std::vector< double > &points, shared\_ptr< LRSplineSurface > &surf, std::vector< double > &limits, double &max\_above, double &max\_below, double &avdist, int &nmb\_points, std::vector< int > &classification, std::vector< int > &nmb\_group)
- void [Go::LRAproxApp::categorizeCloudFromDist\\_omp](#) (std::vector< double > &points, shared\_ptr< LRSplineSurface > &surf, std::vector< double > &limits, double &max\_above, double &max\_below, double &avdist, int &nmb\_points, std::vector< int > &classification, std::vector< int > &nmb\_group)







## Namespaces

- [Go](#)
- [Go::LRBSpline2DUtils](#)

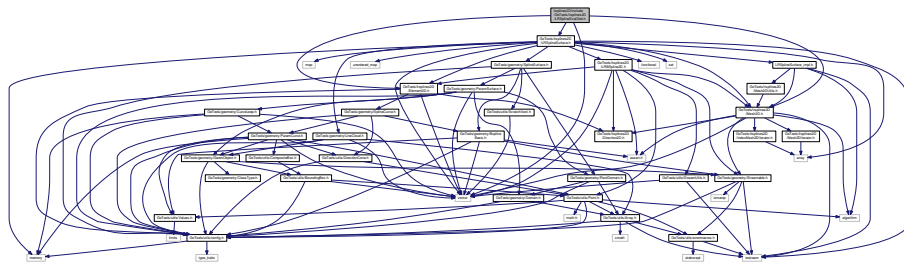
## Functions

- `std::vector< int >` [Go::LRBSpline2DUtils::derive\\_knots](#) (`const Mesh2D &m`, `Direction2D d`, `int beg`, `int end`, `int orto_min`, `int orto_max`)
- `void` [Go::LRBSpline2DUtils::split\\_several](#) (`const double *knotvals`, `const std::vector< int > &k_vec_in`, `const std::vector< int > &new_knots`, `std::vector< int > &k_vec_out`, `std::vector< double > &b_spline_weights`)
- `void` [Go::LRBSpline2DUtils::split\\_function](#) (`const LRBSpline2D &orig`, `const Mesh2D &mesh`, `Direction2D d`, `const double *const knotvalues`, `int new_knot_ix`, `LRBSpline2D *&new_1`, `LRBSpline2D *&new_2`)
- `bool` [Go::LRBSpline2DUtils::try\\_split\\_once](#) (`const LRBSpline2D &b`, `const Mesh2D &mesh`, `LRBSpline2D *&b1`, `LRBSpline2D *&b2`)

## 30.2122 Irsplines2D/include/GoTools/Irsplines2D/LRSplineEvalGrid.h File Reference

```
#include "GoTools/lrsplines2D/LRSplineSurface.h"
#include "GoTools/lrsplines2D/Element2D.h"
#include "GoTools/lrsplines2D/Mesh2D.h"
#include <vector>
```

Include dependency graph for LRSplineEvalGrid.h:



## Classes

- class [Go::LRBSplineEvalGrid](#)

## Namespaces

- [Go](#)

## Macros

- `#define` [\\_LRSPLINEEVAVGRID\\_H](#)

## Typedefs

- typedef `Element2D` [Go::simpleElement](#)

### 30.2122.1 Macro Definition Documentation

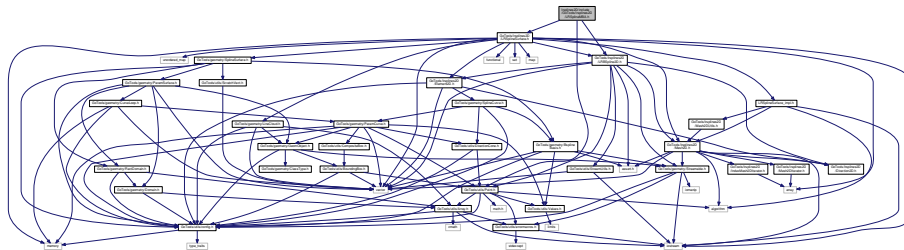
#### 30.2122.1.1 #define \_LRSPLINEEVAVGRID\_H

Definition at line 41 of file LRSplineEvalGrid.h.

### 30.2123 Irsplines2D/include/GoTools/Irsplines2D/LRSplineMBA.h File Reference

```
#include "GoTools/lrsplines2D/LRSplineSurface.h"
#include "GoTools/lrsplines2D/LRBSpline2D.h"
#include "GoTools/utils/Point.h"
```

Include dependency graph for LRSplineMBA.h:



### Namespaces

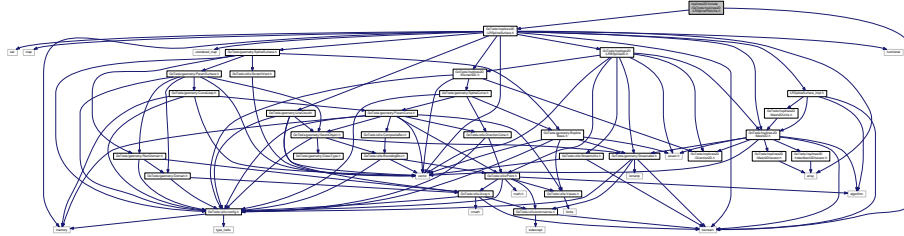
- [Go](#)
- [Go::LRSplineMBA](#)

### Functions

- void [Go::LRSplineMBA::MBADistAndUpdate](#) (LRSplineSurface \*srf)
- void [Go::LRSplineMBA::MBADistAndUpdate\\_omp](#) (LRSplineSurface \*srf)
- void [Go::LRSplineMBA::MBAUpdate](#) (LRSplineSurface \*srf)
- void [Go::LRSplineMBA::MBAUpdate\\_omp](#) (LRSplineSurface \*srf)
- void [Go::LRSplineMBA::MBAUpdate](#) (LRSplineSurface \*srf, std::vector< Element2D \* > &elems, std::vector< Element2D \* > &elems2)
- void [Go::LRSplineMBA::add\\_contribution](#) (int dim, std::map< const LRBSpline2D \*, Array< double, 2 > > &target, const LRBSpline2D \*bspline, double nom[], double denom)
- void [Go::LRSplineMBA::add\\_contribution2](#) (int dim, std::map< const LRBSpline2D \*, Array< double, 4 > > &target, const LRBSpline2D \*bspline, double nom[], double denom)

## 30.2124 Irsplines2D/include/GoTools/Irsplines2D/LRSplinePlotUtils.h File Reference

```
#include "GoTools/lrsplines2D/LRSplineSurface.h"
#include <iostream>
Include dependency graph for LRSplinePlotUtils.h:
```



### Namespaces

- [Go](#)

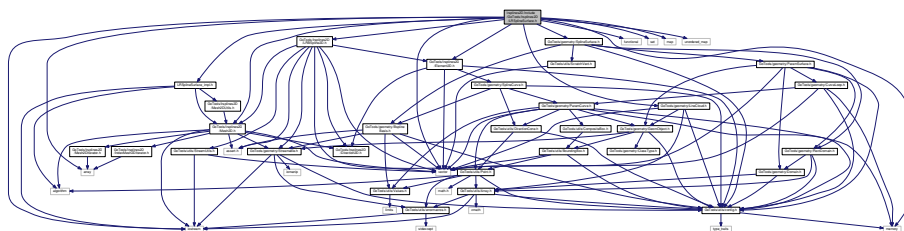
### Functions

- void [Go::writePostscriptMesh](#) ([Go::LRSplineSurface](#) &lr\_spline\_sf, std::ostream &out, [const bool](#) close=true)

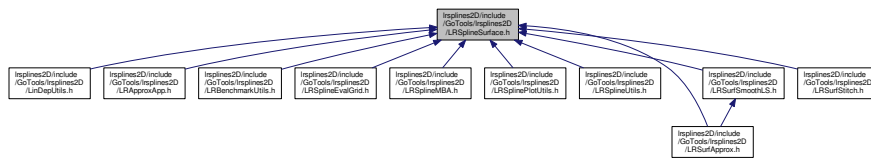
## 30.2125 Irsplines2D/include/GoTools/Irsplines2D/Irsplines2d-doxymain.h File Reference

## 30.2126 Irsplines2D/include/GoTools/Irsplines2D/LRSplineSurface.h File Reference

```
#include <array>
#include <functional>
#include <set>
#include <map>
#include <vector>
#include <unordered_map>
#include <iostream>
#include <memory>
#include "GoTools/geometry/SplineSurface.h"
#include "GoTools/geometry/LineCloud.h"
#include "GoTools/lrsplines2D/Mesh2D.h"
#include "GoTools/lrsplines2D/LRBSpline2D.h"
#include "GoTools/lrsplines2D/Element2D.h"
#include "LRSplineSurface_impl.h"
Include dependency graph for LRSplineSurface.h:
```



This graph shows which files directly or indirectly include this file:



## Classes

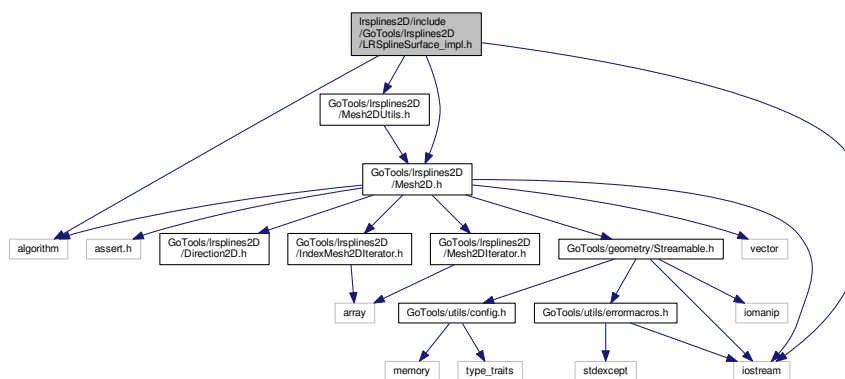
- class [Go::LRSplineSurface](#)
- struct [Go::LRSplineSurface::Refinement2D](#)
- struct [Go::LRSplineSurface::BSKey](#)
- struct [Go::LRSplineSurface::double\\_pair\\_hash](#)
- struct [Go::LRSplineSurface::ElemKey](#)

## Namespaces

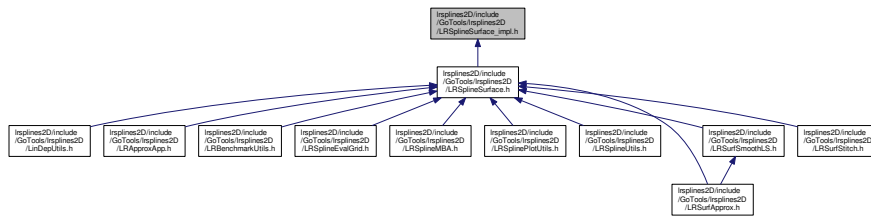
- [Go](#)

## 30.2127 Irsplines2D/include/GoTools/Irsplines2D/LRSplineSurface\_impl.h File Reference

```
#include <algorithm>
#include <iostream>
#include "GoTools/lrsplines2D/Mesh2DUtils.h"
#include "GoTools/lrsplines2D/Mesh2D.h"
Include dependency graph for LRSplineSurface_impl.h:
```



This graph shows which files directly or indirectly include this file:



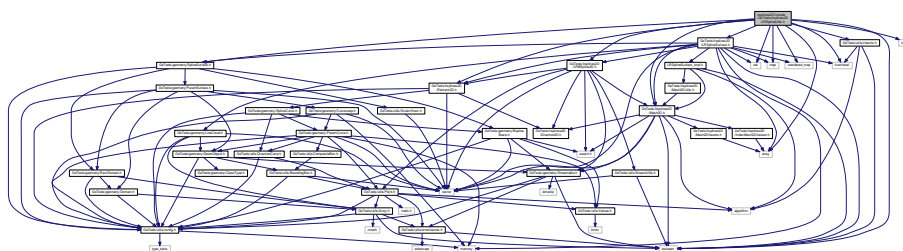
## Namespaces

- [Go](#)

## 30.2128 Irsplines2D/include/GoTools/Irsplines2D/LRSplineUtils.h File Reference

```
#include <array>
#include <functional>
#include <set>
#include <map>
#include <unordered_map>
#include <tuple>
#include <iostream>
#include "GoTools/utils/checks.h"
#include "GoTools/geometry/SplineSurface.h"
#include "GoTools/lrsplines2D/LRSplineSurface.h"
#include "GoTools/lrsplines2D/Mesh2D.h"
#include "GoTools/lrsplines2D/LRSpline2D.h"
#include "GoTools/lrsplines2D/Element2D.h"
```

Include dependency graph for LRSplineUtils.h:



## Classes

- struct [Go::LRSplineUtils::support\\_compare](#)

## Namespaces

- [Go](#)
- [Go::LRSplineUtils](#)

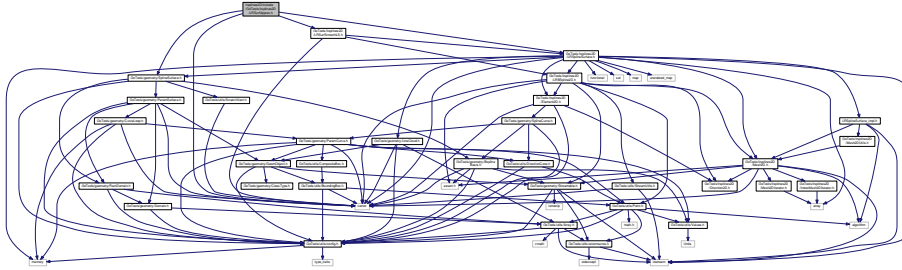
## Functions

- LRSplineSurface::ElementMap [Go::LRSplineUtils::identify\\_elements\\_from\\_mesh](#) (const Mesh2D &m)
- void [Go::LRSplineUtils::update\\_elements\\_with\\_single\\_bspline](#) (LRBSpline2D \*b, LRSplineSurface::↔ ElementMap &emap, const Mesh2D &mesh, bool remove)
- int [Go::LRSplineUtils::locate\\_interval](#) (const Mesh2D &m, Direction2D d, double value, double other\_value, bool at\_end)
- void [Go::LRSplineUtils::increment\\_knotvec\\_indices](#) (LRSplineSurface::BSplineMap &bmap, Direction2D d, int from\_ix)
- LRBSpline2D \* [Go::LRSplineUtils::insert\\_basis\\_function](#) (std::unique\_ptr< LRBSpline2D > &b, const Mesh2D &mesh, LRSplineSurface::BSplineMap &bmap)
- std::vector< int > [Go::LRSplineUtils::set\\_uniform\\_meshlines](#) (Direction2D d, Mesh2D &mesh)
- bool [Go::LRSplineUtils::all\\_meshlines\\_uniform](#) (Direction2D d, const Mesh2D &m)
- double [Go::LRSplineUtils::compute\\_greville](#) (const std::vector< int > &v\_ixs, const double \*const vals)
- std::vector< double > [Go::LRSplineUtils::compute\\_greville](#) (int deg, const std::vector< int > &k\_vec\_in, const double \*const knotvals)
- std::vector< int > [Go::LRSplineUtils::knots\\_to\\_insert](#) (const std::vector< int > &ref, const std::vector< int > &mults)
- double [Go::LRSplineUtils::compute\\_alpha](#) (int degree, const int \*const oldvec\_ix, const int \*const newvec\_ix, const double \*const kval)
- std::tuple< std::vector< double >, std::vector< int > > [Go::LRSplineUtils::insert\\_knots](#) (const std::vector< int > &new\_knots, std::unique\_ptr< LRBSpline2D > &bfun, const Direction2D d, const double \*const kval)
- void [Go::LRSplineUtils::tensor\\_split](#) (std::unique\_ptr< LRBSpline2D > &bfun, const std::vector< int > &x↔ \_mults, const std::vector< int > &y\_mults, const Mesh2D &tensor\_mesh, LRSplineSurface::BSplineMap &bmap)
- void [Go::LRSplineUtils::iteratively\\_split](#) (std::vector< std::unique\_ptr< LRBSpline2D > > &bfun, const Mesh2D &mesh)
- void [Go::LRSplineUtils::iteratively\\_split2](#) (std::vector< LRBSpline2D \* > &bsplines, const Mesh2D &mesh, LRSplineSurface::BSplineMap &bmap, double domain[])
- std::tuple< int, int, int, int > [Go::LRSplineUtils::refine\\_mesh](#) (Direction2D d, double fixed\_val, double start, double end, int mult, bool absolute, int spline\_degree, double knot\_tol, Mesh2D &mesh, LRSplineSurface::↔ BSplineMap &bmap)
- bool [Go::LRSplineUtils::support\\_equal](#) (const LRBSpline2D \*b1, const LRBSpline2D \*b2)
- bool [Go::LRSplineUtils::elementOK](#) (const Element2D \*elem, const Mesh2D &m)
- void [Go::LRSplineUtils::insertParameterFunctions](#) (LRSplineSurface \*lr\_spline\_sf)
- SplineSurface \* [Go::LRSplineUtils::fullTensorProductSurface](#) (const LRSplineSurface &lr\_spline\_sf)
- LRBSpline2D \* [Go::LRSplineUtils::mostComparableBspline](#) (LRSplineSurface \*lr\_spline\_sf, Point pos)
- std::vector< std::vector< double > > [Go::LRSplineUtils::elementLineClouds](#) (const LRSplineSurface &lr\_↔ spline\_sf)
- void [Go::LRSplineUtils::distributeDataPoints](#) (LRSplineSurface \*srf, std::vector< double > &points, bool add\_distance\_field=false, bool primary\_points=true)

### 30.2129 Irsplines2D/include/GoTools/Irsplines2D/LRSurfApprox.h File Reference

```
#include "GoTools/geometry/SplineSurface.h"
#include "GoTools/lrsplines2D/LRSplineSurface.h"
#include "GoTools/lrsplines2D/LRSurfSmoothLS.h"
#include <vector>
```

Include dependency graph for LRSurfApprox.h:



## Classes

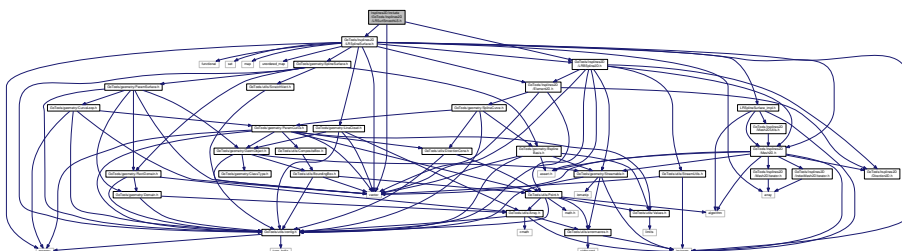
- class [Go::LRSurfApprox](#)

## Namespaces

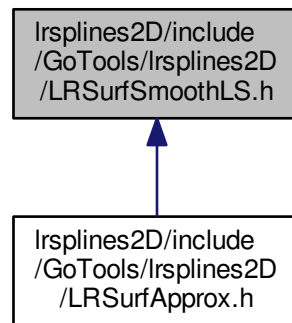
- [Go](#)

## 30.2130 Irsplines2D/include/GoTools/Irsplines2D/LRSurfSmoothLS.h File Reference

```
#include <vector>
#include "GoTools/Irsplines2D/LRSplineSurface.h"
#include "GoTools/Irsplines2D/LRSpline2D.h"
Include dependency graph for LRSurfSmoothLS.h:
```



This graph shows which files directly or indirectly include this file:



## Classes

- class [Go::LRSurfSmoothLS](#)

## Namespaces

- [Go](#)

## Typedefs

- typedef `std::map< LRBSpline2D *, size_t >` [Go::BsplineIndexMap](#)

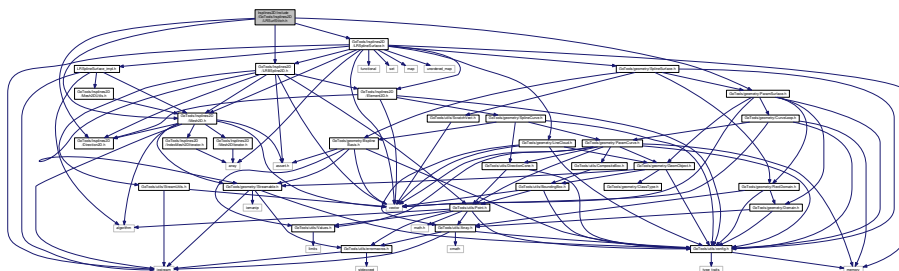
## 30.2131 Irsplines2D/include/GoTools/Irsplines2D/LRSurfStitch.h File Reference

```

#include "GoTools/geometry/ParamSurface.h"
#include "GoTools/lrsplines2D/LRSplineSurface.h"
#include "GoTools/lrsplines2D/Direction2D.h"
#include "GoTools/lrsplines2D/Mesh2D.h"
#include "GoTools/lrsplines2D/LRBSpline2D.h"

```

Include dependency graph for LRSurfStitch.h:





## Namespaces

- [Go](#)
- [Go::LRSurfStitch](#)

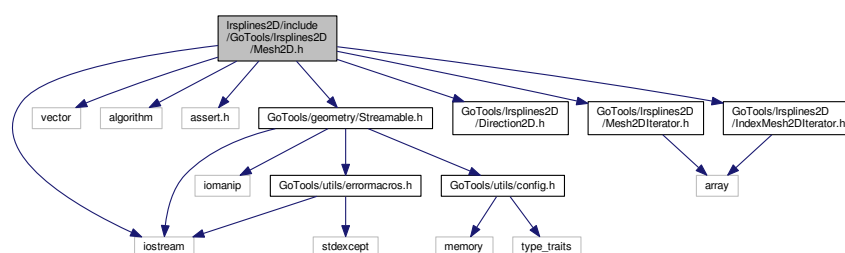
## Functions

- `int` [Go::LRSurfStitch::averageCorner](#) (`std::vector< std::pair< shared_ptr< ParamSurface >, int > >` &sfs, `double` tol)
- `bool` [Go::LRSurfStitch::averageEdge](#) (`shared_ptr< ParamSurface >` surf1, `int` edge1, `shared_ptr< ParamSurface >` surf2, `int` edge2, `double` tol)
- `bool` [Go::LRSurfStitch::averageEdge](#) (`shared_ptr< LRSplineSurface >` surf1, `int` edge1, `shared_ptr< LRSplineSurface >` surf2, `int` edge2, `double` tol)
- `void` [Go::LRSurfStitch::fetchEdgeCorners](#) (`shared_ptr< LRSplineSurface >` surf, `int` edge, `double` &u1, `double` &v1, `double` &u2, `double` &v2)
- `void` [Go::LRSurfStitch::extractMissingKnots](#) (`std::vector< double >` &union\_vec, `std::vector< double >` &vec, `double` tol, `int` order, `std::vector< double >` &resvec)
- `void` [Go::LRSurfStitch::defineRefinements](#) (`const` Mesh2D &mesh, `Direction2D` dir, `int` edge, `int` ix, `std::vector< double >` &knot\_vals, `int` element\_width, `std::vector< LRSplineSurface::Refinement2D >` &refs)
- `void` [Go::LRSurfStitch::extractBoundaryBsplines](#) (`shared_ptr< LRSplineSurface >` surf, `int` edge, `std::vector< LRBSpline2D * >` &bsplines)

## 30.2132 Irsplines2D/include/GoTools/Irsplines2D/Mesh2D.h File Reference

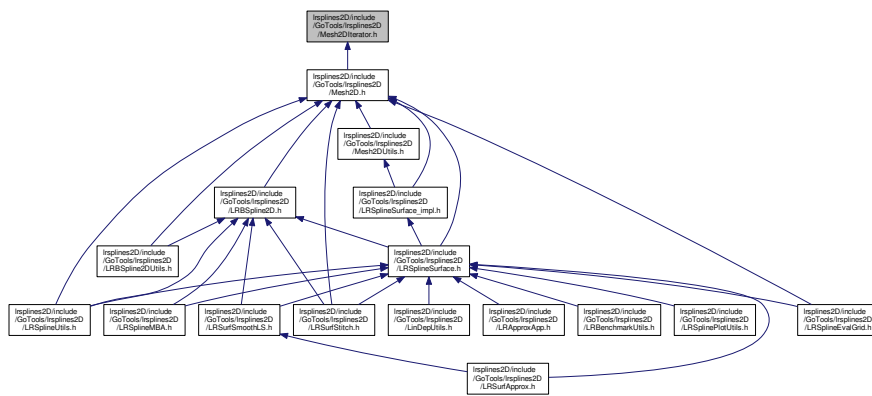
```
#include <iostream>
#include <vector>
#include <algorithm>
#include <assert.h>
#include "GoTools/geometry/Streamable.h"
#include "GoTools/lrsplines2D/Direction2D.h"
#include "GoTools/lrsplines2D/Mesh2DIterator.h"
#include "GoTools/lrsplines2D/IndexMesh2DIterator.h"
```

Include dependency graph for Mesh2D.h:





This graph shows which files directly or indirectly include this file:



Classes

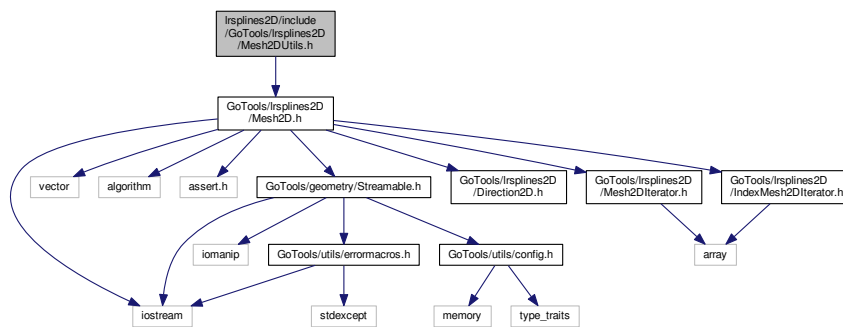
- class [Go::Mesh2DIterator](#)

Namespaces

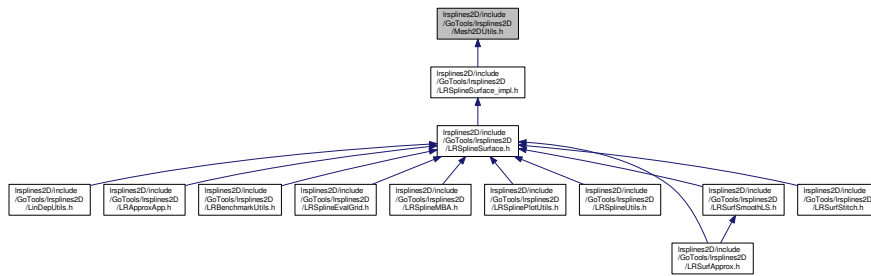
- [Go](#)

30.2134 Irsplines2D/include/GoTools/Irsplines2D/Mesh2DUtils.h File Reference

```
#include "GoTools/Irsplines2D/Mesh2D.h"
Include dependency graph for Mesh2DUtils.h:
```



This graph shows which files directly or indirectly include this file:



## Namespaces

- [Go](#)
- [Go::Mesh2DUtills](#)

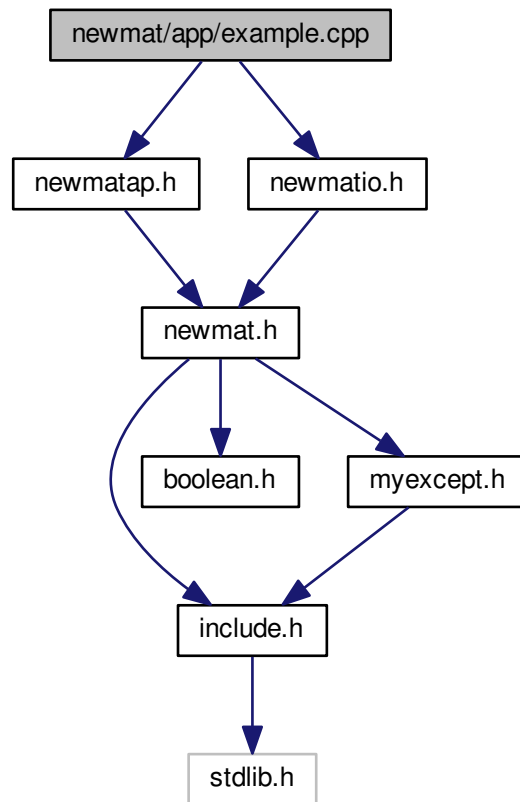
## Functions

- `int Go::Mesh2DUtills::last_nonlarger_knotvalue_ix (const Mesh2D &m, Direction2D d, double par)`
- `int Go::Mesh2DUtills::first_larger_knotvalue_ix (const Mesh2D &m, Direction2D d, double par)`
- `bool Go::Mesh2DUtills::identify_patch_lower_left (const Mesh2D &m, double u, double v, int &x_ix, int &y_ix)`
- `bool Go::Mesh2DUtills::identify_patch_upper_right (const Mesh2D &m, double u, double v, int &x_ix, int &y_ix)`
- `int Go::Mesh2DUtills::search_downwards_for_nonzero_multiplicity (const Mesh2D &m, Direction2D d, int start_ix, int other_ix)`
- `int Go::Mesh2DUtills::search_upwards_for_nonzero_multiplicity (const Mesh2D &m, Direction2D d, int start_ix, int other_ix)`
- `int Go::Mesh2DUtills::search_downwards_for_nonzero_multiplicity (const Mesh2D &m, Direction2D d, int start_ix, int ix1, int ix2)`
- `int Go::Mesh2DUtills::search_upwards_for_nonzero_multiplicity (const Mesh2D &m, Direction2D d, int start_ix, int ix1, int ix2)`

## 30.2135 newmat/app/example.cpp File Reference

```
#include "newmatap.h"
#include "newmatio.h"
```

Include dependency graph for example.cpp:



## Macros

- `#define WANT_STREAM`
- `#define WANT_MATH`

## Functions

- void `test1` (`Real *y`, `Real *x1`, `Real *x2`, `int nobs`, `int npred`)
- void `test2` (`Real *y`, `Real *x1`, `Real *x2`, `int nobs`, `int npred`)
- void `test3` (`Real *y`, `Real *x1`, `Real *x2`, `int nobs`, `int npred`)
- void `test4` (`Real *y`, `Real *x1`, `Real *x2`, `int nobs`, `int npred`)
- void `test5` (`Real *y`, `Real *x1`, `Real *x2`, `int nobs`, `int npred`)
- int `main` ()

### 30.2135.1 Macro Definition Documentation

#### 30.2135.1.1 `#define WANT_MATH`

Definition at line 4 of file `example.cpp`.

30.2135.1.2 `#define WANT_STREAM`

Definition at line 3 of file example.cpp.

## 30.2135.2 Function Documentation

30.2135.2.1 `int main ( )`

Definition at line 288 of file example.cpp.

30.2135.2.2 `void test1 ( Real * y, Real * x1, Real * x2, int nobs, int npred )`

Definition at line 19 of file example.cpp.

30.2135.2.3 `void test2 ( Real * y, Real * x1, Real * x2, int nobs, int npred )`

Definition at line 78 of file example.cpp.

30.2135.2.4 `void test3 ( Real * y, Real * x1, Real * x2, int nobs, int npred )`

Definition at line 157 of file example.cpp.

30.2135.2.5 `void test4 ( Real * y, Real * x1, Real * x2, int nobs, int npred )`

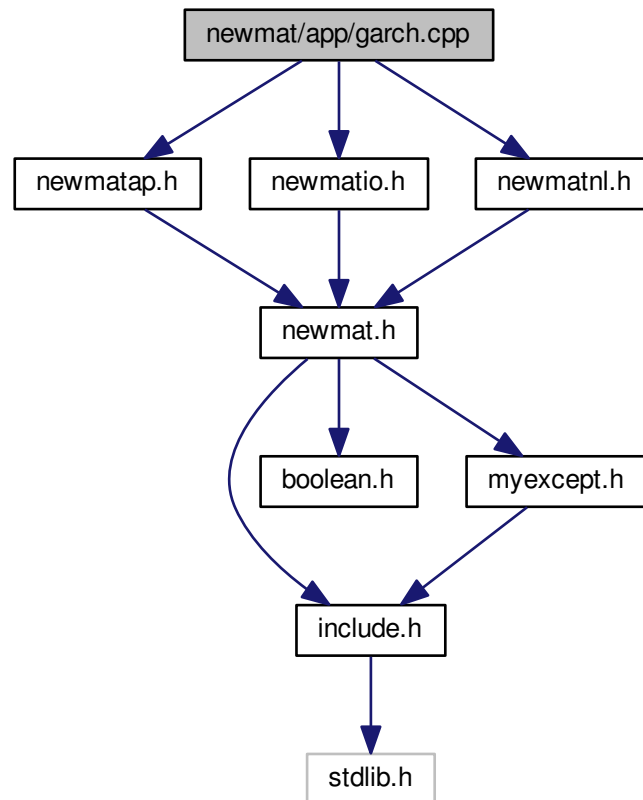
Definition at line 213 of file example.cpp.

30.2135.2.6 `void test5 ( Real * y, Real * x1, Real * x2, int nobs, int npred )`

Definition at line 251 of file example.cpp.

## 30.2136 newmat/app/garch.cpp File Reference

```
#include "newmatap.h"
#include "newmatio.h"
#include "newmatnl.h"
Include dependency graph for garch.cpp:
```



### Classes

- class [GARCH11\\_LL](#)

### Macros

- #define [WANT\\_STREAM](#)
- #define [WANT\\_MATH](#)
- #define [WANT\\_FSTREAM](#)

### Functions

- [Real square](#) (Real x)
- int [main](#) ()

### 30.2136.1 Macro Definition Documentation

#### 30.2136.1.1 `#define WANT_FSTREAM`

Definition at line 4 of file garch.cpp.

#### 30.2136.1.2 `#define WANT_MATH`

Definition at line 3 of file garch.cpp.

#### 30.2136.1.3 `#define WANT_STREAM`

Definition at line 2 of file garch.cpp.

### 30.2136.2 Function Documentation

#### 30.2136.2.1 `int main ( )`

Definition at line 141 of file garch.cpp.

#### 30.2136.2.2 `Real square ( Real x ) [inline]`

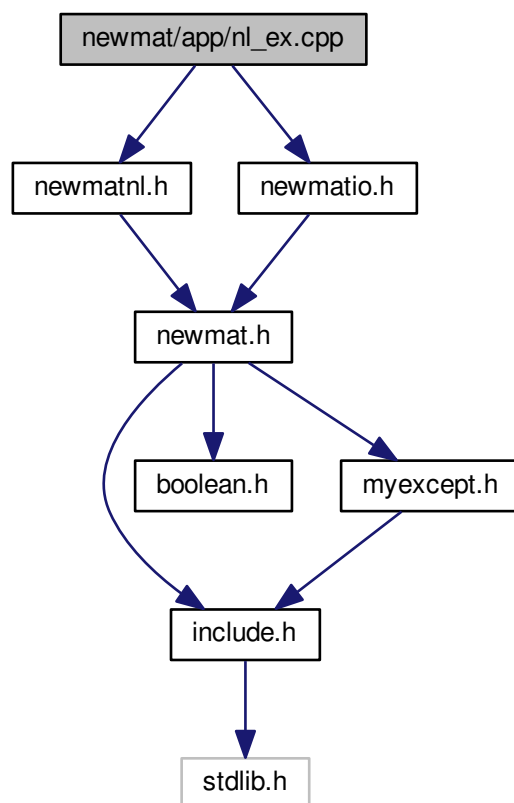
Definition at line 27 of file garch.cpp.

### 30.2137 newmat/app/nl\_ex.cpp File Reference

```
#include "newmatnl.h"
#include "newmatio.h"
```



Include dependency graph for nl\_ex.cpp:



## Classes

- class [Model\\_3pe](#)

## Macros

- `#define` [WANT\\_STREAM](#)
- `#define` [WANT\\_MATH](#)

## Functions

- int [main](#) ()

### 30.2137.1 Macro Definition Documentation

#### 30.2137.1.1 `#define` WANT\_MATH

Definition at line 9 of file `nl_ex.cpp`.

30.2137.1.2 `#define WANT_STREAM`

Definition at line 8 of file `nl_ex.cpp`.

## 30.2137.2 Function Documentation

30.2137.2.1 `int main ( )`

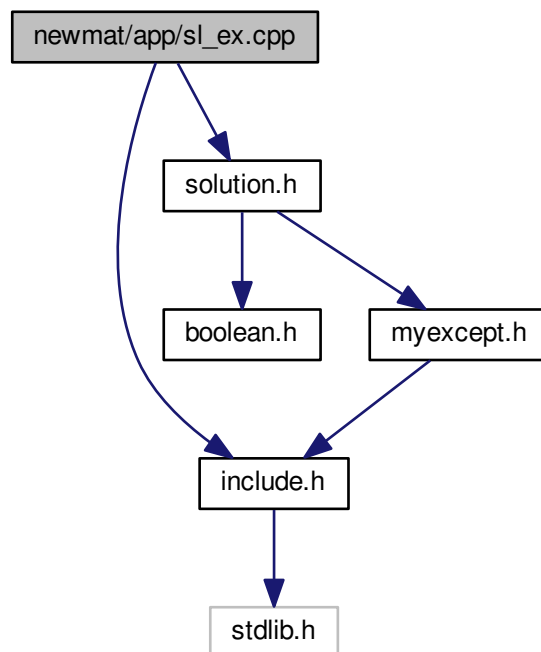
Definition at line 45 of file `nl_ex.cpp`.

## 30.2138 `newmat/app/sl_ex.cpp` File Reference

```
#include "include.h"
```

```
#include "solution.h"
```

Include dependency graph for `sl_ex.cpp`:



## Classes

- class [Cube](#)

## Macros

- #define [WANT\\_STREAM](#)
- #define [WANT\\_MATH](#)

## Functions

- int [main](#) ()

### 30.2138.1 Macro Definition Documentation

#### 30.2138.1.1 #define WANT\_MATH

Definition at line 7 of file sl\_ex.cpp.

#### 30.2138.1.2 #define WANT\_STREAM

Definition at line 6 of file sl\_ex.cpp.

### 30.2138.2 Function Documentation

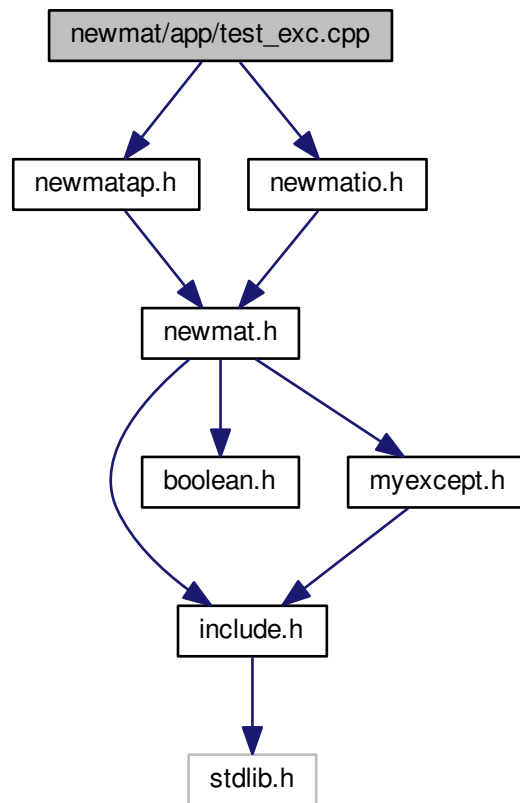
#### 30.2138.2.1 int main ( )

Definition at line 22 of file sl\_ex.cpp.

## 30.2139 newmat/app/test\_exc.cpp File Reference

```
#include "newmatap.h"
#include "newmatio.h"
```

Include dependency graph for test\_exc.cpp:



## Macros

- `#define WANT_STREAM`

## Functions

- `int main ()`

### 30.2139.1 Macro Definition Documentation

#### 30.2139.1.1 `#define WANT_STREAM`

Definition at line 1 of file `test_exc.cpp`.

## 30.2139.2 Function Documentation

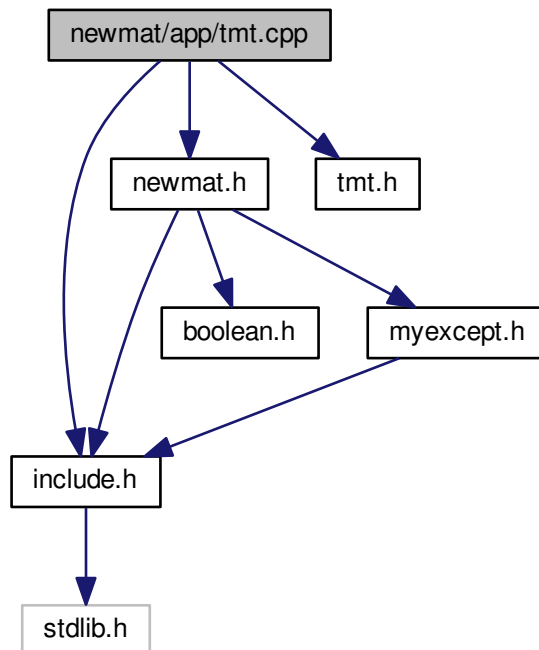
### 30.2139.2.1 int main ( )

Definition at line 15 of file test\_exc.cpp.

## 30.2140 newmat/app/tmt.cpp File Reference

```
#include "include.h"
#include "newmat.h"
#include "tmt.h"
```

Include dependency graph for tmt.cpp:



## Classes

- class [PrintCounter](#)

## Macros

- `#define` [WANT\\_STREAM](#)

## Functions

- [PrintCounter PCZ](#) ("Number of non-zero matrices (should be 1) = ")
- [PrintCounter PCN](#) ("Number of matrices tested = ")
- void [Print](#) (const [Matrix](#) &X)
- void [Print](#) (const [UpperTriangularMatrix](#) &X)
- void [Print](#) (const [DiagonalMatrix](#) &X)
- void [Print](#) (const [SymmetricMatrix](#) &X)
- void [Print](#) (const [LowerTriangularMatrix](#) &X)
- void [Clean](#) ([Matrix](#) &A, Real c)
- void [Clean](#) ([DiagonalMatrix](#) &A, Real c)
- void [PentiumCheck](#) (Real N, Real D)
- void [TestTypeAdd](#) ()
- void [TestTypeMult](#) ()
- void [TestTypeConcat](#) ()
- void [TestTypeSP](#) ()
- void [TestTypeKP](#) ()
- void [TestTypeOrder](#) ()
- int [main](#) ()

### 30.2140.1 Macro Definition Documentation

#### 30.2140.1.1 #define WANT\_STREAM

Definition at line 1 of file tmt.cpp.

### 30.2140.2 Function Documentation

#### 30.2140.2.1 void Clean ( [Matrix](#) & A, Real c )

Definition at line 121 of file tmt.cpp.

#### 30.2140.2.2 void Clean ( [DiagonalMatrix](#) & A, Real c )

Definition at line 131 of file tmt.cpp.

#### 30.2140.2.3 int main ( )

Definition at line 162 of file tmt.cpp.

#### 30.2140.2.4 [PrintCounter](#) PCN ( )

#### 30.2140.2.5 [PrintCounter](#) PCZ ( )

#### 30.2140.2.6 void [PentiumCheck](#) ( Real N, Real D )

Definition at line 138 of file tmt.cpp.

30.2140.2.7 void Print ( const Matrix & X )

Definition at line 35 of file tmt.cpp.

30.2140.2.8 void Print ( const UpperTriangularMatrix & X )

Definition at line 51 of file tmt.cpp.

30.2140.2.9 void Print ( const DiagonalMatrix & X )

Definition at line 69 of file tmt.cpp.

30.2140.2.10 void Print ( const SymmetricMatrix & X )

Definition at line 86 of file tmt.cpp.

30.2140.2.11 void Print ( const LowerTriangularMatrix & X )

Definition at line 104 of file tmt.cpp.

30.2140.2.12 void TestTypeAdd ( )

Definition at line 258 of file tmt.cpp.

30.2140.2.13 void TestTypeConcat ( )

Definition at line 315 of file tmt.cpp.

30.2140.2.14 void TestTypeKP ( )

Definition at line 373 of file tmt.cpp.

30.2140.2.15 void TestTypeMult ( )

Definition at line 286 of file tmt.cpp.

30.2140.2.16 void TestTypeOrder ( )

Definition at line 402 of file tmt.cpp.

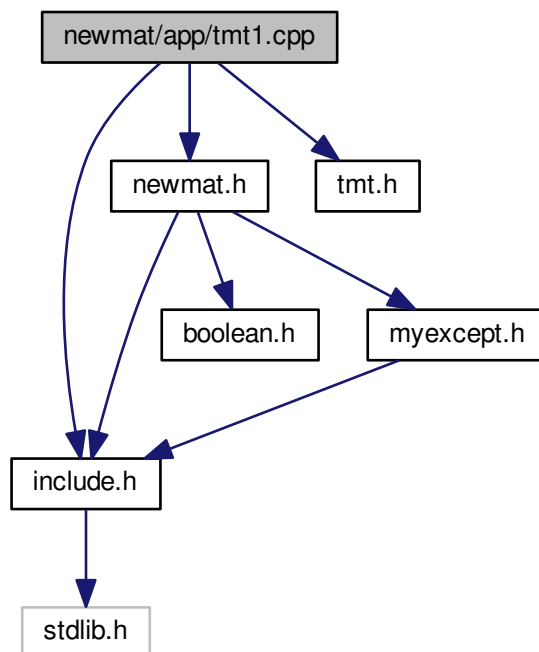
30.2140.2.17 void TestTypeSP ( )

Definition at line 344 of file tmt.cpp.

## 30.2141 newmat/app/tmt1.cpp File Reference

```
#include "include.h"
#include "newmat.h"
#include "tmt.h"
```

Include dependency graph for tmt1.cpp:



### Macros

- #define [WANT\\_STREAM](#)

### Functions

- void [trymat1](#) ()

## 30.2141.1 Macro Definition Documentation

### 30.2141.1.1 #define WANT\_STREAM

Definition at line 2 of file tmt1.cpp.



### 30.2141.2 Function Documentation

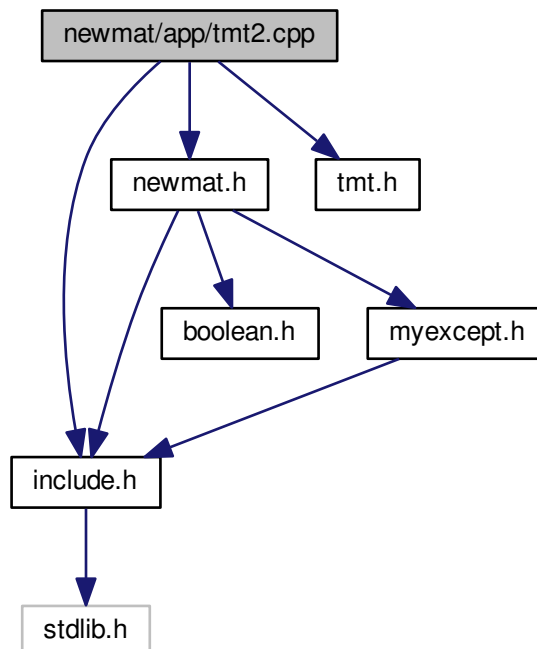
#### 30.2141.2.1 void trymat1 ( )

Definition at line 20 of file tmt1.cpp.

### 30.2142 newmat/app/tmt2.cpp File Reference

```
#include "include.h"
#include "newmat.h"
#include "tmt.h"
```

Include dependency graph for tmt2.cpp:



### Functions

- void [trymat2](#) ( )

### 30.2142.1 Function Documentation

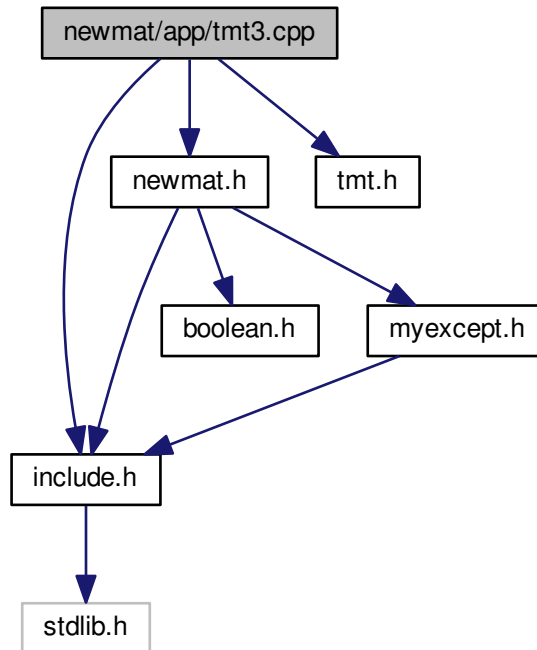
#### 30.2142.1.1 void trymat2 ( )

Definition at line 19 of file tmt2.cpp.

## 30.2143 newmat/app/tmt3.cpp File Reference

```
#include "include.h"
#include "newmat.h"
#include "tmt.h"
```

Include dependency graph for tmt3.cpp:



### Functions

- void [trymat3](#) ( )

### 30.2143.1 Function Documentation

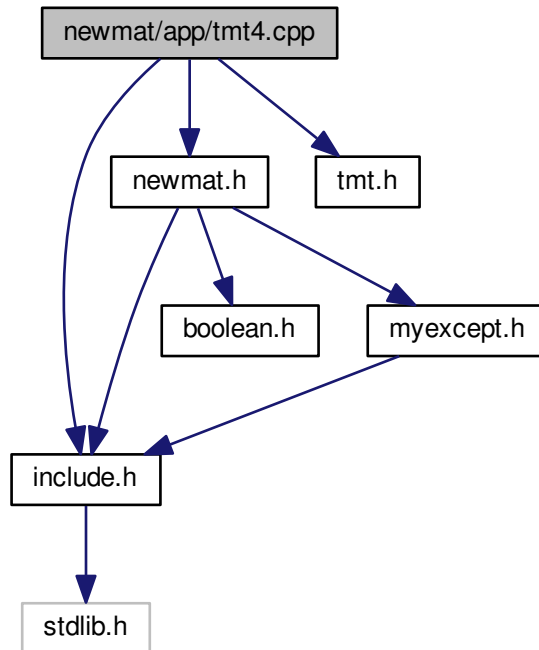
#### 30.2143.1.1 void [trymat3](#) ( )

Definition at line 17 of file `tmt3.cpp`.

## 30.2144 newmat/app/tmt4.cpp File Reference

```
#include "include.h"
#include "newmat.h"
#include "tmt.h"
```

Include dependency graph for tmt4.cpp:



### Functions

- void [trymat4](#) ()

#### 30.2144.1 Function Documentation

##### 30.2144.1.1 void [trymat4](#) ( )

Definition at line 19 of file `tmt4.cpp`.

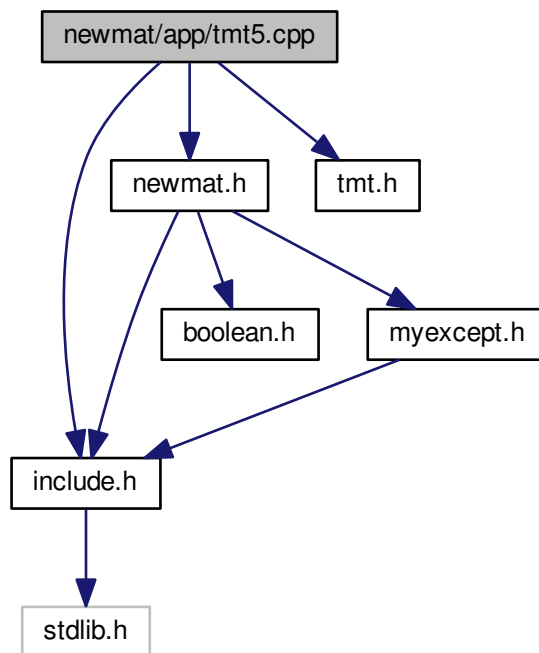
## 30.2145 newmat/app/tmt5.cpp File Reference

```
#include "include.h"
```

```
#include "newmat.h"
```

```
#include "tmt.h"
```

Include dependency graph for tmt5.cpp:



### Functions

- [ReturnMatrix Returner0](#) (`const GenericMatrix &GM`)
- [ReturnMatrix Returner1](#) (`const GenericMatrix &GM`)
- [ReturnMatrix Returner2](#) (`const GenericMatrix &GM`)
- [ReturnMatrix Returner3](#) (`const GenericMatrix &GM`)
- [ReturnMatrix Returner4](#) (`const GenericMatrix &GM`)
- [ReturnMatrix Returner5](#) (`const GenericMatrix &GM`)
- [ReturnMatrix Returner6](#) (`const GenericMatrix &GM`)
- [ReturnMatrix Returner7](#) (`const GenericMatrix &GM`)
- `void trymat5 ()`

### 30.2145.1 Function Documentation

#### 30.2145.1.1 [ReturnMatrix Returner0](#) (`const GenericMatrix & GM`)

Definition at line 19 of file `tmt5.cpp`.

**30.2145.1.2 ReturnMatrix Returner1 ( const GenericMatrix & GM )**

Definition at line 22 of file tmt5.cpp.

**30.2145.1.3 ReturnMatrix Returner2 ( const GenericMatrix & GM )**

Definition at line 25 of file tmt5.cpp.

**30.2145.1.4 ReturnMatrix Returner3 ( const GenericMatrix & GM )**

Definition at line 28 of file tmt5.cpp.

**30.2145.1.5 ReturnMatrix Returner4 ( const GenericMatrix & GM )**

Definition at line 31 of file tmt5.cpp.

**30.2145.1.6 ReturnMatrix Returner5 ( const GenericMatrix & GM )**

Definition at line 34 of file tmt5.cpp.

**30.2145.1.7 ReturnMatrix Returner6 ( const GenericMatrix & GM )**

Definition at line 37 of file tmt5.cpp.

**30.2145.1.8 ReturnMatrix Returner7 ( const GenericMatrix & GM )**

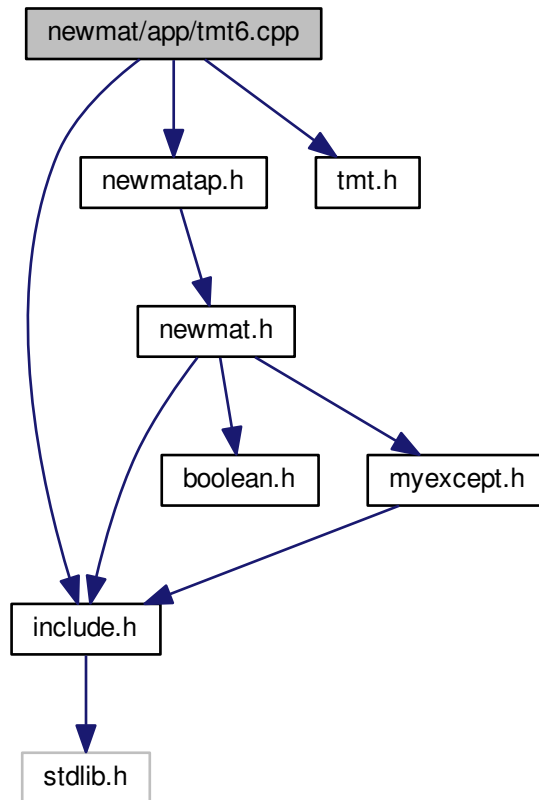
Definition at line 40 of file tmt5.cpp.

**30.2145.1.9 void trymat5 ( )**

Definition at line 43 of file tmt5.cpp.

## 30.2146 newmat/app/tmt6.cpp File Reference

```
#include "include.h"
#include "newmatap.h"
#include "tmt.h"
Include dependency graph for tmt6.cpp:
```



### Macros

- `#define` [WANT\\_MATH](#)

### Functions

- `void` [trymat6](#) ()

### 30.2146.1 Macro Definition Documentation

#### 30.2146.1.1 `#define` [WANT\\_MATH](#)

Definition at line 3 of file `tmt6.cpp`.

## 30.2146.2 Function Documentation

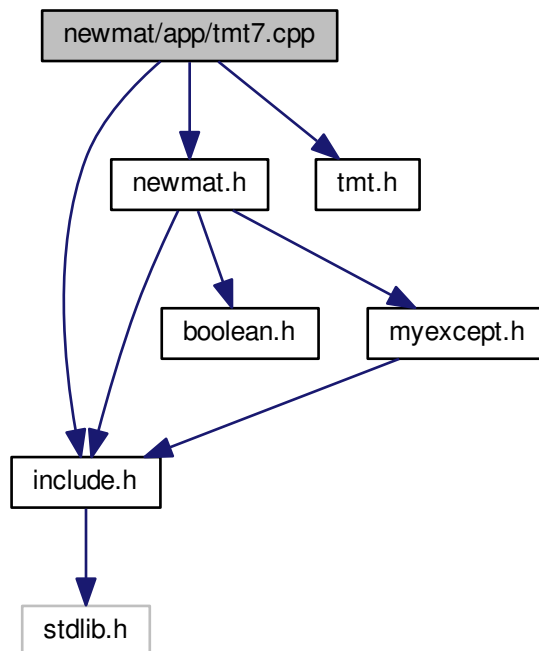
### 30.2146.2.1 void trymat6 ( )

Definition at line 60 of file tmt6.cpp.

## 30.2147 newmat/app/tmt7.cpp File Reference

```
#include "include.h"
#include "newmat.h"
#include "tmt.h"
```

Include dependency graph for tmt7.cpp:



## Functions

- void [trymat7](#) ( )

## 30.2147.1 Function Documentation

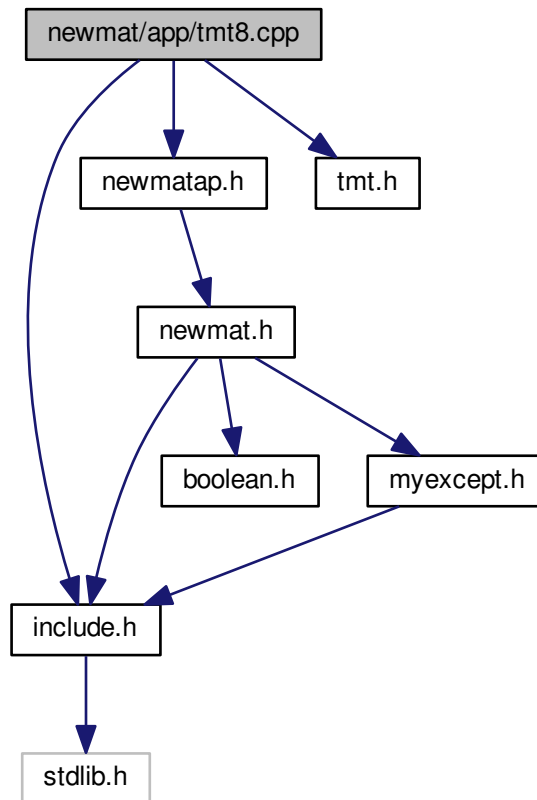
### 30.2147.1.1 void trymat7 ( )

Definition at line 18 of file tmt7.cpp.

## 30.2148 newmat/app/tmt8.cpp File Reference

```
#include "include.h"
#include "newmatap.h"
#include "tmt.h"
```

Include dependency graph for tmt8.cpp:



### Functions

- void `Transposer` (`const GenericMatrix &GM1`, `GenericMatrix &GM2`)
- `ReturnMatrix TestReturn` (`const GeneralMatrix &gm`)
- void `trymat8` ()

### 30.2148.1 Function Documentation

#### 30.2148.1.1 `ReturnMatrix TestReturn` ( `const GeneralMatrix & gm` )

Definition at line 33 of file `tmt8.cpp`.



30.2148.1.2 void Transposer ( const GenericMatrix & GM1, GenericMatrix & GM2 )

Definition at line 19 of file tmt8.cpp.

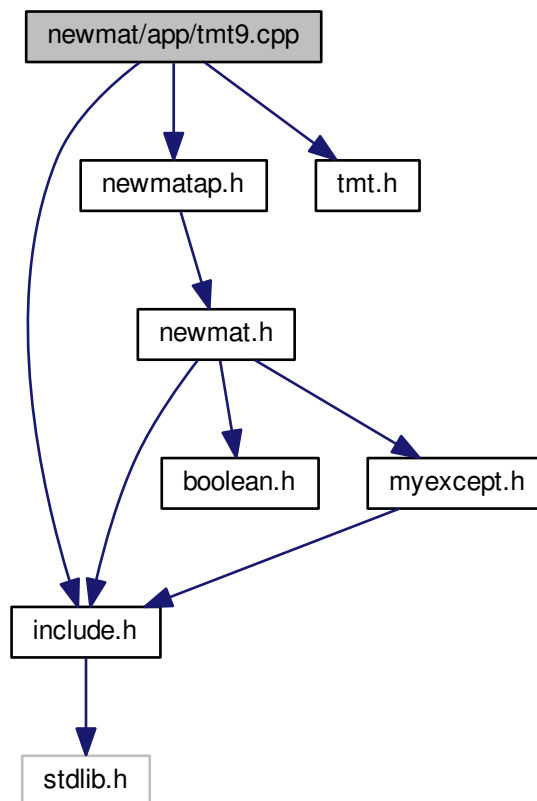
30.2148.1.3 void trymat8 ( )

Definition at line 35 of file tmt8.cpp.

## 30.2149 newmat/app/tmt9.cpp File Reference

```
#include "include.h"
#include "newmatap.h"
#include "tmt.h"
```

Include dependency graph for tmt9.cpp:



## Functions

- void [trymat9](#) ( )

### 30.2149.1 Function Documentation

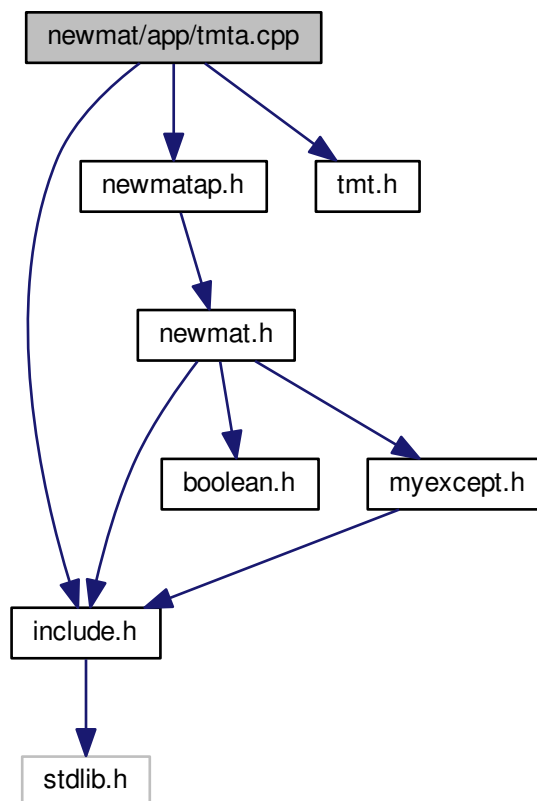
#### 30.2149.1.1 void trymat9 ( )

Definition at line 16 of file tmt9.cpp.

### 30.2150 newmat/app/tmta.cpp File Reference

```
#include "include.h"
#include "newmatap.h"
#include "tmt.h"
```

Include dependency graph for tmta.cpp:



### Functions

- void [trymata](#) ()

### 30.2150.1 Function Documentation

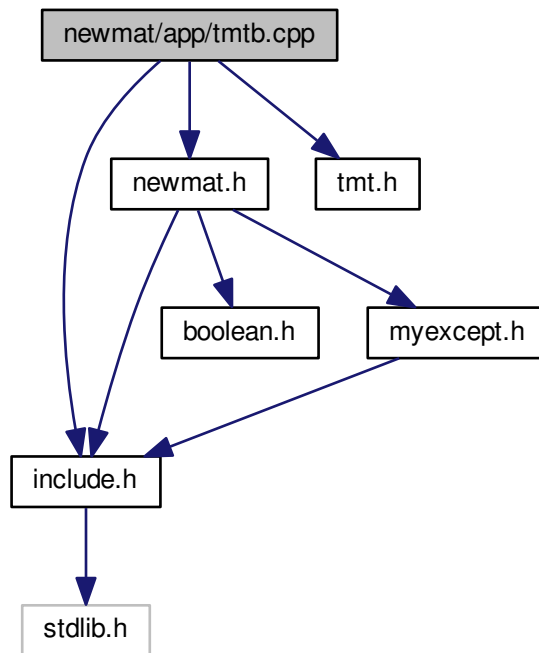
#### 30.2150.1.1 void trymata ( )

Definition at line 32 of file tmta.cpp.

### 30.2151 newmat/app/tmtb.cpp File Reference

```
#include "include.h"
#include "newmat.h"
#include "tmt.h"
```

Include dependency graph for tmtb.cpp:



#### Classes

- class [TestClass](#)

#### Functions

- void [trymatb](#) ()

### 30.2151.1 Function Documentation

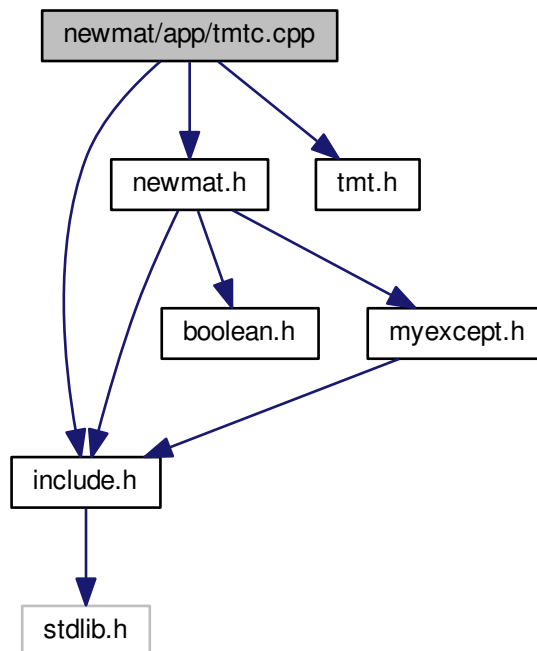
#### 30.2151.1.1 void trymatb ( )

Definition at line 43 of file tmtb.cpp.

### 30.2152 newmat/app/tmtc.cpp File Reference

```
#include "include.h"
#include "newmat.h"
#include "tmt.h"
```

Include dependency graph for tmtc.cpp:



### Functions

- void [trymatc](#) ( )

### 30.2152.1 Function Documentation

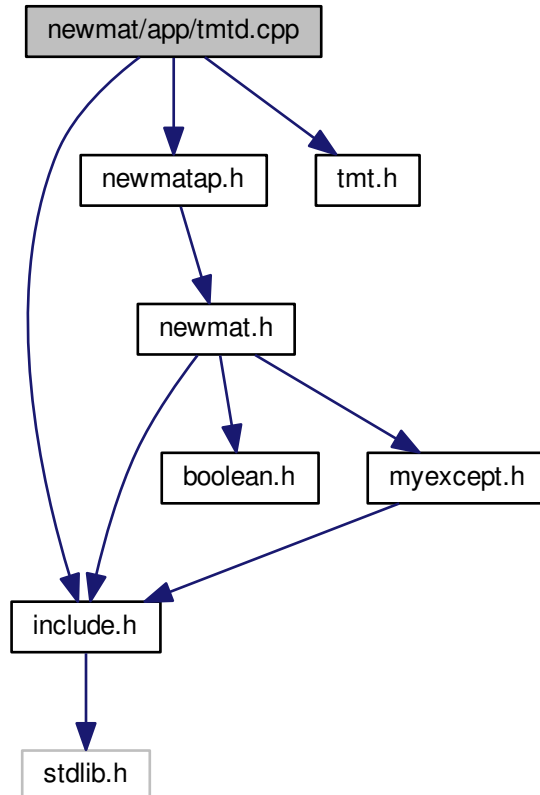
#### 30.2152.1.1 void trymatc ( )

Definition at line 17 of file tmtc.cpp.

## 30.2153 newmat/app/tmtd.cpp File Reference

```
#include "include.h"
#include "newmatap.h"
#include "tmt.h"
```

Include dependency graph for tmtd.cpp:



### Functions

- [ReturnMatrix Inverter](#) (`const CroutMatrix &X`)
- void [trymatd](#) ()

### 30.2153.1 Function Documentation

#### 30.2153.1.1 ReturnMatrix Inverter ( `const CroutMatrix & X` )

Definition at line 13 of file `tmtd.cpp`.

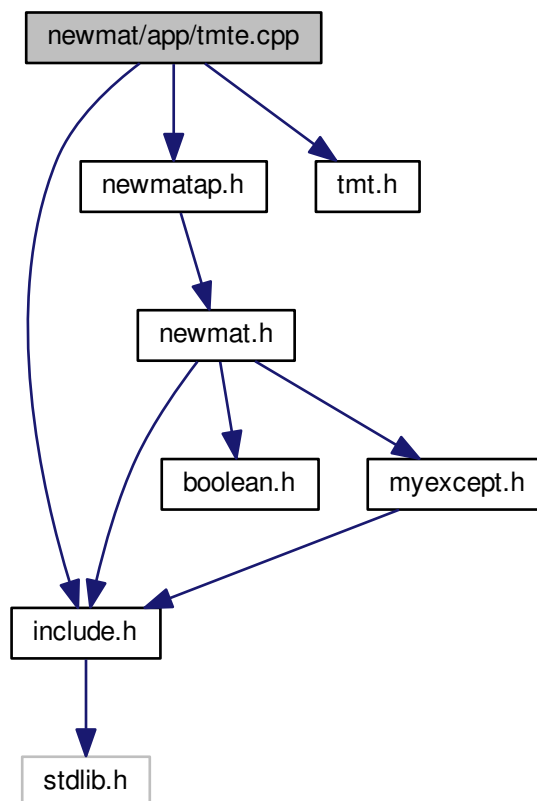
30.2153.1.2 void trymatd ( )

Definition at line 21 of file tmtd.cpp.

## 30.2154 newmat/app/tmte.cpp File Reference

```
#include "include.h"
#include "newmatap.h"
#include "tmt.h"
```

Include dependency graph for tmte.cpp:



### Macros

- #define [WANT\\_MATH](#)

### Functions

- void [CheckIsSorted](#) (const [DiagonalMatrix](#) &D, bool ascending=false)
- void [trymate](#) ()

### 30.2154.1 Macro Definition Documentation

#### 30.2154.1.1 #define WANT\_MATH

Definition at line 3 of file tmt.cpp.

### 30.2154.2 Function Documentation

#### 30.2154.2.1 void CheckIsSorted ( const DiagonalMatrix & D, bool ascending = false )

Definition at line 17 of file tmt.cpp.

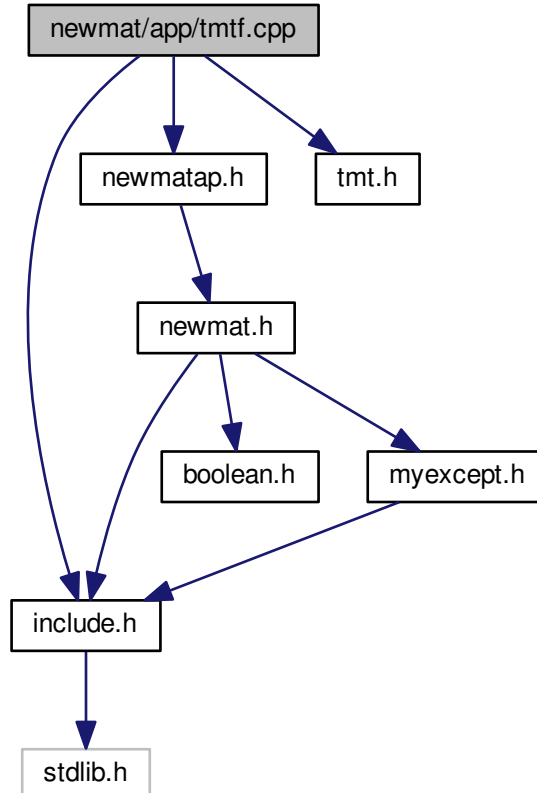
#### 30.2154.2.2 void trymate ( )

Definition at line 26 of file tmt.cpp.

## 30.2155 newmat/app/tmf.cpp File Reference

```
#include "include.h"
#include "newmatap.h"
#include "tmt.h"
```

Include dependency graph for tmf.cpp:



## Macros

- #define [WANT\\_MATH](#)

## Functions

- void [trymatf](#) ()

### 30.2155.1 Macro Definition Documentation

#### 30.2155.1.1 #define WANT\_MATH

Definition at line 3 of file tmtf.cpp.

### 30.2155.2 Function Documentation

#### 30.2155.2.1 void trymatf ( )

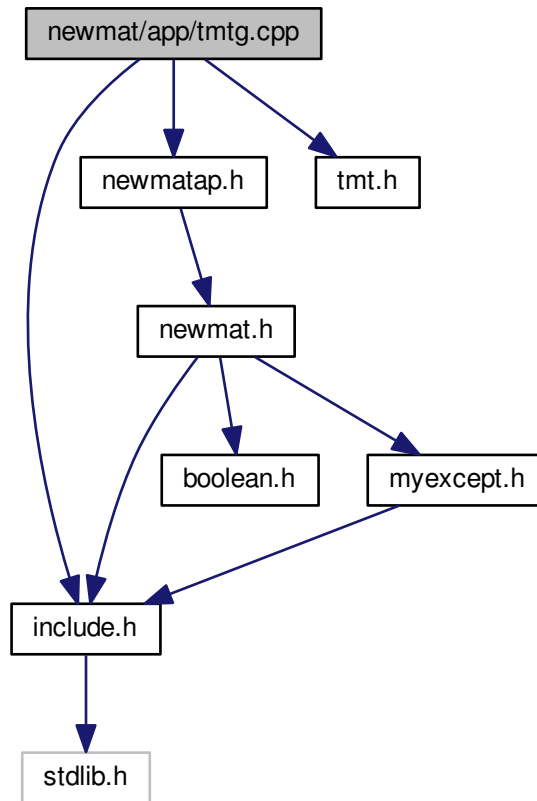
Definition at line 295 of file tmtf.cpp.

## 30.2156 newmat/app/tmtg.cpp File Reference

```
#include "include.h"
#include "newmatap.h"
#include "tmt.h"
```



Include dependency graph for tmtg.cpp:



## Macros

- `#define` [WANT\\_MATH](#)

## Functions

- `void` [trymatg](#) ()

### 30.2156.1 Macro Definition Documentation

#### 30.2156.1.1 `#define` WANT\_MATH

Definition at line 4 of file `tmtg.cpp`.

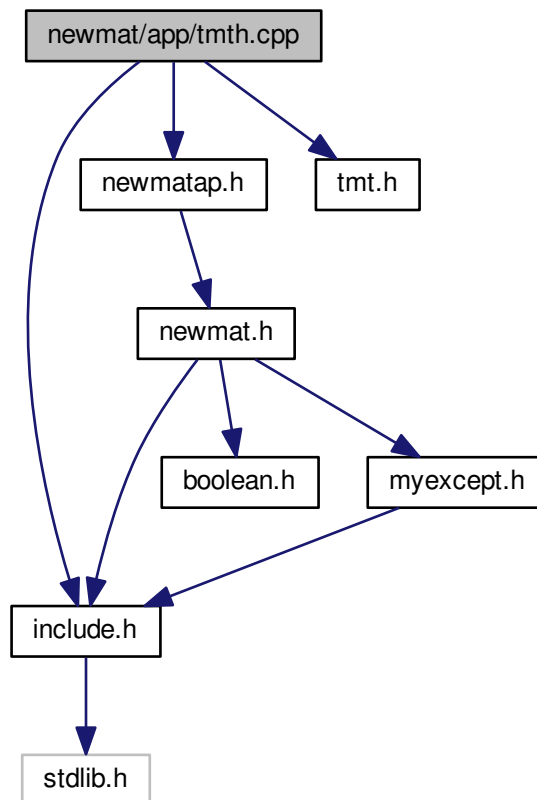
## 30.2156.2 Function Documentation

### 30.2156.2.1 void trymatg ( )

Definition at line 18 of file tmtg.cpp.

## 30.2157 newmat/app/tmth.cpp File Reference

```
#include "include.h"
#include "newmatap.h"
#include "tmt.h"
Include dependency graph for tmth.cpp:
```



## Functions

- void [BandFunctions](#) (int l1, int u1, int l2, int u2)
- void [LowerBandFunctions](#) (int l1, int l2)
- void [UpperBandFunctions](#) (int u1, int u2)
- void [SymmetricBandFunctions](#) (int l1, int l2)
- void [trymath](#) ()

### 30.2157.1 Function Documentation

#### 30.2157.1.1 void BandFunctions ( int *l1*, int *u1*, int *l2*, int *u2* )

Definition at line 20 of file tmth.cpp.

#### 30.2157.1.2 void LowerBandFunctions ( int *l1*, int *l2* )

Definition at line 98 of file tmth.cpp.

#### 30.2157.1.3 void SymmetricBandFunctions ( int *l1*, int *l2* )

Definition at line 248 of file tmth.cpp.

#### 30.2157.1.4 void trymath ( )

Definition at line 365 of file tmth.cpp.

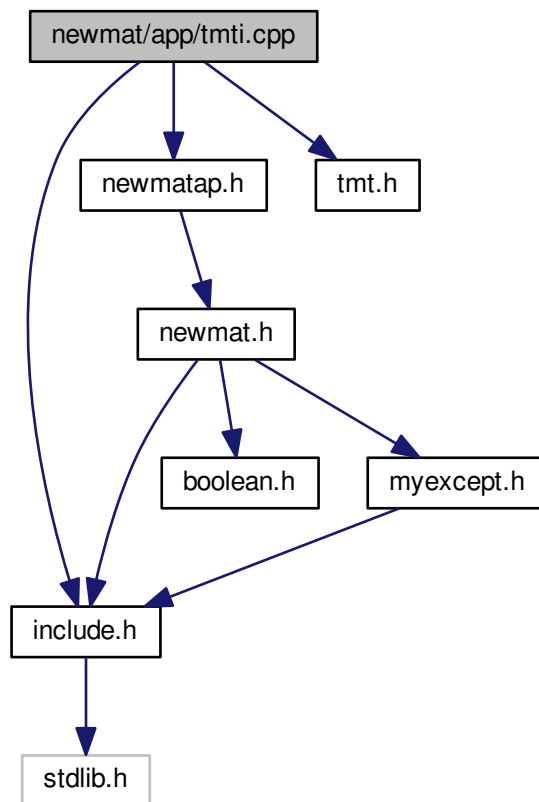
#### 30.2157.1.5 void UpperBandFunctions ( int *u1*, int *u2* )

Definition at line 173 of file tmth.cpp.

## 30.2158 newmat/app/tmti.cpp File Reference

```
#include "include.h"
#include "newmatap.h"
#include "tmt.h"
```

Include dependency graph for tmti.cpp:



## Functions

- void [WillNotConverge](#) ()
- void [ReSizeMatrix](#) (Matrix &A)
- void [trymati](#) ()

### 30.2158.1 Function Documentation

#### 30.2158.1.1 void [ReSizeMatrix](#) ( Matrix & A )

Definition at line 22 of file `tmti.cpp`.

#### 30.2158.1.2 void [trymati](#) ( )

Definition at line 26 of file `tmti.cpp`.

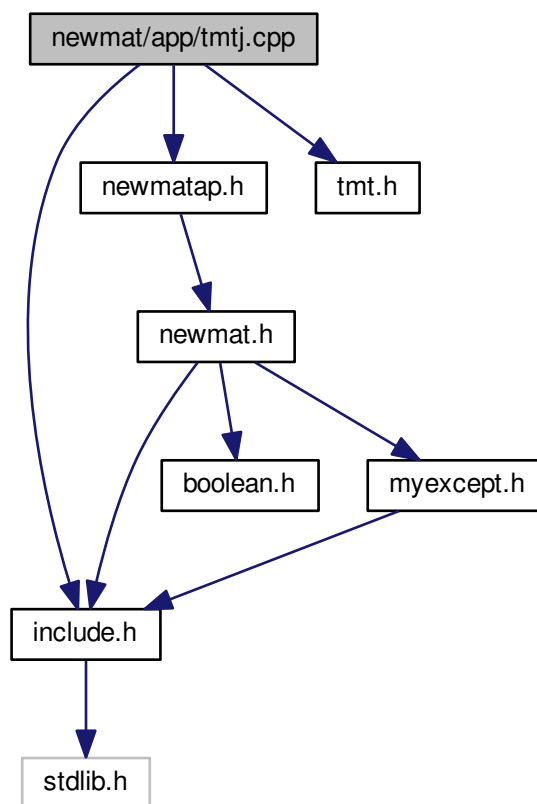
30.2158.1.3 void WillNotConverge ( )

Definition at line 16 of file tmti.cpp.

## 30.2159 newmat/app/tmtj.cpp File Reference

```
#include "include.h"
#include "newmatap.h"
#include "tmt.h"
```

Include dependency graph for tmtj.cpp:



### Functions

- void [trymatj](#) ( )

### 30.2159.1 Function Documentation

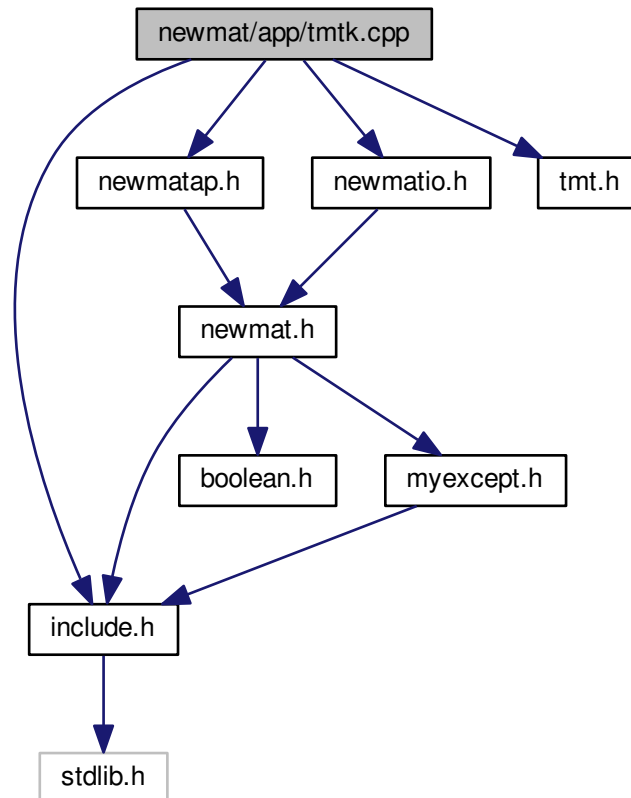
30.2159.1.1 void [trymatj](#) ( )

Definition at line 16 of file `tmtj.cpp`.

## 30.2160 newmat/app/tmtk.cpp File Reference

```
#include "include.h"
#include "newmatap.h"
#include "newmatio.h"
#include "tmt.h"
```

Include dependency graph for tmtk.cpp:



### Macros

- `#define WANT_STREAM`

### Functions

- `void trymatk ()`

### 30.2160.1 Macro Definition Documentation

#### 30.2160.1.1 `#define WANT_STREAM`

Definition at line 2 of file `tmtk.cpp`.

## 30.2160.2 Function Documentation

### 30.2160.2.1 void trymatk ( )

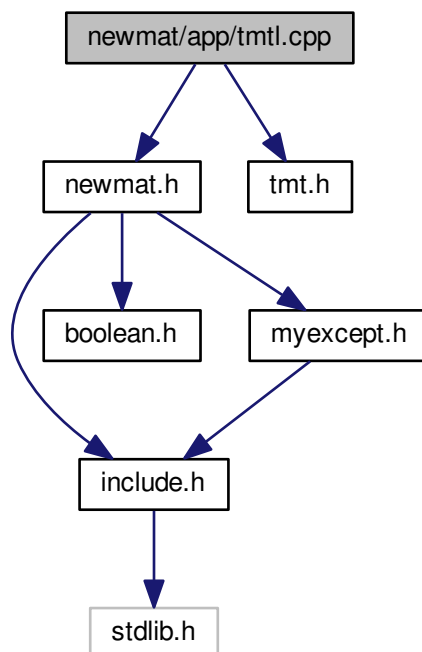
Definition at line 180 of file tmtk.cpp.

## 30.2161 newmat/app/tmtl.cpp File Reference

```
#include "newmat.h"
```

```
#include "tmt.h"
```

Include dependency graph for tmtl.cpp:



## Macros

- #define [WANT\\_STREAM](#)
- #define [WANT\\_MATH](#)

## Functions

- Real [TestMax](#) (`const GenericMatrix &GM`)
- void [trymatl](#) ( )

### 30.2161.1 Macro Definition Documentation

#### 30.2161.1.1 #define WANT\_MATH

Definition at line 4 of file tmtl.cpp.

#### 30.2161.1.2 #define WANT\_STREAM

Definition at line 2 of file tmtl.cpp.

### 30.2161.2 Function Documentation

#### 30.2161.2.1 Real TestMax ( const GenericMatrix & GM )

Definition at line 18 of file tmtl.cpp.

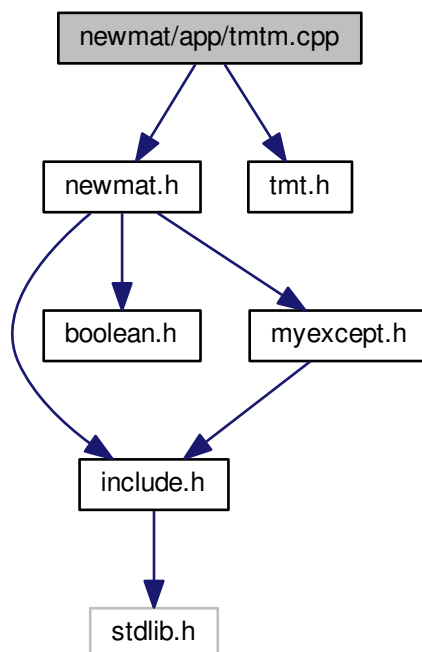
#### 30.2161.2.2 void trymatl ( )

Definition at line 132 of file tmtl.cpp.

## 30.2162 newmat/app/tmtm.cpp File Reference

```
#include "newmat.h"
#include "tmt.h"
```

Include dependency graph for tmtm.cpp:





## Macros

- #define [WANT\\_STREAM](#)
- #define [WANT\\_MATH](#)

## Functions

- void [trymatm](#) ()

### 30.2162.1 Macro Definition Documentation

#### 30.2162.1.1 #define WANT\_MATH

Definition at line 4 of file tmtm.cpp.

#### 30.2162.1.2 #define WANT\_STREAM

Definition at line 2 of file tmtm.cpp.

### 30.2162.2 Function Documentation

#### 30.2162.2.1 void trymatm ( )

Definition at line 19 of file tmtm.cpp.

## 30.2163 newmat/include/boolean.h File Reference

This graph shows which files directly or indirectly include this file:



## Classes

- class [bool](#)

## Macros

- #define [bool\\_LIB](#) 0

## Variables

- [const bool true](#) = 1
- [const bool false](#) = 0

### 30.2163.1 Macro Definition Documentation

#### 30.2163.1.1 #define bool\_LIB 0

Definition at line 6 of file boolean.h.

### 30.2163.2 Variable Documentation

#### 30.2163.2.1 const bool false = 0

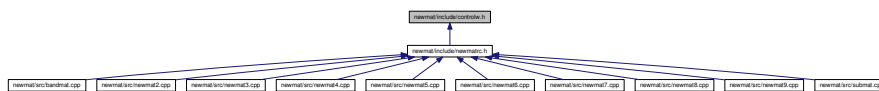
Definition at line 26 of file boolean.h.

#### 30.2163.2.2 const bool true = 1

Definition at line 25 of file boolean.h.

## 30.2164 newmat/include/controlw.h File Reference

This graph shows which files directly or indirectly include this file:



## Classes

- class [ControlWord](#)

## Macros

- #define [CONTROL\\_WORD\\_LIB](#) 0

### 30.2164.1 Macro Definition Documentation

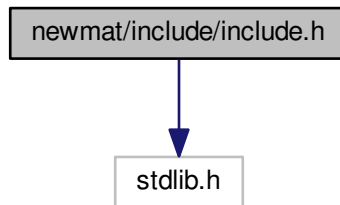
#### 30.2164.1.1 #define CONTROL\_WORD\_LIB 0

Definition at line 4 of file controlw.h.

## 30.2165 newmat/include/include.h File Reference

```
#include <stdlib.h>
```

Include dependency graph for include.h:



This graph shows which files directly or indirectly include this file:



### Namespaces

- [RBD\\_COMMON](#)
- [RBD\\_LIBRARIES](#)

### Macros

- `#define` [use\\_namespace](#)
- `#define` [UseExceptions](#)
- `#define` [USING\\_DOUBLE](#)
- `#define` [bool\\_LIB](#) 0
- `#define` [TypeDefException](#)
- `#define` [DEFAULT\\_HEADER](#)
- `#define` [ATandT](#)

### Typedefs

- `typedef` [double](#) [RBD\\_COMMON::Real](#)
- `typedef` [long](#) [double](#) [RBD\\_COMMON::long\\_Real](#)

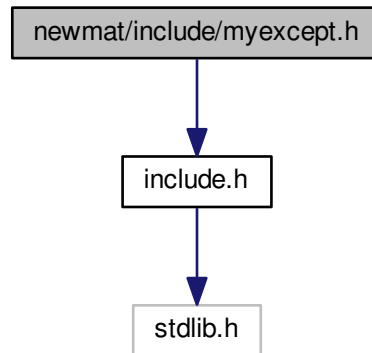
### 30.2165.1 Detailed Description

Set options and and details of include files.

## 30.2166 newmat/include/myexcept.h File Reference

```
#include "include.h"
```

Include dependency graph for myexcept.h:



This graph shows which files directly or indirectly include this file:



### Classes

- class [RBD\\_COMMON::Tracer](#)
- class [RBD\\_COMMON::BaseException](#)
- class [RBD\\_COMMON::Janitor](#)
- class [RBD\\_COMMON::Logic\\_error](#)
- class [RBD\\_COMMON::Runtime\\_error](#)
- class [RBD\\_COMMON::Domain\\_error](#)
- class [RBD\\_COMMON::Invalid\\_argument](#)
- class [RBD\\_COMMON::Length\\_error](#)
- class [RBD\\_COMMON::Out\\_of\\_range](#)
- class [RBD\\_COMMON::Range\\_error](#)
- class [RBD\\_COMMON::Overflow\\_error](#)
- class [RBD\\_COMMON::Bad\\_alloc](#)

### Namespaces

- [RBD\\_COMMON](#)

## Macros

- #define [Try](#) try
- #define [Throw](#)(E) throw E
- #define [ReThrow](#) throw
- #define [Catch](#) catch
- #define [CatchAll](#) catch(...)
- #define [CatchAndThrow](#) {}
- #define [FREE\\_CHECK](#)(Class) public:
- #define [MONITOR\\_REAL\\_NEW](#)(Operation, Size, Pointer) {}
- #define [MONITOR\\_INT\\_NEW](#)(Operation, Size, Pointer) {}
- #define [MONITOR\\_REAL\\_DELETE](#)(Operation, Size, Pointer) {}
- #define [MONITOR\\_INT\\_DELETE](#)(Operation, Size, Pointer) {}
- #define [NEW\\_DELETE](#)(Class) [FREE\\_CHECK](#)(Class)

## Typedefs

- typedef BaseException [RBD\\_COMMON::Exception](#)

## Functions

- void [RBD\\_COMMON::Terminate](#) ()

### 30.2166.1 Detailed Description

Exception handler. The low level classes for

- my exception class hierarchy
- the functions needed for my simulated exceptions
- the Tracer mechanism
- routines for checking whether new and delete calls are balanced

### 30.2166.2 Macro Definition Documentation

#### 30.2166.2.1 #define [Catch](#) catch

Definition at line 193 of file myexcept.h.

#### 30.2166.2.2 #define [CatchAll](#) catch(...)

Definition at line 194 of file myexcept.h.

#### 30.2166.2.3 #define [CatchAndThrow](#) {}

Definition at line 195 of file myexcept.h.

30.2166.2.4 `#define FREE_CHECK( Class ) public:`

Definition at line 328 of file myexcept.h.

30.2166.2.5 `#define MONITOR_INT_DELETE( Operation, Size, Pointer ) {}`

Definition at line 332 of file myexcept.h.

30.2166.2.6 `#define MONITOR_INT_NEW( Operation, Size, Pointer ) {}`

Definition at line 330 of file myexcept.h.

30.2166.2.7 `#define MONITOR_REAL_DELETE( Operation, Size, Pointer ) {}`

Definition at line 331 of file myexcept.h.

30.2166.2.8 `#define MONITOR_REAL_NEW( Operation, Size, Pointer ) {}`

Definition at line 329 of file myexcept.h.

30.2166.2.9 `#define NEW_DELETE( Class ) FREE_CHECK(Class)`

Definition at line 350 of file myexcept.h.

30.2166.2.10 `#define ReThrow throw`

Definition at line 192 of file myexcept.h.

30.2166.2.11 `#define Throw( E ) throw E`

Definition at line 191 of file myexcept.h.

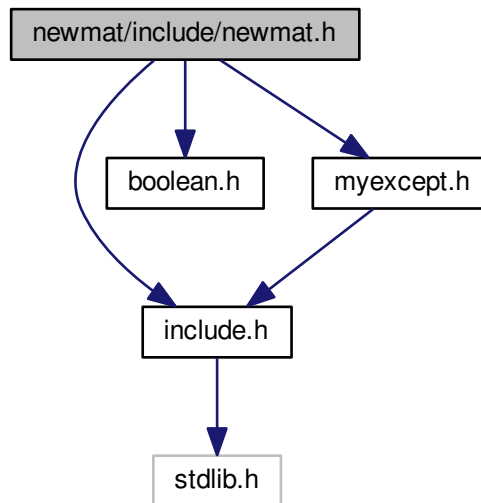
30.2166.2.12 `#define Try try`

Definition at line 190 of file myexcept.h.

## 30.2167 newmat/include/newmat.h File Reference

```
#include "include.h"
#include "boolean.h"
#include "myexcept.h"
```

Include dependency graph for newmat.h:



This graph shows which files directly or indirectly include this file:



### Classes

- class [NEWMAT::LogAndSign](#)
- class [NEWMAT::MatrixType](#)
- class [NEWMAT::MatrixBandWidth](#)
- class [NEWMAT::ArrayLengthSpecifier](#)
- class [NEWMAT::BaseMatrix](#)
- class [NEWMAT::GeneralMatrix](#)
- class [NEWMAT::Matrix](#)
- class [NEWMAT::nricMatrix](#)
- class [NEWMAT::SymmetricMatrix](#)
- class [NEWMAT::UpperTriangularMatrix](#)
- class [NEWMAT::LowerTriangularMatrix](#)
- class [NEWMAT::DiagonalMatrix](#)
- class [NEWMAT::RowVector](#)
- class [NEWMAT::ColumnVector](#)
- class [NEWMAT::CroutMatrix](#)

- class [NEWMAT::BandMatrix](#)
- class [NEWMAT::UpperBandMatrix](#)
- class [NEWMAT::LowerBandMatrix](#)
- class [NEWMAT::SymmetricBandMatrix](#)
- class [NEWMAT::BandLUMatrix](#)
- class [NEWMAT::IdentityMatrix](#)
- class [NEWMAT::GenericMatrix](#)
- class [NEWMAT::MultipliedMatrix](#)
- class [NEWMAT::AddedMatrix](#)
- class [NEWMAT::SPMatrix](#)
- class [NEWMAT::KPMatrix](#)
- class [NEWMAT::ConcatenatedMatrix](#)
- class [NEWMAT::StackedMatrix](#)
- class [NEWMAT::SolvedMatrix](#)
- class [NEWMAT::SubtractedMatrix](#)
- class [NEWMAT::ShiftedMatrix](#)
- class [NEWMAT::NegShiftedMatrix](#)
- class [NEWMAT::ScaledMatrix](#)
- class [NEWMAT::NegatedMatrix](#)
- class [NEWMAT::TransposedMatrix](#)
- class [NEWMAT::ReversedMatrix](#)
- class [NEWMAT::InvertedMatrix](#)
- class [NEWMAT::RowedMatrix](#)
- class [NEWMAT::ColedMatrix](#)
- class [NEWMAT::DiagedMatrix](#)
- class [NEWMAT::MatedMatrix](#)
- class [NEWMAT::ReturnMatrixX](#)
- class [NEWMAT::GetSubMatrix](#)
- class [NEWMAT::LinearEquationSolver](#)
- class [NEWMAT::MatrixInput](#)
- class [NEWMAT::SimpleIntArray](#)
- class [NEWMAT::NPDException](#)
- class [NEWMAT::ConvergenceException](#)
- class [NEWMAT::SingularException](#)
- class [NEWMAT::OverflowException](#)
- class [NEWMAT::ProgramException](#)
- class [NEWMAT::IndexException](#)
- class [NEWMAT::VectorException](#)
- class [NEWMAT::NotSquareException](#)
- class [NEWMAT::SubMatrixDimensionException](#)
- class [NEWMAT::IncompatibleDimensionsException](#)
- class [NEWMAT::NotDefinedException](#)
- class [NEWMAT::CannotBuildException](#)
- class [NEWMAT::InternalException](#)

## Namespaces

- [NEWMAT](#)
- [RBD\\_LIBRARIES](#)

## Macros

- `#define NEWMAT_LIB 0`
- `#define ReturnMatrix ReturnMatrixX`
- `#define MatrixTypeUnSp 0`



## Functions

- void `NEWMAT::MatrixErrorNoSpace` (void \*)
- bool `NEWMAT::operator==` (const `GeneralMatrix` &A, const `GeneralMatrix` &B)
- bool `NEWMAT::operator==` (const `BaseMatrix` &A, const `BaseMatrix` &B)
- bool `NEWMAT::operator!=` (const `GeneralMatrix` &A, const `GeneralMatrix` &B)
- bool `NEWMAT::operator!=` (const `BaseMatrix` &A, const `BaseMatrix` &B)
- bool `NEWMAT::operator<=` (const `BaseMatrix` &A, const `BaseMatrix` &B)
- bool `NEWMAT::operator>=` (const `BaseMatrix` &A, const `BaseMatrix` &B)
- bool `NEWMAT::operator<` (const `BaseMatrix` &A, const `BaseMatrix` &B)
- bool `NEWMAT::operator>` (const `BaseMatrix` &A, const `BaseMatrix` &B)
- bool `NEWMAT::IsZero` (const `BaseMatrix` &A)
- bool `NEWMAT::Rectangular` (`MatrixType` a, `MatrixType` b, `MatrixType` c)
- bool `NEWMAT::Compare` (const `MatrixType` &, `MatrixType` &)
- `Real` `NEWMAT::DotProduct` (const `Matrix` &A, const `Matrix` &B)
- `SPMatrix` `NEWMAT::SP` (const `BaseMatrix` &, const `BaseMatrix` &)
- `KPMatrix` `NEWMAT::KP` (const `BaseMatrix` &, const `BaseMatrix` &)
- `ShiftedMatrix` `NEWMAT::operator+` (`Real` f, const `BaseMatrix` &BM)
- `NegShiftedMatrix` `NEWMAT::operator-` (`Real`, const `BaseMatrix` &)
- `ScaledMatrix` `NEWMAT::operator*` (`Real` f, const `BaseMatrix` &BM)
- `LogAndSign` `NEWMAT::LogDeterminant` (const `BaseMatrix` &B)
- `Real` `NEWMAT::Determinant` (const `BaseMatrix` &B)
- `Real` `NEWMAT::SumSquare` (const `BaseMatrix` &B)
- `Real` `NEWMAT::NormFrobenius` (const `BaseMatrix` &B)
- `Real` `NEWMAT::Trace` (const `BaseMatrix` &B)
- `Real` `NEWMAT::SumAbsoluteValue` (const `BaseMatrix` &B)
- `Real` `NEWMAT::Sum` (const `BaseMatrix` &B)
- `Real` `NEWMAT::MaximumAbsoluteValue` (const `BaseMatrix` &B)
- `Real` `NEWMAT::MinimumAbsoluteValue` (const `BaseMatrix` &B)
- `Real` `NEWMAT::Maximum` (const `BaseMatrix` &B)
- `Real` `NEWMAT::Minimum` (const `BaseMatrix` &B)
- `Real` `NEWMAT::Norm1` (const `BaseMatrix` &B)
- `Real` `NEWMAT::Norm1` (`RowVector` &RV)
- `Real` `NEWMAT::NormInfinity` (const `BaseMatrix` &B)
- `Real` `NEWMAT::NormInfinity` (`ColumnVector` &CV)
- bool `NEWMAT::IsZero` (const `GeneralMatrix` &A)

### 30.2167.1 Macro Definition Documentation

#### 30.2167.1.1 #define `MatrixTypeUnSp` 0

Definition at line 271 of file `newmat.h`.

#### 30.2167.1.2 #define `NEWMAT_LIB` 0

Definition at line 6 of file `newmat.h`.

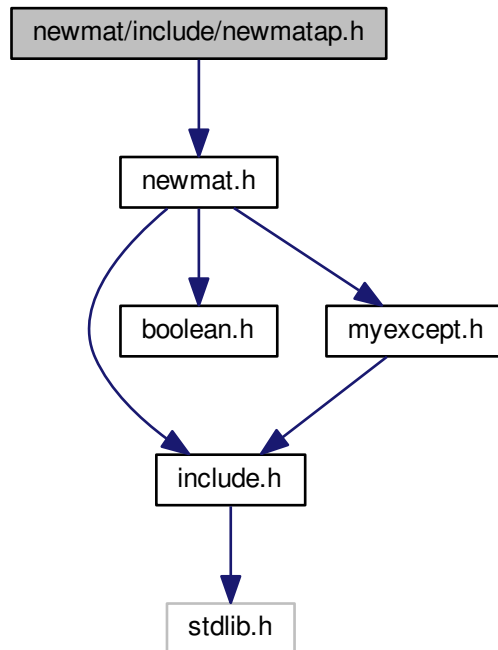
#### 30.2167.1.3 #define `ReturnMatrix` `ReturnMatrixX`

Definition at line 35 of file `newmat.h`.

## 30.2168 newmat/include/newmatap.h File Reference

```
#include "newmat.h"
```

Include dependency graph for newmatap.h:



This graph shows which files directly or indirectly include this file:



### Classes

- class [NEWMAT::SymmetricEigenAnalysis](#)
- class [NEWMAT::FFT\\_Controller](#)
- class [NEWMAT::MultiRadixCounter](#)

### Namespaces

- [NEWMAT](#)

### Macros

- `#define` [NEWMATAP\\_LIB](#) 0

## Functions

- void [NEWMAT::QRZT](#) (Matrix &, LowerTriangularMatrix &)
- void [NEWMAT::QRZT](#) (const Matrix &, Matrix &, Matrix &)
- void [NEWMAT::QRZ](#) (Matrix &, UpperTriangularMatrix &)
- void [NEWMAT::QRZ](#) (const Matrix &, Matrix &, Matrix &)
- void [NEWMAT::HHDecompose](#) (Matrix &X, LowerTriangularMatrix &L)
- void [NEWMAT::HHDecompose](#) (const Matrix &X, Matrix &Y, Matrix &M)
- [ReturnMatrix](#) [NEWMAT::Cholesky](#) (const SymmetricMatrix &)
- [ReturnMatrix](#) [NEWMAT::Cholesky](#) (const SymmetricBandMatrix &)
- void [NEWMAT::SVD](#) (const Matrix &, DiagonalMatrix &, Matrix &, Matrix &, bool=true, bool=true)
- void [NEWMAT::SVD](#) (const Matrix &, DiagonalMatrix &)
- void [NEWMAT::SVD](#) (const Matrix &A, DiagonalMatrix &D, Matrix &U, bool withU=true)
- void [NEWMAT::SortSV](#) (DiagonalMatrix &D, Matrix &U, bool ascending=false)
- void [NEWMAT::SortSV](#) (DiagonalMatrix &D, Matrix &U, Matrix &V, bool ascending=false)
- void [NEWMAT::Jacobi](#) (const SymmetricMatrix &, DiagonalMatrix &)
- void [NEWMAT::Jacobi](#) (const SymmetricMatrix &, DiagonalMatrix &, SymmetricMatrix &)
- void [NEWMAT::Jacobi](#) (const SymmetricMatrix &, DiagonalMatrix &, Matrix &)
- void [NEWMAT::Jacobi](#) (const SymmetricMatrix &, DiagonalMatrix &, SymmetricMatrix &, Matrix &, bool=true)
- void [NEWMAT::EigenValues](#) (const SymmetricMatrix &, DiagonalMatrix &)
- void [NEWMAT::EigenValues](#) (const SymmetricMatrix &, DiagonalMatrix &, SymmetricMatrix &)
- void [NEWMAT::EigenValues](#) (const SymmetricMatrix &, DiagonalMatrix &, Matrix &)
- void [NEWMAT::SortAscending](#) (GeneralMatrix &)
- void [NEWMAT::SortDescending](#) (GeneralMatrix &)
- void [NEWMAT::FFT](#) (const ColumnVector &, const ColumnVector &, ColumnVector &, ColumnVector &)
- void [NEWMAT::FFTI](#) (const ColumnVector &, const ColumnVector &, ColumnVector &, ColumnVector &)
- void [NEWMAT::RealFFT](#) (const ColumnVector &, ColumnVector &, ColumnVector &)
- void [NEWMAT::RealFFTI](#) (const ColumnVector &, const ColumnVector &, ColumnVector &)
- void [NEWMAT::DCT\\_II](#) (const ColumnVector &, ColumnVector &)
- void [NEWMAT::DCT\\_II\\_inverse](#) (const ColumnVector &, ColumnVector &)
- void [NEWMAT::DST\\_II](#) (const ColumnVector &, ColumnVector &)
- void [NEWMAT::DST\\_II\\_inverse](#) (const ColumnVector &, ColumnVector &)
- void [NEWMAT::DCT](#) (const ColumnVector &, ColumnVector &)
- void [NEWMAT::DCT\\_inverse](#) (const ColumnVector &, ColumnVector &)
- void [NEWMAT::DST](#) (const ColumnVector &, ColumnVector &)
- void [NEWMAT::DST\\_inverse](#) (const ColumnVector &, ColumnVector &)

### 30.2168.1 Macro Definition Documentation

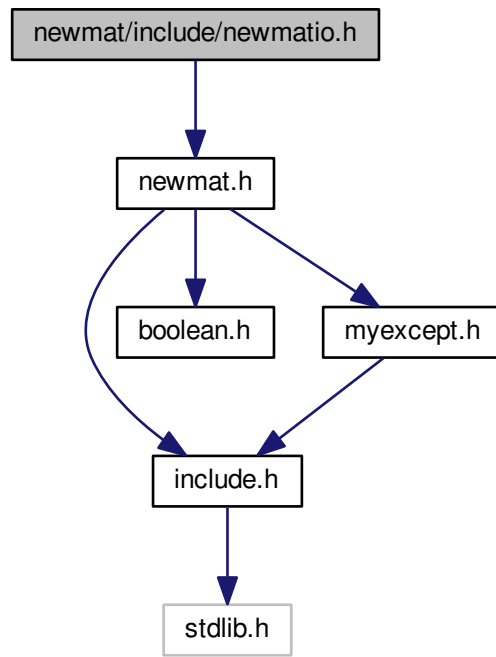
#### 30.2168.1.1 #define NEWMATAP\_LIB 0

Definition at line 6 of file newmatap.h.

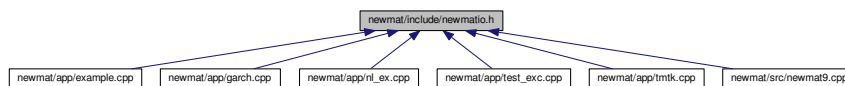
## 30.2169 newmat/include/newmatio.h File Reference

```
#include "newmat.h"
```

Include dependency graph for newmatio.h:



This graph shows which files directly or indirectly include this file:



### Namespaces

- [NEWMAT](#)

### Macros

- `#define` [NEWMATIO\\_LIB](#) 0

### Functions

- `std::ostream &` [NEWMAT::operator<<](#) (`std::ostream &`, `const BaseMatrix &`)
- `std::ostream &` [NEWMAT::operator<<](#) (`std::ostream &`, `const GeneralMatrix &`)

### 30.2169.1 Macro Definition Documentation

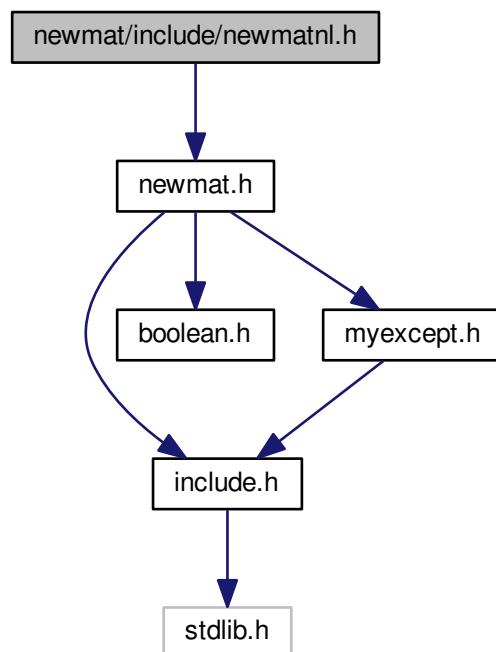
#### 30.2169.1.1 #define NEWMATIO\_LIB 0

Definition at line 6 of file newmatio.h.

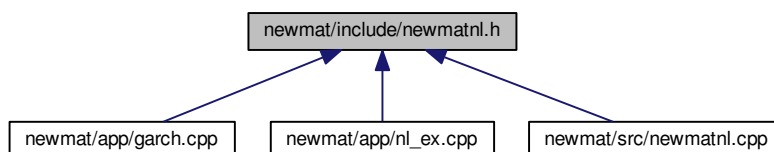
### 30.2170 newmat/include/newmatnl.h File Reference

```
#include "newmat.h"
```

Include dependency graph for newmatnl.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [NEWMAT::FindMaximum2](#)
- class [NEWMAT::R1\\_Col\\_I\\_D](#)
- class [NEWMAT::NonLinearLeastSquares](#)
- class [NEWMAT::LL\\_D\\_FI](#)
- class [NEWMAT::MLE\\_D\\_FI](#)

## Namespaces

- [NEWMAT](#)

## Macros

- `#define NEWMATNL_LIB 0`

### 30.2170.1 Macro Definition Documentation

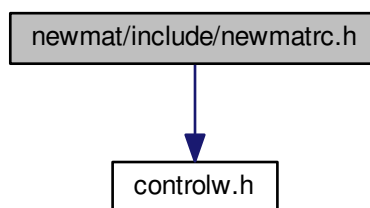
#### 30.2170.1.1 `#define NEWMATNL_LIB 0`

Definition at line 6 of file `newmatnl.h`.

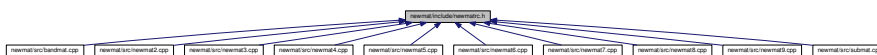
### 30.2171 `newmat/include/newmatrc.h` File Reference

```
#include "controlw.h"
```

Include dependency graph for `newmatrc.h`:



This graph shows which files directly or indirectly include this file:



## Classes

- class [LoadAndStoreFlag](#)
- class [MatrixRowCol](#)
- class [MatrixRow](#)
- class [MatrixCol](#)
- class [MatrixColX](#)

## Macros

- #define [NEWMATRC\\_LIB](#) 0

## Enumerations

- enum [LSF](#) {  
  [LoadOnEntry](#) =1, [StoreOnExit](#) =2, [DirectPart](#) =4, [StoreHere](#) =8,  
  [HaveStore](#) =16 }

### 30.2171.1 Macro Definition Documentation

#### 30.2171.1.1 #define NEWMATRC\_LIB 0

Definition at line 6 of file newmatrc.h.

### 30.2171.2 Enumeration Type Documentation

#### 30.2171.2.1 enum LSF

Enumerator

***LoadOnEntry***

***StoreOnExit***

***DirectPart***

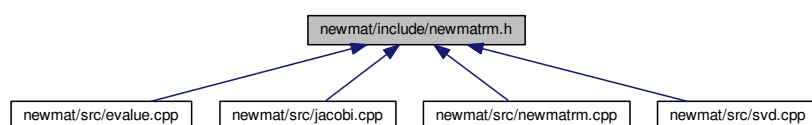
***StoreHere***

***HaveStore***

Definition at line 21 of file newmatrc.h.

## 30.2172 newmat/include/newmatrm.h File Reference

This graph shows which files directly or indirectly include this file:



## Classes

- class [RectMatrixRowCol](#)
- class [RectMatrixRow](#)
- class [RectMatrixCol](#)
- class [RectMatrixDiag](#)

## Macros

- `#define NEWMATRM\_LIB 0`

## Functions

- Real [square](#) (Real x)
- Real [sign](#) (Real x, Real y)

### 30.2172.1 Macro Definition Documentation

#### 30.2172.1.1 `#define NEWMATRM\_LIB 0`

Definition at line 6 of file `newmatrm.h`.

### 30.2172.2 Function Documentation

#### 30.2172.2.1 Real `sign ( Real x, Real y )` [`inline`]

Definition at line 106 of file `newmatrm.h`.

#### 30.2172.2.2 Real `square ( Real x )` [`inline`]

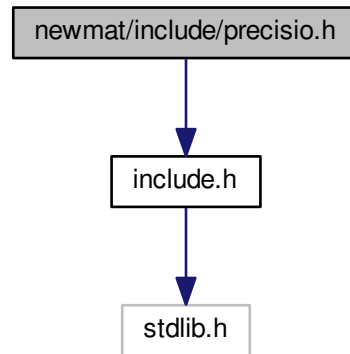
Definition at line 105 of file `newmatrm.h`.



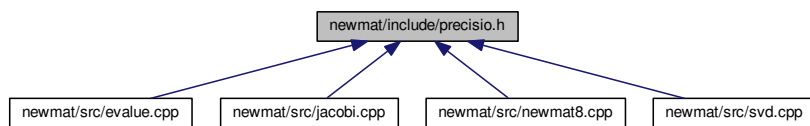
## 30.2173 newmat/include/precisio.h File Reference

```
#include "include.h"
```

Include dependency graph for precisio.h:



This graph shows which files directly or indirectly include this file:



### Classes

- class [NEWMAT::FloatingPointPrecision](#)  
*Floating point precision (type double).*

### Namespaces

- [NEWMAT](#)

### Macros

- #define [PRECISION\\_LIB](#) 0
- #define [WANT\\_MATH](#)

### 30.2173.1 Detailed Description

Floating point precision constants.

### 30.2173.2 Macro Definition Documentation

#### 30.2173.2.1 `#define PRECISION_LIB 0`

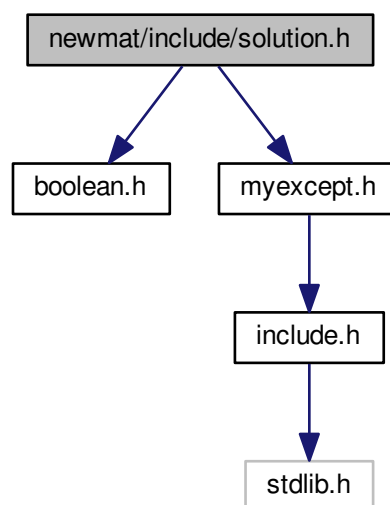
Definition at line 8 of file `precisio.h`.

#### 30.2173.2.2 `#define WANT_MATH`

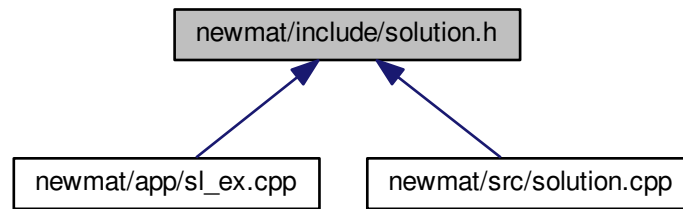
Definition at line 10 of file `precisio.h`.

## 30.2174 `newmat/include/solution.h` File Reference

```
#include "boolean.h"
#include "myexcept.h"
Include dependency graph for solution.h:
```



This graph shows which files directly or indirectly include this file:



### Classes

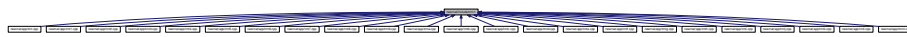
- class [RBD\\_COMMON::R1\\_R1](#)
- class [RBD\\_COMMON::SolutionException](#)
- class [RBD\\_COMMON::OneDimSolve](#)

### Namespaces

- [RBD\\_COMMON](#)

## 30.2175 newmat/include/tmt.h File Reference

This graph shows which files directly or indirectly include this file:



### Classes

- class [time\\_lapse](#)

### Functions

- void [Print](#) ([const Matrix](#) &X)
- void [Print](#) ([const UpperTriangularMatrix](#) &X)
- void [Print](#) ([const DiagonalMatrix](#) &X)
- void [Print](#) ([const SymmetricMatrix](#) &X)
- void [Print](#) ([const LowerTriangularMatrix](#) &X)
- void [Clean](#) ([Matrix](#) &, [Real](#))
- void [Clean](#) ([DiagonalMatrix](#) &, [Real](#))
- void [trymat1](#) ()
- void [trymat2](#) ()

- void [trymat3](#) ()
- void [trymat4](#) ()
- void [trymat5](#) ()
- void [trymat6](#) ()
- void [trymat7](#) ()
- void [trymat8](#) ()
- void [trymat9](#) ()
- void [trymata](#) ()
- void [trymatb](#) ()
- void [trymatc](#) ()
- void [trymatd](#) ()
- void [trymate](#) ()
- void [trymatf](#) ()
- void [trymatg](#) ()
- void [trymath](#) ()
- void [trymati](#) ()
- void [trymatj](#) ()
- void [trymatk](#) ()
- void [trymatl](#) ()
- void [trymatm](#) ()

### 30.2175.1 Function Documentation

#### 30.2175.1.1 void Clean ( Matrix & , Real )

Definition at line 121 of file tmt.cpp.

#### 30.2175.1.2 void Clean ( DiagonalMatrix & , Real )

Definition at line 131 of file tmt.cpp.

#### 30.2175.1.3 void Print ( const Matrix & X )

Definition at line 35 of file tmt.cpp.

#### 30.2175.1.4 void Print ( const UpperTriangularMatrix & X )

Definition at line 51 of file tmt.cpp.

#### 30.2175.1.5 void Print ( const DiagonalMatrix & X )

Definition at line 69 of file tmt.cpp.

#### 30.2175.1.6 void Print ( const SymmetricMatrix & X )

Definition at line 86 of file tmt.cpp.

30.2175.1.7 void Print ( const LowerTriangularMatrix & X )

Definition at line 104 of file tmt.cpp.

30.2175.1.8 void trymat1 ( )

Definition at line 20 of file tmt1.cpp.

30.2175.1.9 void trymat2 ( )

Definition at line 19 of file tmt2.cpp.

30.2175.1.10 void trymat3 ( )

Definition at line 17 of file tmt3.cpp.

30.2175.1.11 void trymat4 ( )

Definition at line 19 of file tmt4.cpp.

30.2175.1.12 void trymat5 ( )

Definition at line 43 of file tmt5.cpp.

30.2175.1.13 void trymat6 ( )

Definition at line 60 of file tmt6.cpp.

30.2175.1.14 void trymat7 ( )

Definition at line 18 of file tmt7.cpp.

30.2175.1.15 void trymat8 ( )

Definition at line 35 of file tmt8.cpp.

30.2175.1.16 void trymat9 ( )

Definition at line 16 of file tmt9.cpp.

30.2175.1.17 void trymata ( )

Definition at line 32 of file tmta.cpp.

30.2175.1.18 void trymatb ( )

Definition at line 43 of file tmtb.cpp.

30.2175.1.19 void trymatc ( )

Definition at line 17 of file tmtc.cpp.

30.2175.1.20 void trymatd ( )

Definition at line 21 of file tmtd.cpp.

30.2175.1.21 void trymate ( )

Definition at line 26 of file tmte.cpp.

30.2175.1.22 void trymatf ( )

Definition at line 295 of file tmtf.cpp.

30.2175.1.23 void trymatg ( )

Definition at line 18 of file tmtg.cpp.

30.2175.1.24 void trymath ( )

Definition at line 365 of file tmth.cpp.

30.2175.1.25 void trymati ( )

Definition at line 26 of file tmti.cpp.

30.2175.1.26 void trymatj ( )

Definition at line 16 of file tmtj.cpp.

30.2175.1.27 void trymatk ( )

Definition at line 180 of file tmtk.cpp.

30.2175.1.28 void trymatl ( )

Definition at line 132 of file tmtl.cpp.

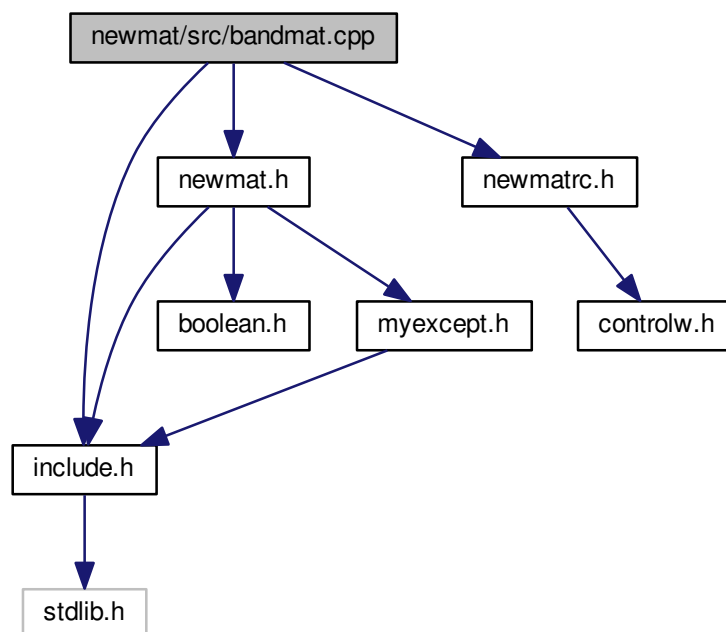
30.2175.1.29 void trymatm ( )

Definition at line 19 of file tmtm.cpp.

## 30.2176 newmat/src/bandmat.cpp File Reference

```
#include "include.h"
#include "newmat.h"
#include "newmatrc.h"
```

Include dependency graph for bandmat.cpp:



### Macros

- #define `WANT_MATH`
- #define `REPORT {}`

## Functions

- Real [square](#) (Real x)

### 30.2176.1 Macro Definition Documentation

#### 30.2176.1.1 `#define REPORT {}`

Definition at line 23 of file bandmat.cpp.

#### 30.2176.1.2 `#define WANT_MATH`

Definition at line 5 of file bandmat.cpp.

### 30.2176.2 Function Documentation

#### 30.2176.2.1 `Real square ( Real x )` `[inline]`

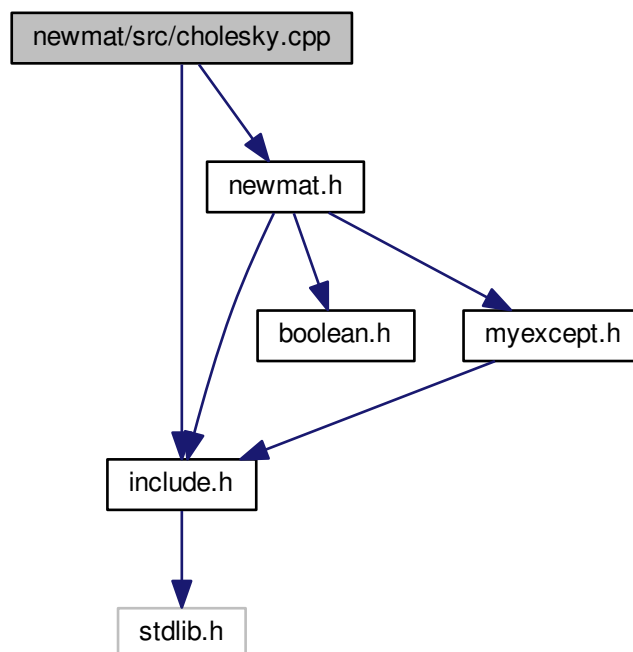
Definition at line 532 of file bandmat.cpp.

## 30.2177 newmat/src/cholesky.cpp File Reference

```
#include "include.h"
```

```
#include "newmat.h"
```

Include dependency graph for cholesky.cpp:





## Macros

- #define [WANT\\_MATH](#)
- #define [REPORT](#) {}

## Functions

- Real [square](#) (Real x)
- [ReturnMatrix Cholesky](#) (const SymmetricMatrix &S)
- [ReturnMatrix Cholesky](#) (const SymmetricBandMatrix &S)

### 30.2177.1 Macro Definition Documentation

#### 30.2177.1.1 #define REPORT {}

Definition at line 19 of file cholesky.cpp.

#### 30.2177.1.2 #define WANT\_MATH

Definition at line 5 of file cholesky.cpp.

### 30.2177.2 Function Documentation

#### 30.2177.2.1 ReturnMatrix Cholesky ( const SymmetricMatrix & S )

Definition at line 29 of file cholesky.cpp.

#### 30.2177.2.2 ReturnMatrix Cholesky ( const SymmetricBandMatrix & S )

Definition at line 54 of file cholesky.cpp.

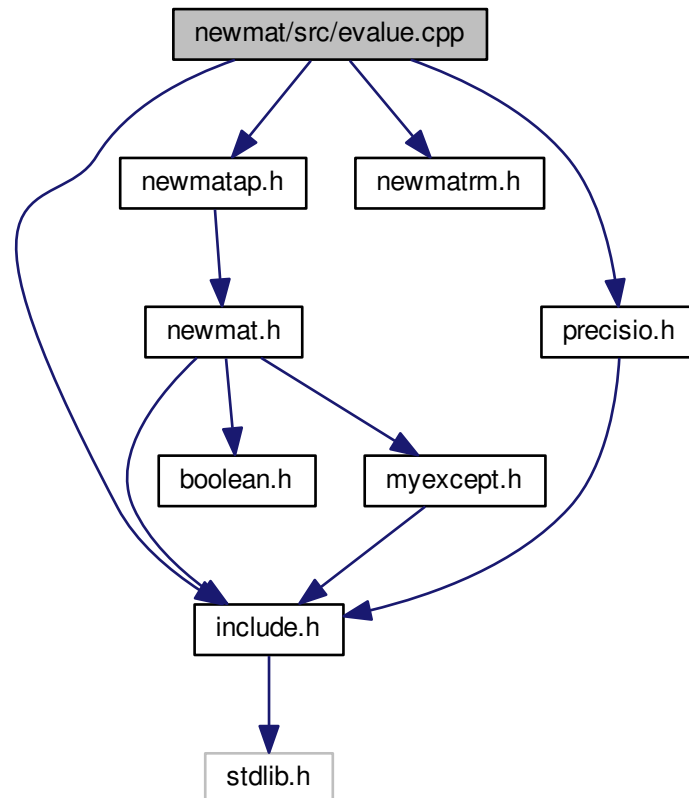
#### 30.2177.2.3 Real square ( Real x ) [inline]

Definition at line 27 of file cholesky.cpp.

## 30.2178 newmat/src/evalvalue.cpp File Reference

```
#include "include.h"
#include "newmatap.h"
#include "newmatrm.h"
#include "precisio.h"
```

Include dependency graph for evalvalue.cpp:



### Macros

- `#define WANT_MATH`
- `#define REPORT {}`

### Functions

- void `EigenValues` (`const` SymmetricMatrix &A, DiagonalMatrix &D, Matrix &Z)
- void `EigenValues` (`const` SymmetricMatrix &X, DiagonalMatrix &D)
- void `EigenValues` (`const` SymmetricMatrix &X, DiagonalMatrix &D, SymmetricMatrix &A)

### 30.2178.1 Macro Definition Documentation

#### 30.2178.1.1 #define REPORT {}

Definition at line 19 of file evaluate.cpp.

#### 30.2178.1.2 #define WANT\_MATH

Definition at line 5 of file evaluate.cpp.

### 30.2178.2 Function Documentation

#### 30.2178.2.1 void EigenValues ( const SymmetricMatrix & A, DiagonalMatrix & D, Matrix & Z )

Definition at line 283 of file evaluate.cpp.

#### 30.2178.2.2 void EigenValues ( const SymmetricMatrix & X, DiagonalMatrix & D )

Definition at line 286 of file evaluate.cpp.

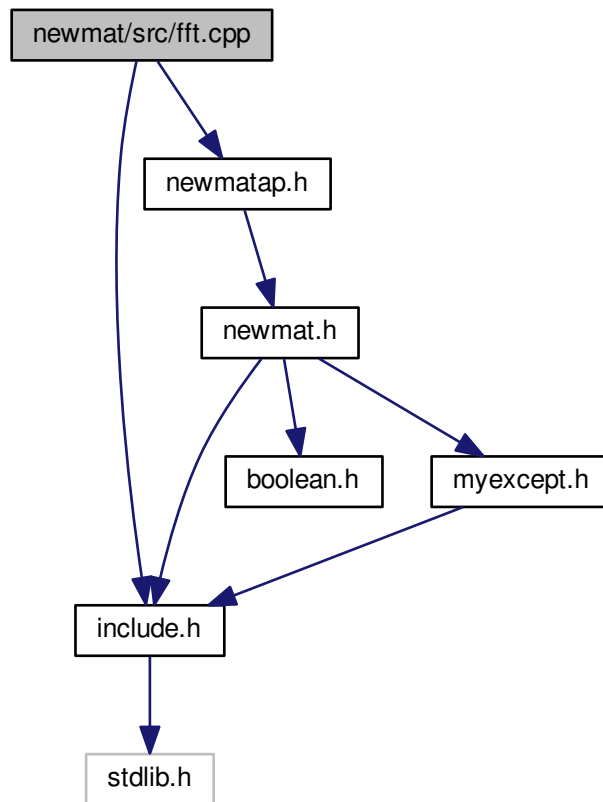
#### 30.2178.2.3 void EigenValues ( const SymmetricMatrix & X, DiagonalMatrix & D, SymmetricMatrix & A )

Definition at line 289 of file evaluate.cpp.

### 30.2179 newmat/src/fft.cpp File Reference

```
#include "include.h"
#include "newmatap.h"
```

Include dependency graph for fft.cpp:



## Macros

- `#define WANT_MATH`
- `#define REPORT {}`

## Functions

- void `FFTI` (`const` ColumnVector &U, `const` ColumnVector &V, ColumnVector &X, ColumnVector &Y)
- void `RealFFT` (`const` ColumnVector &U, ColumnVector &X, ColumnVector &Y)
- void `RealFFTI` (`const` ColumnVector &A, `const` ColumnVector &B, ColumnVector &U)
- void `FFT` (`const` ColumnVector &U, `const` ColumnVector &V, ColumnVector &X, ColumnVector &Y)
- void `DCT_II` (`const` ColumnVector &U, ColumnVector &V)
- void `DCT_II_inverse` (`const` ColumnVector &V, ColumnVector &U)
- void `DST_II` (`const` ColumnVector &U, ColumnVector &V)
- void `DST_II_inverse` (`const` ColumnVector &V, ColumnVector &U)
- void `DCT_inverse` (`const` ColumnVector &V, ColumnVector &U)
- void `DCT` (`const` ColumnVector &U, ColumnVector &V)
- void `DST_inverse` (`const` ColumnVector &V, ColumnVector &U)
- void `DST` (`const` ColumnVector &U, ColumnVector &V)

## 30.2179.1 Macro Definition Documentation

### 30.2179.1.1 #define REPORT {}

Definition at line 22 of file fft.cpp.

### 30.2179.1.2 #define WANT\_MATH

Definition at line 6 of file fft.cpp.

## 30.2179.2 Function Documentation

### 30.2179.2.1 void DCT ( const ColumnVector & *U*, ColumnVector & *V* )

Definition at line 398 of file fft.cpp.

### 30.2179.2.2 void DCT\_II ( const ColumnVector & *U*, ColumnVector & *V* )

Definition at line 254 of file fft.cpp.

### 30.2179.2.3 void DCT\_II\_inverse ( const ColumnVector & *V*, ColumnVector & *U* )

Definition at line 282 of file fft.cpp.

### 30.2179.2.4 void DCT\_inverse ( const ColumnVector & *V*, ColumnVector & *U* )

Definition at line 364 of file fft.cpp.

### 30.2179.2.5 void DST ( const ColumnVector & *U*, ColumnVector & *V* )

Definition at line 435 of file fft.cpp.

### 30.2179.2.6 void DST\_II ( const ColumnVector & *U*, ColumnVector & *V* )

Definition at line 309 of file fft.cpp.

### 30.2179.2.7 void DST\_II\_inverse ( const ColumnVector & *V*, ColumnVector & *U* )

Definition at line 337 of file fft.cpp.

30.2179.2.8 void DST\_inverse ( const ColumnVector & V, ColumnVector & U )

Definition at line 407 of file fft.cpp.

30.2179.2.9 void FFT ( const ColumnVector & U, const ColumnVector & V, ColumnVector & X, ColumnVector & Y )

Definition at line 196 of file fft.cpp.

30.2179.2.10 void FFTI ( const ColumnVector & U, const ColumnVector & V, ColumnVector & X, ColumnVector & Y )

Definition at line 115 of file fft.cpp.

30.2179.2.11 void RealFFT ( const ColumnVector & U, ColumnVector & X, ColumnVector & Y )

Definition at line 125 of file fft.cpp.

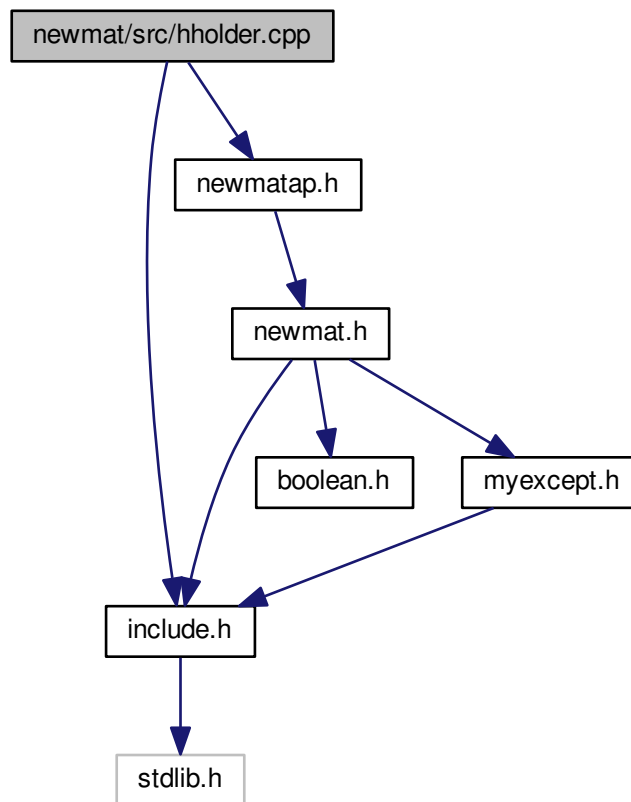
30.2179.2.12 void RealFFTI ( const ColumnVector & A, const ColumnVector & B, ColumnVector & U )

Definition at line 161 of file fft.cpp.

## 30.2180 newmat/src/hholder.cpp File Reference

```
#include "include.h"
#include "newmatap.h"
```

Include dependency graph for hholder.cpp:



## Macros

- `#define WANT_MATH`
- `#define REPORT {}`

## Functions

- Real `square` (Real `x`)
- void `QRZT` (Matrix &`X`, LowerTriangularMatrix &`L`)
- void `QRZT` (`const` Matrix &`X`, Matrix &`Y`, Matrix &`M`)
- void `QRZ` (Matrix &`X`, UpperTriangularMatrix &`U`)
- void `QRZ` (`const` Matrix &`X`, Matrix &`Y`, Matrix &`M`)

### 30.2180.1 Macro Definition Documentation

#### 30.2180.1.1 `#define REPORT {}`

Definition at line 18 of file `hholder.cpp`.

30.2180.1.2 `#define WANT_MATH`

Definition at line 5 of file `holder.cpp`.

## 30.2180.2 Function Documentation

30.2180.2.1 `void QRZ ( Matrix & X, UpperTriangularMatrix & U )`

Definition at line 101 of file `holder.cpp`.

30.2180.2.2 `void QRZ ( const Matrix & X, Matrix & Y, Matrix & M )`

Definition at line 136 of file `holder.cpp`.

30.2180.2.3 `void QRZT ( Matrix & X, LowerTriangularMatrix & L )`

Definition at line 26 of file `holder.cpp`.

30.2180.2.4 `void QRZT ( const Matrix & X, Matrix & Y, Matrix & M )`

Definition at line 50 of file `holder.cpp`.

30.2180.2.5 `Real square ( Real x ) [inline]`

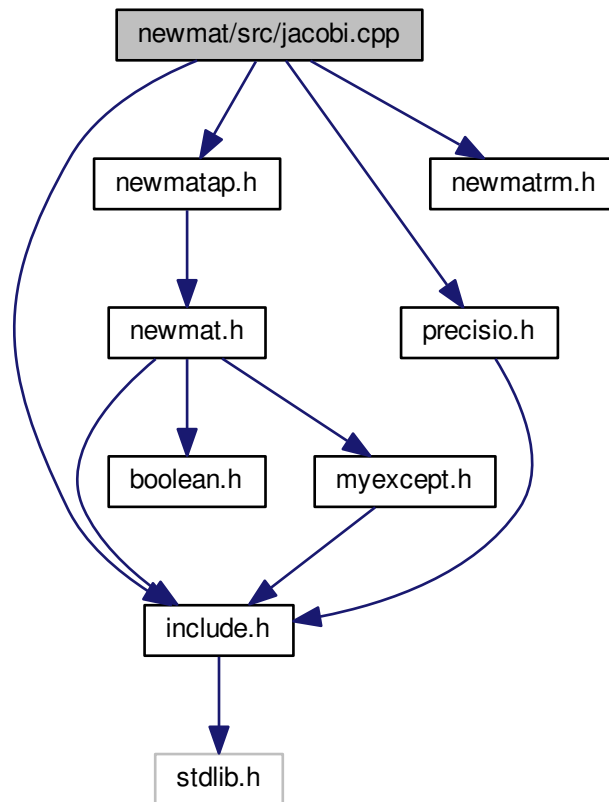
Definition at line 24 of file `holder.cpp`.

## 30.2181 `newmat/src/jacobi.cpp` File Reference

```
#include "include.h"
#include "newmatap.h"
#include "precisio.h"
#include "newmatrm.h"
```



Include dependency graph for jacobi.cpp:



## Macros

- `#define WANT_MATH`
- `#define REPORT {}`

## Functions

- void `Jacobi` (`const` SymmetricMatrix &X, DiagonalMatrix &D, SymmetricMatrix &A, Matrix &V, `bool` eivec)
- void `Jacobi` (`const` SymmetricMatrix &X, DiagonalMatrix &D)
- void `Jacobi` (`const` SymmetricMatrix &X, DiagonalMatrix &D, SymmetricMatrix &A)
- void `Jacobi` (`const` SymmetricMatrix &X, DiagonalMatrix &D, Matrix &V)

### 30.2181.1 Macro Definition Documentation

#### 30.2181.1.1 `#define REPORT {}`

Definition at line 23 of file `jacobi.cpp`.

30.2181.1.2 `#define WANT_MATH`

Definition at line 9 of file `jacobi.cpp`.

### 30.2181.2 Function Documentation

30.2181.2.1 `void Jacobi ( const SymmetricMatrix & X, DiagonalMatrix & D, SymmetricMatrix & A, Matrix & V, bool eivec )`

Definition at line 27 of file `jacobi.cpp`.

30.2181.2.2 `void Jacobi ( const SymmetricMatrix & X, DiagonalMatrix & D )`

Definition at line 110 of file `jacobi.cpp`.

30.2181.2.3 `void Jacobi ( const SymmetricMatrix & X, DiagonalMatrix & D, SymmetricMatrix & A )`

Definition at line 113 of file `jacobi.cpp`.

30.2181.2.4 `void Jacobi ( const SymmetricMatrix & X, DiagonalMatrix & D, Matrix & V )`

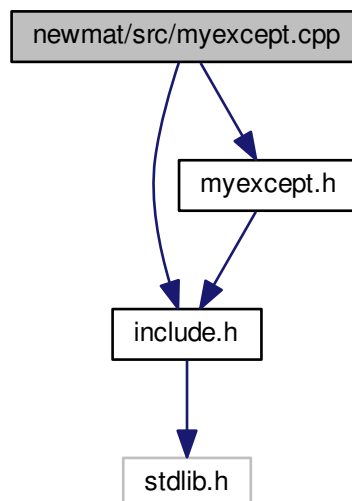
Definition at line 116 of file `jacobi.cpp`.

## 30.2182 `newmat/src/myexcept.cpp` File Reference

```
#include "include.h"
```

```
#include "myexcept.h"
```

Include dependency graph for `myexcept.cpp`:



- `#define WANT_STREAM`
- `#define WANT_STRING`
- `void Terminate ()`

### 30.2182.1 Detailed Description

Exception handler. The low level classes for

- my exception class hierarchy
- the functions needed for my simulated exceptions
- the Tracer mechanism
- routines for checking whether new and delete calls are balanced

### 30.2182.2 Macro Definition Documentation

#### 30.2182.2.1 #define WANT\_STREAM

Definition at line 16 of file myexcept.cpp.

#### 30.2182.2.2 #define WANT\_STRING

Definition at line 17 of file myexcept.cpp.

### 30.2182.3 Function Documentation

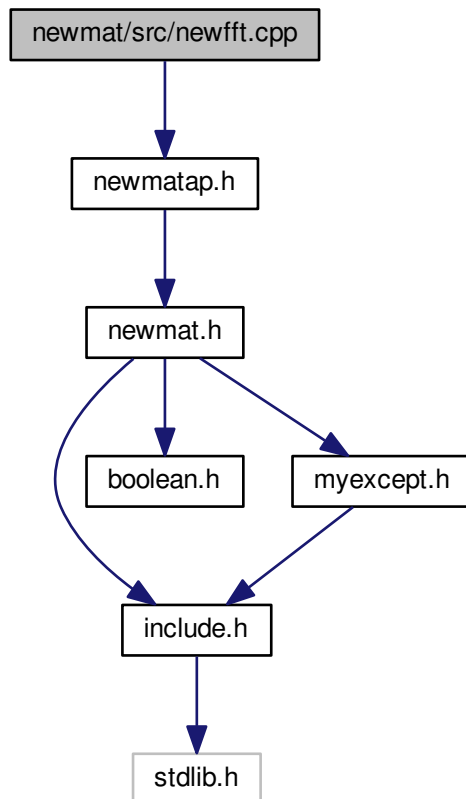
#### 30.2182.3.1 void Terminate ( )

Definition at line 226 of file myexcept.cpp.

## 30.2183 newmat/src/newfft.cpp File Reference

```
#include "newmatap.h"
```

Include dependency graph for newfft.cpp:



### Macros

- #define [WANT\\_STREAM](#)
- #define [WANT\\_MATH](#)
- #define [REPORT](#) {}

### Functions

- Real [square](#) (Real x)
- int [square](#) (int x)

### 30.2183.1 Macro Definition Documentation

#### 30.2183.1.1 #define [REPORT](#) {}

Definition at line 107 of file `newfft.cpp`.

30.2183.1.2 `#define WANT_MATH`

Definition at line 96 of file newfft.cpp.

30.2183.1.3 `#define WANT_STREAM`

Definition at line 94 of file newfft.cpp.

## 30.2183.2 Function Documentation

30.2183.2.1 `Real square ( Real x ) [inline]`

Definition at line 110 of file newfft.cpp.

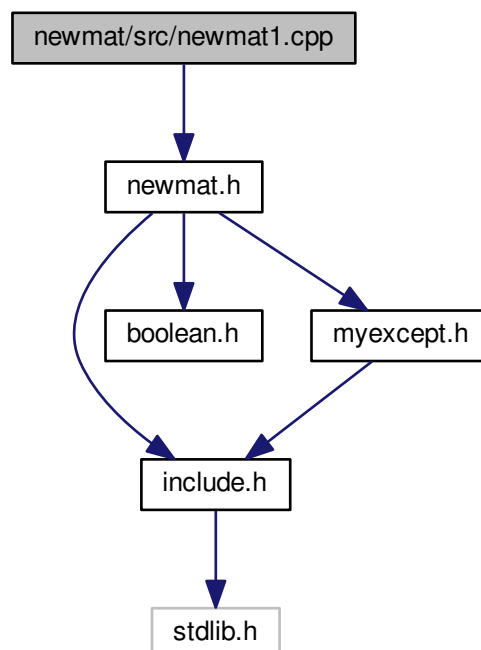
30.2183.2.2 `int square ( int x ) [inline]`

Definition at line 111 of file newfft.cpp.

## 30.2184 newmat/src/newmat1.cpp File Reference

```
#include "newmat.h"
```

Include dependency graph for newmat1.cpp:



## Macros

- `#define REPORT {}`

## Functions

- `bool Rectangular (MatrixType a, MatrixType b, MatrixType c)`

### 30.2184.1 Macro Definition Documentation

#### 30.2184.1.1 `#define REPORT {}`

Definition at line 16 of file newmat1.cpp.

### 30.2184.2 Function Documentation

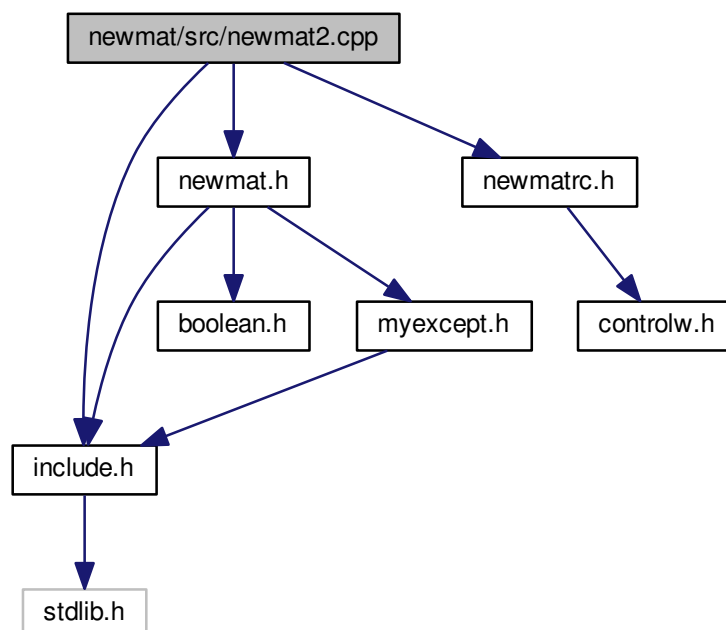
#### 30.2184.2.1 `bool Rectangular ( MatrixType a, MatrixType b, MatrixType c )`

Definition at line 88 of file newmat1.cpp.

## 30.2185 newmat/src/newmat2.cpp File Reference

```
#include "include.h"
#include "newmat.h"
#include "newmatrc.h"
```

Include dependency graph for newmat2.cpp:



## Macros

- #define [WANT\\_MATH](#)
- #define [REPORT](#) {}
- #define [MONITOR](#)(*what*, *store*, *storage*) {}

## Functions

- Real [DotProd](#) ([const MatrixRowCol](#) &*mrc1*, [const MatrixRowCol](#) &*mrc2*)

### 30.2185.1 Macro Definition Documentation

#### 30.2185.1.1 #define [MONITOR](#)( *what*, *store*, *storage* ) {}

Definition at line 25 of file newmat2.cpp.

#### 30.2185.1.2 #define [REPORT](#) {}

Definition at line 20 of file newmat2.cpp.

#### 30.2185.1.3 #define [WANT\\_MATH](#)

Definition at line 5 of file newmat2.cpp.

### 30.2185.2 Function Documentation

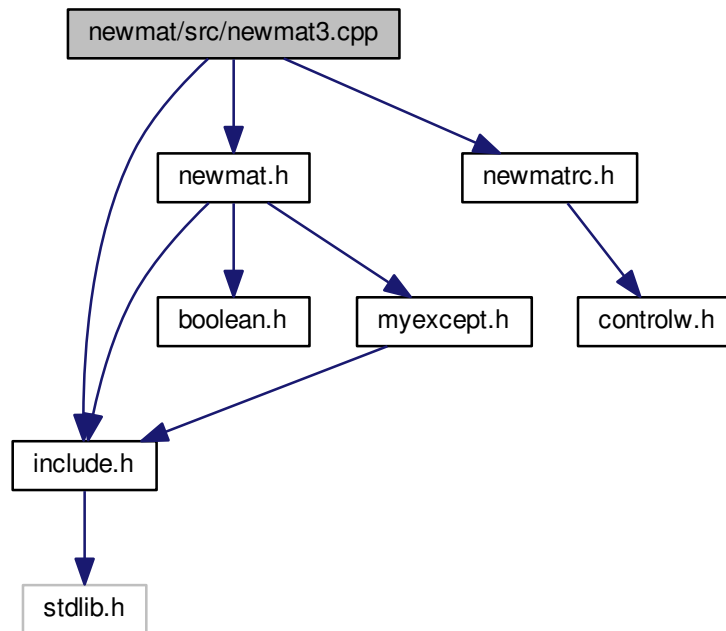
#### 30.2185.2.1 Real [DotProd](#) ( [const MatrixRowCol](#) & *mrc1*, [const MatrixRowCol](#) & *mrc2* )

Definition at line 73 of file newmat2.cpp.

## 30.2186 newmat/src/newmat3.cpp File Reference

```
#include "include.h"
#include "newmat.h"
#include "newmatrc.h"
```

Include dependency graph for newmat3.cpp:



### Macros

- `#define REPORT {}`
- `#define MONITOR(what, store, storage) {}`

### 30.2186.1 Macro Definition Documentation

#### 30.2186.1.1 `#define MONITOR( what, store, storage ) {}`

Definition at line 26 of file `newmat3.cpp`.

#### 30.2186.1.2 `#define REPORT {}`

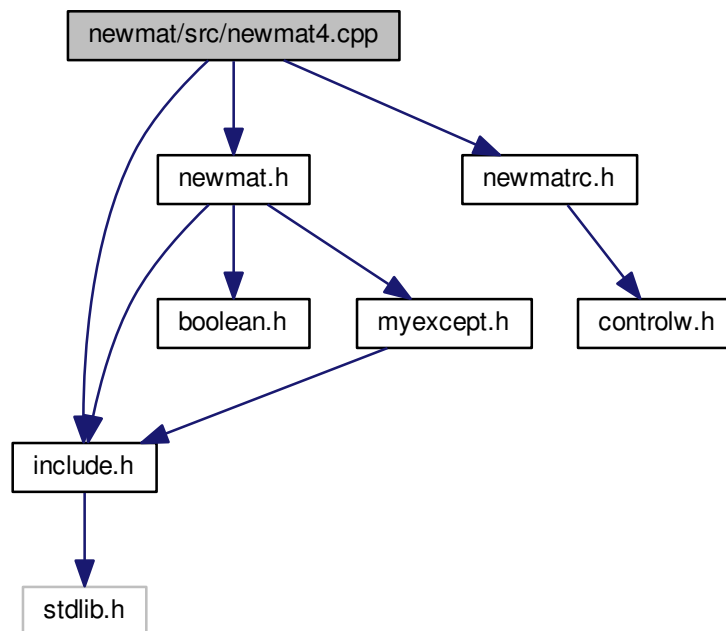
Definition at line 20 of file `newmat3.cpp`.



## 30.2187 newmat/src/newmat4.cpp File Reference

```
#include "include.h"
#include "newmat.h"
#include "newmatrc.h"
```

Include dependency graph for newmat4.cpp:



### Macros

- `#define REPORT {}`
- `#define DO_SEARCH`

### Functions

- `bool Compare (const MatrixType &source, MatrixType &destination)`

#### 30.2187.1 Macro Definition Documentation

##### 30.2187.1.1 `#define DO_SEARCH`

Definition at line 23 of file newmat4.cpp.

30.2187.1.2 `#define REPORT {}`

Definition at line 19 of file newmat4.cpp.

## 30.2187.2 Function Documentation

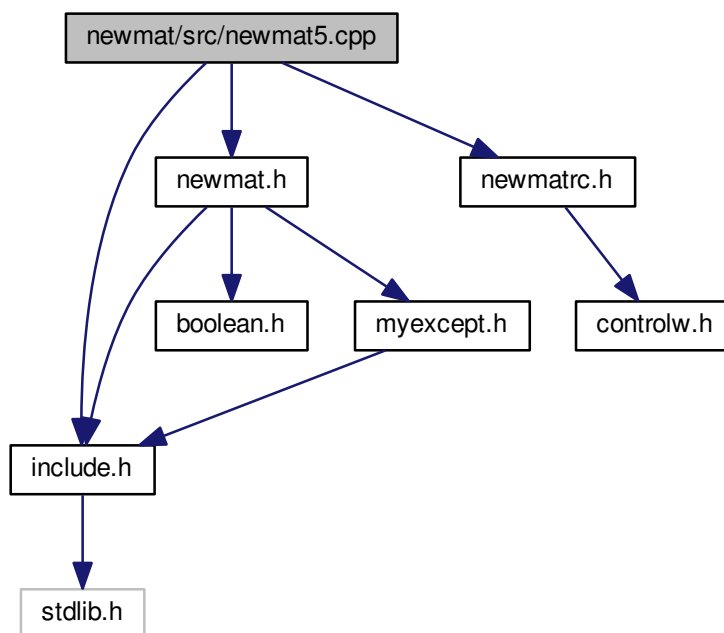
30.2187.2.1 `bool Compare ( const MatrixType & source, MatrixType & destination )`

Definition at line 681 of file newmat4.cpp.

## 30.2188 newmat/src/newmat5.cpp File Reference

```
#include "include.h"
#include "newmat.h"
#include "newmatrc.h"
```

Include dependency graph for newmat5.cpp:



## Macros

- `#define REPORT {}`

### 30.2188.1 Macro Definition Documentation

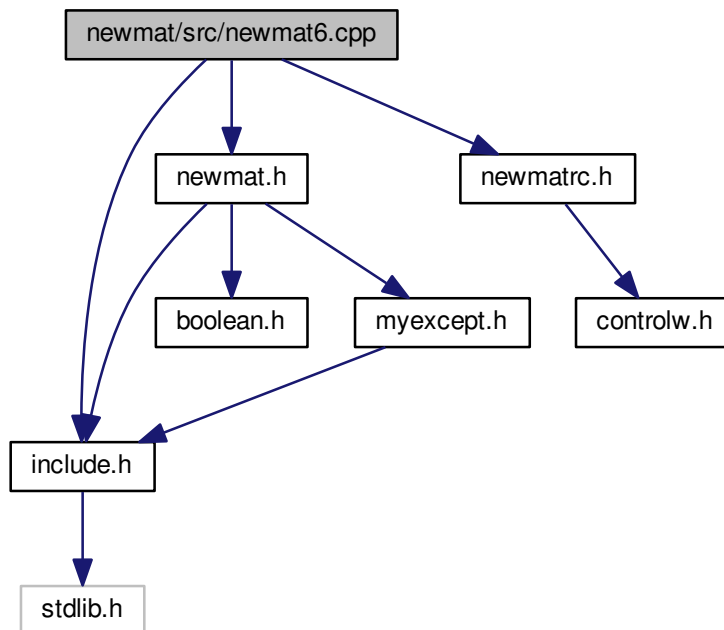
#### 30.2188.1.1 #define REPORT {}

Definition at line 20 of file newmat5.cpp.

### 30.2189 newmat/src/newmat6.cpp File Reference

```
#include "include.h"
#include "newmat.h"
#include "newmatrc.h"
```

Include dependency graph for newmat6.cpp:



#### Macros

- #define [REPORT](#) {}

#### Functions

- SPMatrix [SP](#) ([const](#) BaseMatrix &bm1, [const](#) BaseMatrix &bm2)
- KPMatrix [KP](#) ([const](#) BaseMatrix &bm1, [const](#) BaseMatrix &bm2)
- NegShiftedMatrix [operator-](#) (Real f, [const](#) BaseMatrix &bm)

### 30.2189.1 Macro Definition Documentation

#### 30.2189.1.1 #define REPORT {}

Definition at line 19 of file newmat6.cpp.

### 30.2189.2 Function Documentation

#### 30.2189.2.1 KPMatrix KP ( const BaseMatrix & bm1, const BaseMatrix & bm2 )

Definition at line 451 of file newmat6.cpp.

#### 30.2189.2.2 NegShiftedMatrix operator- ( Real f, const BaseMatrix & bm )

Definition at line 472 of file newmat6.cpp.

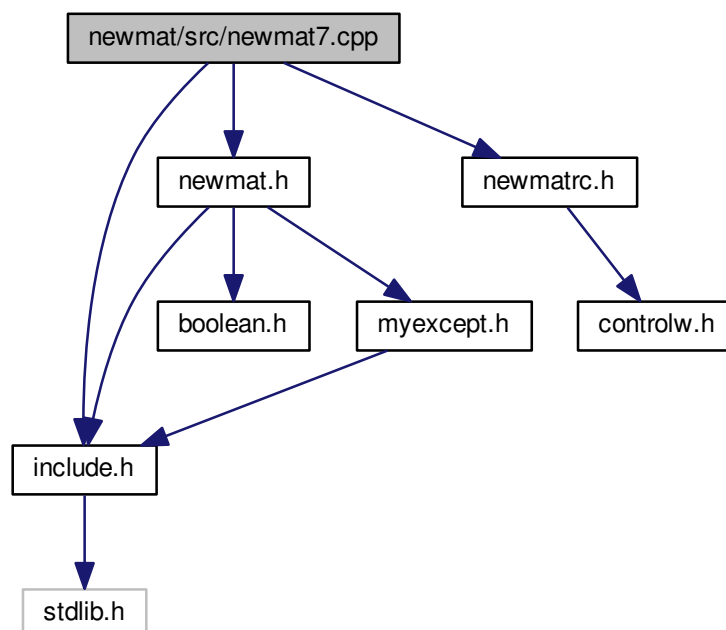
#### 30.2189.2.3 SPMatrix SP ( const BaseMatrix & bm1, const BaseMatrix & bm2 )

Definition at line 448 of file newmat6.cpp.

## 30.2190 newmat/src/newmat7.cpp File Reference

```
#include "include.h"
#include "newmat.h"
#include "newmatrc.h"
```

Include dependency graph for newmat7.cpp:



## Macros

- `#define REPORT {}`

## Functions

- `bool operator==(const BaseMatrix &A, const BaseMatrix &B)`
- `bool operator==(const GeneralMatrix &A, const GeneralMatrix &B)`
- `bool IsZero(const BaseMatrix &A)`

### 30.2190.1 Macro Definition Documentation

#### 30.2190.1.1 `#define REPORT {}`

Definition at line 18 of file newmat7.cpp.

### 30.2190.2 Function Documentation

#### 30.2190.2.1 `bool IsZero ( const BaseMatrix & A )`

Definition at line 982 of file newmat7.cpp.

#### 30.2190.2.2 `bool operator==( const BaseMatrix & A, const BaseMatrix & B )`

Definition at line 905 of file newmat7.cpp.

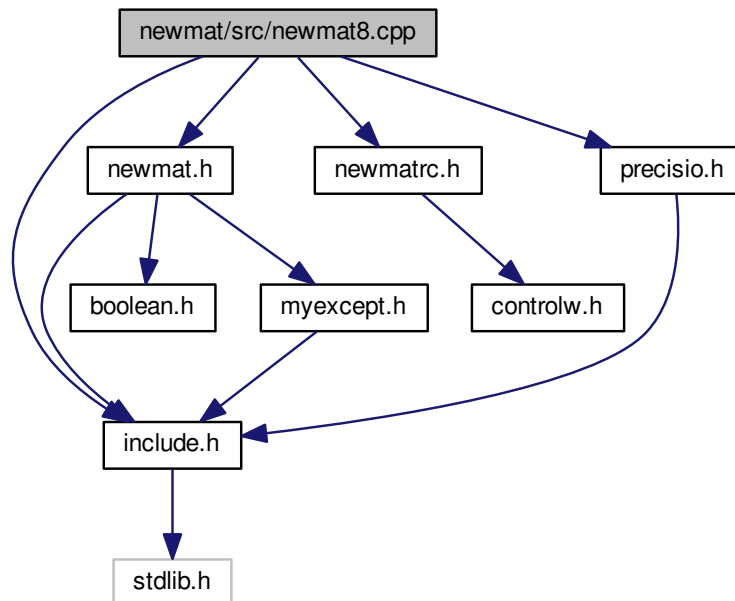
#### 30.2190.2.3 `bool operator==( const GeneralMatrix & A, const GeneralMatrix & B )`

Definition at line 943 of file newmat7.cpp.

## 30.2191 newmat/src/newmat8.cpp File Reference

```
#include "include.h"
#include "newmat.h"
#include "newmatrc.h"
#include "precisio.h"
```

Include dependency graph for newmat8.cpp:



### Macros

- `#define WANT_MATH`
- `#define REPORT {}`

### Functions

- Real `square` (Real x)
- Real `DotProduct` (const Matrix &A, const Matrix &B)

### 30.2191.1 Macro Definition Documentation

#### 30.2191.1.1 `#define REPORT {}`

Definition at line 21 of file `newmat8.cpp`.

30.2191.1.2 `#define WANT_MATH`

Definition at line 5 of file newmat8.cpp.

## 30.2191.2 Function Documentation

30.2191.2.1 `Real DotProduct ( const Matrix & A, const Matrix & B )`

Definition at line 524 of file newmat8.cpp.

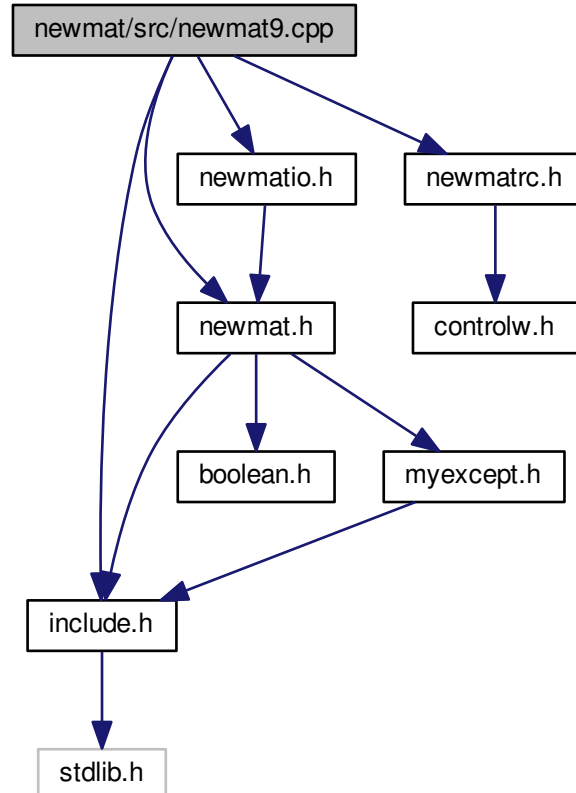
30.2191.2.2 `Real square ( Real x ) [inline]`

Definition at line 148 of file newmat8.cpp.

## 30.2192 newmat/src/newmat9.cpp File Reference

```
#include "include.h"
#include "newmat.h"
#include "newmatio.h"
#include "newmatrc.h"
```

Include dependency graph for newmat9.cpp:



## Macros

- #define `WANT_FSTREAM`
- #define `REPORT {}`
- #define `ios_format_flags` long

## Functions

- ostream & `operator<<` (ostream &s, const BaseMatrix &X)
- ostream & `operator<<` (ostream &s, const GeneralMatrix &X)

### 30.2192.1 Macro Definition Documentation

#### 30.2192.1.1 #define ios\_format\_flags long

Definition at line 28 of file newmat9.cpp.

#### 30.2192.1.2 #define REPORT {}

Definition at line 23 of file newmat9.cpp.

#### 30.2192.1.3 #define WANT\_FSTREAM

Definition at line 6 of file newmat9.cpp.

### 30.2192.2 Function Documentation

#### 30.2192.2.1 ostream& operator<< ( ostream & s, const BaseMatrix & X )

Definition at line 31 of file newmat9.cpp.

#### 30.2192.2.2 ostream& operator<< ( ostream & s, const GeneralMatrix & X )

Definition at line 38 of file newmat9.cpp.

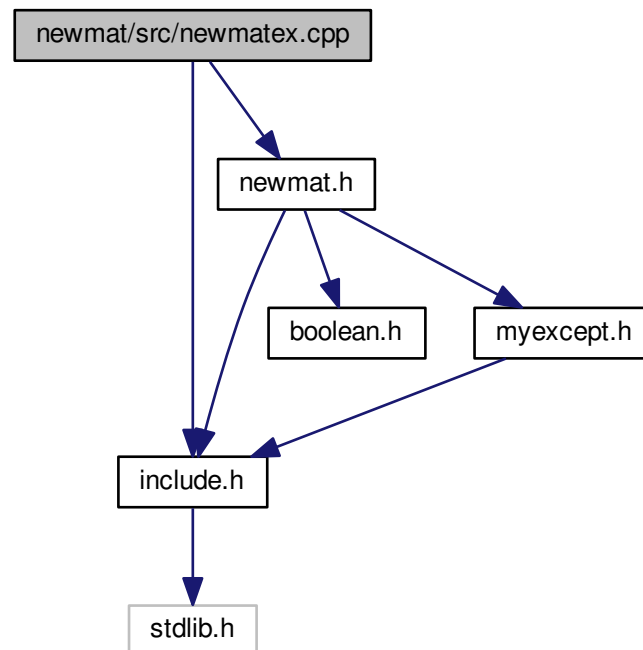


## 30.2193 newmat/src/newmatex.cpp File Reference

```
#include "include.h"
```

```
#include "newmat.h"
```

Include dependency graph for newmatex.cpp:



### Macros

- `#define WANT_STREAM`

### Functions

- `void MatrixErrorNoSpace (void *v)`

### 30.2193.1 Macro Definition Documentation

#### 30.2193.1.1 `#define WANT_STREAM`

Definition at line 5 of file `newmatex.cpp`.

## 30.2193.2 Function Documentation

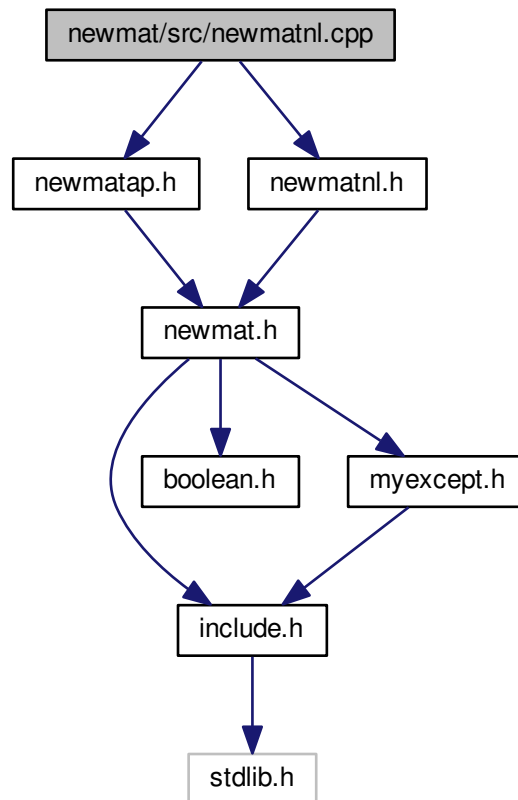
### 30.2193.2.1 void MatrixErrorNoSpace ( void \* v )

Definition at line 275 of file newmatex.cpp.

## 30.2194 newmat/src/newmatnl.cpp File Reference

```
#include "newmatap.h"
#include "newmatnl.h"
```

Include dependency graph for newmatnl.cpp:



### Macros

- #define [WANT\\_MATH](#)
- #define [WANT\\_STREAM](#)

### 30.2194.1 Macro Definition Documentation

#### 30.2194.1.1 #define WANT\_MATH

Definition at line 6 of file newmatnl.cpp.

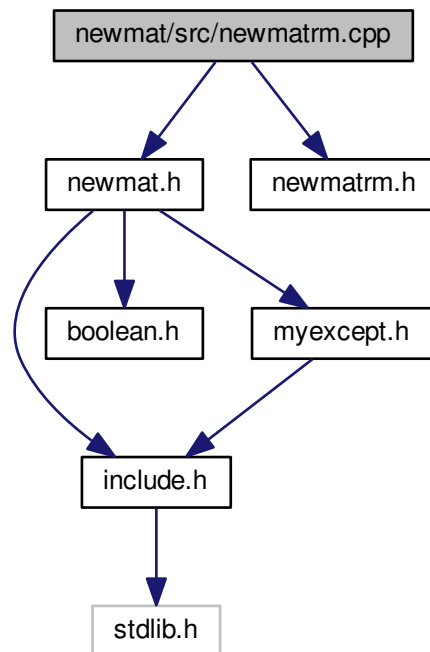
#### 30.2194.1.2 #define WANT\_STREAM

Definition at line 7 of file newmatnl.cpp.

## 30.2195 newmat/src/newmatrm.cpp File Reference

```
#include "newmat.h"
#include "newmatrm.h"
```

Include dependency graph for newmatrm.cpp:



### Macros

- #define `REPORT` {}

## Functions

- void [ComplexScale](#) ([RectMatrixCol](#) &U, [RectMatrixCol](#) &V, Real x, Real y)
- void [Rotate](#) ([RectMatrixCol](#) &U, [RectMatrixCol](#) &V, Real tau, Real s)

### 30.2195.1 Macro Definition Documentation

#### 30.2195.1.1 #define REPORT {}

Definition at line 17 of file newmatrm.cpp.

### 30.2195.2 Function Documentation

#### 30.2195.2.1 void ComplexScale ( [RectMatrixCol](#) & U, [RectMatrixCol](#) & V, Real x, Real y )

Definition at line 131 of file newmatrm.cpp.

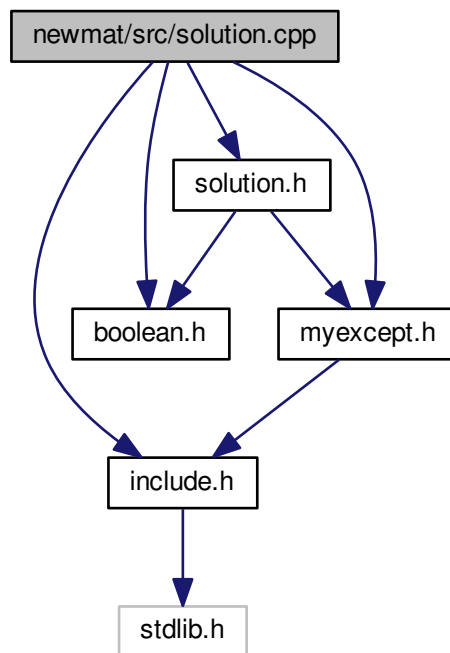
#### 30.2195.2.2 void Rotate ( [RectMatrixCol](#) & U, [RectMatrixCol](#) & V, Real tau, Real s )

Definition at line 155 of file newmatrm.cpp.

## 30.2196 newmat/src/solution.cpp File Reference

```
#include "include.h"
#include "boolean.h"
#include "myexcept.h"
#include "solution.h"
```

Include dependency graph for solution.cpp:



## Macros

- `#define WANT_STREAM`
- `#define WANT_MATH`

## Functions

- Real `square` (Real x)

### 30.2196.1 Macro Definition Documentation

#### 30.2196.1.1 `#define WANT_MATH`

Definition at line 7 of file `solution.cpp`.

#### 30.2196.1.2 `#define WANT_STREAM`

Definition at line 6 of file `solution.cpp`.

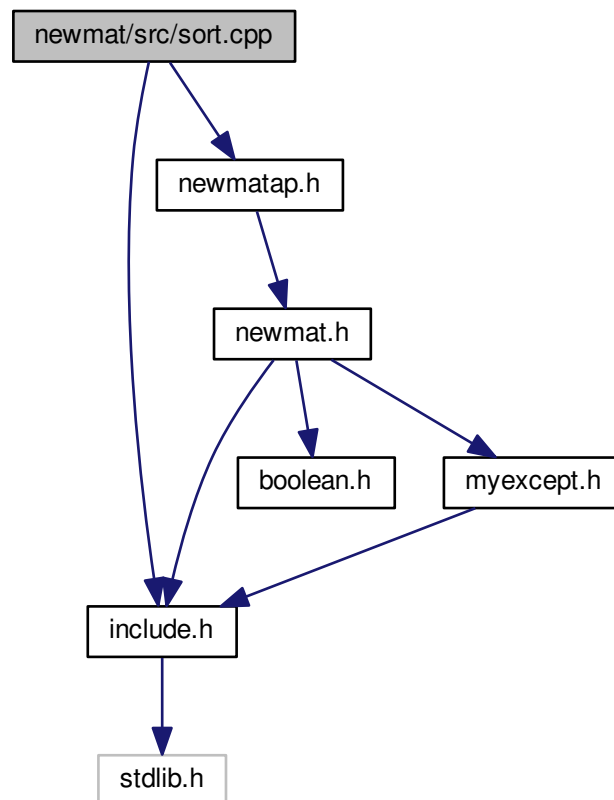
## 30.2196.2 Function Documentation

### 30.2196.2.1 Real square ( Real x ) [inline]

Definition at line 44 of file solution.cpp.

## 30.2197 newmat/src/sort.cpp File Reference

```
#include "include.h"
#include "newmatap.h"
Include dependency graph for sort.cpp:
```



## Macros

- #define [WANT\\_MATH](#)
- #define [REPORT](#) {}
- #define [DoSimpleSort](#) 17
- #define [MaxDepth](#) 50

## Functions

- void [SortDescending](#) (GeneralMatrix &GM)
- void [SortAscending](#) (GeneralMatrix &GM)
- void [SortSV](#) (DiagonalMatrix &D, Matrix &U, [bool](#) ascending)
- void [SortSV](#) (DiagonalMatrix &D, Matrix &U, Matrix &V, [bool](#) ascending)

### 30.2197.1 Macro Definition Documentation

#### 30.2197.1.1 `#define DoSimpleSort 17`

Definition at line 32 of file sort.cpp.

#### 30.2197.1.2 `#define MaxDepth 50`

Definition at line 33 of file sort.cpp.

#### 30.2197.1.3 `#define REPORT {}`

Definition at line 18 of file sort.cpp.

#### 30.2197.1.4 `#define WANT_MATH`

Definition at line 5 of file sort.cpp.

### 30.2197.2 Function Documentation

#### 30.2197.2.1 `void SortAscending ( GeneralMatrix & GM )`

Definition at line 121 of file sort.cpp.

#### 30.2197.2.2 `void SortDescending ( GeneralMatrix & GM )`

Definition at line 45 of file sort.cpp.

#### 30.2197.2.3 `void SortSV ( DiagonalMatrix & D, Matrix & U, bool ascending )`

Definition at line 190 of file sort.cpp.

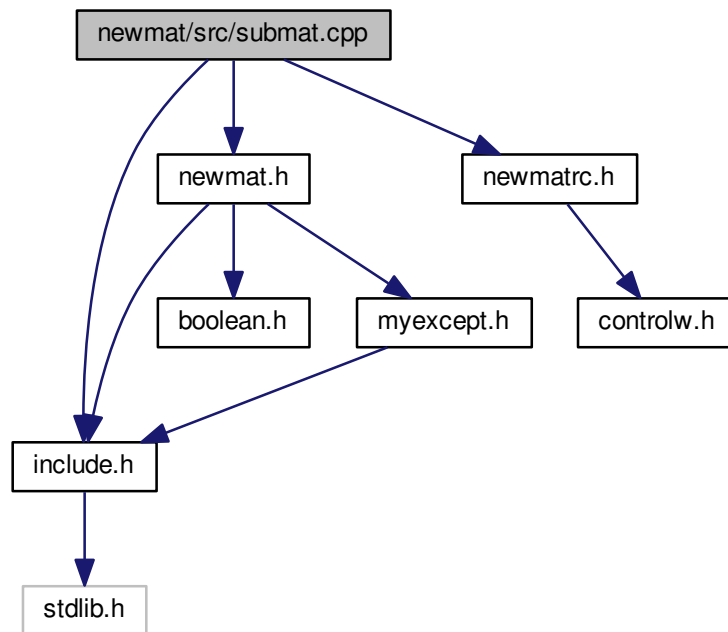
#### 30.2197.2.4 `void SortSV ( DiagonalMatrix & D, Matrix & U, Matrix & V, bool ascending )`

Definition at line 224 of file sort.cpp.

## 30.2198 newmat/src/submat.cpp File Reference

```
#include "include.h"
#include "newmat.h"
#include "newmatrc.h"
```

Include dependency graph for submat.cpp:



### Macros

- `#define REPORT {}`

#### 30.2198.1 Macro Definition Documentation

##### 30.2198.1.1 `#define REPORT {}`

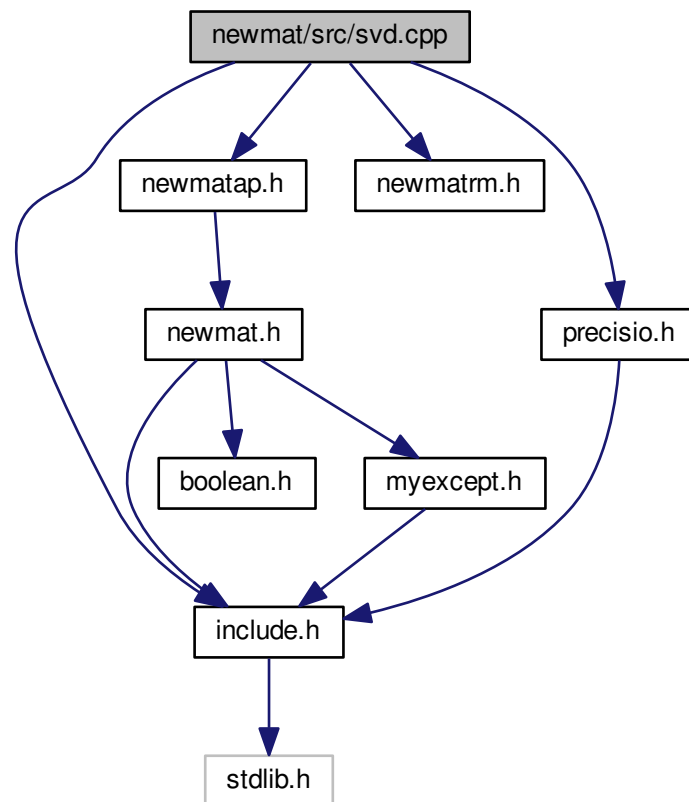
Definition at line 17 of file `submat.cpp`.

## 30.2199 newmat/src/svd.cpp File Reference

```
#include "include.h"
#include "newmatap.h"
#include "newmatrm.h"
#include "precisio.h"
```



Include dependency graph for svd.cpp:



## Macros

- `#define WANT_MATH`
- `#define REPORT {}`

## Functions

- void `SVD` (`const` Matrix &A, DiagonalMatrix &Q, Matrix &U, Matrix &V, `bool` withU, `bool` withV)
- void `SVD` (`const` Matrix &A, DiagonalMatrix &D)

### 30.2199.1 Macro Definition Documentation

#### 30.2199.1.1 `#define REPORT {}`

Definition at line 20 of file `svd.cpp`.

30.2199.1.2 `#define WANT_MATH`

Definition at line 6 of file svd.cpp.

## 30.2199.2 Function Documentation

30.2199.2.1 `void SVD ( const Matrix & A, DiagonalMatrix & Q, Matrix & U, Matrix & V, bool withU, bool withV )`

Definition at line 49 of file svd.cpp.

30.2199.2.2 `void SVD ( const Matrix & A, DiagonalMatrix & D )`

Definition at line 219 of file svd.cpp.

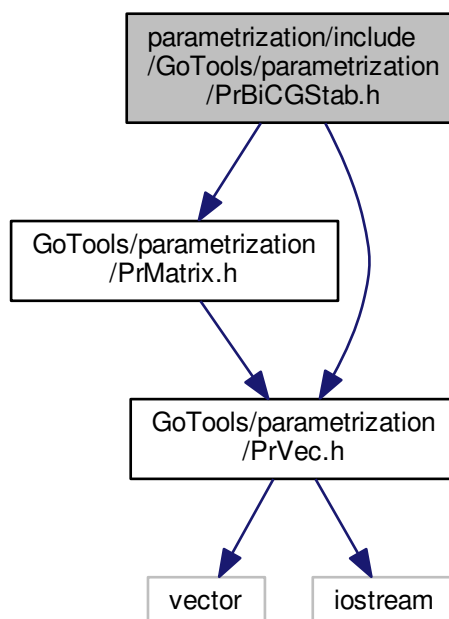
## 30.2200 parametrization/include/GoTools/parametrization/parametrization-doxymain.h File Reference

## 30.2201 parametrization/include/GoTools/parametrization/PrBiCGStab.h File Reference

```
#include "GoTools/parametrization/PrMatrix.h"
```

```
#include "GoTools/parametrization/PrVec.h"
```

Include dependency graph for PrBiCGStab.h:



## Classes

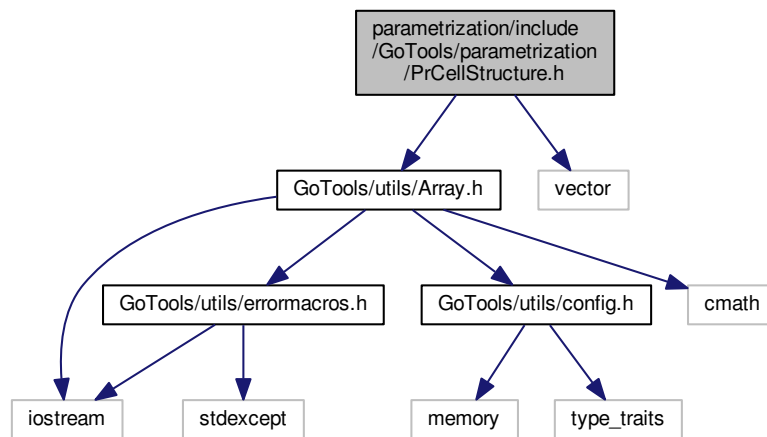
- class [PrBiCGStab](#)

## 30.2202 parametrization/include/GoTools/parametrization/PrCellStructure.h File Reference

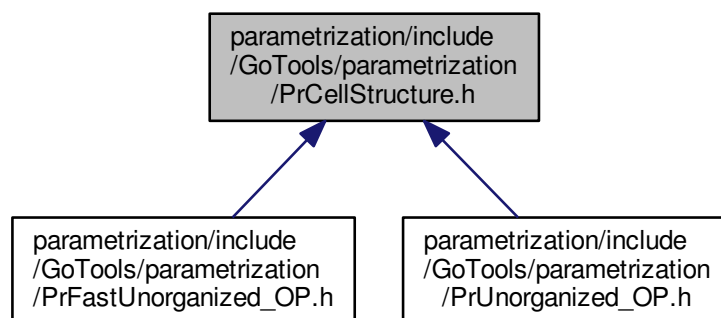
```
#include "GoTools/Utils/Array.h"
```

```
#include <vector>
```

Include dependency graph for PrCellStructure.h:



This graph shows which files directly or indirectly include this file:

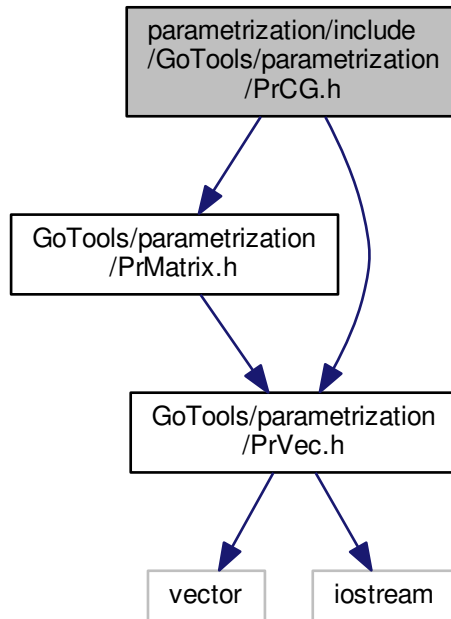


## Classes

- class [PrCellStructure](#)

### 30.2203 parametrization/include/GoTools/parametrization/PrCG.h File Reference

```
#include "GoTools/parametrization/PrMatrix.h"
#include "GoTools/parametrization/PrVec.h"
Include dependency graph for PrCG.h:
```



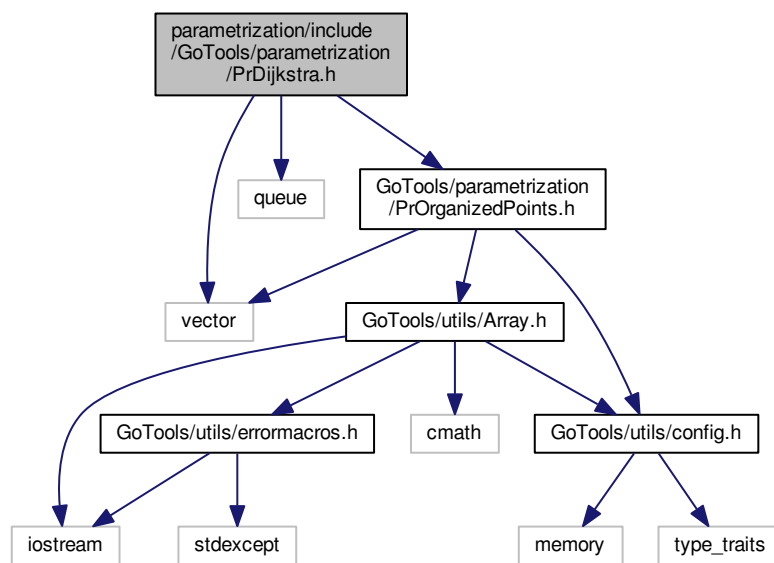
#### Classes

- class [PrCG](#)

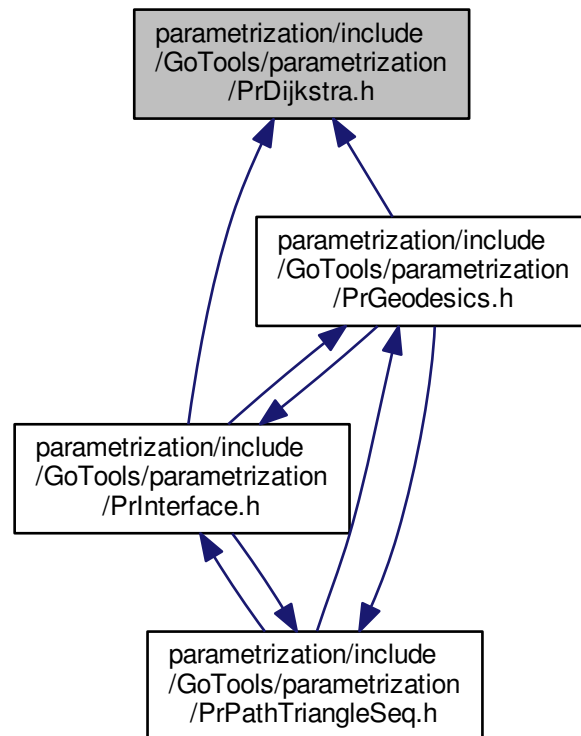
### 30.2204 parametrization/include/GoTools/parametrization/PrDijkstra.h File Reference

```
#include <vector>
#include <queue>
#include "GoTools/parametrization/PrOrganizedPoints.h"
```

Include dependency graph for PrDijkstra.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [HeapNode](#)
- class [Dijkstra](#)

## Macros

- `#define` [GraphType PrOrganizedPoints](#)

## Typedefs

- typedef `std::priority_queue< HeapNode, vector< HeapNode >, std::greater< HeapNode > >` [HeapType](#)

### 30.2204.1 Macro Definition Documentation

#### 30.2204.1.1 `#define` [GraphType PrOrganizedPoints](#)

Definition at line 46 of file `PrDijkstra.h`.

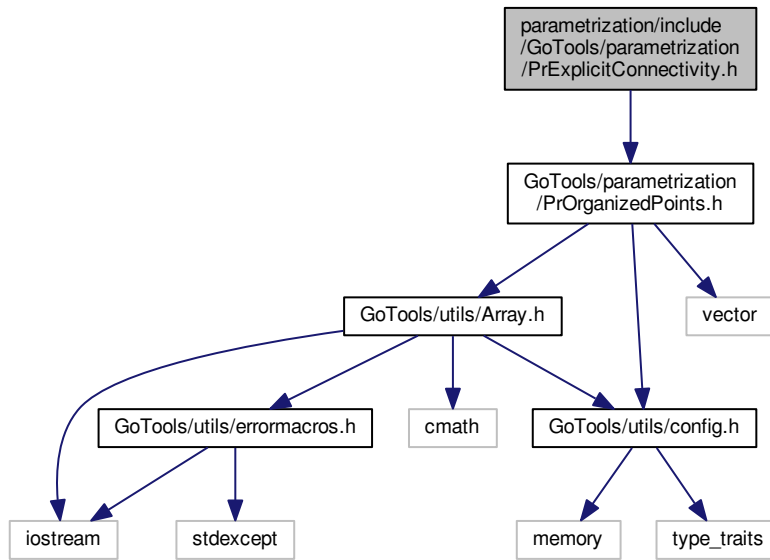
### 30.2204.2 Typedef Documentation

30.2204.2.1 typedef std::priority\_queue< HeapNode, vector<HeapNode>, std::greater<HeapNode>> > HeapType

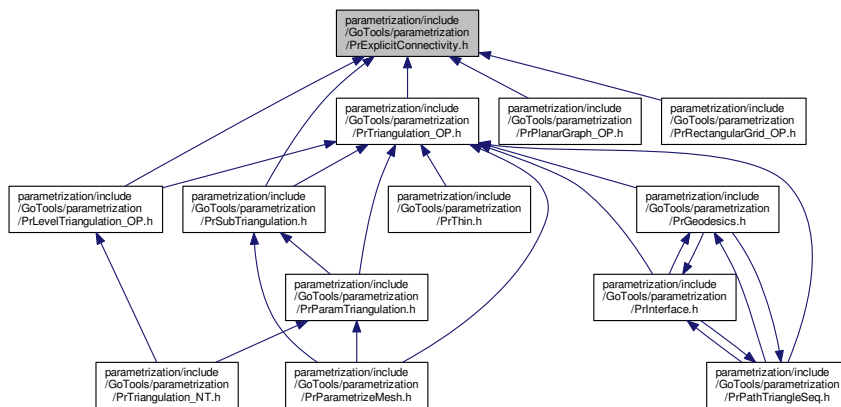
Definition at line 70 of file PrDijkstra.h.

## 30.2205 parametrization/include/GoTools/parametrization/PrExplicitConnectivity.h File Reference

#include "GoTools/parametrization/PrOrganizedPoints.h"  
 Include dependency graph for PrExplicitConnectivity.h:



This graph shows which files directly or indirectly include this file:

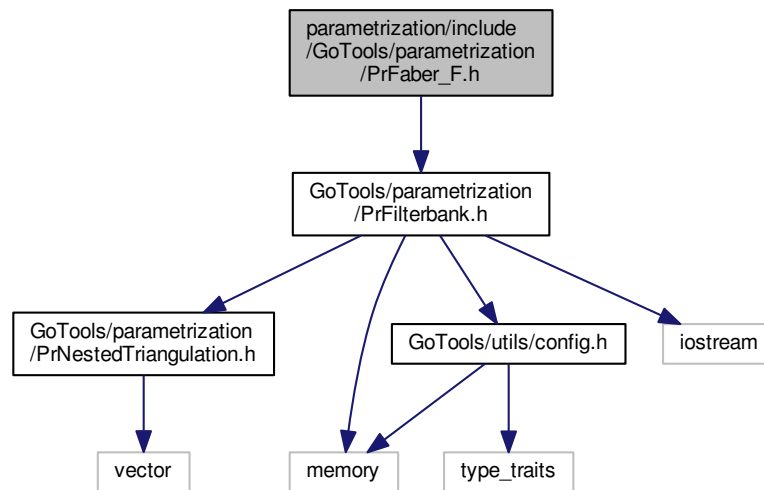


## Classes

- class [PrExplicitConnectivity](#)

## 30.2206 parametrization/include/GoTools/parametrization/PrFaber\_F.h File Reference

```
#include "GoTools/parametrization/PrFilterbank.h"
Include dependency graph for PrFaber_F.h:
```



## Classes

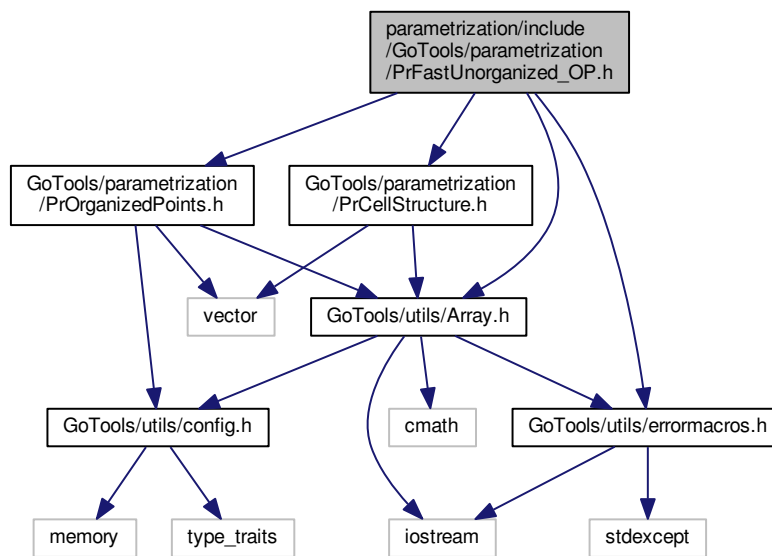
- class [PrFaber\\_F](#)

## 30.2207 parametrization/include/GoTools/parametrization/PrFastUnorganized\_OP.h File Reference

```
#include "GoTools/parametrization/PrOrganizedPoints.h"
#include "GoTools/parametrization/PrCellStructure.h"
#include "GoTools/utils/Array.h"
#include "GoTools/utils/errormacros.h"
```



Include dependency graph for PrFastUnorganized\_OP.h:



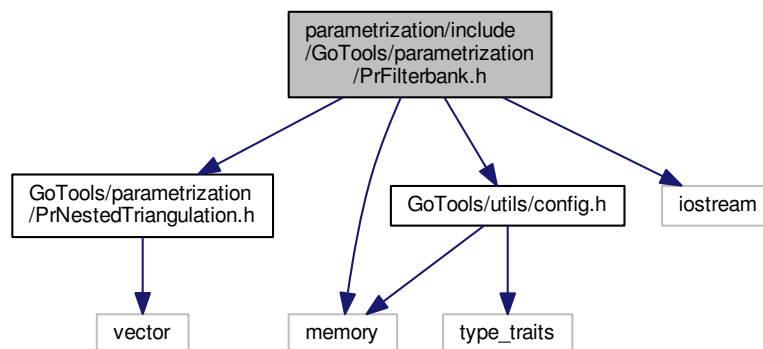
## Classes

- class [PrFastUnorganized\\_OP](#)

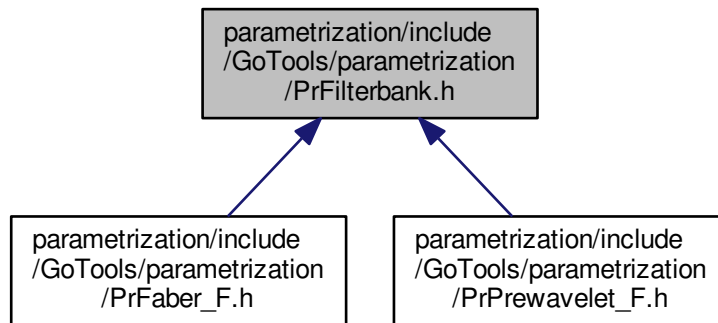
## 30.2208 parametrization/include/GoTools/parametrization/PrFilterbank.h File Reference

```
#include "GoTools/parametrization/PrNestedTriangulation.h"
#include "GoTools/utills/config.h"
#include <memory>
#include <iostream>
```

Include dependency graph for PrFilterbank.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [PrFilterbank](#)

## 30.2209 parametrization/include/GoTools/parametrization/PrGeodesics.h File Reference

```
#include <stdlib.h>
#include <stdio.h>
#include <malloc.h>
#include <math.h>
#include <string.h>
#include <fstream>
#include <iostream>
#include <list>
#include "GoTools/parametrization/PrPathTriangleSeq.h"
#include "GoTools/parametrization/PrInterface.h"
#include "GoTools/parametrization/PrOrganizedPoints.h"
#include "GoTools/parametrization/PrTriangulation_OP.h"
#include "GoTools/parametrization/PrTriangle.h"
#include "GoTools/parametrization/PrDijkstra.h"
#include "GoTools/utils/Array.h"
```



- `vector< Vector3D > point3D_from_indice (PrTriangulation_OP &t, const vector< int > &ind_path)`  
*3D points from their indices in the triangulation*
- `vector< int > tr_crossed_by_path_vertices (PrTriangulation_OP &t, vector< int > &vert_path)`
- `bool update_triangle_sequence_around_pivot (PrTriangulation_OP &t, vector< int > &tr_seq, int pivot)`
- `std::list< int > tr_sequence_around_vertex (PrTriangulation_OP &t, int vertex)`  
*triangles containing a vertex, ordered clockwise.*
- `std::list< int > tr_sequence_around_vertex (PrTriangulation_OP &t, int vertex, int first_tr)`  
*triangles containing a vertex, ordered clockwise, starting from first\_tr.*
- `bool pivot_in_new_list (std::list< int > &list_pivots, std::vector< int >::const_iterator &tried_begin, std::vector< int >::const_iterator &tried_end, std::list< int >::iterator &iter_pivot)`
- `bool pivot_in_new_list (std::list< int > &list_pivots, const std::list< int > &list_pivots_prec, int nb_steps, const vector< double > &lengths, std::list< int >::iterator &iter_pivot, int pivot)`
- `bool next_pivot (std::list< int > &list_pivots, std::list< int >::iterator &iter_pivot, int prec_pivot)`  
*the treated pivot belongs to the boundary, consider the next one in the list.*
- `bool decreasing_length (const int nb_steps, const vector< double > &lengths)`
- `double dist3D (Vector3D &a, Vector3D &b)`  
*distance between two R3 points*
- `double dist2D (Vector2D &a, Vector2D &b)`  
*distance between two R2 points*
- `double sqr (double x)`
- `double length_polygonal_path (const vector< Vector3D > &path)`

### 30.2209.1 Macro Definition Documentation

#### 30.2209.1.1 #define EPS 1e-8

Definition at line 67 of file PrGeodesics.h.

### 30.2209.2 Function Documentation

#### 30.2209.2.1 bool decreasing\_length ( const int nb\_steps, const vector< double > & lengths )

#### 30.2209.2.2 std::vector<int> DijkstraPT ( PrTriangulation\_OP & triangulation, int sce\_vertex, int dest\_vertex )

**Dijkstra** computes the shortest path between `sce_vertex` and `dest_vertex`, along the edges of the triangulation. It returns the polygonal solution path given by its vertices, vertices of the triangulation.

#### 30.2209.2.3 double dist2D ( Vector2D & a, Vector2D & b )

distance between two R2 points

#### 30.2209.2.4 double dist3D ( Vector3D & a, Vector3D & b )

distance between two R3 points

30.2209.2.5 `void GeodesicPT ( PrTriangulation_OP & triangulation, int sce_vertex, int dest_vertex, vector< Vector3D > & current_path, vector< int > & tr_seq3d )`

Computes a geodesic path (ie locally shortest path) between the vertices indexed by `sce_vertex` and `dest_vertex`. iterativ method consisting in :

1. Initialisation of a path with the [Dijkstra](#) method which computes a shortest path following the edges of the triangulation.
2. Improves locally the current path and associated sequence with an update of the sequence around pivot vertices, ie vertices where the path is not locally optimal. properties:
  - decreasing length at each step.
  - finite number of steps.

Returns the geodesic path as a polygonal line on the triangulated surface and the triangle sequence crossed by this path.

30.2209.2.6 `double length_polygonal_path ( const vector< Vector3D > & path )`

30.2209.2.7 `bool next_pivot ( std::list< int > & list_pivots, std::list< int >::iterator & iter_pivot, int prec_pivot )`

the treated pivot belongs to the boundary, consider the next one in the list.

30.2209.2.8 `bool pivot_in_new_list ( std::list< int > & list_pivots, std::vector< int >::const_iterator & tried_begin, std::vector< int >::const_iterator & tried_end, std::list< int >::iterator & iter_pivot )`

30.2209.2.9 `bool pivot_in_new_list ( std::list< int > & list_pivots, const std::list< int > & list_pivots_prec, int nb_steps, const vector< double > & lengths, std::list< int >::iterator & iter_pivot, int pivot )`

decide if there are pivot vertices in the new sequence set the pivot vertex where the sequence is to update in `iter_pivot`. compare the pivot lists before and after updating : if equal : incrementation of the pivot. if not : first pivot in the new list.

30.2209.2.10 `vector<Vector3D> point3D_from_indice ( PrTriangulation_OP & t, const vector< int > & ind_path )`

3D points from their indices in the triangulation

30.2209.2.11 `double sqr ( double x )`

30.2209.2.12 `vector<int> tr_crossed_by_path_vertices ( PrTriangulation_OP & t, vector< int > & vert_path )`

triangle sequence crossed by a path where the path is given by a list of vert. indices

30.2209.2.13 `std::list<int> tr_sequence_around_vertex ( PrTriangulation_OP & t, int vertex )`

triangles containing a vertex, ordered clockwise.

30.2209.2.14 `std::list<int> tr_sequence_around_vertex ( PrTriangulation_OP & t, int vertex, int first_tr )`

triangles containing a vertex, ordered clockwise, starting from `first_tr`.

30.2209.2.15 `bool update_triangle_sequence_around_pivot ( PrTriangulation_OP & t, vector< int > & tr_seq, int pivot )`

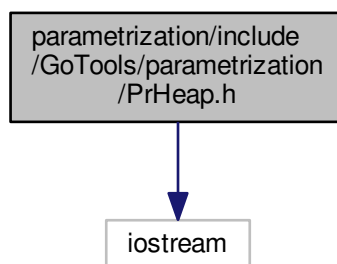
a triangle sequence is modified/updated around a given pivot vertex. returns

- 0 if it is not possible to update (if pivot belongs to the boundary or pivot does not belong to the sequence)
- 1 otherwise

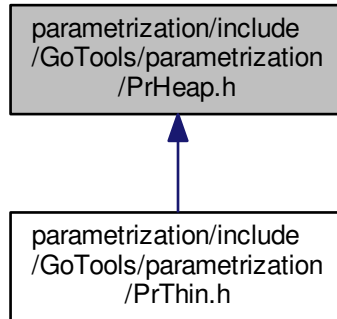
## 30.2210 parametrization/include/GoTools/parametrization/PrHeap.h File Reference

```
#include <iostream>
```

Include dependency graph for PrHeap.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [PrHeap](#)

## 30.2211 parametrization/include/GoTools/parametrization/PrInterface.h File Reference

```
#include <iomanip>
#include <stdlib.h>
#include <malloc.h>
#include <math.h>
#include <string.h>
#include <fstream>
#include <iostream>
#include <list>
#include "GoTools/parametrization/PrPathTriangleSeq.h"
#include "GoTools/parametrization/PrGeodesics.h"
#include "GoTools/parametrization/PrOrganizedPoints.h"
#include "GoTools/parametrization/PrTriangulation_OP.h"
#include "GoTools/parametrization/PrTriangle.h"
#include "GoTools/parametrization/PrDijkstra.h"
#include "GoTools/utils/Array.h"
```





- void [printGeomviewTriangleSequence](#) ([PrTriangulation\\_OP](#) &t, vector< int > tr\_seq)  
*Writing file for the triangle sequence visualisation with geomview.*
- void [printGeomviewTriangleSequence](#) (std::ofstream &os, [PrTriangulation\\_OP](#) &t, vector< int > tr\_seq)
- void [printGeomviewPath](#) (vector< [Vector3D](#) > &path)  
*Writing file for the shortest path visualisation with geomview.*
- void [printGeomviewPath](#) (std::ofstream &os, vector< [Vector3D](#) > &path)
- void [print\\_edges\\_lengths](#) (vector< int > tr\_seq, [PrTriangulation\\_OP](#) &t)
- void [print\\_lengths](#) (const vector< [double](#) > &lengths)

### 30.2211.1 Function Documentation

30.2211.1.1 void print ( vector< int > & v )

30.2211.1.2 void print ( vector< double > & v )

30.2211.1.3 void print ( vector< [Vector3D](#) > & v )

30.2211.1.4 void print ( vector< [Vector2D](#) > & v )

30.2211.1.5 void print ( vector< [PrTriangle](#) > & t )

30.2211.1.6 void print\_edges\_lengths ( vector< int > tr\_seq, [PrTriangulation\\_OP](#) & t )

30.2211.1.7 void print\_lengths ( const vector< double > & lengths )

30.2211.1.8 void printGeomviewPath ( vector< [Vector3D](#) > & path )

Writing file for the shortest path visualisation with geomview.

30.2211.1.9 void printGeomviewPath ( std::ofstream & os, vector< [Vector3D](#) > & path )

Writing file for the shortest path visualisation with geomview. Problem in giving the name function as parameter.

30.2211.1.10 void printGeomviewTriangleSequence ( [PrTriangulation\\_OP](#) & t, vector< int > tr\_seq )

Writing file for the triangle sequence visualisation with geomview.

30.2211.1.11 void printGeomviewTriangleSequence ( std::ofstream & os, [PrTriangulation\\_OP](#) & t, vector< int > tr\_seq )

Writing file for the triangle sequence visualisation with geomview. Problem in giving the name function as parameter.

30.2211.1.12 void printGeomviewTriangulation ( [PrTriangulation\\_OP](#) & t )

Writing file for the triangulation visualisation with geomview.

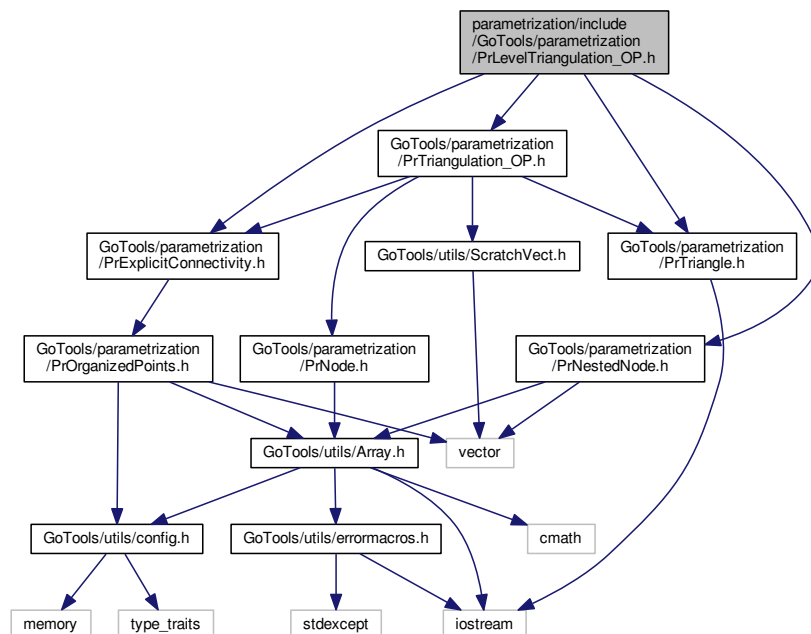
```
30.2211.1.13 void printGeomviewTriangulation (std::ofstream & os, PrTriangulation_OP & t)
```

Writing file for the triangulation visualisation with geomview. Problem in giving the name function as parameter.

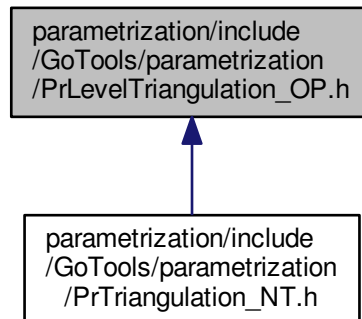
### 30.2212 parametrization/include/GoTools/parametrization/PrLevelTriangulation\_OP.h File Reference

```
#include "GoTools/parametrization/PrExplicitConnectivity.h"
#include "GoTools/parametrization/PrTriangulation_OP.h"
#include "GoTools/parametrization/PrNestedNode.h"
#include "GoTools/parametrization/PrTriangle.h"
```

Include dependency graph for PrLevelTriangulation\_OP.h:



This graph shows which files directly or indirectly include this file:



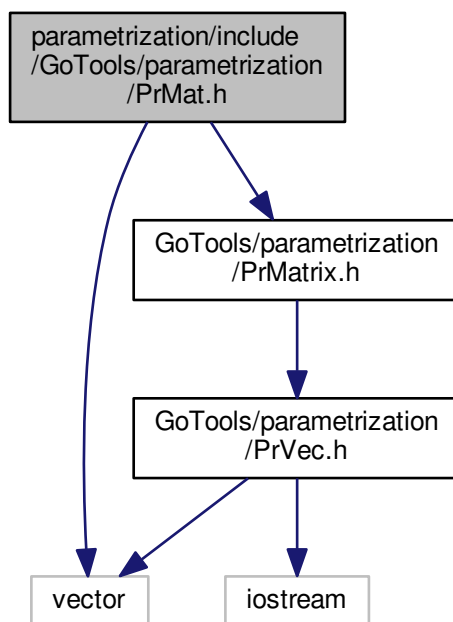
## Classes

- class [PrLevelTriangulation\\_OP](#)

## 30.2213 parametrization/include/GoTools/parametrization/PrMat.h File Reference

```
#include <vector>
#include "GoTools/parametrization/PrMatrix.h"
```

Include dependency graph for PrMat.h:



## Classes

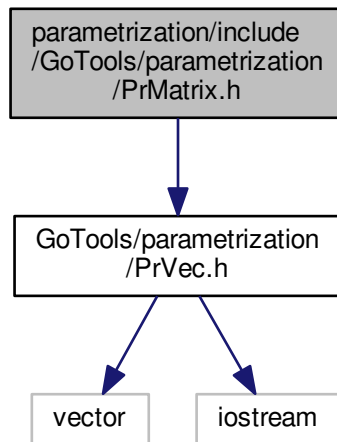
- class [PrMat](#)

*This class implements a matrix.*

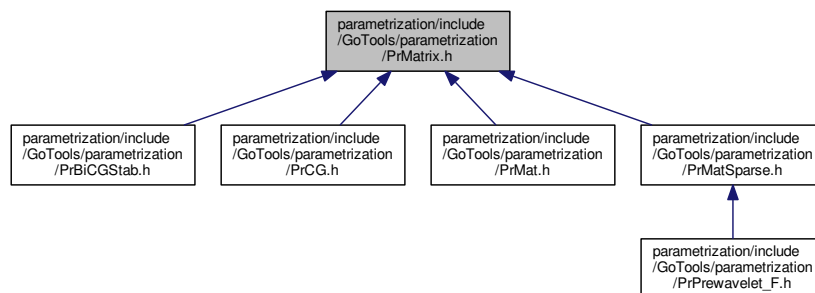
## 30.2214 parametrization/include/GoTools/parametrization/PrMatrix.h File Reference

```
#include "GoTools/parametrization/PrVec.h"
```

Include dependency graph for PrMatrix.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [PrMatrix](#)

*This class implements a matrix.*

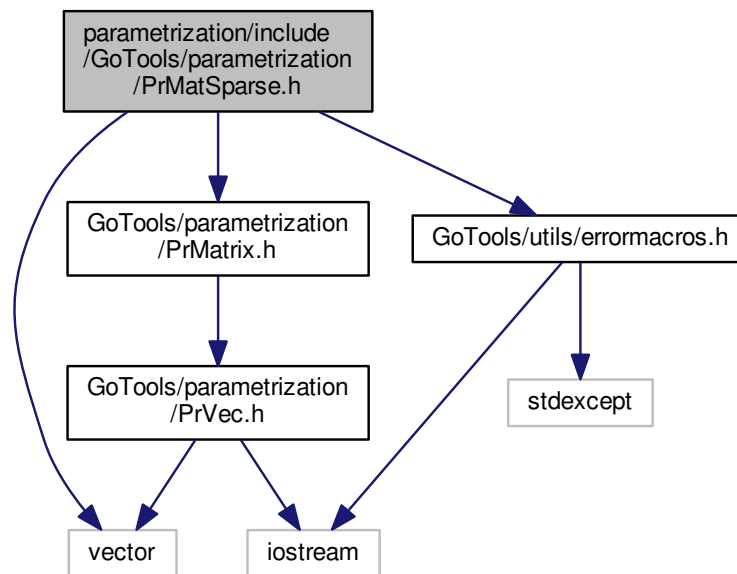
## 30.2215 parametrization/include/GoTools/parametrization/PrMatSparse.h File Reference

```

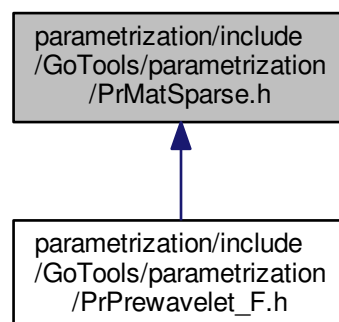
#include "GoTools/parametrization/PrMatrix.h"
#include "GoTools/utils/errormacros.h"
#include <vector>

```

Include dependency graph for PrMatSparse.h:



This graph shows which files directly or indirectly include this file:

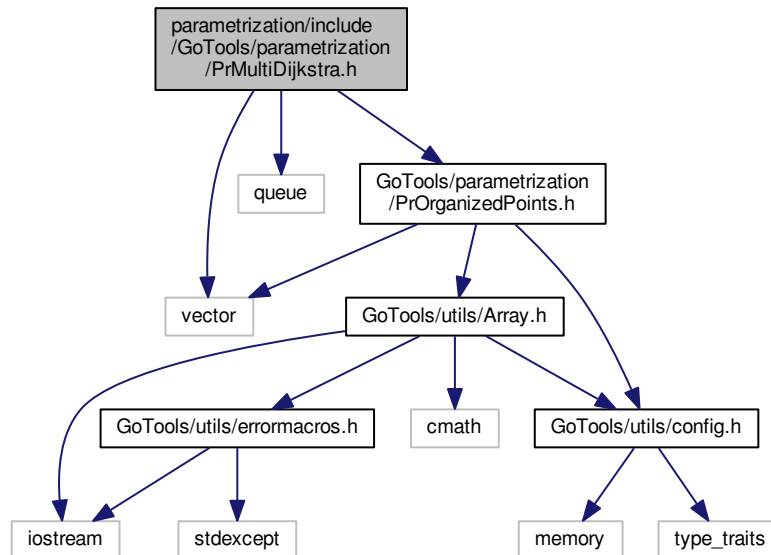


## Classes

- class [PrMatSparse](#)

## 30.2216 parametrization/include/GoTools/parametrization/PrMultiDijkstra.h File Reference

```
#include <vector>
#include <queue>
#include "GoTools/parametrization/PrOrganizedPoints.h"
Include dependency graph for PrMultiDijkstra.h:
```



### Classes

- class [HeapNode2](#)
- class [MultiDijkstra](#)

### Macros

- `#define` [GraphType](#) [PrOrganizedPoints](#)
- `#define` [Point](#) [Vector3D](#)

### Typedefs

- typedef `std::priority_queue< HeapNode2, vector< HeapNode2 >, std::greater< HeapNode2 > >` [HeapType2](#)

### 30.2216.1 Macro Definition Documentation

#### 30.2216.1.1 `#define` [GraphType](#) [PrOrganizedPoints](#)

Definition at line 47 of file PrMultiDijkstra.h.

30.2216.1.2 `#define Point Vector3D`

Definition at line 50 of file PrMultiDijkstra.h.

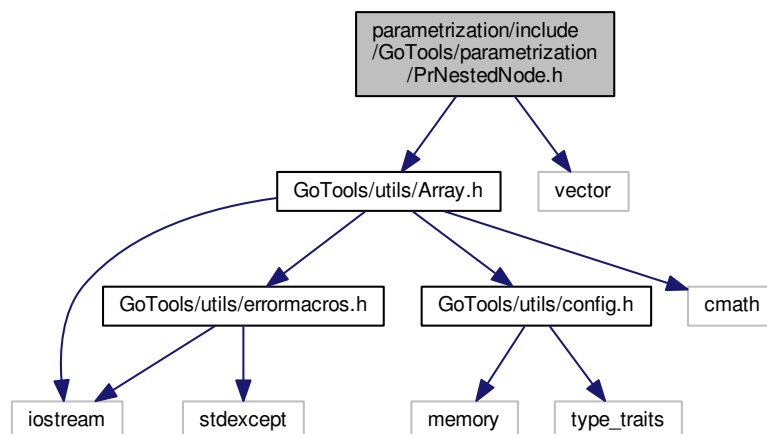
### 30.2216.2 Typedef Documentation

30.2216.2.1 `typedef std::priority_queue< HeapNode2, vector<HeapNode2>, std::greater<HeapNode2>> > HeapType2`

Definition at line 70 of file PrMultiDijkstra.h.

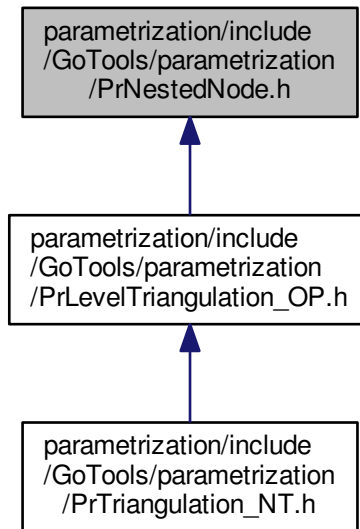
### 30.2217 parametrization/include/GoTools/parametrization/PrNestedNode.h File Reference

```
#include "GoTools/Utils/Array.h"
#include <vector>
Include dependency graph for PrNestedNode.h:
```





This graph shows which files directly or indirectly include this file:



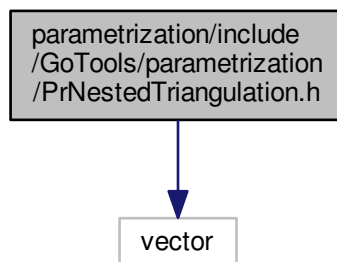
## Classes

- class [PrNestedNode](#)

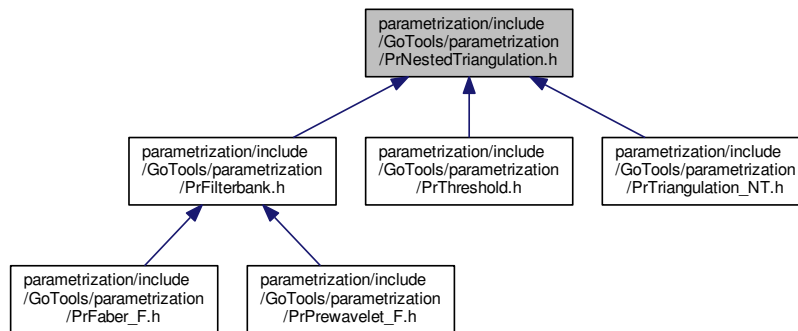
## 30.2218 parametrization/include/GoTools/parametrization/PrNestedTriangulation.h File Reference

```
#include <vector>
```

Include dependency graph for PrNestedTriangulation.h:



This graph shows which files directly or indirectly include this file:



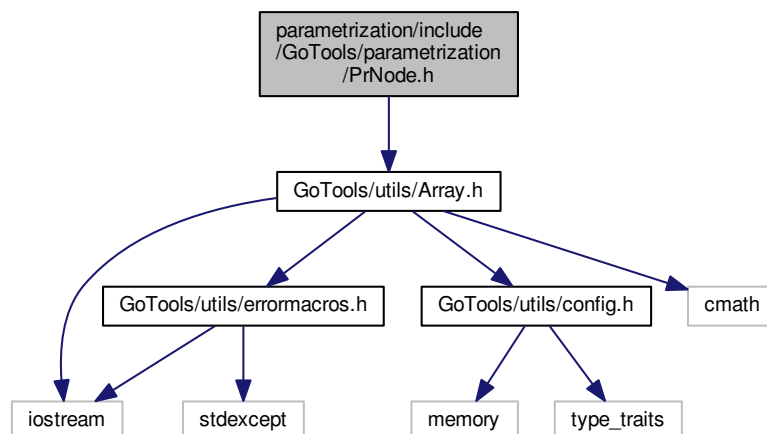
## Classes

- class [PrNestedTriangulation](#)

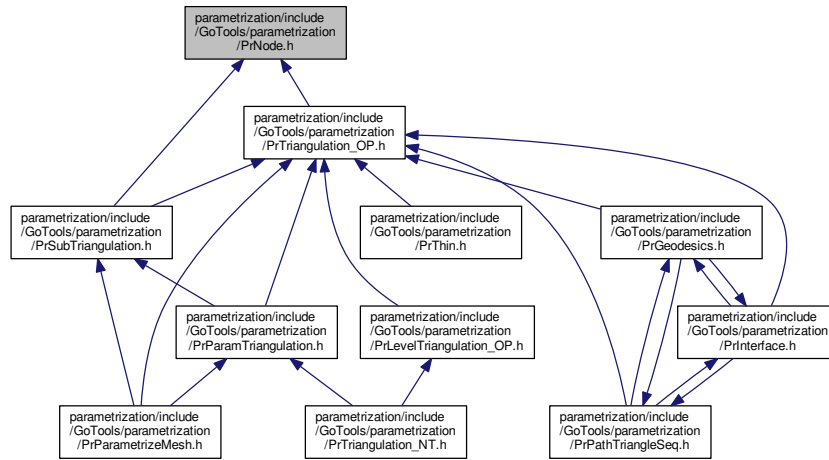
## 30.2219 parametrization/include/GoTools/parametrization/PrNode.h File Reference

```
#include "GoTools/utils/Array.h"
```

Include dependency graph for PrNode.h:



This graph shows which files directly or indirectly include this file:



Classes

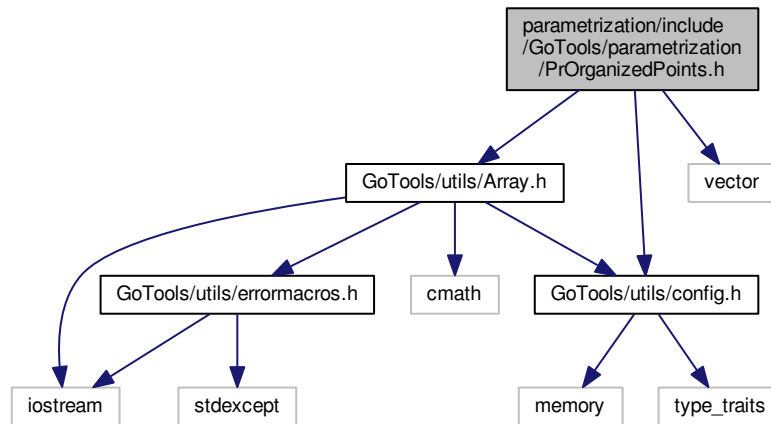
- class [PrNode](#)

30.2220 parametrization/include/GoTools/parametrization/PrOrganizedPoints.h File Reference

```

#include "GoTools/utils/Array.h"
#include <vector>
#include "GoTools/utils/config.h"
Include dependency graph for PrOrganizedPoints.h:

```



This graph shows which files directly or indirectly include this file:



## Classes

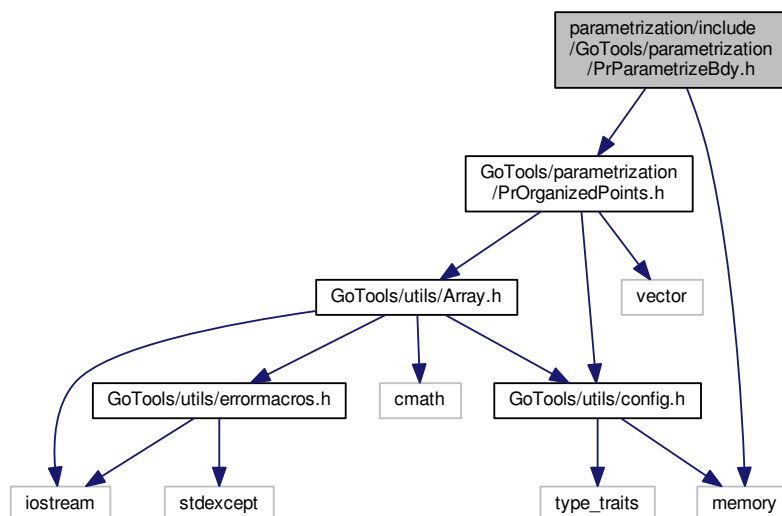
- class [PrOrganizedPoints](#)

## 30.2221 parametrization/include/GoTools/parametrization/PrParametrizeBdy.h File Reference

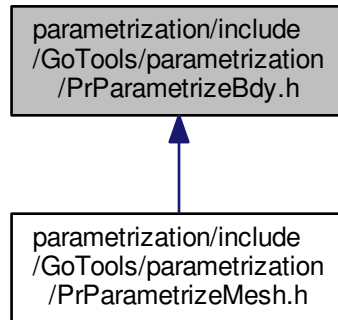
```
#include "GoTools/parametrization/PrOrganizedPoints.h"
```

```
#include <memory>
```

Include dependency graph for PrParametrizeBdy.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [PrParametrizeBdy](#)

## Enumerations

- enum [PrBdyParamKind](#) { [PrCHORDLENGTHBDY](#) = 1, [PrCENTRIPETAL](#) = 2, [PrUNIFBDY](#) = 3 }

### 30.2221.1 Enumeration Type Documentation

#### 30.2221.1.1 enum PrBdyParamKind

Enumerator

***PrCHORDLENGTHBDY***

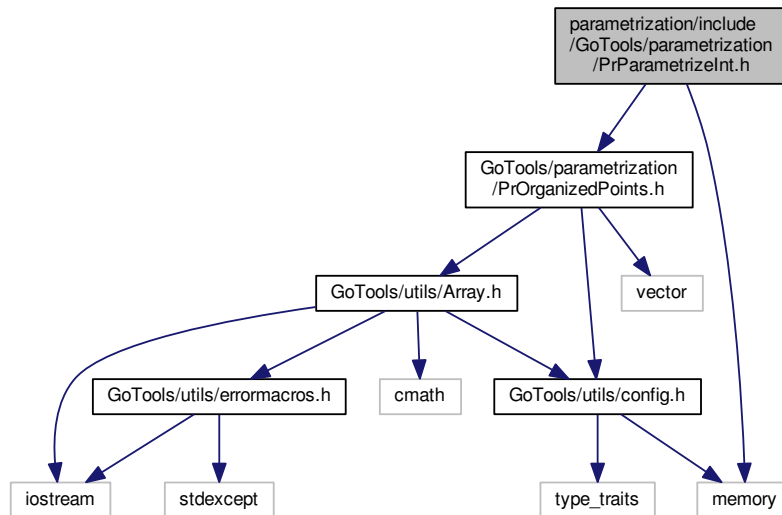
***PrCENTRIPETAL***

***PrUNIFBDY***

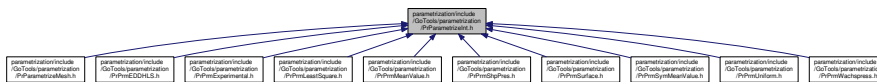
Definition at line 49 of file PrParametrizeBdy.h.

## 30.2222 parametrization/include/GoTools/parametrization/PrParametrizeInt.h File Reference

```
#include "GoTools/parametrization/PrOrganizedPoints.h"
#include <memory>
Include dependency graph for PrParametrizeInt.h:
```



This graph shows which files directly or indirectly include this file:



### Classes

- class [PrParametrizeInt](#)

### Enumerations

- enum [PrParamStartVector](#) { [PrBARYCENTRE](#) = 1, [PrFROMUV](#) = 2 }

### 30.2222.1 Enumeration Type Documentation

#### 30.2222.1.1 enum PrParamStartVector

##### Enumerator

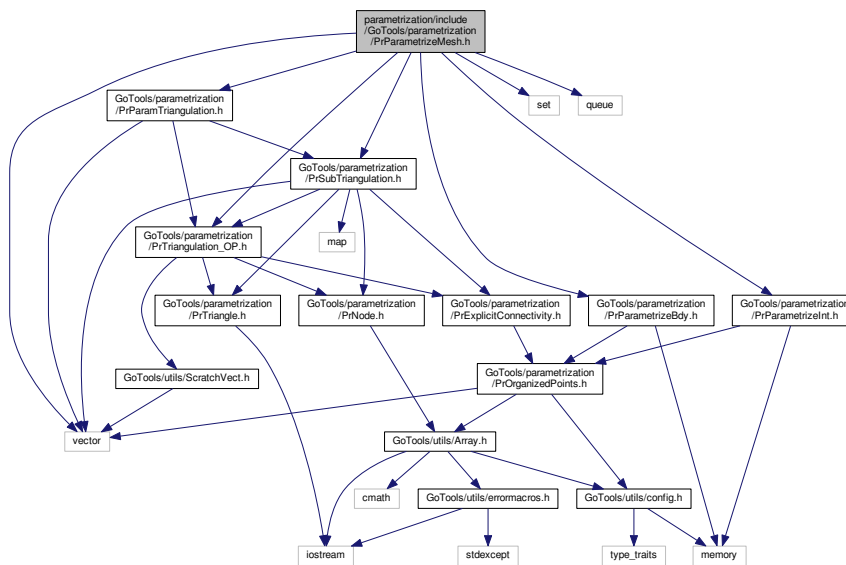
***PrBARYCENTRE***  
***PrFROMUV***

Definition at line 48 of file PrParametrizeInt.h.

## 30.2223 parametrization/include/GoTools/parametrization/PrParametrizeMesh.h File Reference

```
#include "GoTools/parametrization/PrTriangulation_OP.h"
#include "GoTools/parametrization/PrSubTriangulation.h"
#include "GoTools/parametrization/PrParamTriangulation.h"
#include "GoTools/parametrization/PrParametrizeInt.h"
#include "GoTools/parametrization/PrParametrizeBdy.h"
#include <vector>
#include <set>
#include <queue>
```

Include dependency graph for PrParametrizeMesh.h:



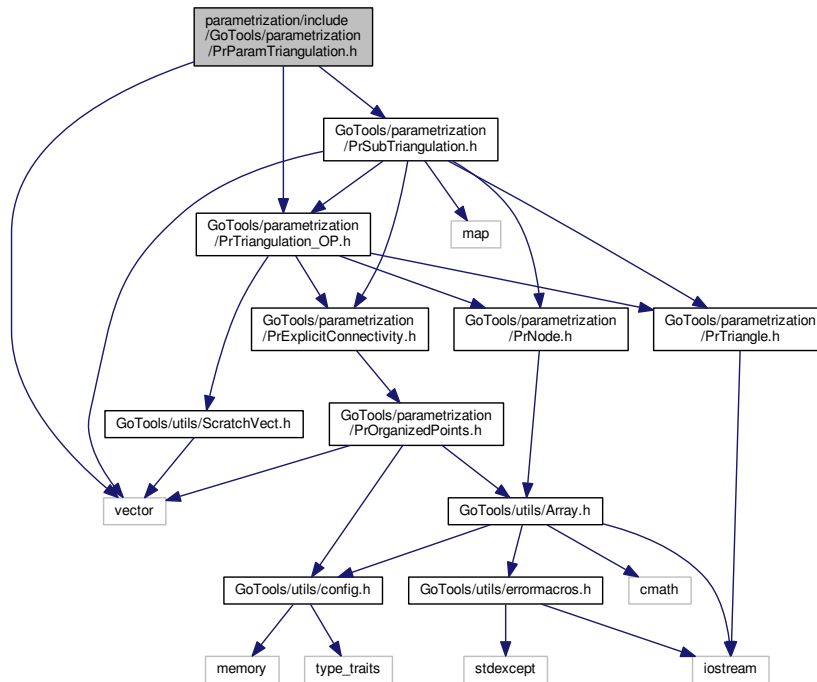
### Classes

- class [PrParametrizeMesh](#)

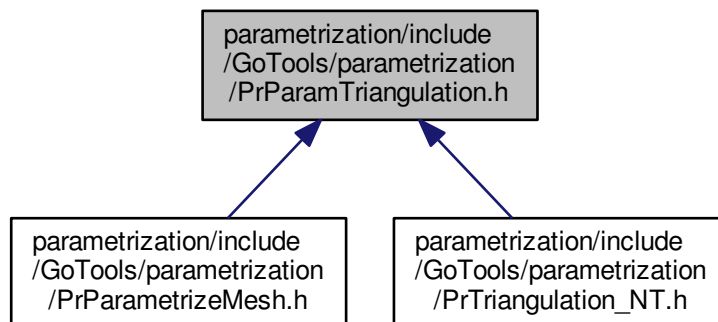
## 30.2224 parametrization/include/GoTools/parametrization/PrParamTriangulation.h File Reference

```
#include "GoTools/parametrization/PrTriangulation_OP.h"
#include "GoTools/parametrization/PrSubTriangulation.h"
#include <vector>
```

Include dependency graph for PrParamTriangulation.h:



This graph shows which files directly or indirectly include this file:



## Classes

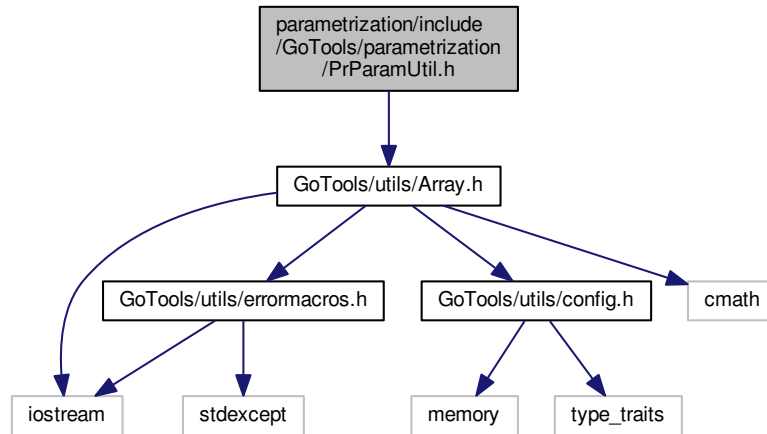
- class [PrParamTriangulation](#)



## 30.2225 parametrization/include/GoTools/parametrization/PrParamUtil.h File Reference

```
#include "GoTools/Utils/Array.h"
```

Include dependency graph for PrParamUtil.h:



## Functions

- void `baryCoords` (`double x`, `double y`, `double x0`, `double y0`, `double x1`, `double y1`, `double x2`, `double y2`, `double &tau0`, `double &tau1`, `double &tau2`)
- void `baryCoords0` (`double &u0`, `double &v0`, `double &u1`, `double &v1`, `double &u2`, `double &v2`, `double &tau0`, `double &tau1`, `double &tau2`)
- `double det` (`const double &u1`, `const double &v1`, `const double &u2`, `const double &v2`)
- `double area` (`const double &x0`, `const double &y0`, `const double &x1`, `const double &y1`, `const double &x2`, `const double &y2`)
- `double area` (`const Vector3D &a`, `const Vector3D &b`, `const Vector3D &c`)  
*Find the positive area of a 3D triangle.*
- void `polarCoords` (`Vector2D v`, `double &r`, `double &theta`)
- void `polarCoords` (`double u`, `double v`, `double &r`, `double &theta`)
- `double tanThetaOverTwo` (`const Vector3D &a`, `const Vector3D &b`, `const Vector3D &c`)
- `double cotangent` (`const Vector3D &a`, `const Vector3D &b`, `const Vector3D &c`)

### 30.2225.1 Function Documentation

30.2225.1.1 `double area ( const double & x0, const double & y0, const double & x1, const double & y1, const double & x2, const double & y2 )`

Find the signed area of triangle  $[(x_0,y_0),(x_1,y_1),(x_2,y_2)]$ , where  $(x_0,y_0),(x_1,y_1),(x_2,y_2)$  are assumed to be ordered anticlockwise.

30.2225.1.2 `double area ( const Vector3D & a, const Vector3D & b, const Vector3D & c )`

Find the positive area of a 3D triangle.

30.2225.1.3 `void baryCoords ( double x, double y, double x0, double y0, double x1, double y1, double x2, double y2, double & tau0, double & tau1, double & tau2 )`

Find the three barycentric coordinates of the point (x,y) with respect to the triangle with vertices (x0,y0), (x1,y1), (x2,y2). The three vertices should not be collinear!

30.2225.1.4 `void baryCoords0 ( double & u0, double & v0, double & u1, double & v1, double & u2, double & v2, double & tau0, double & tau1, double & tau2 )`

Find the barycentric coordinates of the origin (0,0) with respect to the triangle formed by three vectors (u0,v0),(u0,v1),(u0,v2). This is bit more efficient than calling baryCoords.

30.2225.1.5 `double cotangent ( const Vector3D & a, const Vector3D & b, const Vector3D & c )`

30.2225.1.6 `double det ( const double & u1, const double & v1, const double & u2, const double & v2 )`

30.2225.1.7 `void polarCoords ( Vector2D v, double & r, double & theta )`

Represent a vector v in polar coordinates  $r(\cos(\theta), \sin(\theta))$  where  $0 \leq r < 2\pi$ .

30.2225.1.8 `void polarCoords ( double u, double v, double & r, double & theta )`

Represent a vector (u,v) in polar coordinates  $r(\cos(\theta), \sin(\theta))$  where  $0 \leq r < 2\pi$ .

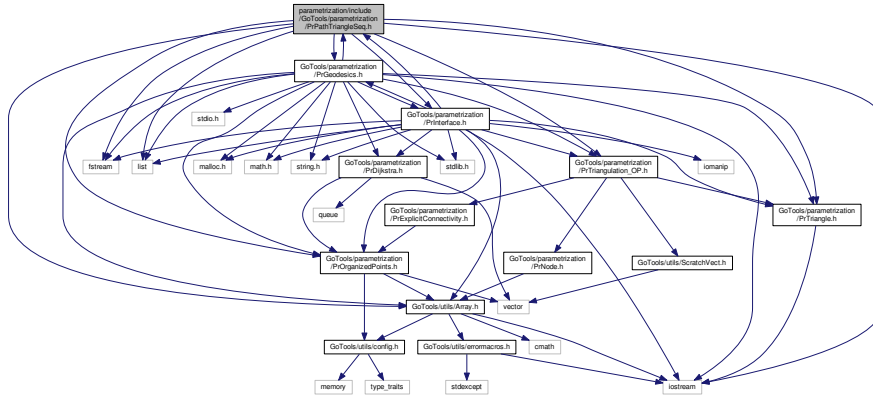
30.2225.1.9 `double tanThetaOverTwo ( const Vector3D & a, const Vector3D & b, const Vector3D & c )`

Return tangent of half the angle between vectors b-a and c-a without using trig functions. Use fact that  $\tan(\alpha/2) = (1 - \cos(\alpha)) / \sin(\alpha)$ . and use scalar and dot products to get  $\cos(\alpha)$  and  $\sin(\alpha)$ .

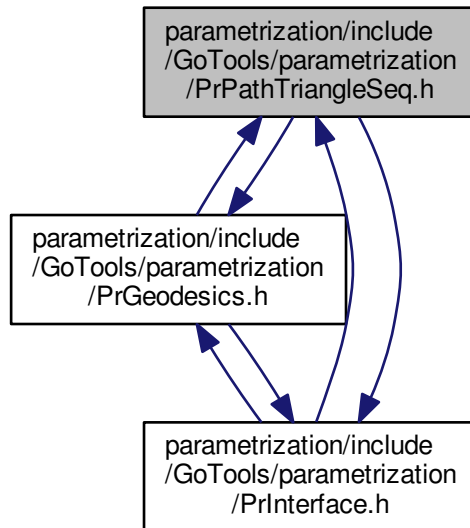
## 30.2226 parametrization/include/GoTools/parametrization/PrPathTriangleSeq.h File Reference

```
#include <fstream>
#include <iostream>
#include <list>
#include "GoTools/parametrization/PrOrganizedPoints.h"
#include "GoTools/parametrization/PrTriangulation_OP.h"
#include "GoTools/parametrization/PrGeodesics.h"
#include "GoTools/parametrization/PrInterface.h"
#include "GoTools/parametrization/PrTriangle.h"
#include "GoTools/utils/Array.h"
```

Include dependency graph for PrPathTriangleSeq.h:



This graph shows which files directly or indirectly include this file:



**Classes**

- class [EdgeType](#)
- class [UnfNodeType](#)
- class [PathType](#)

**Functions**

- void [printNode](#) (vector< [UnfNodeType](#) > &v)

- `vector< Vector3D > shortest_path_triangle_sequence (const PrTriangulation_OP &t, int sce_vertex, int dest_vertex, const vector< int > &tr_seq3d, list< int > &list_pivots, double &length)`
- `void shortest_path_triangle_sequence (const PrTriangulation_OP &t, const int &sce_vertex, const int &dest_←_vertex, vector< int > &tr_seq3d, list< int > &list_pivots, std::vector< EdgeType > &edge_seq3d, std←::vector< double > &ratio_on_edge, double &length)`
- `void path_3Dpts_from_ratio_on_edge (const PrTriangulation_OP &t, std::vector< Go::Vector3D > &sh_path, const std::vector< double > &ratio_on_edge, const std::vector< EdgeType > &edge_seq3d)`
- `Go::Vector3D pt3D_from_ratio_on_edge (double r, Go::Vector3D A, Go::Vector3D B)`  
*Computes a 3D point from its position (ratio) on an edge.*
- `void path_2Dpts_from_ratio_on_edge (const vector< UnfNodeType > &nodes_unf, vector< Vector2D > &path, const vector< double > &ratio_on_edge, const vector< EdgeType > &edge_seq2d)`
- `Vector2D pt2D_from_ratio_on_edge (double r, Vector2D a, Vector2D b)`  
*Computes a 2D point from its position (ratio) on an edge.*
- `list< int > pivots3D_from_pivots_in_unf_seq (const list< int > &list_pivots_unf, const vector< UnfNodeType > &nodes_unf)`
- `list< double > deviation_from_pivots (const list< int > &list_pivots_unf, const vector< UnfNodeType > &nodes_unf)`
- `double abs_val (double a)`  
*absolute value*
- `vector< double > shortest_path_triangle_sequence_2D (const PrTriangulation_OP &t, const int sce_vertex, const int dest_vertex, const int sce_vertexUnf, int dest_vertexUnf, const std::vector< int > &tr_seq3d, std←::vector< PrTriangle > &tr_seq_unf, std::list< int > &list_pivots_unf, std::vector< UnfNodeType > nodes_←unf, double &length)`
- `void print_list_int (std::list< int > &list_pivots_unf)`
- `void print_deviation (std::list< double > &list_deviation)`
- `double length_polygonal_path (const list< int > &list_pivots_unf, vector< UnfNodeType > &nodes_unf)`
- `std::vector< double > path_ratio_on_edge_from_pivots (std::list< int > &list_pivots_unf, std::vector< EdgeType > &edge_seq_unf, const std::vector< UnfNodeType > &nodes_unf)`
- `double segment_intersection (Go::Vector2D a, Go::Vector2D b, Go::Vector2D c, Go::Vector2D d)`
- `int line_intersection (Go::Vector2D a, Go::Vector2D b, Go::Vector2D c, Go::Vector2D d, Go::Vector2D &i)`
- `void path_to_edge (const int sce_vertex_unf, int i, std::vector< PathType > &p, const std::vector< Unf←NodeType > &nodes_unf, const std::vector< PrTriangle > &tr_seq_unf, std::vector< EdgeType > &edge_←seq_unf)`
- `std::vector< EdgeType > edge_sequence (const std::vector< int > &tr_seq, const PrTriangulation_OP &t)`
- `std::vector< EdgeType > edge_sequence (std::vector< PrTriangle > tr_seq)`
- `EdgeType common_edge (PrTriangle t1, PrTriangle t2)`  
*common edge between two triangles, with the nodes given in the order of t1*
- `void unfolding_triangle_sequence (const PrTriangulation_OP &t, const vector< int > &tr_seq3d, vector< PrTriangle > &tr_seq_unf, const vector< EdgeType > &edge_seq3d, vector< EdgeType > &edge_seq_unf, vector< UnfNodeType > &nodes_unf)`
- `void unfolding_first_triangle (const PrTriangulation_OP &t, const vector< int > &tr_seq, vector< PrTriangle > &tr_seq_unf, vector< UnfNodeType > &nodes_unf)`
- `void unfolding_vertex (const PrTriangulation_OP &t, const vector< int > &tr_seq3d, vector< PrTriangle > &tr_seq_unf, const int vertex, int &vertex_unf)`
- `void local_frame (Go::Vector3D a, Go::Vector3D b, Go::Vector3D c, Go::Vector3D &U, Go::Vector3D &V, Go←::Vector3D &W)`  
*local frame of triangle ABC*
- `Go::Vector3D vector_W (Go::Vector3D a, Go::Vector3D b, Go::Vector3D c)`  
*axis and normal vector in the current triangle plane for the local frame*
- `Go::Vector3D vector_U (const Go::Vector3D W)`  
*first axis/vector in the current triangle plane for the local frame*
- `Go::Vector3D vector_V (const Go::Vector3D W, const Go::Vector3D U)`  
*second axis/vector in the current triangle plane for the local frame*
- `Go::Vector3D local_coordinates (const Go::Vector3D a, const Go::Vector3D &U, const Go::Vector3D &V, const Go::Vector3D &W)`

- local coordinates of a vector A in an axis frame given by (U, V, W)*
- `double angle (const Go::Vector2D &a, const Go::Vector2D &b, const Go::Vector2D &c, const Go::Vector2D &d)`
- angle between two vectors AB and CD in the plane*
- `void rotation2D (const Go::Vector2D a, const double theta, const Go::Vector2D c, Go::Vector2D &vect)`
  - `void axial_symmetry (const Go::Vector2D c, const Go::Vector2D a, const Go::Vector2D b, Go::Vector2D &v)`
  - `int third_vertex (PrTriangle Tr, EdgeType e)`
- third vertex of triangle t not belonging to edge e*
- `int same_side (const Go::Vector2D pt1, const Go::Vector2D pt2, const Go::Vector2D ptA, const Go::Vector2D ptB)`
- check if pt1 and pt2 are both on the same side of [ptA ptB]*
- `double vect_prod2D (Go::Vector2D a, Go::Vector2D b, Go::Vector2D c, Go::Vector2D d)`
  - `void printUnfoldedSequence (std::ofstream &os, std::vector< UnfNodeType > &nodes_unf, std::vector< PrTriangle > &tr_seq_unf)`
  - `void printUnfoldedPath (std::ofstream &os, std::vector< UnfNodeType > &nodes_unf, std::list< int > &list_pivots_unf)`
  - `void printUnfoldedPath (std::ofstream &os, std::vector< Go::Vector2D > path)`
  - `void printUnfoldedObjects (std::ofstream &os, std::vector< UnfNodeType > &nodes_unf, std::vector< PrTriangle > &tr_seq_unf, std::list< int > &list_pivots_unf)`
  - `void print_edges_lengths (std::vector< PrTriangle > tr, std::vector< UnfNodeType > nodes_unf)`
- print the lengths of the edges of the triangles in the unfolded sequence*
- `void print_triangle_sequence (std::vector< PrTriangle > tr, std::vector< UnfNodeType > nodes_unf)`
  - `void print_tr_vertices (std::vector< int > &tr, PrTriangulation_OP &t)`

### 30.2226.1 Function Documentation

#### 30.2226.1.1 `double abs_val ( double a )`

absolute value

#### 30.2226.1.2 `double angle ( const Go::Vector2D & a, const Go::Vector2D & b, const Go::Vector2D & c, const Go::Vector2D & d )`

angle between two vectors AB and CD in the plane

#### 30.2226.1.3 `void axial_symmetry ( const Go::Vector2D c, const Go::Vector2D a, const Go::Vector2D b, Go::Vector2D & v )`

#### 30.2226.1.4 `EdgeType common_edge ( PrTriangle t1, PrTriangle t2 )`

common edge between two triangles, with the nodes given in the order of t1

#### 30.2226.1.5 `list<double> deviation_from_pivots ( const list< int > & list_pivots_unf, const vector< UnfNodeType > & nodes_unf )`

At each interior pivot, the path is deviated from the straight line. This function computes the deviation angles.

```
30.2226.1.6 std::vector<EdgeType> edge_sequence (const std::vector< int > & tr_seq, const PrTriangulation_OP
 & t)
```

given a triangle sequence, computes an edge sequence consisting of the common edges to two successive triangles in the sequence

```
30.2226.1.7 std::vector<EdgeType> edge_sequence (std::vector< PrTriangle > tr_seq)
```

given a triangle sequence, computes an edge sequence consisting of the common edges to two successive triangles in the sequence

```
30.2226.1.8 double length_polygonal_path (const list< int > & list_pivots_unf, vector< UnfNodeType > & nodes_unf)
```

```
30.2226.1.9 int line_intersection (Go::Vector2D a, Go::Vector2D b, Go::Vector2D c, Go::Vector2D d,
 Go::Vector2D & i)
```

computes i, the intersection between (ab) and (cd) returns 1 if there is a unique solution, 2 if the lines are the same, 0 if they are parallel

```
30.2226.1.10 Go::Vector3D local_coordinates (const Go::Vector3D a, const Go::Vector3D & U, const
 Go::Vector3D & V, const Go::Vector3D & W)
```

local coordinates of a vector A in an axis frame given by (U, V, W)

```
30.2226.1.11 void local_frame (Go::Vector3D a, Go::Vector3D b, Go::Vector3D c, Go::Vector3D & U,
 Go::Vector3D & V, Go::Vector3D & W)
```

local frame of triangle ABC

```
30.2226.1.12 void path_2Dpts_from_ratio_on_edge (const vector< UnfNodeType > & nodes_unf, vector< Vector2D > &
 path, const vector< double > & ratio_on_edge, const vector< EdgeType > & edge_seq2d)
```

Computes the 2D points of the path from their position (ratio) on the edges of the sequence

```
30.2226.1.13 void path_3Dpts_from_ratio_on_edge (const PrTriangulation_OP & t, std::vector< Go::Vector3D > &
 sh_path, const std::vector< double > & ratio_on_edge, const std::vector< EdgeType > & edge_seq3d)
```

Computes the 3D points of the path from their position (ratio) on the edges of the sequence.

```
30.2226.1.14 std::vector<double> path_ratio_on_edge_from_pivots (std::list< int > & list_pivots_unf, std::vector<
 EdgeType > & edge_seq_unf, const std::vector< UnfNodeType > & nodes_unf)
```

computes the intersection of the path, given by the pivots, with the edge sequence, returns the ratio/position on each edge

```
30.2226.1.15 void path_to_edge (const int sce_vertex_unf, int i, std::vector< PathType > & p, const std::vector<
 UnfNodeType > & nodes_unf, const std::vector< PrTriangle > & tr_seq_unf, std::vector< EdgeType >
 & edge_seq_unf)
```

Computes the path from a source point to an edge of the 2d sequence:. This path is given by:.

- a *l\_path\_*, polygonal chain of vertices to the left vertex of the edge.
- a *r\_path\_*, polygonal chain of vertices to the right vertex of the edge.
- *funnel\_*, common part of the path.

```
30.2226.1.16 list<int> pivots3D_from_pivots_in_unf_seq (const list< int > & list_pivots_unf, const vector<
 UnfNodeType > & nodes_unf)
```

computes the 3d pivot list corresponding to the pivot in the unfolded sequence. The pivots are the vertices of the shortest path. The pivots in the unfolded sequence contain all the vertices of the path (source and dest. too). The 3d pivots do not contain the source and dest. vertices.

```
30.2226.1.17 void print_deviation (std::list< double > & list_deviation)
```

```
30.2226.1.18 void print_edges_lengths (std::vector< PrTriangle > tr, std::vector< UnfNodeType > nodes_unf)
```

print the lengths of the edges of the triangles in the unfolded sequence

```
30.2226.1.19 void print_list_int (std::list< int > & list_pivots_unf)
```

```
30.2226.1.20 void print_tr_vertices (std::vector< int > & tr, PrTriangulation_OP & t)
```

```
30.2226.1.21 void print_triangle_sequence (std::vector< PrTriangle > tr, std::vector< UnfNodeType > nodes_unf)
```

```
30.2226.1.22 void printNode (vector< UnfNodeType > & v)
```

```
30.2226.1.23 void printUnfoldedObjects (std::ofstream & os, std::vector< UnfNodeType > & nodes_unf, std::vector<
 PrTriangle > & tr_seq_unf, std::list< int > & list_pivots_unf)
```

```
30.2226.1.24 void printUnfoldedPath (std::ofstream & os, std::vector< UnfNodeType > & nodes_unf, std::list< int > &
 list_pivots_unf)
```

```
30.2226.1.25 void printUnfoldedPath (std::ofstream & os, std::vector< Go::Vector2D > path)
```

```
30.2226.1.26 void printUnfoldedSequence (std::ofstream & os, std::vector< UnfNodeType > & nodes_unf, std::vector<
 PrTriangle > & tr_seq_unf)
```

```
30.2226.1.27 Vector2D pt2D_from_ratio_on_edge (double r, Vector2D a, Vector2D b)
```

Computes a 2D point from its position (ratio) on an edge.

30.2226.1.28 `Go::Vector3D pt3D_from_ratio_on_edge ( double r, Go::Vector3D A, Go::Vector3D B )`

Computes a 3D point from its position (ratio) on an edge.

30.2226.1.29 `void rotation2D ( const Go::Vector2D a, const double theta, const Go::Vector2D c, Go::Vector2D & vect )`

30.2226.1.30 `int same_side ( const Go::Vector2D pt1, const Go::Vector2D pt2, const Go::Vector2D ptA, const Go::Vector2D ptB )`

check if pt1 and pt2 are both on the same side of [ptA ptB]

30.2226.1.31 `double segment_intersection ( Go::Vector2D a, Go::Vector2D b, Go::Vector2D c, Go::Vector2D d )`

intersection between [ab] and [cd]. returns the position of the intersection on the oriented segment [cd] given by a ratio

30.2226.1.32 `vector<Vector3D> shortest_path_triangle_sequence ( const PrTriangulation_OP & t, int sce_vertex, int dest_vertex, const vector< int > & tr_seq3d, list< int > & list_pivots, double & length )`

input: triangulation and triangle sequence. output:

- shortest path in that sequence = polygonal line given by its 3D vertices.
- list of pivot vertices, ie vertices of the triangulation belonging to this shortest path.
- length of this shortest path.

30.2226.1.33 `void shortest_path_triangle_sequence ( const PrTriangulation_OP & t, const int & sce_vertex, const int & dest_vertex, vector< int > & tr_seq3d, list< int > & list_pivots, std::vector< EdgeType > & edge_seq3d, std::vector< double > & ratio_on_edge, double & length )`

input: triangulation and triangle sequence. output:

- shortest path in that sequence = polygonal line given by its 3D vertices.
- list of pivot vertices, ie vertices of the triangulation. belonging to this shortest path.
- length of this shortest path.

30.2226.1.34 `vector<double> shortest_path_triangle_sequence_2D ( const PrTriangulation_OP & t, const int sce_vertex, const int dest_vertex, const int sce_vertexUnf, int dest_vertexUnf, const std::vector< int > & tr_seq3d, std::vector< PrTriangle > & tr_seq_unf, std::list< int > & list_pivots_unf, std::vector< UnfNodeType > nodes_unf, double & length )`

computes the shortest path in an ordered 2d sequence of triangles, corresponding to a 3d sequence that was unfolded



30.2226.1.35 `int third_vertex ( PrTriangle Tr, EdgeType e )`

third vertex of triangle *t* not belonging to edge *e*

30.2226.1.36 `void unfolding_first_triangle ( const PrTriangulation_OP & t, const vector< int > & tr_seq, vector< PrTriangle > & tr_seq_unf, vector< UnfNodeType > & nodes_unf )`

first triangle unfolded with 1st vertex = (0,0), 2nd on the x-axis, 3rd in the  $y > 0$  half plane

30.2226.1.37 `void unfolding_triangle_sequence ( const PrTriangulation_OP & t, const vector< int > & tr_seq3d, vector< PrTriangle > & tr_seq_unf, const vector< EdgeType > & edge_seq3d, vector< EdgeType > & edge_seq_unf, vector< UnfNodeType > & nodes_unf )`

unfolding of a sequence of triangle 3D from a triangulation. returns a set of unfolded nodes, a sequence of unfolded triangles and a sequence of unfolded edges

30.2226.1.38 `void unfolding_vertex ( const PrTriangulation_OP & t, const vector< int > & tr_seq3d, vector< PrTriangle > & tr_seq_unf, const int vertex, int & vertex_unf )`

associates the unfolded node to a vertex of the sequence (for example for the source and destination vertices)

30.2226.1.39 `double vect_prod2D ( Go::Vector2D a, Go::Vector2D b, Go::Vector2D c, Go::Vector2D d )`

30.2226.1.40 `Go::Vector3D vector_U ( const Go::Vector3D W )`

first axis/vector in the current triangle plane for the local frame

30.2226.1.41 `Go::Vector3D vector_V ( const Go::Vector3D W, const Go::Vector3D U )`

second axis/vector in the current triangle plane for the local frame

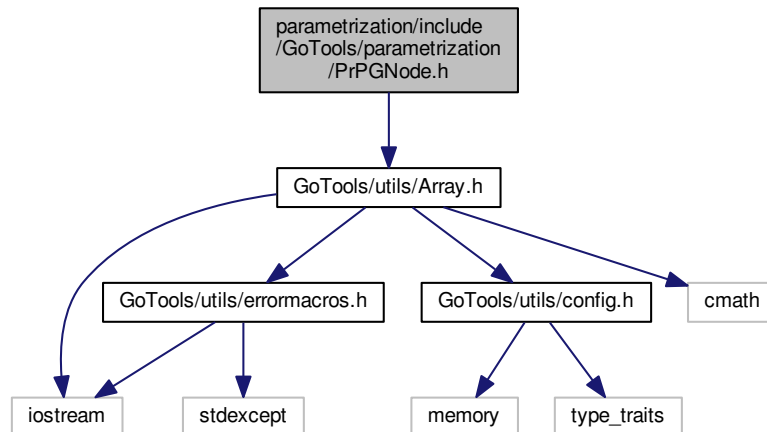
30.2226.1.42 `Go::Vector3D vector_W ( Go::Vector3D a, Go::Vector3D b, Go::Vector3D c )`

axis and normal vector in the current triangle plane for the local frame

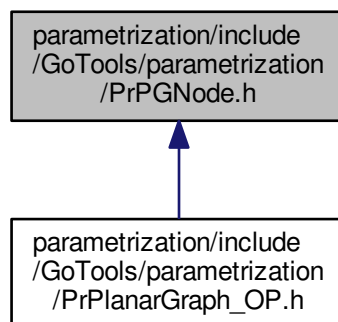
## 30.2227 parametrization/include/GoTools/parametrization/PrPGNode.h File Reference

```
#include "GoTools/Utils/Array.h"
```

Include dependency graph for PrPGNode.h:



This graph shows which files directly or indirectly include this file:

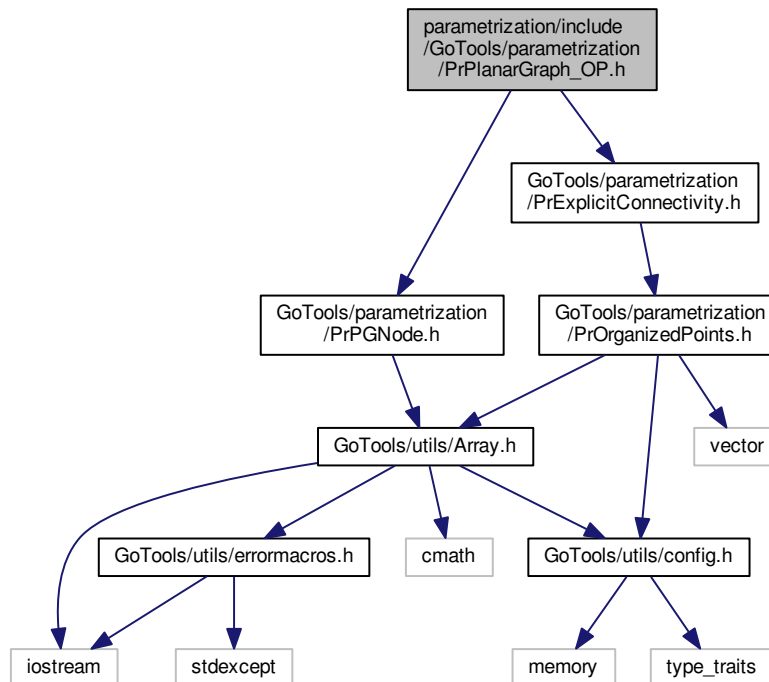


## Classes

- class [PrPGNode](#)

## 30.2228 parametrization/include/GoTools/parametrization/PrPlanarGraph\_OP.h File Reference

```
#include "GoTools/parametrization/PrExplicitConnectivity.h"
#include "GoTools/parametrization/PrPGNode.h"
Include dependency graph for PrPlanarGraph_OP.h:
```



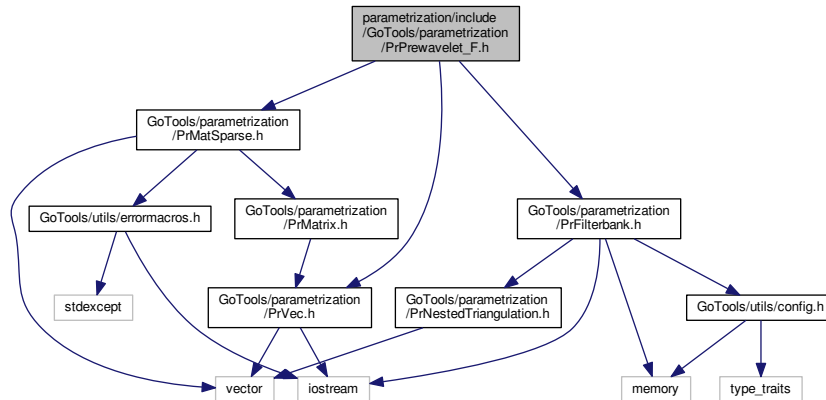
### Classes

- class [PrPlanarGraph\\_OP](#)

## 30.2229 parametrization/include/GoTools/parametrization/PrPrewavelet\_F.h File Reference

```
#include "GoTools/parametrization/PrFilterbank.h"
#include "GoTools/parametrization/PrMatSparse.h"
#include "GoTools/parametrization/PrVec.h"
```

Include dependency graph for PrPrewavelet\_F.h:



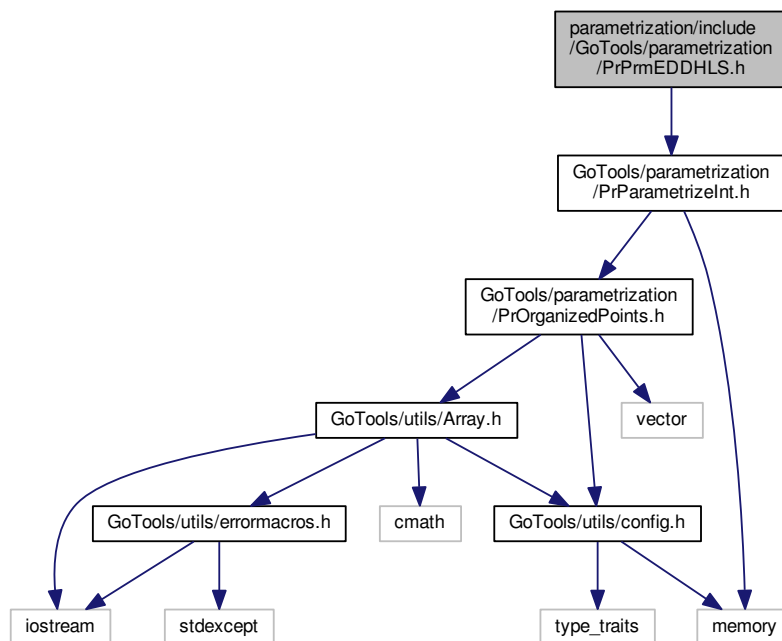
## Classes

- class [PrPrewavelet\\_F](#)

## 30.2230 parametrization/include/GoTools/parametrization/PrPrmEDDHLS.h File Reference

```
#include "GoTools/parametrization/PrParametrizeInt.h"
```

Include dependency graph for PrPrmEDDHLS.h:

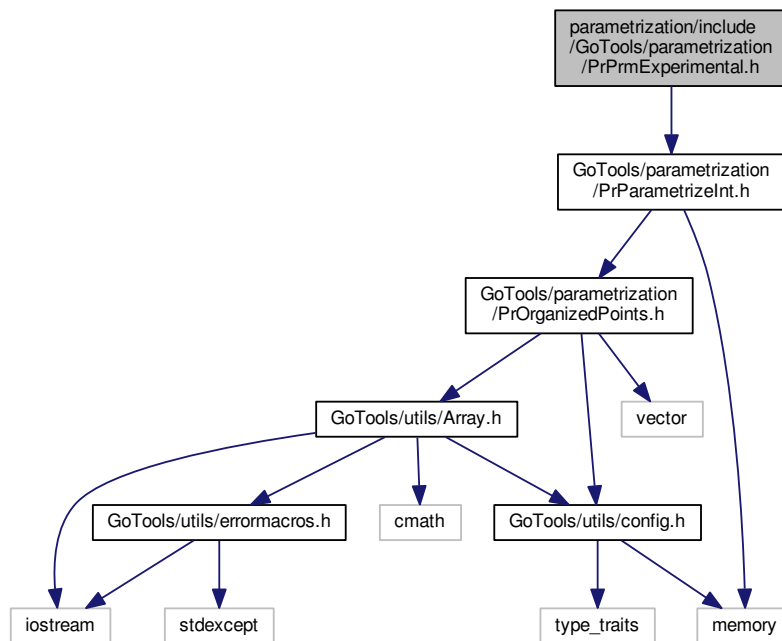


## Classes

- class [PrPrmEDDHLS](#)

### 30.2231 parametrization/include/GoTools/parametrization/PrPrmExperimental.h File Reference

```
#include "GoTools/parametrization/PrParametrizeInt.h"
Include dependency graph for PrPrmExperimental.h:
```



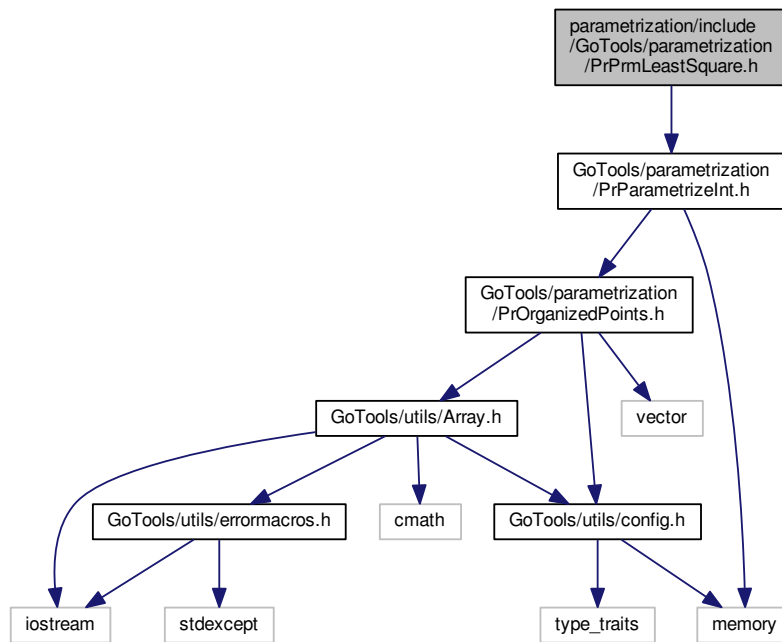
## Classes

- class [PrPrmExperimental](#)

### 30.2232 parametrization/include/GoTools/parametrization/PrPrmLeastSquare.h File Reference

```
#include "GoTools/parametrization/PrParametrizeInt.h"
```

Include dependency graph for PrPrmLeastSquare.h:



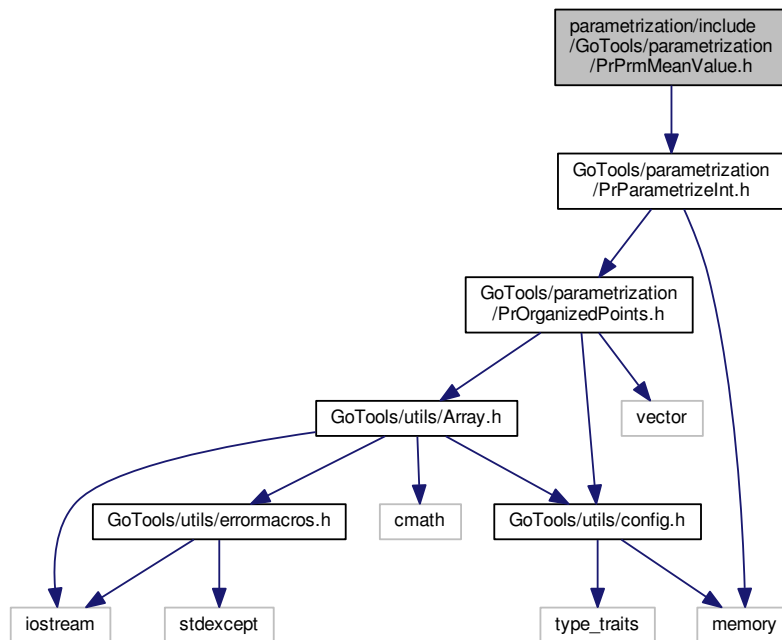
## Classes

- class [PrPrmLeastSquare](#)

## 30.2233 parametrization/include/GoTools/parametrization/PrPrmMeanValue.h File Reference

```
#include "GoTools/parametrization/PrParametrizeInt.h"
```

Include dependency graph for PrPrmMeanValue.h:



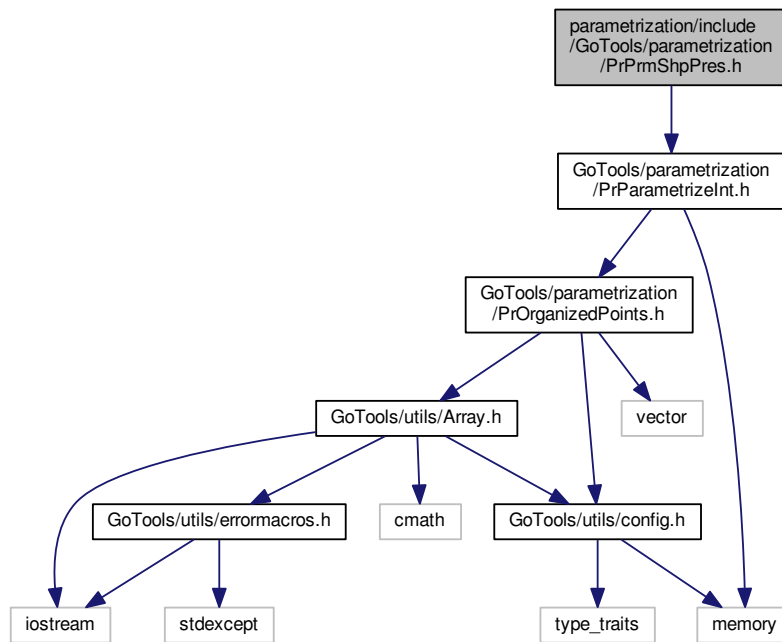
## Classes

- class [PrPrmMeanValue](#)

## 30.2234 parametrization/include/GoTools/parametrization/PrPrmShpPres.h File Reference

```
#include "GoTools/parametrization/PrParametrizeInt.h"
```

Include dependency graph for PrPmShpPres.h:



## Classes

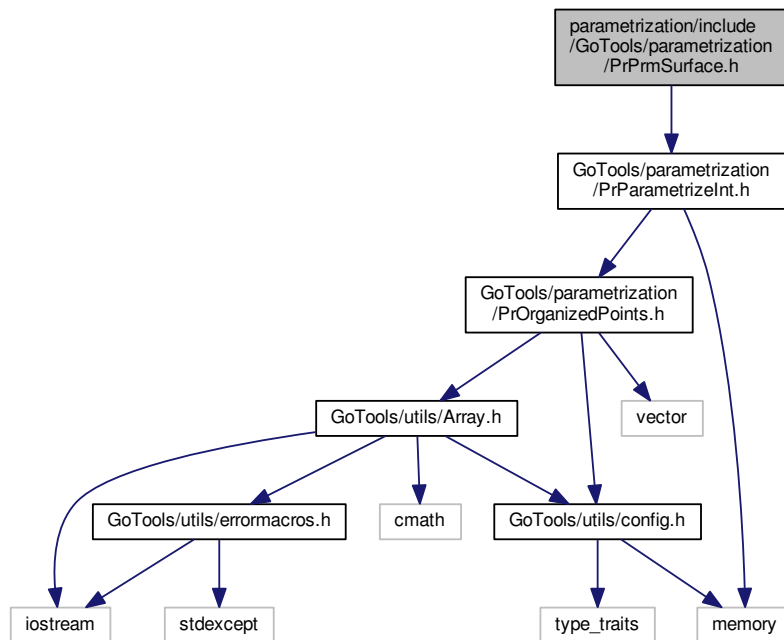
- class [PrPmShpPres](#)

**30.2235** [parametrization/include/GoTools/parametrization/PrPmSurface.h](#) **File Reference**

```
#include "GoTools/parametrization/PrParametrizeInt.h"
```



Include dependency graph for PrPrmSurface.h:



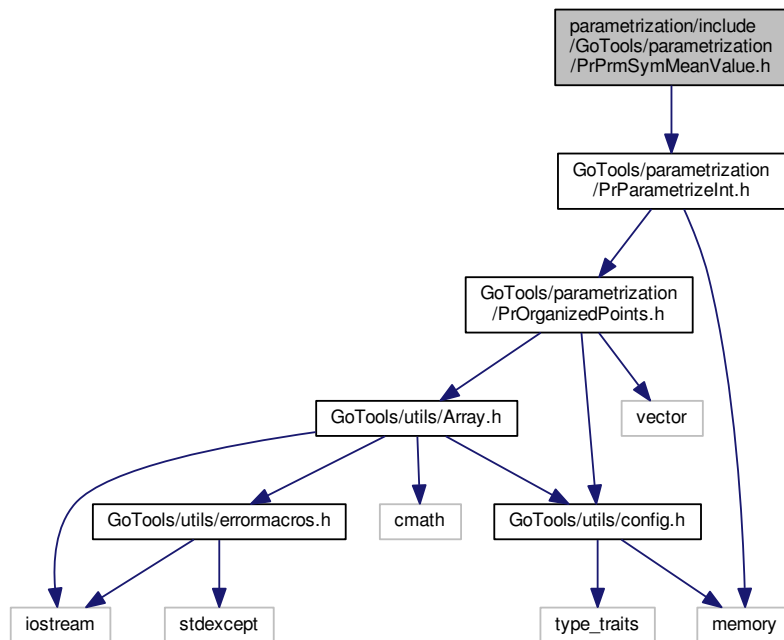
## Classes

- class [PrPrmSurface](#)

## 30.2236 parametrization/include/GoTools/parametrization/PrPrmSymMeanValue.h File Reference

```
#include "GoTools/parametrization/PrParametrizeInt.h"
```

Include dependency graph for PrPrmSymMeanValue.h:



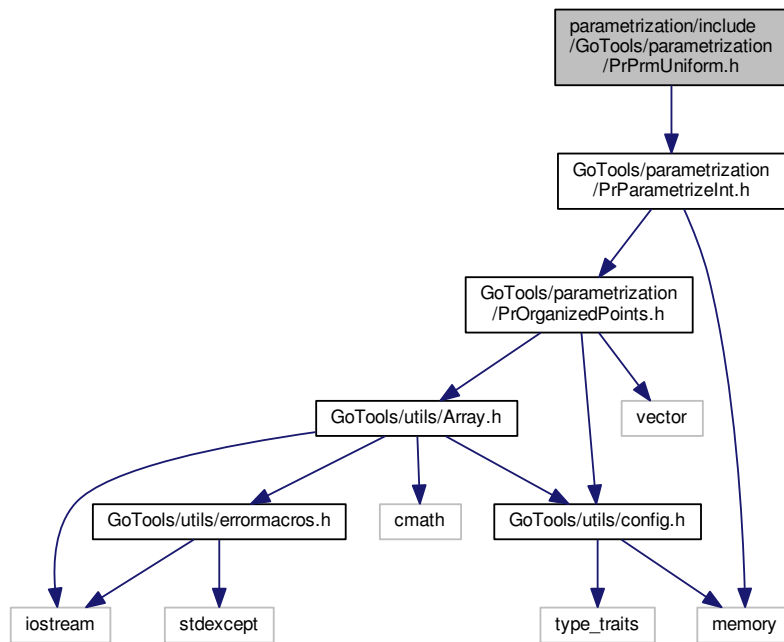
## Classes

- class [PrPrmSymMeanValue](#)

**30.2237** [parametrization/include/GoTools/parametrization/PrPrmUniform.h](#) File Reference

```
#include "GoTools/parametrization/PrParametrizeInt.h"
```

Include dependency graph for PrPrmUniform.h:



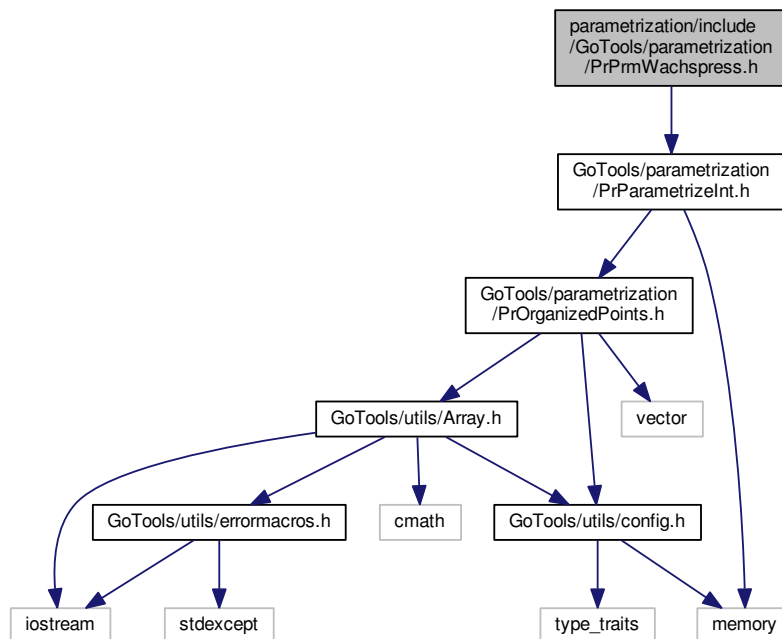
## Classes

- class [PrPrmUniform](#)

## 30.2238 parametrization/include/GoTools/parametrization/PrPrmWachspress.h File Reference

```
#include "GoTools/parametrization/PrParametrizeInt.h"
```

Include dependency graph for PrPrmWachspress.h:



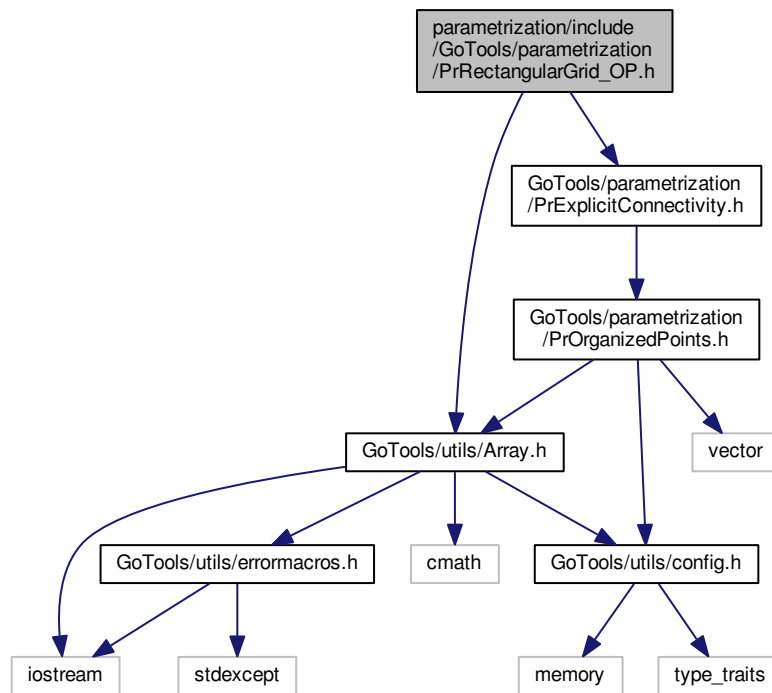
## Classes

- class [PrPrmWachspress](#)

## 30.2239 parametrization/include/GoTools/parametrization/PrRectangularGrid\_OP.h File Reference

```
#include "GoTools/parametrization/PrExplicitConnectivity.h"
#include "GoTools/Utils/Array.h"
```

Include dependency graph for PrRectangularGrid\_OP.h:



## Classes

- class [PrRectangularGrid\\_OP](#)

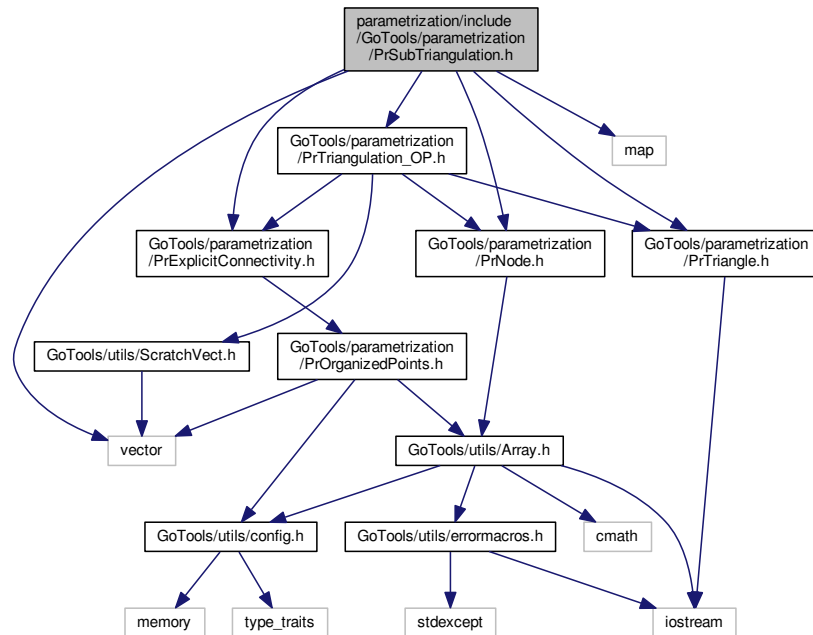
## 30.2240 parametrization/include/GoTools/parametrization/PrSubTriangulation.h File Reference

```

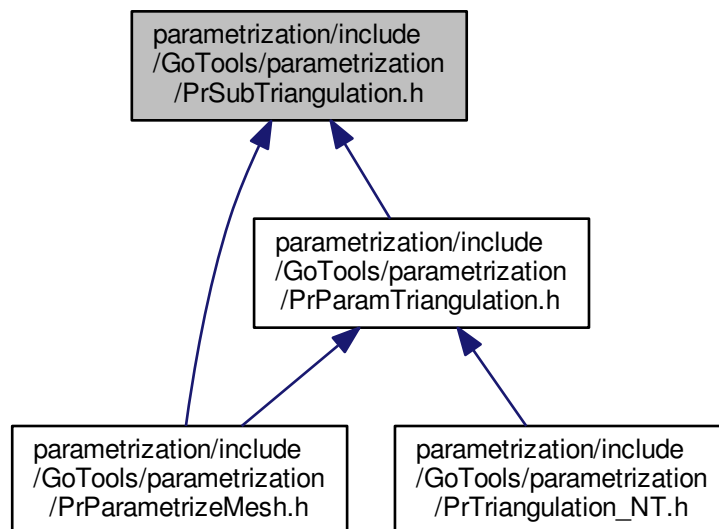
#include "GoTools/parametrization/PrExplicitConnectivity.h"
#include "GoTools/parametrization/PrNode.h"
#include "GoTools/parametrization/PrTriangle.h"
#include "GoTools/parametrization/PrTriangulation_OP.h"
#include <vector>
#include <map>

```

Include dependency graph for PrSubTriangulation.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [PrSubTriangulation](#)

## 30.2241 parametrization/include/GoTools/parametrization/PrTexture.h File Reference

### Functions

- void [printTexture](#) (std::ostream &os, [PrOrganizedPoints](#) &triang)

Method of texture mapping in the class [PrOrganizedPoints](#).

### 30.2241.1 Function Documentation

30.2241.1.1 void [printTexture](#) ( std::ostream & os, [PrOrganizedPoints](#) & triang )

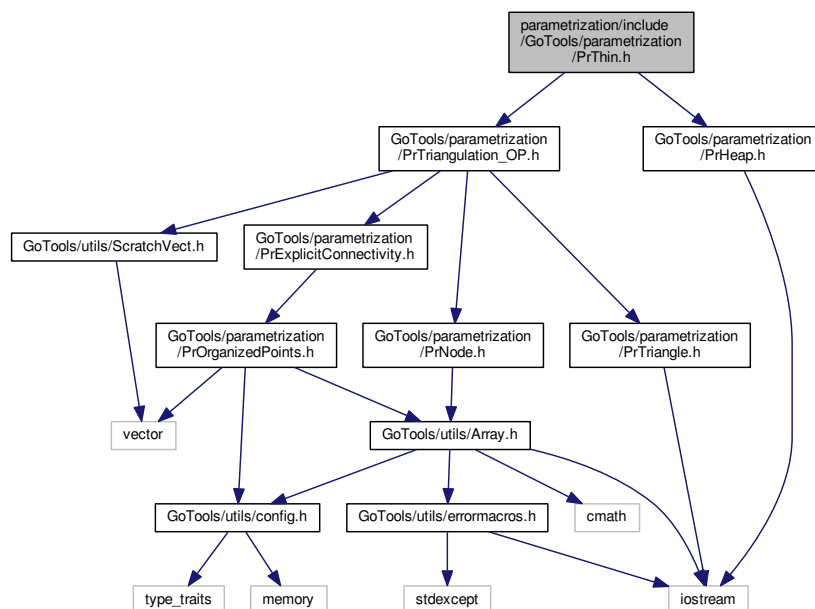
Method of texture mapping in the class [PrOrganizedPoints](#).

## 30.2242 parametrization/include/GoTools/parametrization/PrThin.h File Reference

```
#include "GoTools/parametrization/PrTriangulation_OP.h"
```

```
#include "GoTools/parametrization/PrHeap.h"
```

Include dependency graph for PrThin.h:



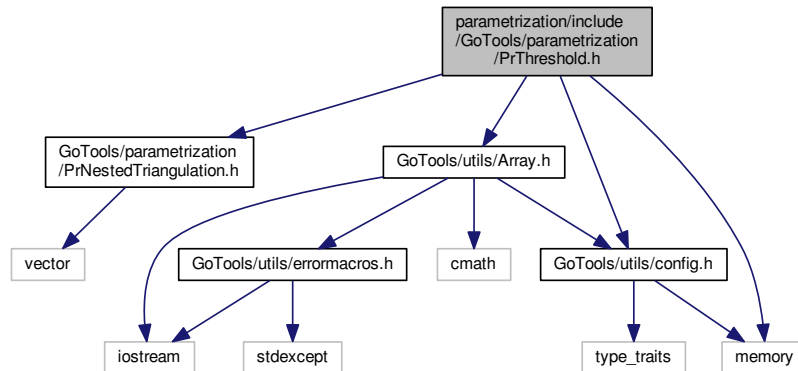
### Classes

- class [PrThin](#)

### 30.2243 parametrization/include/GoTools/parametrization/PrThreshold.h File Reference

```
#include "GoTools/parametrization/PrNestedTriangulation.h"
#include "GoTools/utils/Array.h"
#include "GoTools/utils/config.h"
#include <memory>
```

Include dependency graph for PrThreshold.h:



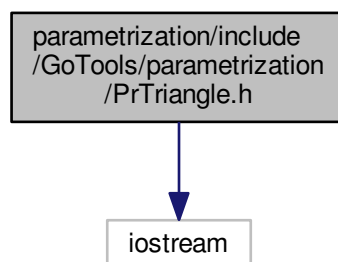
#### Classes

- class [PrThreshold](#)

### 30.2244 parametrization/include/GoTools/parametrization/PrTriangle.h File Reference

```
#include <iostream>
```

Include dependency graph for PrTriangle.h:







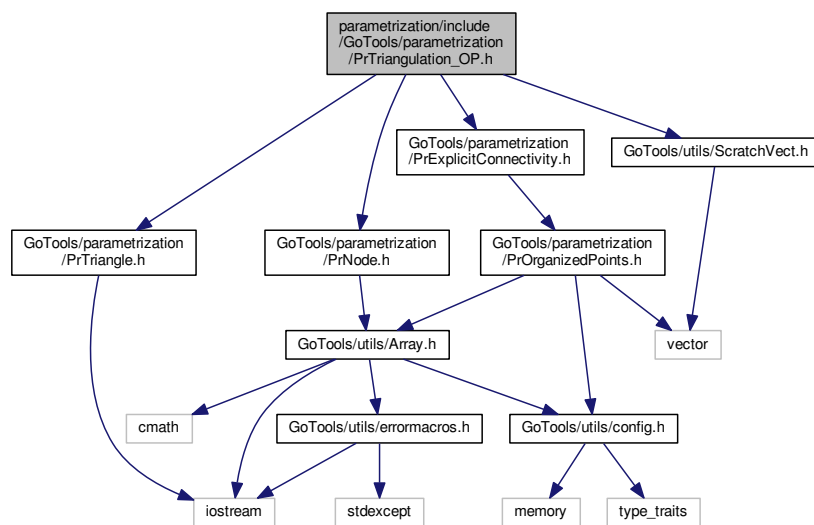
## Classes

- class [PrTriangulation\\_NT](#)

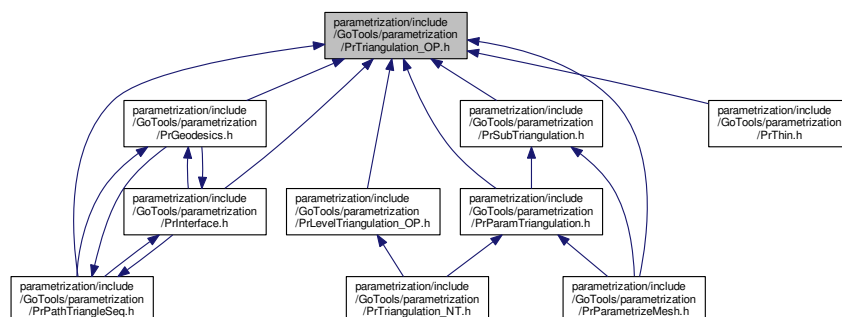
### 30.2246 parametrization/include/GoTools/parametrization/PrTriangulation\_OP.h File Reference

```
#include "GoTools/parametrization/PrExplicitConnectivity.h"
#include "GoTools/parametrization/PrNode.h"
#include "GoTools/parametrization/PrTriangle.h"
#include "GoTools/utils/ScratchVect.h"
```

Include dependency graph for PrTriangulation\_OP.h:



This graph shows which files directly or indirectly include this file:



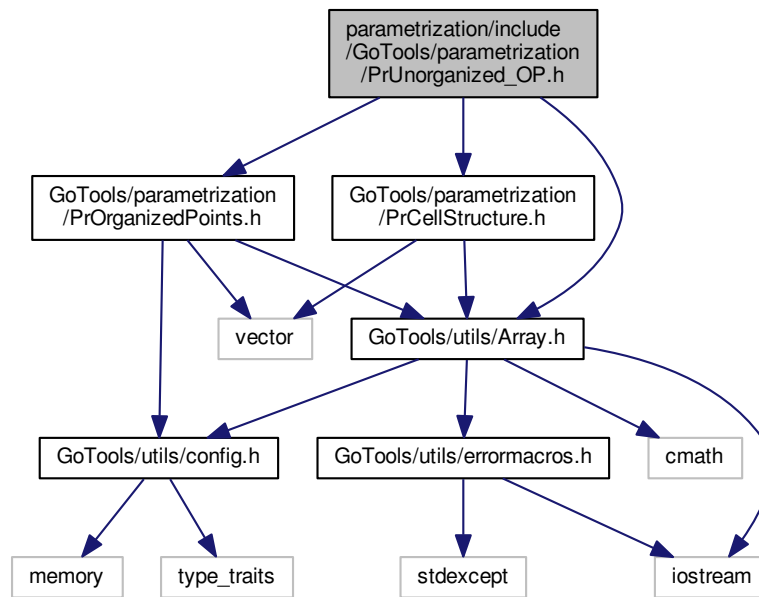
## Classes

- class [PrTriangulation\\_OP](#)

## 30.2247 parametrization/include/GoTools/parametrization/PrUnorganized\_OP.h File Reference

```
#include "GoTools/parametrization/PrOrganizedPoints.h"
#include "GoTools/parametrization/PrCellStructure.h"
#include "GoTools/utils/Array.h"
```

Include dependency graph for PrUnorganized\_OP.h:



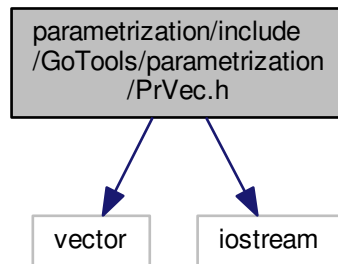
### Classes

- class [PrUnorganized\\_OP](#)

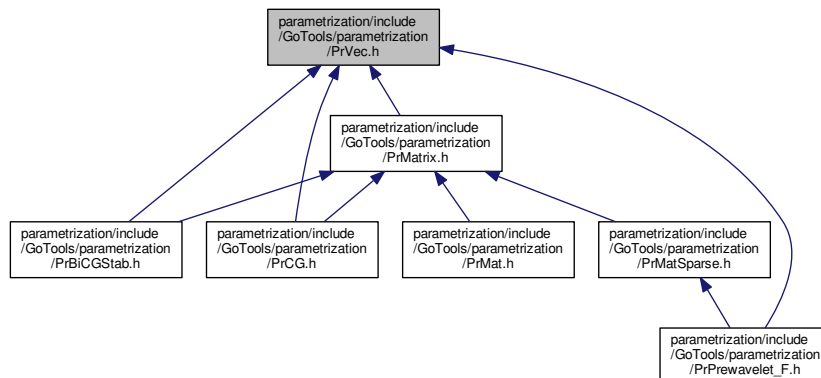
## 30.2248 parametrization/include/GoTools/parametrization/PrVec.h File Reference

```
#include <vector>
#include <iostream>
```

Include dependency graph for PrVec.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [PrVec](#)

## 30.2249 parametrization/include/GoTools/parametrization/PrWaveletUtil.h File Reference

## Functions

- [double theta](#) (int j, int k, int i, int deg, bool isBoundary)

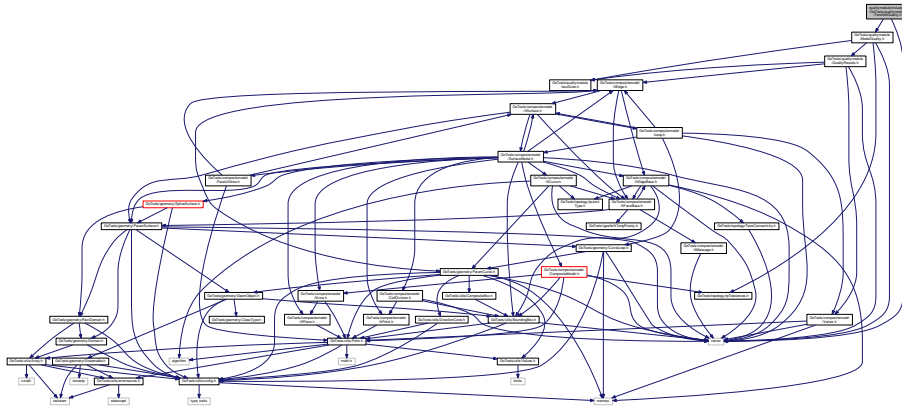
### 30.2249.1 Function Documentation

30.2249.1.1 `double theta ( int j, int k, int i, int deg, bool isBoundary )`

## 30.2250 qualitymodule/include/GoTools/qualitymodule/FaceSetQuality.h File Reference

```
#include "GoTools/qualitymodule/ModelQuality.h"
#include <vector>
```

Include dependency graph for FaceSetQuality.h:



### Classes

- class [Go::FaceSetQuality](#)

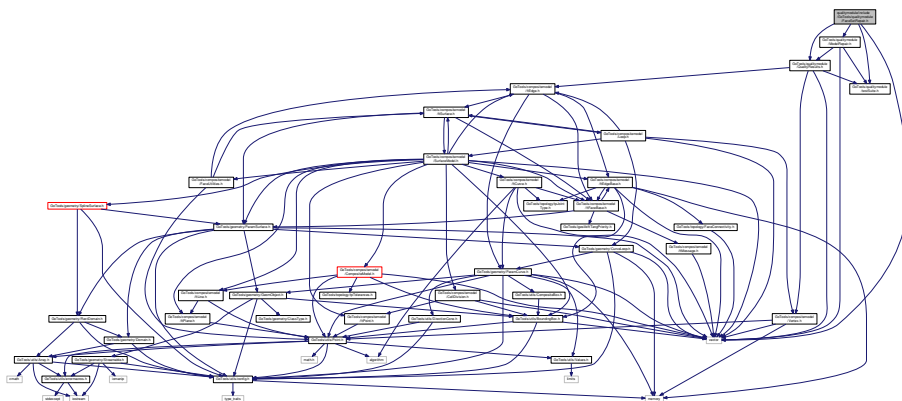
### Namespaces

- [Go](#)

## 30.2251 qualitymodule/include/GoTools/qualitymodule/FaceSetRepair.h File Reference

```
#include "GoTools/qualitymodule/ModelRepair.h"
#include "GoTools/qualitymodule/QualityResults.h"
#include "GoTools/qualitymodule/testSuite.h"
#include <vector>
```

Include dependency graph for FaceSetRepair.h:



## Classes

- class [Go::FaceSetRepair](#)

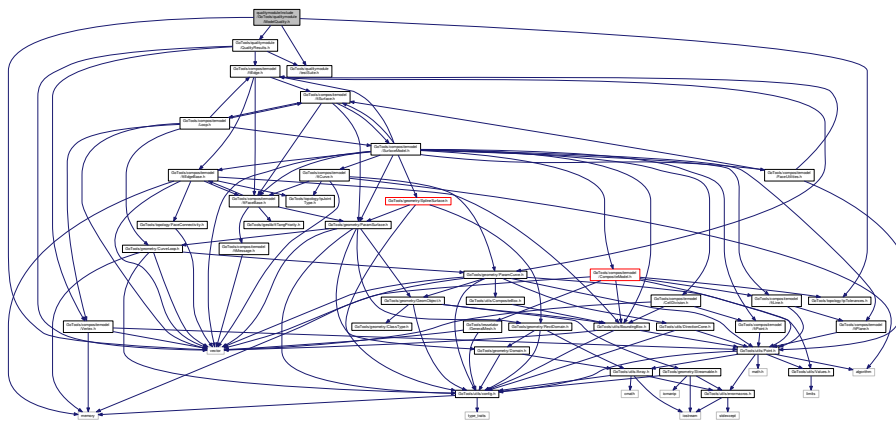
## Namespaces

- [Go](#)

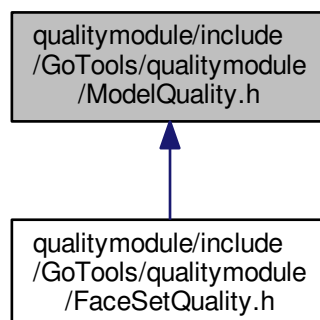
## 30.2252 qualitymodule/include/GoTools/qualitymodule/ModelQuality.h File Reference

```
#include "GoTools/topology/tpTolerances.h"
#include "GoTools/qualitymodule/QualityResults.h"
#include "GoTools/qualitymodule/testSuite.h"
#include <vector>
```

Include dependency graph for ModelQuality.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [Go::ModelQuality](#)

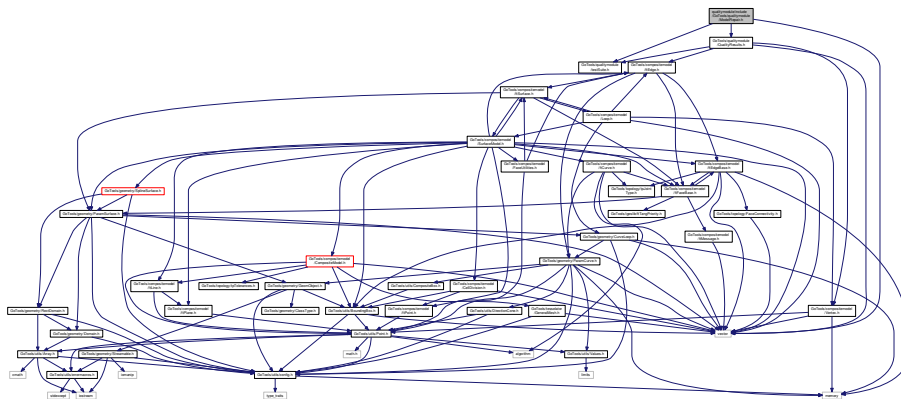
## Namespaces

- [Go](#)

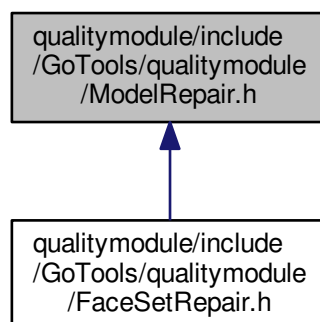
## 30.2253 qualitymodule/include/GoTools/qualitymodule/ModelRepair.h File Reference

```
#include "GoTools/qualitymodule/QualityResults.h"
#include "GoTools/qualitymodule/testSuite.h"
#include <vector>
```

Include dependency graph for ModelRepair.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [Go::ModelRepair](#)

## Namespaces

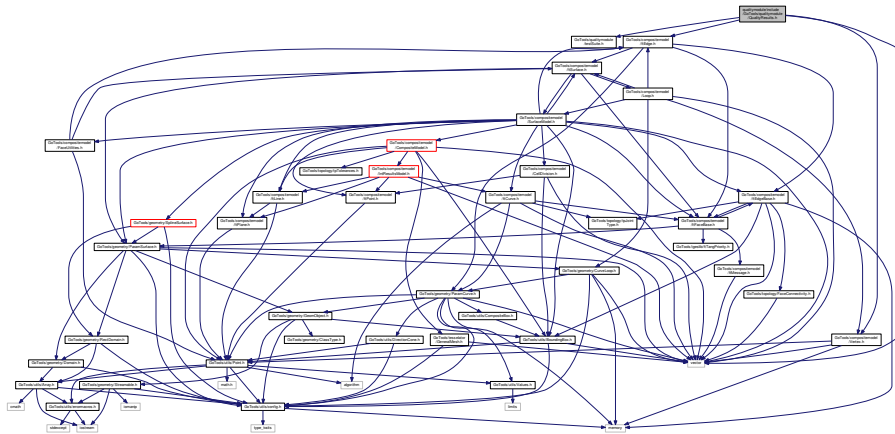
- [Go](#)

### 30.2254 qualitymodule/include/GoTools/qualitymodule/qualitymodule-doxymain.h File Reference

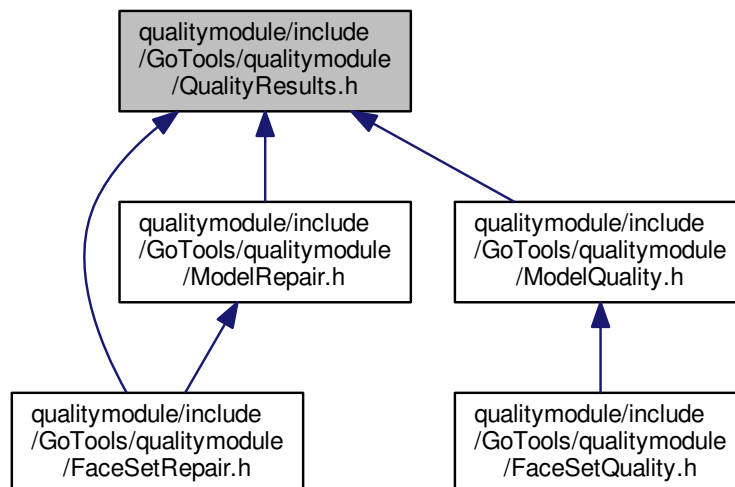
### 30.2255 qualitymodule/include/GoTools/qualitymodule/QualityResults.h File Reference

```
#include "GoTools/qualitymodule/testSuite.h"
#include "GoTools/compositemodel/ftEdge.h"
#include "GoTools/compositemodel/Vertex.h"
#include <vector>
```

Include dependency graph for QualityResults.h:



This graph shows which files directly or indirectly include this file:





## Classes

- class [Go::QualityResults](#)

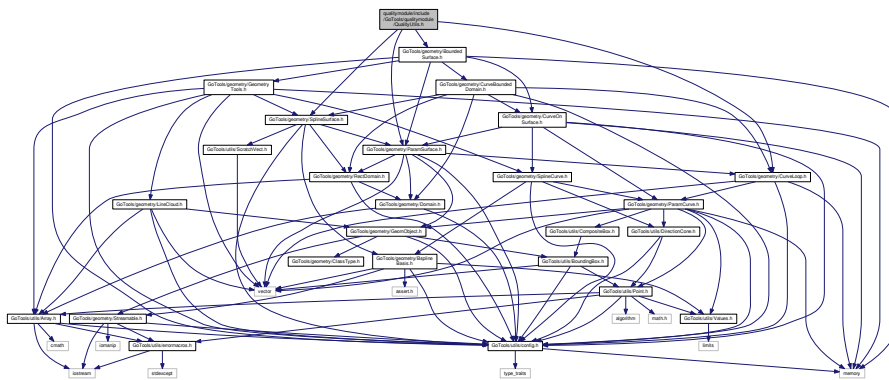
## Namespaces

- [Go](#)

## 30.2256 qualitymodule/include/GoTools/qualitymodule/QualityUtils.h File Reference

```
#include "GoTools/geometry/ParamSurface.h"
#include "GoTools/geometry/SplineSurface.h"
#include "GoTools/geometry/BoundedSurface.h"
#include "GoTools/geometry/CurveLoop.h"
```

Include dependency graph for QualityUtils.h:



## Namespaces

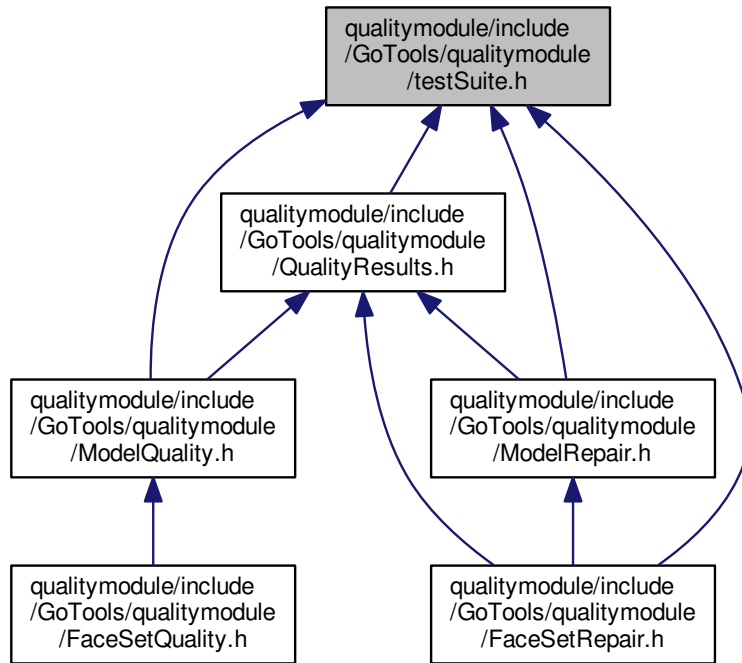
- [Go](#)
- [Go::qualityUtils](#)

## Functions

- [bool Go::qualityUtils::isSliverFace](#) (shared\_ptr< ParamSurface >, double thickness, double factor=2.0)
- [bool Go::qualityUtils::isSliverFace](#) (const SplineSurface &sf, double thickness, double factor=2.0)
- [bool Go::qualityUtils::isSliverFace](#) (const BoundedSurface &sf, double thickness, double factor=2.0)
- [bool Go::qualityUtils::isSliverFace2](#) (const BoundedSurface &sf, double thickness, double factor=2.0)
- [bool Go::qualityUtils::hasIndistinctKnots](#) (shared\_ptr< ParamSurface > surf, double tol, std::vector< shared\_ptr< ParamCurve > > &trim\_cv\_knots)
- [double Go::qualityUtils::estimateArea](#) (shared\_ptr< ParamSurface > surf)
- [double Go::qualityUtils::estimateLoopArea](#) (shared\_ptr< CurveLoop > loop)

## 30.2257 qualitymodule/include/GoTools/qualitymodule/testSuite.h File Reference

This graph shows which files directly or indirectly include this file:



### Namespaces

- [Go](#)

### Macros

- `#define TEST_SUITE_SIZE 36`

### Enumerations

- `enum Go::testSuite {`  
`Go::IDENTICAL_VERTICES = 0, Go::IDENTICAL_EDGES = 1, Go::EMBEDDED_EDGES = 2, Go::IDENTICAL_FACES = 3,`  
`Go::EMBEDDED_FACES = 4, Go::MINI_CURVE = 5, Go::MINI_SURFACE = 6, Go::MINI_EDGE = 7,`  
`Go::MINI_FACE = 8, Go::SLIVER_FACE = 9, Go::NARROW_REGION = 10, Go::DEGEN_SRF_BD = 11,`  
`Go::DEGEN_SRF_CORNER = 12, Go::VANISHING_TANGENT = 13, Go::VANISHING_NORMAL = 14,`

```

Go::EDGE_VERTEX_DISTANCE = 15,
Go::FACE_VERTEX_DISTANCE = 16, Go::FACE_EDGE_DISTANCE = 17, Go::EDGE_POSITION_DISC←
ONT = 18, Go::EDGE_TANGENTIAL_DISCONT = 19,
Go::FACE_POSITION_DISCONT = 20, Go::FACE_TANGENTIAL_DISCONT = 21, Go::LOOP_CONSIST←
ENCY = 22, Go::LOOP_ORIENTATION = 23,
Go::FACE_ORIENTATION = 24, Go::CV_G1DISCONT = 25, Go::CV_C1DISCONT = 26, Go::SF_G1DIS←
CONT = 27,
Go::SF_C1DISCONT = 28, Go::CV_CURVATURE_RADIUS = 29, Go::SF_CURVATURE_RADIUS = 30,
Go::EDGE_ACUTE_ANGLE = 31,
Go::FACE_ACUTE_ANGLE = 32, Go::LOOP_INTERSECTION = 33, Go::LOOP_SELF_INTERSECTION =
34, Go::INDISTINCT_KNOTS = 35 }

```

### 30.2257.1 Macro Definition Documentation

#### 30.2257.1.1 #define TEST\_SUITE\_SIZE 36

Definition at line 90 of file testSuite.h.

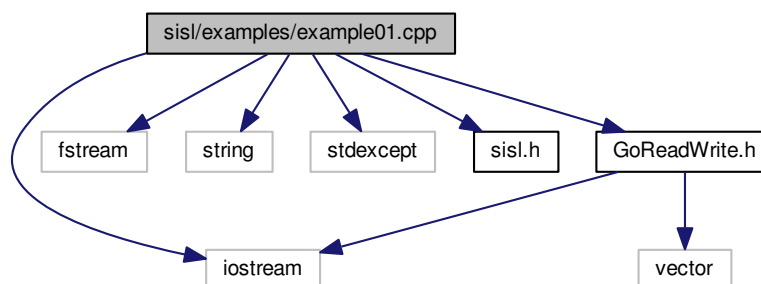
## 30.2258 sisl/examples/example01.cpp File Reference

```

#include <iostream>
#include <fstream>
#include <string>
#include <stdexcept>
#include "sisl.h"
#include "GoReadWrite.h"

```

Include dependency graph for example01.cpp:



### Functions

- int [main](#) (int avnum, char \*\*vararg)

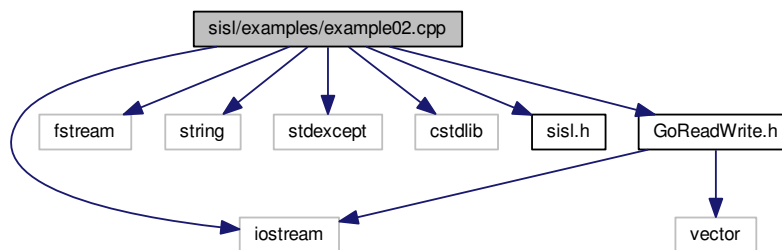
### 30.2258.1 Function Documentation

#### 30.2258.1.1 int main ( int avnum, char \*\* vararg )

Definition at line 81 of file example01.cpp.

## 30.2259 sisl/examples/example02.cpp File Reference

```
#include <iostream>
#include <fstream>
#include <string>
#include <stdexcept>
#include <cstdlib>
#include "sisl.h"
#include "GoReadWrite.h"
Include dependency graph for example02.cpp:
```



### Functions

- int [main](#) (int avnum, char \*\*vararg)

#### 30.2259.1 Function Documentation

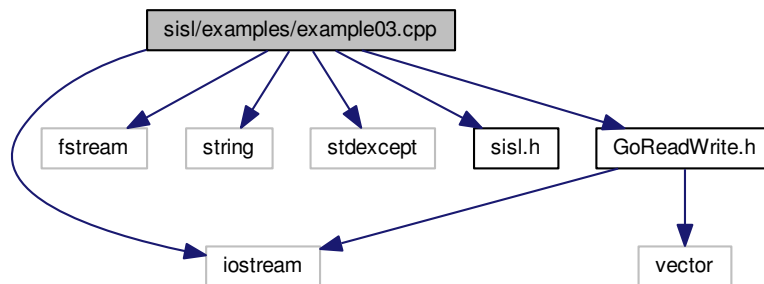
30.2259.1.1 int `main` ( int *avnum*, char \*\* *vararg* )

Definition at line 79 of file `example02.cpp`.

## 30.2260 sisl/examples/example03.cpp File Reference

```
#include <iostream>
#include <fstream>
#include <string>
#include <stdexcept>
#include "sisl.h"
#include "GoReadWrite.h"
```

Include dependency graph for example03.cpp:



## Functions

- int `main` (int `avnum`, char \*\*`vararg`)

### 30.2260.1 Function Documentation

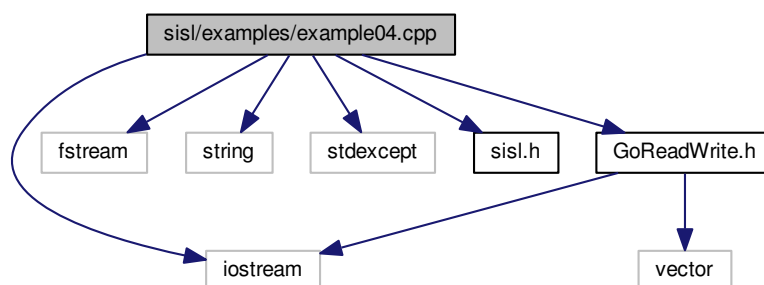
30.2260.1.1 int `main` ( int `avnum`, char \*\* `vararg` )

Definition at line 66 of file `example03.cpp`.

## 30.2261 sisl/examples/example04.cpp File Reference

```
#include <iostream>
#include <fstream>
#include <string>
#include <stdexcept>
#include "sisl.h"
#include "GoReadWrite.h"
```

Include dependency graph for example04.cpp:



## Functions

- int [main](#) (int avnum, char \*\*vararg)

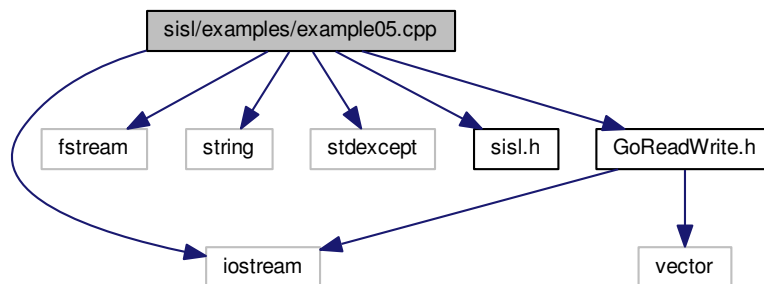
### 30.2261.1 Function Documentation

30.2261.1.1 int main ( int *avnum*, char \*\* *vararg* )

Definition at line 66 of file example04.cpp.

### 30.2262 sisl/examples/example05.cpp File Reference

```
#include <iostream>
#include <fstream>
#include <string>
#include <stdexcept>
#include "sisl.h"
#include "GoReadWrite.h"
Include dependency graph for example05.cpp:
```



## Functions

- int [main](#) (int avnum, char \*\*vararg)

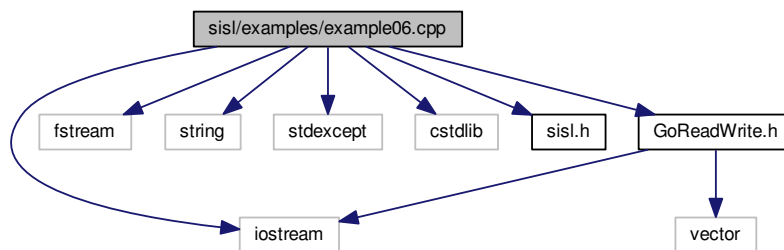
### 30.2262.1 Function Documentation

30.2262.1.1 int main ( int *avnum*, char \*\* *vararg* )

Definition at line 80 of file example05.cpp.

## 30.2263 sisl/examples/example06.cpp File Reference

```
#include <iostream>
#include <fstream>
#include <string>
#include <stdexcept>
#include <cstdlib>
#include "sisl.h"
#include "GoReadWrite.h"
Include dependency graph for example06.cpp:
```



### Functions

- int `main` (int `avnum`, char \*\*`vararg`)

#### 30.2263.1 Function Documentation

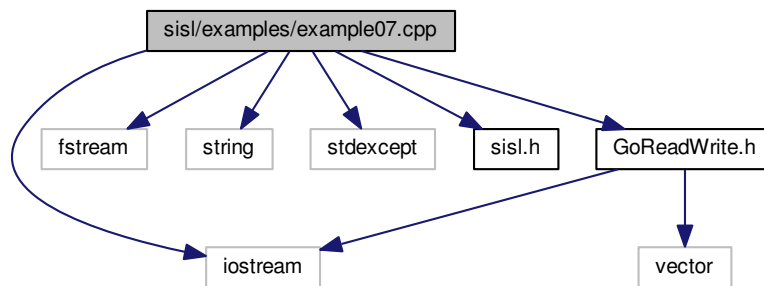
30.2263.1.1 int `main` ( int `avnum`, char \*\* `vararg` )

Definition at line 91 of file `example06.cpp`.

## 30.2264 sisl/examples/example07.cpp File Reference

```
#include <iostream>
#include <fstream>
#include <string>
#include <stdexcept>
#include "sisl.h"
#include "GoReadWrite.h"
```

Include dependency graph for example07.cpp:



## Functions

- int `main` (int `avnum`, char \*\*`vararg`)

### 30.2264.1 Function Documentation

30.2264.1.1 int `main` ( int `avnum`, char \*\* `vararg` )

Definition at line 61 of file `example07.cpp`.

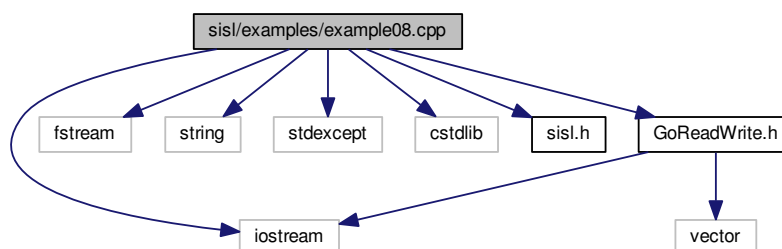
### 30.2265 sisl/examples/example08.cpp File Reference

```

#include <iostream>
#include <fstream>
#include <string>
#include <stdexcept>
#include <cstdlib>
#include "sisl.h"
#include "GoReadWrite.h"

```

Include dependency graph for example08.cpp:





## Functions

- int [main](#) (int avnum, char \*\*vararg)

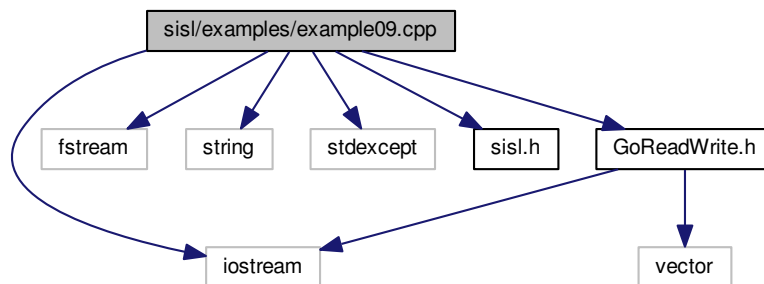
### 30.2265.1 Function Documentation

#### 30.2265.1.1 int main ( int *avnum*, char \*\* *vararg* )

Definition at line 90 of file example08.cpp.

## 30.2266 sisl/examples/example09.cpp File Reference

```
#include <iostream>
#include <fstream>
#include <string>
#include <stdexcept>
#include "sisl.h"
#include "GoReadWrite.h"
Include dependency graph for example09.cpp:
```



## Functions

- int [main](#) (int avnum, char \*\*vararg)

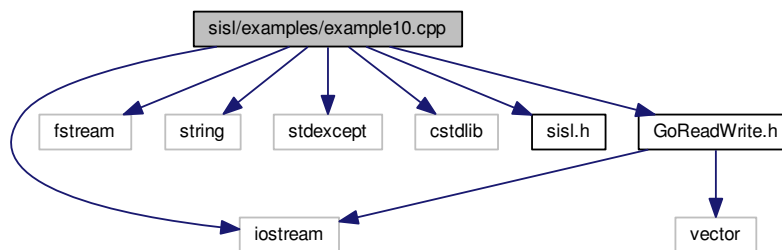
### 30.2266.1 Function Documentation

#### 30.2266.1.1 int main ( int *avnum*, char \*\* *vararg* )

Definition at line 92 of file example09.cpp.

## 30.2267 sisl/examples/example10.cpp File Reference

```
#include <iostream>
#include <fstream>
#include <string>
#include <stdexcept>
#include <cstdlib>
#include "sisl.h"
#include "GoReadWrite.h"
Include dependency graph for example10.cpp:
```



### Functions

- int [main](#) (int avnum, char \*\*vararg)

### 30.2267.1 Function Documentation

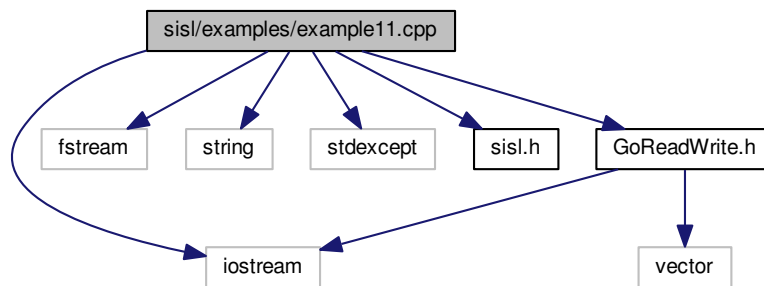
30.2267.1.1 int main ( int *avnum*, char \*\* *vararg* )

Definition at line 88 of file example10.cpp.

## 30.2268 sisl/examples/example11.cpp File Reference

```
#include <iostream>
#include <fstream>
#include <string>
#include <stdexcept>
#include "sisl.h"
#include "GoReadWrite.h"
```

Include dependency graph for example11.cpp:



## Functions

- int [main](#) (int avnum, char \*\*vararg)

### 30.2268.1 Function Documentation

30.2268.1.1 int [main](#) ( int *avnum*, char \*\* *vararg* )

Definition at line 68 of file example11.cpp.

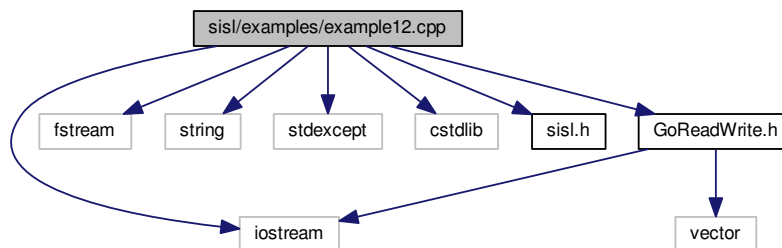
## 30.2269 sisl/examples/example12.cpp File Reference

```

#include <iostream>
#include <fstream>
#include <string>
#include <stdexcept>
#include <cstdlib>
#include "sisl.h"
#include "GoReadWrite.h"

```

Include dependency graph for example12.cpp:



## Functions

- int [main](#) (int avnum, char \*\*vararg)

### 30.2269.1 Function Documentation

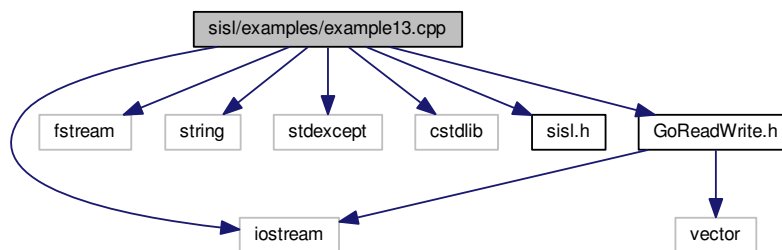
#### 30.2269.1.1 int main ( int *avnum*, char \*\* *vararg* )

Definition at line 68 of file example12.cpp.

### 30.2270 sisl/examples/example13.cpp File Reference

```
#include <iostream>
#include <fstream>
#include <string>
#include <stdexcept>
#include <cstdlib>
#include "sisl.h"
#include "GoReadWrite.h"
```

Include dependency graph for example13.cpp:



## Functions

- int [main](#) (int avnum, char \*\*vararg)

### 30.2270.1 Function Documentation

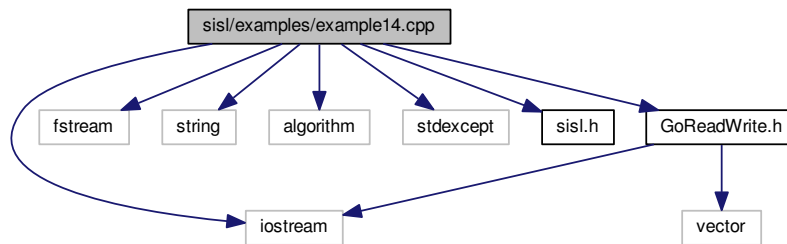
#### 30.2270.1.1 int main ( int *avnum*, char \*\* *vararg* )

Definition at line 70 of file example13.cpp.

## 30.2271 sisl/examples/example14.cpp File Reference

```
#include <iostream>
#include <fstream>
#include <string>
#include <algorithm>
#include <stdexcept>
#include "sisl.h"
#include "GoReadWrite.h"
```

Include dependency graph for example14.cpp:



### Functions

- int [main](#) (int avnum, char \*\*vararg)

#### 30.2271.1 Function Documentation

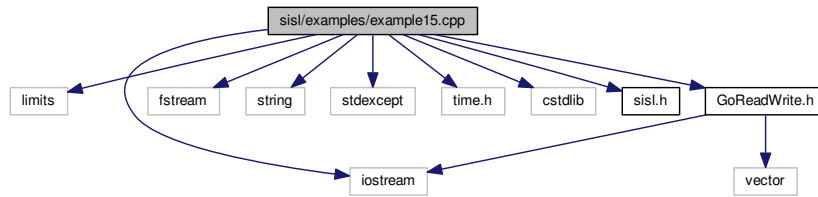
##### 30.2271.1.1 int main ( int *avnum*, char \*\* *vararg* )

Definition at line 85 of file example14.cpp.

## 30.2272 sisl/examples/example15.cpp File Reference

```
#include <limits>
#include <iostream>
#include <fstream>
#include <string>
#include <stdexcept>
#include <time.h>
#include <cstdlib>
#include "sisl.h"
#include "GoReadWrite.h"
```

Include dependency graph for example15.cpp:



## Functions

- int [main](#) (int varnum, char \*\*vararg)

### 30.2272.1 Function Documentation

30.2272.1.1 int main ( int *varnum*, char \*\* *vararg* )

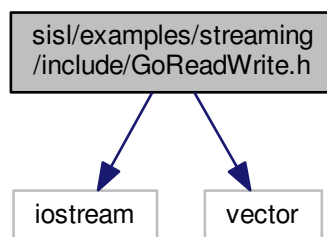
Definition at line 106 of file example15.cpp.

### 30.2273 sis/examples/streaming/include/GoReadWrite.h File Reference

```
#include <iostream>
```

```
#include <vector>
```

Include dependency graph for GoReadWrite.h:



This graph shows which files directly or indirectly include this file:



## Functions

- `SISLCurve * readGoCurve (std::istream &go_stream)`
- `SISLSurf * readGoSurface (std::istream &go_stream)`
- `void writeGoCurve (SISLCurve *curve, std::ostream &go_stream)`
- `void writeGoSurface (SISLSurf *surf, std::ostream &go_stream)`
- `void writeGoPoints (int num_points, double *coords, std::ostream &go_stream)`
- `void readGoPoints (std::vector< double > &coords, std::istream &go_stream)`

### 30.2273.1 Function Documentation

30.2273.1.1 `SISLCurve*` `readGoCurve ( std::istream & go_stream )`

30.2273.1.2 `void` `readGoPoints ( std::vector< double > & coords, std::istream & go_stream )`

Definition at line 248 of file `GoReadWrite.cpp`.

30.2273.1.3 `SISLSurf*` `readGoSurface ( std::istream & go_stream )`

30.2273.1.4 `void` `writeGoCurve ( SISLCurve * curve, std::ostream & go_stream )`

Definition at line 161 of file `GoReadWrite.cpp`.

30.2273.1.5 `void` `writeGoPoints ( int num_points, double * coords, std::ostream & go_stream )`

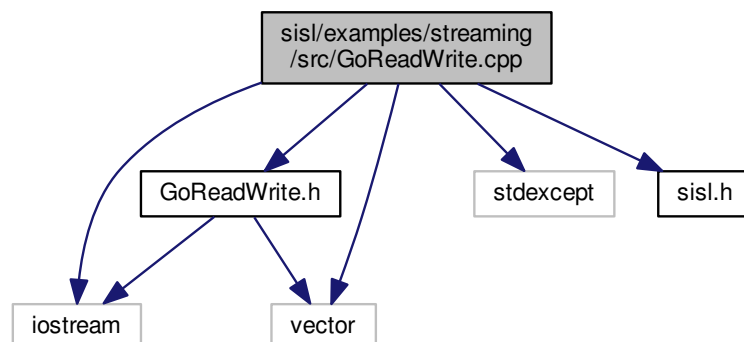
Definition at line 222 of file `GoReadWrite.cpp`.

30.2273.1.6 `void` `writeGoSurface ( SISLSurf * surf, std::ostream & go_stream )`

## 30.2274 sisl/examples/streaming/src/GoReadWrite.cpp File Reference

```
#include "GoReadWrite.h"
#include <vector>
#include <iostream>
#include <stdexcept>
#include "sisl.h"
```

Include dependency graph for `GoReadWrite.cpp`:



## Functions

- `SISLSurf * readGoSurface` (istream &go\_stream)
- void `writeGoSurface` (SISLSurf \*surf, ostream &go\_stream)
- void `writeGoCurve` (SISLCurve \*curve, std::ostream &go\_stream)
- `SISLCurve * readGoCurve` (istream &go\_stream)
- void `writeGoPoints` (int num\_points, double \*coords, std::ostream &go\_stream)
- void `readGoPoints` (std::vector< double > &coords, std::istream &go\_stream)

### 30.2274.1 Function Documentation

#### 30.2274.1.1 SISLCurve\* readGoCurve ( istream & go\_stream )

Definition at line 191 of file GoReadWrite.cpp.

#### 30.2274.1.2 void readGoPoints ( std::vector< double > & coords, std::istream & go\_stream )

Definition at line 248 of file GoReadWrite.cpp.

#### 30.2274.1.3 SISLSurf\* readGoSurface ( istream & go\_stream )

Definition at line 89 of file GoReadWrite.cpp.

#### 30.2274.1.4 void writeGoCurve ( SISLCurve \* curve, std::ostream & go\_stream )

Definition at line 161 of file GoReadWrite.cpp.

#### 30.2274.1.5 void writeGoPoints ( int num\_points, double \* coords, std::ostream & go\_stream )

Definition at line 222 of file GoReadWrite.cpp.

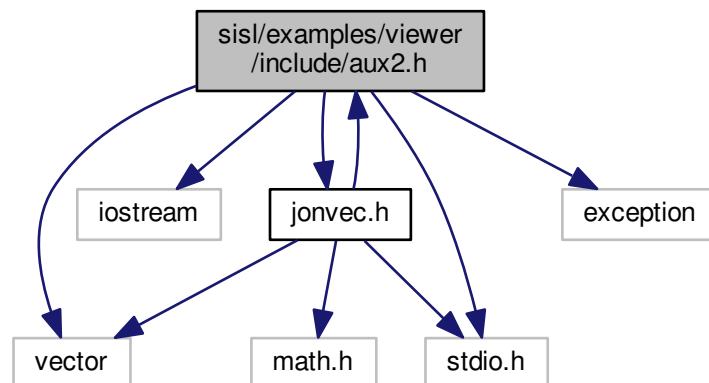
#### 30.2274.1.6 void writeGoSurface ( SISLSurf \* surf, ostream & go\_stream )

Definition at line 130 of file GoReadWrite.cpp.

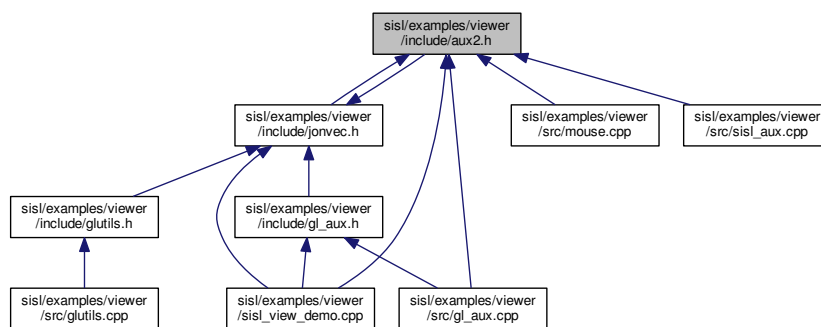


## 30.2275 sisl/examples/viewer/include/aux2.h File Reference

```
#include <vector>
#include <iostream>
#include <stdio.h>
#include "jonvec.h"
#include <exception>
Include dependency graph for aux2.h:
```



This graph shows which files directly or indirectly include this file:



## Macros

- `#define CRIT_ERR(stmt)`
- `#define ASSERT2(a, b) if (!(a)) CRIT_ERR(b)`
- `#define _ERRORMACROS_H`
- `#define REPORT cout << "\nIn file " << __FILE__ << ", line " << __LINE__ << endl`
- `#define MESSAGE(x) cout << "\nIn file " << __FILE__ << ", line " << __LINE__ << ": " << x << endl`
- `#define THROW(x) MESSAGE(x), throw std::exception()`

Usage: `THROW("Error message string.");`

- `#define ASSERT(cond) if (!(cond)) THROW("Assertion \"' #cond \"' failed.")`
- `#define ASSERT3(cond, x) if (!(cond)) THROW(x)`
- `#define ERROR_IF(cond, x) if (cond) THROW(x)`
- `#define DEQUALX(a, b, tol) (fabs((a)-(b))<=(tol) * std::max(std::max(fabs(a), fabs(b)), 1.0))`
- `#define DEQUAL(a, b) (fabs((a)-(b))<=1e-12 * std::max(std::max(fabs(a), fabs(b)), 1.0))`
- `#define NDEQUAL(a, b) (!DEQUAL(a, b))`
- `#define DNEQUAL(a, b) (NDEQUAL(a, b))`
- `#define DLESS(a, b) (((a)<(b)) && (NDEQUAL((a), (b))))`
- `#define DGREATER(a, b) (((a)>(b)) && (NDEQUAL((a), (b))))`
- `#define DLESSEQ(a, b) (((a)<(b)) || (DEQUAL((a), (b))))`
- `#define DGREATEREQ(a, b) (((a)>(b)) || (DEQUAL((a), (b))))`
- `#define DEQUAL2(a, b) (fabs((a)-(b))<=1e-8 * std::max(std::max(fabs(a), fabs(b)), 1.0))`
- `#define DEQUAL3(a, b) (fabs((a)-(b))<=1e-6 * std::max(std::max(fabs(a), fabs(b)), 1.0))`
- `#define NDEQUAL2(a, b) (!DEQUAL2(a, b))`
- `#define DNEQUAL2(a, b) (NDEQUAL2(a, b))`
- `#define DLESSEQ2(a, b) (((a)<(b)) || (DEQUAL2((a), (b))))`
- `#define DGREATEREQ2(a, b) (((a)>(b)) || (DEQUAL2((a), (b))))`
- `#define DEQUAL4(a, b) (fabs((a)-(b))<=1e-14 * std::max(std::max(fabs(a), fabs(b)), 1.0))`
- `#define SQR(a) ((a)*(a))`
- `#define PI 3.1415926535`
- `#define MIN(a, b) ((a)<(b)? (a):(b))`
- `#define MAX(a, b) ((a)>(b)? (a):(b))`
- `#define AUX2_H_INCLUDED`

## Functions

- `int eps_equal (const double &a, const double &b)`
- `int eps_less (const double &a, const double &b)`
- `int eps_greater (const double &a, const double &b)`
- `int eps_less_eq (const double &a, const double &b)`
- `int eps_greater_eq (const double &a, const double &b)`
- `void tic (void)`
- `void toc (void)`

### 30.2275.1 Macro Definition Documentation

#### 30.2275.1.1 `#define _ERRORMACROS_H`

Definition at line 68 of file aux2.h.

#### 30.2275.1.2 `#define ASSERT( cond ) if (!(cond)) THROW("Assertion \"' #cond \"' failed.")`

Usage: `ASSERT(condition)` Usage: `ASSERT2(condition, "Error message string.")` Usage: `ERROR_IF(condition, "Error message string.")`;

Definition at line 99 of file aux2.h.

30.2275.1.3 `#define ASSERT2( a, b ) if (!(a)) CRIT_ERR(b)`

Definition at line 65 of file aux2.h.

30.2275.1.4 `#define ASSERT3( cond, x ) if (!(cond)) THROW(x)`

Definition at line 101 of file aux2.h.

30.2275.1.5 `#define AUX2_H_INCLUDED`

Definition at line 185 of file aux2.h.

30.2275.1.6 `#define CRIT_ERR( stmt )`

**Value:**

```
printf("\nIn file %s, line %d:\n ", __FILE__, __LINE__), \
 (stmt), exit(0)
```

Definition at line 60 of file aux2.h.

30.2275.1.7 `#define DEQUAL( a, b ) (fabs((a)-(b))<=1e-12 * std::max(std::max(fabs(a), fabs(b)), 1.0))`

Definition at line 124 of file aux2.h.

30.2275.1.8 `#define DEQUAL2( a, b ) (fabs((a)-(b))<=1e-8 * std::max(std::max(fabs(a), fabs(b)), 1.0))`

Definition at line 138 of file aux2.h.

30.2275.1.9 `#define DEQUAL3( a, b ) (fabs((a)-(b))<=1e-6 * std::max(std::max(fabs(a), fabs(b)), 1.0))`

Definition at line 140 of file aux2.h.

30.2275.1.10 `#define DEQUAL4( a, b ) (fabs((a)-(b))<=1e-14 * std::max(std::max(fabs(a), fabs(b)), 1.0))`

Definition at line 147 of file aux2.h.

30.2275.1.11 `#define DEQUALX( a, b, tol ) (fabs((a)-(b))<=(tol) * std::max(std::max(fabs(a), fabs(b)), 1.0))`

Definition at line 122 of file aux2.h.

30.2275.1.12 **#define DGREATER( a, b )(((a)>(b)) && (NDEQUAL((a), (b))))**

Definition at line 128 of file aux2.h.

30.2275.1.13 **#define DGREATEREQ( a, b )(((a)>(b)) || (DEQUAL((a), (b))))**

Definition at line 130 of file aux2.h.

30.2275.1.14 **#define DGREATEREQ2( a, b )(((a)>(b)) || (DEQUAL2((a), (b))))**

Definition at line 144 of file aux2.h.

30.2275.1.15 **#define DLESS( a, b )(((a)<(b)) && (NDEQUAL((a), (b))))**

Definition at line 127 of file aux2.h.

30.2275.1.16 **#define DLESSEQ( a, b )(((a)<(b)) || (DEQUAL((a), (b))))**

Definition at line 129 of file aux2.h.

30.2275.1.17 **#define DLESSEQ2( a, b )(((a)<(b)) || (DEQUAL2((a), (b))))**

Definition at line 143 of file aux2.h.

30.2275.1.18 **#define DNEQUAL( a, b ) (NDEQUAL(a, b))**

Definition at line 126 of file aux2.h.

30.2275.1.19 **#define DNEQUAL2( a, b ) (NDEQUAL2(a, b))**

Definition at line 142 of file aux2.h.

30.2275.1.20 **#define ERROR\_IF( cond, x ) if (cond) THROW(x)**

Definition at line 102 of file aux2.h.

30.2275.1.21 **#define MAX( a, b )((a)>(b)? (a):(b))**

Definition at line 163 of file aux2.h.

30.2275.1.22 `#define MESSAGE( x ) cout << "\nIn file " << __FILE__ << ", line " << __LINE__ << ": " << x << endl`

Definition at line 80 of file aux2.h.

30.2275.1.23 `#define MIN( a, b ) ((a)<(b)? (a):(b))`

Definition at line 160 of file aux2.h.

30.2275.1.24 `#define NDEQUAL( a, b ) (!DEQUAL(a, b))`

Definition at line 125 of file aux2.h.

30.2275.1.25 `#define NDEQUAL2( a, b ) (!DEQUAL2(a, b))`

Definition at line 141 of file aux2.h.

30.2275.1.26 `#define PI 3.1415926535`

Definition at line 156 of file aux2.h.

30.2275.1.27 `#define REPORT cout << "\nIn file " << __FILE__ << ", line " << __LINE__ << endl`

Usage: REPORT; Usage: MESSAGE("Message string.");

Definition at line 79 of file aux2.h.

30.2275.1.28 `#define SQR( a ) ((a)*(a))`

Definition at line 153 of file aux2.h.

30.2275.1.29 `#define THROW( x ) MESSAGE(x), throw std::exception()`

Usage: THROW("Error message string.");

Definition at line 84 of file aux2.h.

### 30.2275.2 Function Documentation

30.2275.2.1 `int eps_equal ( const double & a, const double & b )`

30.2275.2.2 `int eps_greater ( const double & a, const double & b )`

30.2275.2.3 `int eps_greater_eq ( const double & a, const double & b )`

30.2275.2.4 `int eps_less ( const double & a, const double & b )`

30.2275.2.5 `int eps_less_eq ( const double & a, const double & b )`

30.2275.2.6 `void tic ( void )`

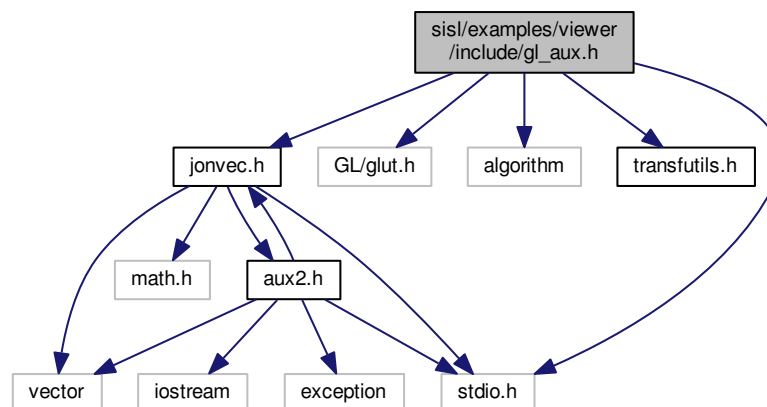
Definition at line 100 of file aux2.cpp.

30.2275.2.7 `void toc ( void )`

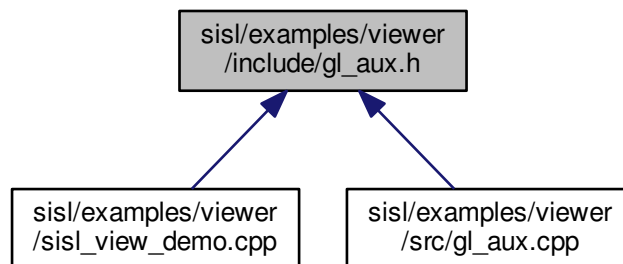
Definition at line 106 of file aux2.cpp.

### 30.2276 sisl/examples/viewer/include/gl\_aux.h File Reference

```
#include <stdio.h>
#include <GL/glut.h>
#include <algorithm>
#include "transfutils.h"
#include "jonvec.h"
Include dependency graph for gl_aux.h:
```



This graph shows which files directly or indirectly include this file:



## Classes

- class [material\\_appearance](#)

## Functions

- void [draw\\_cylinder](#) (double *x0*, double *y0*, double *z0*, double *x1*, double *y1*, double *z1*, double *radius*, double *radius2*, int *n*)
- void [draw\\_gl\\_axes\\_old](#) (int *n*=10, double *r*=0.7, double *radius*=0.01, double *rim*=0.04, double *l*=0.1)
- void [draw\\_grid](#) (const int *n1*, const int *n2*, const int *n3*)
- void [draw\\_grid\\_planes](#) (const int *n1*, const int *n2*, const int *n3*)
- void [gl\\_init](#) (int *argc*, char \**argv*[], const int *xs*, const int *ys*, const int *x0*, const int *y0*, const int *doubleBuffer*, const int *two\_sided*, const int *lighting*, const int *normalize*, const int *smooth*, const double *xtrans*, const double *ytrans*, const double *ztrans*, const double *xscale*, const double *yscale*, const double *zscale*, int &*tx*, int &*ty*, const int *texture\_mode*, const char \**const* *texture\_file*=NULL)
- void [print\\_gl\\_matrix](#) (const int *m*)
- void [reshape\\_window](#) (int *width*, int *height*)
- void [write\\_gl\\_matrices](#) (FILE \**f*)
- void [read\\_gl\\_matrices](#) (FILE \**f*)

## Variables

- const int [predefined\\_colours](#)
- const double [predef\\_col\\_table](#) []
- const double [col\\_setting\\_back\\_old](#) []
- const double [col\\_setting\\_back](#) []

### 30.2276.1 Function Documentation

- 30.2276.1.1 void [draw\\_cylinder](#) ( double *x0*, double *y0*, double *z0*, double *x1*, double *y1*, double *z1*, double *radius*, double *radius2*, int *n* )

Definition at line 91 of file `gl_aux.cpp`.

30.2276.1.2 `void draw_gl_axes_old ( int n = 10, double r = 0.7, double radius = 0.01, double rim = 0.04, double l = 0.1 )`

Definition at line 174 of file `gl_aux.cpp`.

30.2276.1.3 `void draw_grid ( const int n1, const int n2, const int n3 )`

Definition at line 223 of file `gl_aux.cpp`.

30.2276.1.4 `void draw_grid_planes ( const int n1, const int n2, const int n3 )`

Definition at line 273 of file `gl_aux.cpp`.

30.2276.1.5 `void gl_init ( int argc, char * argv[], const int xs, const int ys, const int x0, const int y0, const int doubleBuffer, const int two_sided, const int lighting, const int normalize, const int smooth, const double xtrans, const double ytrans, const double ztrans, const double xscale, const double yscale, const double zscale, int & tx, int & ty, const int texture_mode, const char *const texfile = NULL )`

Definition at line 329 of file `gl_aux.cpp`.

30.2276.1.6 `void print_gl_matrix ( const int m )`

Definition at line 659 of file `gl_aux.cpp`.

30.2276.1.7 `void read_gl_matrices ( FILE * f )`

Definition at line 781 of file `gl_aux.cpp`.

30.2276.1.8 `void reshape_window ( int width, int height )`

Definition at line 741 of file `gl_aux.cpp`.

30.2276.1.9 `void write_gl_matrices ( FILE * f )`

Definition at line 761 of file `gl_aux.cpp`.

## 30.2276.2 Variable Documentation

30.2276.2.1 `const double col_setting_back[]`

30.2276.2.2 `const double col_setting_back_old[]`

30.2276.2.3 `const double predef_col_table[]`

Definition at line 56 of file `gl_aux.cpp`.



## 30.2276.2.4 const int predefined\_colours

Definition at line 54 of file gl\_aux.cpp.

## 30.2277 sisl/examples/viewer/include/glutils.h File Reference

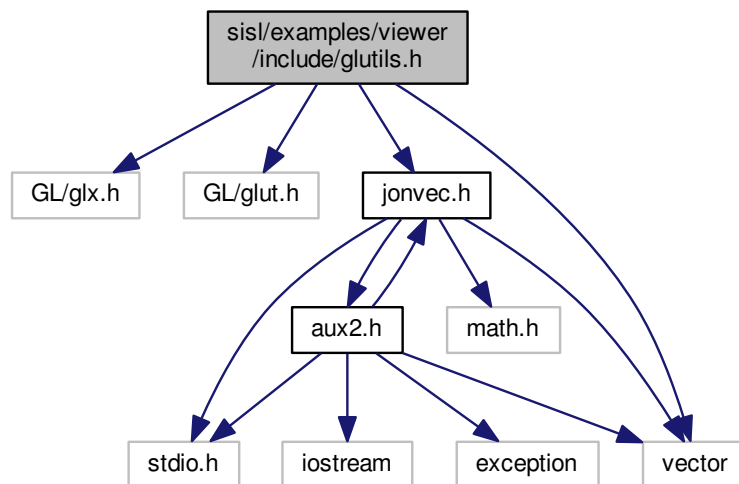
```
#include <GL/glx.h>
```

```
#include "GL/glut.h"
```

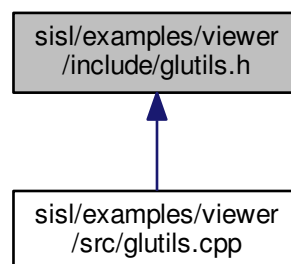
```
#include <vector>
```

```
#include "jonvec.h"
```

Include dependency graph for glutils.h:



This graph shows which files directly or indirectly include this file:



## Macros

- #define `ASSERT_GL assert_gl_m(__LINE__, __FILE__)`
- #define `ASSERT_GL_DONT_EXIT assert_gl_m(__LINE__, __FILE__, false)`
- #define `assert_gl assert_gl_m(__LINE__, __FILE__); assert_gl_dummy_and_empty`
- #define `GLUTILS_H_INCLUDED`

## Functions

- void `assert_gl_m` (int line, char \*file, const bool do\_exit=true)
- void `assert_gl_dummy_and_empty` (void)
- void `list_FBConfigs` (GLXFBConfig \*config, int nelements)
- void `draw_sphere` (const vector3t< float > &pos, const float r, const int n, const int slot, const vector3t< float > &col, const bool wiremode)

### 30.2277.1 Macro Definition Documentation

#### 30.2277.1.1 #define ASSERT\_GL assert\_gl\_m(\_\_LINE\_\_, \_\_FILE\_\_)

Definition at line 70 of file glutils.h.

#### 30.2277.1.2 #define assert\_gl assert\_gl\_m(\_\_LINE\_\_, \_\_FILE\_\_); assert\_gl\_dummy\_and\_empty

Definition at line 76 of file glutils.h.

#### 30.2277.1.3 #define ASSERT\_GL\_DONT\_EXIT assert\_gl\_m(\_\_LINE\_\_, \_\_FILE\_\_, false)

Definition at line 71 of file glutils.h.

#### 30.2277.1.4 #define GLUTILS\_H\_INCLUDED

Definition at line 92 of file glutils.h.

### 30.2277.2 Function Documentation

#### 30.2277.2.1 void assert\_gl\_dummy\_and\_empty ( void )

Definition at line 166 of file glutils.cpp.

#### 30.2277.2.2 void assert\_gl\_m ( int line, char \* file, const bool do\_exit = true )

Definition at line 60 of file glutils.cpp.

30.2277.2.3 `void draw_sphere ( const vector3t< float > & pos, const float r, const int n, const int slot, const vector3t< float > & col, const bool wiremode )`

Definition at line 327 of file glutils.cpp.

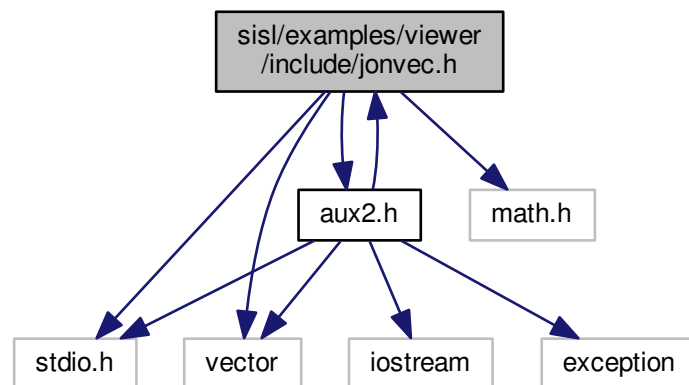
30.2277.2.4 `void list_FBConfigs ( GLXFBConfig * config, int nelements )`

Definition at line 186 of file glutils.cpp.

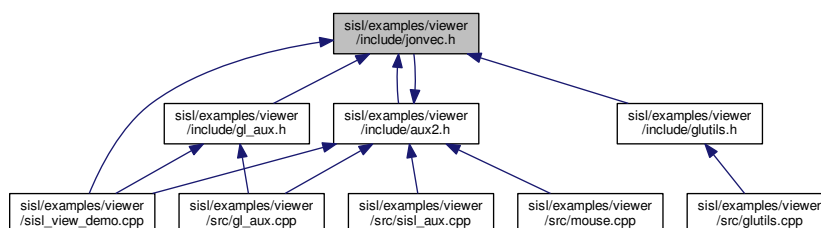
## 30.2278 sisl/examples/viewer/include/jonvec.h File Reference

```
#include <stdio.h>
#include <math.h>
#include <vector>
#include "aux2.h"
```

Include dependency graph for jonvec.h:



This graph shows which files directly or indirectly include this file:

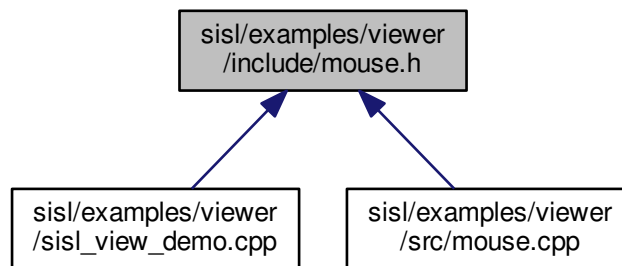


## Classes

- class [vector3t< T >](#)

## 30.2279 sisl/examples/viewer/include/mouse.h File Reference

This graph shows which files directly or indirectly include this file:



## Macros

- `#define` [MOUSE\\_H\\_INCLUDED](#)

## Functions

- int [transversal\\_rotation](#) (int x, int y, int last\_xx, int last\_yy)
- void [MouseRotate](#) (int x, int y)
- void [MouseZoom](#) (int x, int y)
- void [MouseTranslate](#) (int x, int y)
- void [Mouse](#) (int butt, int state, int x, int y)

## Variables

- GLboolean [allow\\_zrot](#)
- int [last\\_x](#)
- int [last\\_y](#)
- int [last\\_x0](#)
- int [last\\_y0](#)
- bool [mouse\\_movement](#)

### 30.2279.1 Macro Definition Documentation

#### 30.2279.1.1 `#define` [MOUSE\\_H\\_INCLUDED](#)

Definition at line 62 of file mouse.h.

## 30.2279.2 Function Documentation

30.2279.2.1 void Mouse ( int *butt*, int *state*, int *x*, int *y* )

Definition at line 222 of file mouse.cpp.

30.2279.2.2 void MouseRotate ( int *x*, int *y* )

Definition at line 151 of file mouse.cpp.

30.2279.2.3 void MouseTranslate ( int *x*, int *y* )

Definition at line 209 of file mouse.cpp.

30.2279.2.4 void MouseZoom ( int *x*, int *y* )

Definition at line 196 of file mouse.cpp.

30.2279.2.5 int transversal\_rotation ( int *x*, int *y*, int *last\_xx*, int *last\_yy* )

Definition at line 115 of file mouse.cpp.

## 30.2279.3 Variable Documentation

30.2279.3.1 GLboolean allow\_zrot

Definition at line 69 of file mouse.cpp.

30.2279.3.2 int last\_x

Definition at line 70 of file mouse.cpp.

30.2279.3.3 int last\_x0

Definition at line 70 of file mouse.cpp.

30.2279.3.4 int last\_y

Definition at line 70 of file mouse.cpp.

### 30.2279.3.5 int last\_y0

Definition at line 70 of file mouse.cpp.

### 30.2279.3.6 bool mouse\_movement

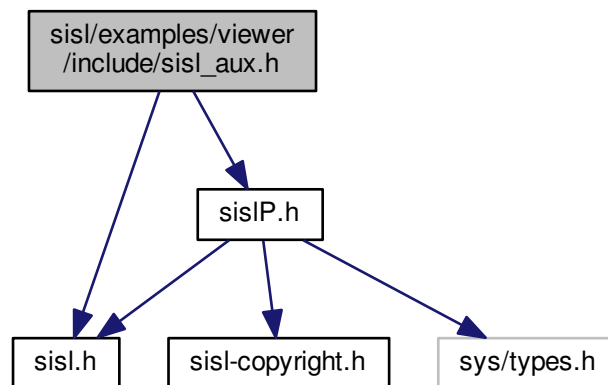
Definition at line 71 of file mouse.cpp.

## 30.2280 sisl/examples/viewer/include/sisl\_aux.h File Reference

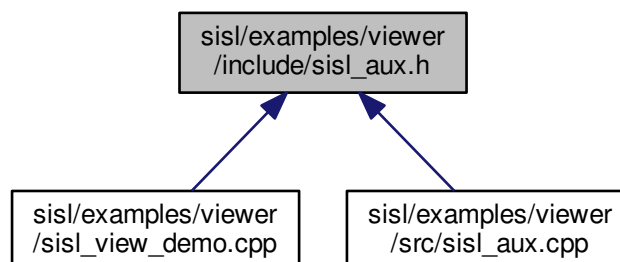
```
#include "sisl.h"
```

```
#include "sislP.h"
```

Include dependency graph for sisl\_aux.h:



This graph shows which files directly or indirectly include this file:



## Macros

- #define [SISL\\_AUX\\_H\\_INCLUDED](#)

## Functions

- void [lower\\_degree\\_to\\_linear](#) (SISLSurf \*\*srf, double e)
- void [lower\\_degree\\_and\\_subdivide](#) (SISLSurf \*\*srf, int new\_knots\_per\_interval, int max\_number\_of\_knots)
- void [compute\\_surface\\_normals](#) (SISLSurf \*srf, double \*\*ngrid)

### 30.2280.1 Macro Definition Documentation

#### 30.2280.1.1 #define SISL\_AUX\_H\_INCLUDED

Definition at line 59 of file sisl\_aux.h.

### 30.2280.2 Function Documentation

#### 30.2280.2.1 void compute\_surface\_normals ( SISLSurf \* srf, double \*\* ngrid )

Definition at line 231 of file sisl\_aux.cpp.

#### 30.2280.2.2 void lower\_degree\_and\_subdivide ( SISLSurf \*\* srf, int new\_knots\_per\_interval, int max\_number\_of\_knots )

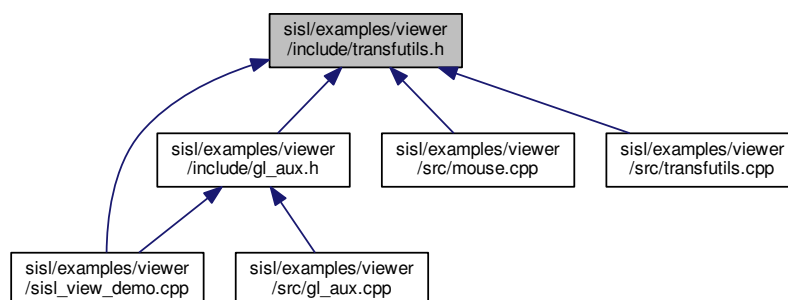
Definition at line 81 of file sisl\_aux.cpp.

#### 30.2280.2.3 void lower\_degree\_to\_linear ( SISLSurf \*\* srf, double e )

Definition at line 62 of file sisl\_aux.cpp.

## 30.2281 sisl/examples/viewer/include/transfutils.h File Reference

This graph shows which files directly or indirectly include this file:



## Macros

- `#define TRANSFUTILS_H_INCLUDED`

## Functions

- void `rotate` (double `y_ang`, double `x_ang`, double `z_ang`)
- void `translate` (double `x`, double `y`, double `z`)
- void `scale` (double `x`, double `y`, double `z`)

## Variables

- double `xtrans`
- double `ytrans`
- double `ztrans`
- double `xrot_eps`
- double `yrot_eps`
- double `zrot_eps`
- double `xscale`
- double `yscale`
- double `zscale`
- double `xrot`
- double `yrot`
- double `zrot`

### 30.2281.1 Macro Definition Documentation

#### 30.2281.1.1 `#define TRANSFUTILS_H_INCLUDED`

Definition at line 63 of file `transfutils.h`.

### 30.2281.2 Function Documentation

#### 30.2281.2.1 void `rotate` ( double `y_ang`, double `x_ang`, double `z_ang` )

Definition at line 68 of file `transfutils.cpp`.

#### 30.2281.2.2 void `scale` ( double `x`, double `y`, double `z` )

Definition at line 127 of file `transfutils.cpp`.

#### 30.2281.2.3 void `translate` ( double `x`, double `y`, double `z` )

Definition at line 108 of file `transfutils.cpp`.



### 30.2281.3 Variable Documentation

#### 30.2281.3.1 `double xrot`

Definition at line 61 of file `transfutils.cpp`.

#### 30.2281.3.2 `double xrot_eps`

Definition at line 59 of file `transfutils.cpp`.

#### 30.2281.3.3 `double xscale` `[property]`, `[bound]`, `[constrained]`

Definition at line 92 of file `all_8.js`.

#### 30.2281.3.4 `double xtrans` `[property]`, `[bound]`, `[constrained]`

Definition at line 92 of file `all_8.js`.

#### 30.2281.3.5 `double yrot`

Definition at line 61 of file `transfutils.cpp`.

#### 30.2281.3.6 `double yrot_eps`

Definition at line 59 of file `transfutils.cpp`.

#### 30.2281.3.7 `double yscale` `[property]`, `[bound]`, `[constrained]`

Definition at line 92 of file `all_8.js`.

#### 30.2281.3.8 `double ytrans` `[property]`, `[bound]`, `[constrained]`

Definition at line 92 of file `all_8.js`.

#### 30.2281.3.9 `double zrot`

Definition at line 61 of file `transfutils.cpp`.

#### 30.2281.3.10 `double zrot_eps`

Definition at line 59 of file `transfutils.cpp`.

30.2281.3.11 **double zscale** [property], [bound], [constrained]

Definition at line 92 of file all\_8.js.

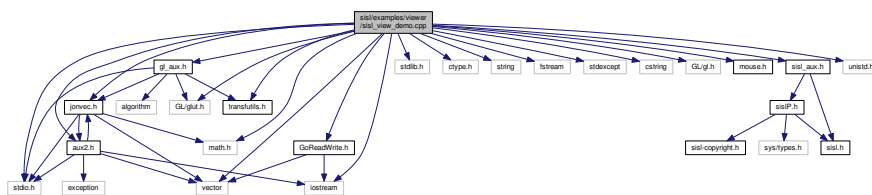
30.2281.3.12 **double ztrans** [property], [bound], [constrained]

Definition at line 92 of file all\_8.js.

## 30.2282 sisl/examples/viewer/sisl\_view\_demo.cpp File Reference

```
#include <stdio.h>
#include <math.h>
#include <stdlib.h>
#include <ctype.h>
#include <string>
#include <iostream>
#include <fstream>
#include <stdexcept>
#include <vector>
#include <cstring>
#include <GL/gl.h>
#include <GL/glut.h>
#include "gl_aux.h"
#include "transfutils.h"
#include "mouse.h"
#include "sisl_aux.h"
#include "GoReadWrite.h"
#include "aux2.h"
#include "jonvec.h"
#include <unistd.h>
```

Include dependency graph for sisl\_view\_demo.cpp:



## Macros

- #define **MAX\_SURFACES** 100
- #define **MAX\_CURVES** 2000
- #define **MAX\_LINES** 100
- #define **CURVE\_EVALUATIONS** 500
- #define **ESC\_STRING** "<ESC>"
- #define **TAB\_STRING** "<TAB>"
- #define **CTRL\_STRING** "<CTRL>"

## Functions

- void `draw_all_curves` (void)
- int `main` (int argc, char \*argv[ ])

## Variables

- char `init_key_string` [1000]
- double `axis_thickness` =0.3
- double `axis_length` =0.7
- double `marker_size` =1.0
- int `draw_edges` =0
- float `wire_col` =0.0
- float `background_col` =0.0
- int `draw_axes` =1
- int `frames_without_movement` =0
- `SISLSurf` \* `surface` [MAX\_SURFACES]
- int `surfaces` =0
- int `selected_surface` =-1
- int `surface_highlights` =0
- int `curve_highlights` =0
- int `selected_curve` =-1
- double \* `normal` [MAX\_SURFACES]
- int `surf_enabled` [MAX\_SURFACES]
- int `curve_enabled` [MAX\_CURVES]
- char \* `surface_name` [MAX\_SURFACES]
- char \* `curve_name` [MAX\_CURVES]
- `SISLCurve` \* `curve` [MAX\_CURVES]
- int `curves` =0
- double \* `discr_curve` [MAX\_CURVES]
- vector< vector3t< float > > `pcloud`
- const int `predef_colours` =6
- const double `col_setting` [3 \*predef\_colours]
- const double `col_x_setting_back` [3 \*predef\_colours]

### 30.2282.1 Macro Definition Documentation

30.2282.1.1 `#define CTRL_STRING "<CTRL>"`

30.2282.1.2 `#define CURVE_EVALUATIONS 500`

30.2282.1.3 `#define ESC_STRING "<ESC>"`

30.2282.1.4 `#define MAX_CURVES 2000`

Definition at line 85 of file `sisl_view_demo.cpp`.

30.2282.1.5 `#define MAX_LINES 100`

Definition at line 86 of file `sisl_view_demo.cpp`.

30.2282.1.6 `#define MAX_SURFACES 100`

Definition at line 84 of file `sisl_view_demo.cpp`.

30.2282.1.7 `#define TAB_STRING "<TAB>"`

## 30.2282.2 Function Documentation

30.2282.2.1 `void draw_all_curves ( void )`

Definition at line 372 of file `sisl_view_demo.cpp`.

30.2282.2.2 `int main ( int argc, char * argv[] )`

Definition at line 1287 of file `sisl_view_demo.cpp`.

## 30.2282.3 Variable Documentation

30.2282.3.1 `double axis_length =0.7`

Definition at line 78 of file `sisl_view_demo.cpp`.

30.2282.3.2 `double axis_thickness =0.3`

Definition at line 78 of file `sisl_view_demo.cpp`.

30.2282.3.3 `float background_col =0.0`

Definition at line 80 of file `sisl_view_demo.cpp`.

30.2282.3.4 `const double col_setting[3 *predef_colours]`

### Initial value:

```
={1.0, 0.0, 0.0,
 0.0, 1.0, 0.0,
 0.0, 0.0, 1.0,
 1.0, 1.0, 0.0,
 0.0, 1.0, 1.0,
 1.0, 0.0, 1.0}
```

Definition at line 108 of file `sisl_view_demo.cpp`.

**30.2282.3.5 const double col\_x\_setting\_back[3 \*predef\_colours]****Initial value:**

```
={0.6, 0.2, 0.0,
 0.0, 0.6, 0.2,
 0.2, 0.0, 0.6,
 0.6, 0.6, 0.2,
 0.2, 0.6, 0.6,
 0.6, 0.2, 0.6}
```

Definition at line 115 of file sisl\_view\_demo.cpp.

**30.2282.3.6 SISLCurve\* curve[MAX\_CURVES]**

Definition at line 96 of file sisl\_view\_demo.cpp.

**30.2282.3.7 int curve\_enabled[MAX\_CURVES]**

Definition at line 93 of file sisl\_view\_demo.cpp.

**30.2282.3.8 int curve\_highlights =0**

Definition at line 89 of file sisl\_view\_demo.cpp.

**30.2282.3.9 char\* curve\_name[MAX\_CURVES]**

Definition at line 95 of file sisl\_view\_demo.cpp.

**30.2282.3.10 int curves =0**

Definition at line 97 of file sisl\_view\_demo.cpp.

**30.2282.3.11 double\* discr\_curve[MAX\_CURVES]**

Definition at line 98 of file sisl\_view\_demo.cpp.

**30.2282.3.12 int draw\_axes =1**

Definition at line 81 of file sisl\_view\_demo.cpp.

**30.2282.3.13 int draw\_edges =0**

Definition at line 79 of file sisl\_view\_demo.cpp.

30.2282.3.14 `int frames_without_movement =0`

Definition at line 82 of file `sisl_view_demo.cpp`.

30.2282.3.15 `char init_key_string[1000]`

Definition at line 77 of file `sisl_view_demo.cpp`.

30.2282.3.16 `double marker_size =1.0`

Definition at line 78 of file `sisl_view_demo.cpp`.

30.2282.3.17 `double* normal[MAX_SURFACES]`

Definition at line 91 of file `sisl_view_demo.cpp`.

30.2282.3.18 `vector< vector3t<float> > pcloud`

Definition at line 99 of file `sisl_view_demo.cpp`.

30.2282.3.19 `const int predef_colours =6`

Definition at line 106 of file `sisl_view_demo.cpp`.

30.2282.3.20 `int selected_curve =-1`

Definition at line 90 of file `sisl_view_demo.cpp`.

30.2282.3.21 `int selected_surface =-1`

Definition at line 89 of file `sisl_view_demo.cpp`.

30.2282.3.22 `int surf_enabled[MAX_SURFACES]`

Definition at line 92 of file `sisl_view_demo.cpp`.

30.2282.3.23 `SISLSurf* surface[MAX_SURFACES]`

Definition at line 88 of file `sisl_view_demo.cpp`.

30.2282.3.24 `int surface_highlights =0`

Definition at line 89 of file `sisl_view_demo.cpp`.

30.2282.3.25 `char* surface_name[MAX_SURFACES]`

Definition at line 94 of file `sisl_view_demo.cpp`.

30.2282.3.26 `int surfaces =0`

Definition at line 89 of file `sisl_view_demo.cpp`.

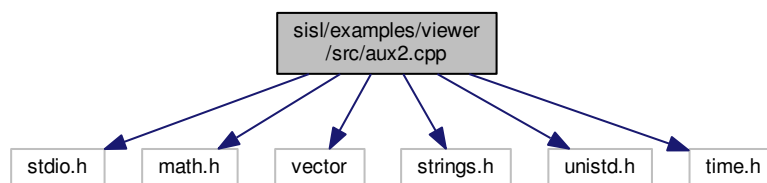
30.2282.3.27 `float wire_col =0.0`

Definition at line 80 of file `sisl_view_demo.cpp`.

## 30.2283 sisl/examples/viewer/src/aux2.cpp File Reference

```
#include <stdio.h>
#include <math.h>
#include <vector>
#include <strings.h>
#include <unistd.h>
#include <time.h>
```

Include dependency graph for `aux2.cpp`:



### Functions

- void `tic` (void)
- void `toc` (void)

### Variables

- double `jon_timer`

### 30.2283.1 Function Documentation

#### 30.2283.1.1 void tic ( void )

Definition at line 100 of file aux2.cpp.

#### 30.2283.1.2 void toc ( void )

Definition at line 106 of file aux2.cpp.

### 30.2283.2 Variable Documentation

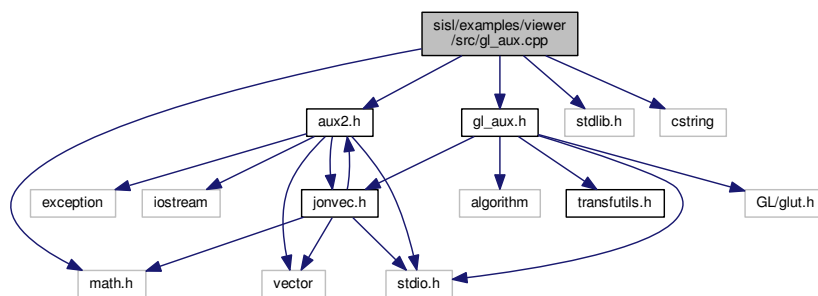
#### 30.2283.2.1 double jon\_timer

Definition at line 70 of file aux2.cpp.

## 30.2284 sisl/examples/viewer/src/gl\_aux.cpp File Reference

```
#include <math.h>
#include <stdlib.h>
#include <cstring>
#include "aux2.h"
#include "gl_aux.h"
```

Include dependency graph for gl\_aux.cpp:



### Macros

- #define PI 3.1415926536



## Functions

- void `draw_cylinder` (double *x0*, double *y0*, double *z0*, double *x1*, double *y1*, double *z1*, double *radius*, double *radius2*, int *n*)  
*Draw a cylinder, or possibly a cone.*
- void `draw_gl_axes_old` (int *n*, double *r*, double *radius*, double *rim*, double *l*)
- void `draw_grid` (const int *n1*, const int *n2*, const int *n3*)
- void `draw_grid_planes` (const int *n1*, const int *n2*, const int *n3*)
- void `gl_init` (int *argc*, char \**argv*[], const int *xs*, const int *ys*, const int *x0*, const int *y0*, const int *doubleBuffer*, const int *two\_sided*, const int *lighting*, const int *normalize*, const int *smooth*, const double *xtrans*, const double *ytrans*, const double *ztrans*, const double *xscale*, const double *yscale*, const double *zscale*, int &*tx*, int &*ty*, const int *texture\_mode*, const char \**const* *texturefile*)
- void `transpose_matrix` (double \**const* *d*)
- void `print_gl_matrix` (const int *m*)
- void `reshape_window` (int *width*, int *height*)
- void `write_gl_matrices` (FILE \**f*)
- void `read_gl_matrices` (FILE \**f*)

## Variables

- const int `predefined_colours` =24
- const double `predef_col_table` [3 \*`predefined_colours`]

### 30.2284.1 Macro Definition Documentation

30.2284.1.1 `#define PI 3.1415926536`

Definition at line 50 of file `gl_aux.cpp`.

### 30.2284.2 Function Documentation

30.2284.2.1 void `draw_cylinder` ( double *x0*, double *y0*, double *z0*, double *x1*, double *y1*, double *z1*, double *radius*, double *radius2*, int *n* )

Draw a cylinder, or possibly a cone.

Definition at line 91 of file `gl_aux.cpp`.

30.2284.2.2 void `draw_gl_axes_old` ( int *n*, double *r*, double *radius*, double *rim*, double *l* )

Definition at line 174 of file `gl_aux.cpp`.

30.2284.2.3 void `draw_grid` ( const int *n1*, const int *n2*, const int *n3* )

Definition at line 223 of file `gl_aux.cpp`.

30.2284.2.4 void draw\_grid\_planes ( const int *n1*, const int *n2*, const int *n3* )

Definition at line 273 of file gl\_aux.cpp.

30.2284.2.5 void gl\_init ( int *argc*, char \* *argv*[], const int *xs*, const int *ys*, const int *x0*, const int *y0*, const int *doubleBuffer*, const int *two\_sided*, const int *lighting*, const int *normalize*, const int *smooth*, const double *xtrans*, const double *ytrans*, const double *ztrans*, const double *xscale*, const double *yscale*, const double *zscale*, int & *tx*, int & *ty*, const int *texture\_mode*, const char \*const *texture\_file* )

Definition at line 329 of file gl\_aux.cpp.

30.2284.2.6 void print\_gl\_matrix ( const int *m* )

Definition at line 659 of file gl\_aux.cpp.

30.2284.2.7 void read\_gl\_matrices ( FILE \* *f* )

Definition at line 781 of file gl\_aux.cpp.

30.2284.2.8 void reshape\_window ( int *width*, int *height* )

Definition at line 741 of file gl\_aux.cpp.

30.2284.2.9 void transpose\_matrix ( double \*const *d* )

Definition at line 641 of file gl\_aux.cpp.

30.2284.2.10 void write\_gl\_matrices ( FILE \* *f* )

Definition at line 761 of file gl\_aux.cpp.

### 30.2284.3 Variable Documentation

30.2284.3.1 const double predef\_col\_table[3 \*predefined\_colours]

#### Initial value:

```
={1.0, 0.0, 0.0,
 0.0, 1.0, 0.0,
 0.0, 0.0, 1.0,
 1.0, 1.0, 0.0,
 0.0, 1.0, 1.0,
 1.0, 0.0, 1.0,
 0.7, 0.0, 0.0,
 0.0, 0.7, 0.0,
 0.0, 0.0, 0.7,
 0.7, 0.7, 0.0,
 0.0, 0.7, 0.7,
 0.7, 0.0, 0.7,
 0.3, 0.8, 0.8,
 0.8, 0.3, 0.8,
 0.8, 0.8, 0.3,
 0.3, 0.3, 0.8,
 0.8, 0.3, 0.3,
 0.3, 0.8, 0.3,
 0.5, 0.8, 0.8,
 0.8, 0.5, 0.8,
 0.8, 0.8, 0.5,
 0.5, 0.5, 0.8,
 0.8, 0.5, 0.5,
 0.5, 0.8, 0.5}
```

Definition at line 56 of file gl\_aux.cpp.

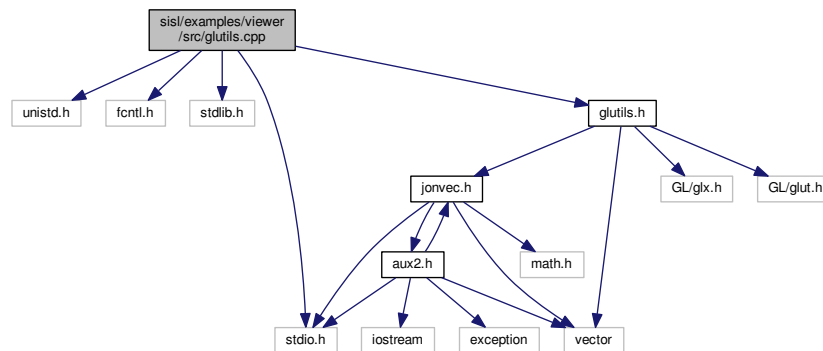
30.2284.3.2 `const int predefined_colours =24`

Definition at line 54 of file `gl_aux.cpp`.

## 30.2285 sisl/examples/viewer/src/glutils.cpp File Reference

```
#include <unistd.h>
#include <fcntl.h>
#include <stdlib.h>
#include <stdio.h>
#include "glutils.h"
```

Include dependency graph for `glutils.cpp`:



## Macros

- `#define M_PI 3.1415926535897932384`

## Functions

- void `assert_gl_m` (int line, char \*file, `const bool` do\_exit)
- void `assert_gl_dummy_and_empty` (void)
- void `list_FBConfigs` (GLXFBCConfig \*config, int nelements)
- void `draw_sphere` (`const vector3t< float >` &pos, `const float` r, `const int` n, `const int` slot, `const vector3t< float >` &col, `const bool` wiremode)

### 30.2285.1 Macro Definition Documentation

#### 30.2285.1.1 `#define M_PI 3.1415926535897932384`

Definition at line 53 of file `glutils.cpp`.

## 30.2285.2 Function Documentation

### 30.2285.2.1 void assert\_gl\_dummy\_and\_empty ( void )

Definition at line 166 of file glutils.cpp.

### 30.2285.2.2 void assert\_gl\_m ( int line, char \* file, const bool do\_exit )

Definition at line 60 of file glutils.cpp.

### 30.2285.2.3 void draw\_sphere ( const vector3t< float > & pos, const float r, const int n, const int slot, const vector3t< float > & col, const bool wiremode )

Definition at line 327 of file glutils.cpp.

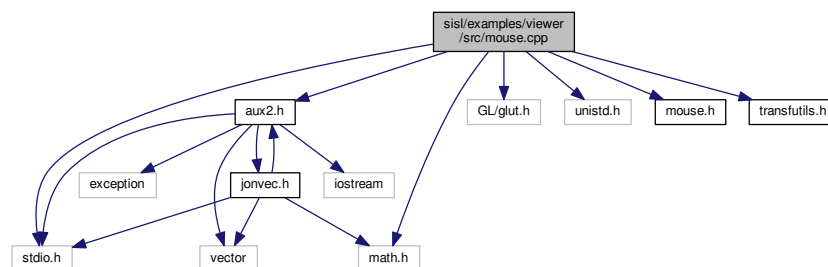
### 30.2285.2.4 void list\_FBConfigs ( GLXFBCConfig \* config, int nelements )

Definition at line 186 of file glutils.cpp.

## 30.2286 sis/examples/viewer/src/mouse.cpp File Reference

```
#include <stdio.h>
#include <GL/glut.h>
#include <math.h>
#include <unistd.h>
#include "mouse.h"
#include "transfutils.h"
#include "aux2.h"
```

Include dependency graph for mouse.cpp:



## Macros

- #define [GLUT\\_DISABLE\\_ATEXIT\\_HACK](#)
- #define [PI](#) 3.1415926536

## Functions

- void [draw\\_cursor](#) (int x, int y)
- int [transversal\\_rotation](#) (int x, int y, int last\_xx, int last\_yy)
- void [MouseRotate](#) (int x, int y)
- void [MouseZoom](#) (int x, int y)
- void [MouseTranslate](#) (int x, int y)
- void [Mouse](#) (int butt, int state, int x, int y)

## Variables

- GLboolean [allow\\_zrot](#) =GL\_TRUE
- int [last\\_x](#) =0
- int [last\\_y](#) =0
- int [last\\_x0](#) =0
- int [last\\_y0](#) =0
- bool [mouse\\_movement](#) =false
- float [curx](#) =0.0
- float [cury](#) =0.0
- float [curz](#) =0.0

### 30.2286.1 Macro Definition Documentation

#### 30.2286.1.1 `#define GLUT_DISABLE_ATEXIT_HACK`

Definition at line 41 of file mouse.cpp.

#### 30.2286.1.2 `#define PI 3.1415926536`

Definition at line 49 of file mouse.cpp.

### 30.2286.2 Function Documentation

#### 30.2286.2.1 `void draw_cursor ( int x, int y )`

Definition at line 86 of file mouse.cpp.

#### 30.2286.2.2 `void Mouse ( int butt, int state, int x, int y )`

Definition at line 222 of file mouse.cpp.

#### 30.2286.2.3 `void MouseRotate ( int x, int y )`

Definition at line 151 of file mouse.cpp.

30.2286.2.4 void MouseTranslate ( int x, int y )

Definition at line 209 of file mouse.cpp.

30.2286.2.5 void MouseZoom ( int x, int y )

Definition at line 196 of file mouse.cpp.

30.2286.2.6 int transversal\_rotation ( int x, int y, int *last\_xx*, int *last\_yy* )

Definition at line 115 of file mouse.cpp.

### 30.2286.3 Variable Documentation

30.2286.3.1 GLboolean allow\_zrot =GL\_TRUE

Definition at line 69 of file mouse.cpp.

30.2286.3.2 float curx =0.0

Definition at line 84 of file mouse.cpp.

30.2286.3.3 float cury =0.0

Definition at line 84 of file mouse.cpp.

30.2286.3.4 float curz =0.0

Definition at line 84 of file mouse.cpp.

30.2286.3.5 int last\_x =0

Definition at line 70 of file mouse.cpp.

30.2286.3.6 int last\_x0 =0

Definition at line 70 of file mouse.cpp.

30.2286.3.7 int last\_y =0

Definition at line 70 of file mouse.cpp.

30.2286.3.8 `int last_y0 =0`

Definition at line 70 of file mouse.cpp.

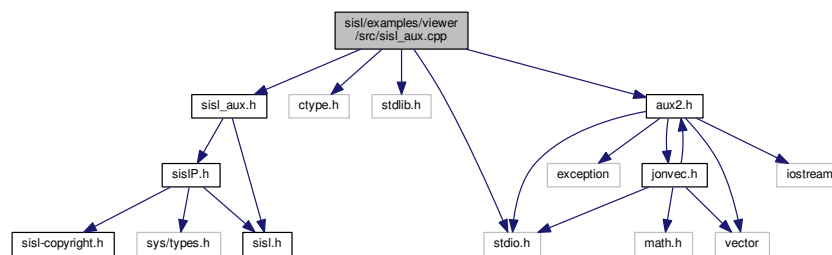
30.2286.3.9 `bool mouse_movement =false`

Definition at line 71 of file mouse.cpp.

## 30.2287 sisl/examples/viewer/src/sisl\_aux.cpp File Reference

```
#include <stdio.h>
#include <ctype.h>
#include <stdlib.h>
#include "sisl_aux.h"
#include "aux2.h"
```

Include dependency graph for sisl\_aux.cpp:



### Macros

- `#define DBGqqq`

### Functions

- void `lower_degree_to_linear` (SISLSurf \*\*srf, double e)
- void `lower_degree_and_subdivide` (SISLSurf \*\*srf, int new\_knots\_per\_interval, int max\_number\_of\_knots)
- void `compute_surface_normals` (SISLSurf \*srf, double \*\*ngrid)

### 30.2287.1 Macro Definition Documentation

30.2287.1.1 `#define DBGqqq`

Definition at line 40 of file sisl\_aux.cpp.

## 30.2287.2 Function Documentation

30.2287.2.1 void `compute_surface_normals` ( `SISLSurf * srf`, `double ** ngrid` )

Definition at line 231 of file `sisl_aux.cpp`.

30.2287.2.2 void `lower_degree_and_subdivide` ( `SISLSurf ** srf`, `int new_knots_per_interval`, `int max_number_of_knots` )

Definition at line 81 of file `sisl_aux.cpp`.

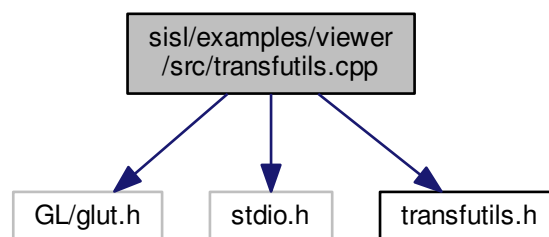
30.2287.2.3 void `lower_degree_to_linear` ( `SISLSurf ** srf`, `double e` )

Definition at line 62 of file `sisl_aux.cpp`.

## 30.2288 sisl/examples/viewer/src/transfutils.cpp File Reference

```
#include <GL/glut.h>
#include <stdio.h>
#include "transfutils.h"
```

Include dependency graph for `transfutils.cpp`:



## Functions

- void `rotate` (`double y_ang`, `double x_ang`, `double z_ang`)
- void `translate` (`double x`, `double y`, `double z`)
- void `scale` (`double x`, `double y`, `double z`)



## Variables

- `double xtrans =0.0`
- `double ytrans =0.0`
- `double ztrans =-6`
- `double xrot_eps =5.0`
- `double yrot_eps =5.0`
- `double zrot_eps =5.0`
- `double xscale =1.0`
- `double yscale =1.0`
- `double zscale =1.0`
- `double xrot`
- `double yrot`
- `double zrot`

### 30.2288.1 Function Documentation

30.2288.1.1 `void rotate ( double y_ang, double x_ang, double z_ang )`

Definition at line 68 of file `transfutils.cpp`.

30.2288.1.2 `void scale ( double x, double y, double z )`

Definition at line 127 of file `transfutils.cpp`.

30.2288.1.3 `void translate ( double x, double y, double z )`

Definition at line 108 of file `transfutils.cpp`.

### 30.2288.2 Variable Documentation

30.2288.2.1 `double xrot`

Definition at line 61 of file `transfutils.cpp`.

30.2288.2.2 `double xrot_eps =5.0`

Definition at line 59 of file `transfutils.cpp`.

30.2288.2.3 `double xscale =1.0` `[property]`, `[bound]`, `[constrained]`

Definition at line 60 of file `transfutils.cpp`.

**30.2288.2.4** `double xtrans =0.0` [property], [bound], [constrained]

Definition at line 58 of file transfutils.cpp.

**30.2288.2.5** `double yrot`

Definition at line 61 of file transfutils.cpp.

**30.2288.2.6** `double yrot_eps =5.0`

Definition at line 59 of file transfutils.cpp.

**30.2288.2.7** `double yscale =1.0` [property], [bound], [constrained]

Definition at line 60 of file transfutils.cpp.

**30.2288.2.8** `double ytrans =0.0` [property], [bound], [constrained]

Definition at line 58 of file transfutils.cpp.

**30.2288.2.9** `double zrot`

Definition at line 61 of file transfutils.cpp.

**30.2288.2.10** `double zrot_eps =5.0`

Definition at line 59 of file transfutils.cpp.

**30.2288.2.11** `double zscale =1.0` [property], [bound], [constrained]

Definition at line 60 of file transfutils.cpp.

**30.2288.2.12** `double ztrans =-6` [property], [bound], [constrained]

Definition at line 58 of file transfutils.cpp.

## 30.2289 sisl/include/sisl-copyright.h File Reference

## 30.2290 sisl/include/sisl.h File Reference

This graph shows which files directly or indirectly include this file:



### Classes

- struct [SISLdir](#)
- struct [SISLbox](#)
- struct [SISLCurve](#)
- struct [SISLSurf](#)
- struct [SISLIntcurve](#)

### Macros

- `#define` [GO\\_API](#)
- `#define` [SISL\\_CRV\\_PERIODIC](#) -1
- `#define` [SISL\\_CRV\\_OPEN](#) 1
- `#define` [SISL\\_CRV\\_CLOSED](#) 0
- `#define` [SISL\\_SURF\\_PERIODIC](#) -1
- `#define` [SISL\\_SURF\\_OPEN](#) 1
- `#define` [SISL\\_SURF\\_CLOSED](#) 0

### Typedefs

- typedef struct [SISLdir](#) [SISLdir](#)
- typedef struct [SISLbox](#) [SISLbox](#)
- typedef struct [SISLCurve](#) [SISLCurve](#)
- typedef struct [SISLSurf](#) [SISLSurf](#)
- typedef struct [SISLIntcurve](#) [SISLIntcurve](#)

### Enumerations

- enum { [SI\\_RIGHT](#) =1, [SI\\_LEFT](#) =2 }
- enum { [SI\\_UNDEF](#), [SI\\_IN](#), [SI\\_OUT](#), [SI\\_ON](#), [SI\\_AT](#) }

## Functions

- [SISLbox \\* newbox \(\)](#)
- [SISLCurve \\* newCurve \(\)](#)
- [SISLCurve \\* copyCurve \(\)](#)
- [SISLdir \\* newdir \(\)](#)
- [SISLIntcurve \\* newIntcurve \(\)](#)
- [SISLSurf \\* newSurf \(\)](#)
- [SISLSurf \\* copySurface \(\)](#)
- [void freeCurve \(\)](#)
- [void freeIntcrvlist \(\)](#)
- [void freeIntcurve \(\)](#)
- [void freeSurf \(\)](#)
- [void s1001 \(\)](#)
- [void s1011 \(\)](#)
- [void s1012 \(\)](#)
- [void s1013 \(\)](#)
- [void s1014 \(\)](#)
- [void s1015 \(\)](#)
- [void s1016 \(\)](#)
- [void s1017 \(\)](#)
- [void s1018 \(\)](#)
- [void s1021 \(\)](#)
- [void s1022 \(\)](#)
- [void s1023 \(\)](#)
- [void s1024 \(\)](#)
- [void s1025 \(\)](#)
- [void s1221 \(\)](#)
- [void s1225 \(\)](#)
- [void s1226 \(\)](#)
- [void s1227 \(\)](#)
- [void s1233 \(\)](#)
- [void s1237 \(\)](#)
- [void s1238 \(\)](#)
- [void s1240 \(\)](#)
- [void s1241 \(\)](#)
- [void s1243 \(\)](#)
- [void s1302 \(\)](#)
- [void s1303 \(\)](#)
- [void s1310 \(\)](#)
- [void s1314 \(\)](#)
- [void s1315 \(\)](#)
- [void s1316 \(\)](#)
- [void s1317 \(\)](#)
- [void s1318 \(\)](#)
- [void s1319 \(\)](#)
- [void s1327 \(\)](#)
- [void s1328 \(\)](#)
- [void s1332 \(\)](#)
- [void s1356 \(\)](#)
- [void s1357 \(\)](#)
- [void s1360 \(\)](#)
- [void s1363 \(\)](#)
- [void s1364 \(\)](#)
- [void s1365 \(\)](#)

- void [s1369](#) ()
- void [s1371](#) ()
- void [s1372](#) ()
- void [s1373](#) ()
- void [s1374](#) ()
- void [s1375](#) ()
- void [s1379](#) ()
- void [s1380](#) ()
- void [s1383](#) ()
- void [s1386](#) ()
- void [s1387](#) ()
- void [s1388](#) ()
- void [s1389](#) ()
- void [s1390](#) ()
- void [s1391](#) ()
- void [s1401](#) ()
- void [s1421](#) ()
- void [s1422](#) ()
- void [s1424](#) ()
- void [s1425](#) ()
- void [s1439](#) ()
- void [s1440](#) ()
- void [s1450](#) ()
- void [s1451](#) ()
- void [s1501](#) ()
- void [s1502](#) ()
- void [s1503](#) ()
- void [s1506](#) ()
- void [s1508](#) ()
- void [s1510](#) ()
- void [s1511](#) ()
- void [s1514](#) ()
- void [s1515](#) ()
- void [s1522](#) ()
- void [s1529](#) ()
- void [s1530](#) ()
- void [s1534](#) ()
- void [s1535](#) ()
- void [s1536](#) ()
- void [s1537](#) ()
- void [s1538](#) ()
- void [s1539](#) ()
- void [s1542](#) ()
- void [s1600](#) ()
- void [s1601](#) ()
- void [s1602](#) ()
- void [s1603](#) ()
- void [s1606](#) ()
- void [s1607](#) ()
- void [s1608](#) ()
- void [s1609](#) ()
- void [s1611](#) ()
- void [s1613](#) ()
- void [s1620](#) ()
- void [s1630](#) ()

- void [s1706](#) ()
- void [s1710](#) ()
- void [s1711](#) ()
- void [s1712](#) ()
- void [s1713](#) ()
- void [s1714](#) ()
- void [s1715](#) ()
- void [s1716](#) ()
- void [s1720](#) ()
- void [s1730](#) ()
- void [s1731](#) ()
- void [s1732](#) ()
- void [s1733](#) ()
- void [s1750](#) ()
- void [s1774](#) ()
- void [s1775](#) ()
- void [s1850](#) ()
- void [s1851](#) ()
- void [s1852](#) ()
- void [s1853](#) ()
- void [s1854](#) ()
- void [s1855](#) ()
- void [s1856](#) ()
- void [s1857](#) ()
- void [s1858](#) ()
- void [s1859](#) ()
- void [s1860](#) ()
- void [s1870](#) ()
- void [s1871](#) ()
- void [s1920](#) ()
- void [s1921](#) ()
- void [s1940](#) ()
- void [s1953](#) ()
- void [s1954](#) ()
- void [s1955](#) ()
- void [s1957](#) ()
- void [s1958](#) ()
- void [s1961](#) ()
- void [s1962](#) ()
- void [s1963](#) ()
- void [s1965](#) ()
- void [s1966](#) ()
- void [s1967](#) ()
- void [s1968](#) ()
- void [s1986](#) ()
- void [s1987](#) ()
- void [s1988](#) ()
- void [s1989](#) ()
- void [s2500](#) ()
- void [s2502](#) ()
- void [s2504](#) ()
- void [s2506](#) ()
- void [s2508](#) ()
- void [s2510](#) ()
- void [s2532](#) ()

- void [s2536](#) ()
- void [s2540](#) ()
- void [s2542](#) ()
- void [s2544](#) ()
- void [s2545](#) ()
- void [s2550](#) ()
- void [s2553](#) ()
- void [s2556](#) ()
- void [s2559](#) ()
- void [s2562](#) ()
- void [s6drawseq](#) ()
- void [make\\_cv\\_cyclic](#) ()

### 30.2290.1 Macro Definition Documentation

#### 30.2290.1.1 #define GO\_API

Definition at line 75 of file sisl.h.

#### 30.2290.1.2 #define SISL\_CRV\_CLOSED 0

Definition at line 263 of file sisl.h.

#### 30.2290.1.3 #define SISL\_CRV\_OPEN 1

Definition at line 262 of file sisl.h.

#### 30.2290.1.4 #define SISL\_CRV\_PERIODIC -1

Definition at line 261 of file sisl.h.

#### 30.2290.1.5 #define SISL\_SURF\_CLOSED 0

Definition at line 267 of file sisl.h.

#### 30.2290.1.6 #define SISL\_SURF\_OPEN 1

Definition at line 266 of file sisl.h.

#### 30.2290.1.7 #define SISL\_SURF\_PERIODIC -1

Definition at line 265 of file sisl.h.

## 30.2290.2 Typedef Documentation

30.2290.2.1 typedef struct SISLbox SISLbox

30.2290.2.2 typedef struct SISLCurve SISLCurve

30.2290.2.3 typedef struct SISLdir SISLdir

30.2290.2.4 typedef struct SISLIntcurve SISLIntcurve

30.2290.2.5 typedef struct SISLSurf SISLSurf

## 30.2290.3 Enumeration Type Documentation

30.2290.3.1 anonymous enum

Enumerator

***SI\_RIGHT***

***SI\_LEFT***

Definition at line 252 of file sisl.h.

30.2290.3.2 anonymous enum

Enumerator

***SI\_UNDEF***

***SI\_IN***

***SI\_OUT***

***SI\_ON***

***SI\_AT***

Definition at line 256 of file sisl.h.



## 30.2290.4 Function Documentation

30.2290.4.1 **SISLCurve\*** copyCurve ( )

30.2290.4.2 **SISLSurf\*** copySurface ( )

30.2290.4.3 void freeCurve ( )

30.2290.4.4 void freeIntcrvlist ( )

30.2290.4.5 void freeIntcurve ( )

30.2290.4.6 void freeSurf ( )

30.2290.4.7 void make\_cv\_cyclic ( )

30.2290.4.8 **SISLbox\*** newbox ( )

30.2290.4.9 **SISLCurve\*** newCurve ( )

30.2290.4.10 **SISLdir\*** newdir ( )

30.2290.4.11 **SISLIntcurve\*** newIntcurve ( )

30.2290.4.12 **SISLSurf\*** newSurf ( )

30.2290.4.13 void s1001 ( )

30.2290.4.14 void s1011 ( )

30.2290.4.15 void s1012 ( )

30.2290.4.16 void s1013 ( )

30.2290.4.17 void s1014 ( )

30.2290.4.18 void s1015 ( )

30.2290.4.19 void s1016 ( )

30.2290.4.20 void s1017 ( )

30.2290.4.21 void s1018 ( )

30.2290.4.22 void s1021 ( )

30.2290.4.23 void s1022 ( )  
30.2290.4.24 void s1023 ( )  
30.2290.4.25 void s1024 ( )  
30.2290.4.26 void s1025 ( )  
30.2290.4.27 void s1221 ( )  
30.2290.4.28 void s1225 ( )  
30.2290.4.29 void s1226 ( )  
30.2290.4.30 void s1227 ( )  
30.2290.4.31 void s1233 ( )  
30.2290.4.32 void s1237 ( )  
30.2290.4.33 void s1238 ( )  
30.2290.4.34 void s1240 ( )  
30.2290.4.35 void s1241 ( )  
30.2290.4.36 void s1243 ( )  
30.2290.4.37 void s1302 ( )  
30.2290.4.38 void s1303 ( )  
30.2290.4.39 void s1310 ( )  
30.2290.4.40 void s1314 ( )  
30.2290.4.41 void s1315 ( )  
30.2290.4.42 void s1316 ( )  
30.2290.4.43 void s1317 ( )  
30.2290.4.44 void s1318 ( )  
30.2290.4.45 void s1319 ( )

30.2290.4.46 void s1327 ( )

30.2290.4.47 void s1328 ( )

30.2290.4.48 void s1332 ( )

30.2290.4.49 void s1356 ( )

30.2290.4.50 void s1357 ( )

30.2290.4.51 void s1360 ( )

30.2290.4.52 void s1363 ( )

30.2290.4.53 void s1364 ( )

30.2290.4.54 void s1365 ( )

30.2290.4.55 void s1369 ( )

30.2290.4.56 void s1371 ( )

30.2290.4.57 void s1372 ( )

30.2290.4.58 void s1373 ( )

30.2290.4.59 void s1374 ( )

30.2290.4.60 void s1375 ( )

30.2290.4.61 void s1379 ( )

30.2290.4.62 void s1380 ( )

30.2290.4.63 void s1383 ( )

30.2290.4.64 void s1386 ( )

30.2290.4.65 void s1387 ( )

30.2290.4.66 void s1388 ( )

30.2290.4.67 void s1389 ( )

30.2290.4.68 void s1390 ( )

30.2290.4.69 void s1391 ( )  
30.2290.4.70 void s1401 ( )  
30.2290.4.71 void s1421 ( )  
30.2290.4.72 void s1422 ( )  
30.2290.4.73 void s1424 ( )  
30.2290.4.74 void s1425 ( )  
30.2290.4.75 void s1439 ( )  
30.2290.4.76 void s1440 ( )  
30.2290.4.77 void s1450 ( )  
30.2290.4.78 void s1451 ( )  
30.2290.4.79 void s1501 ( )  
30.2290.4.80 void s1502 ( )  
30.2290.4.81 void s1503 ( )  
30.2290.4.82 void s1506 ( )  
30.2290.4.83 void s1508 ( )  
30.2290.4.84 void s1510 ( )  
30.2290.4.85 void s1511 ( )  
30.2290.4.86 void s1514 ( )  
30.2290.4.87 void s1515 ( )  
30.2290.4.88 void s1522 ( )  
30.2290.4.89 void s1529 ( )  
30.2290.4.90 void s1530 ( )  
30.2290.4.91 void s1534 ( )

30.2290.4.92 void s1535 ( )

30.2290.4.93 void s1536 ( )

30.2290.4.94 void s1537 ( )

30.2290.4.95 void s1538 ( )

30.2290.4.96 void s1539 ( )

30.2290.4.97 void s1542 ( )

30.2290.4.98 void s1600 ( )

30.2290.4.99 void s1601 ( )

30.2290.4.100 void s1602 ( )

30.2290.4.101 void s1603 ( )

30.2290.4.102 void s1606 ( )

30.2290.4.103 void s1607 ( )

30.2290.4.104 void s1608 ( )

30.2290.4.105 void s1609 ( )

30.2290.4.106 void s1611 ( )

30.2290.4.107 void s1613 ( )

30.2290.4.108 void s1620 ( )

30.2290.4.109 void s1630 ( )

30.2290.4.110 void s1706 ( )

30.2290.4.111 void s1710 ( )

30.2290.4.112 void s1711 ( )

30.2290.4.113 void s1712 ( )

30.2290.4.114 void s1713 ( )

30.2290.4.115 void s1714 ( )  
30.2290.4.116 void s1715 ( )  
30.2290.4.117 void s1716 ( )  
30.2290.4.118 void s1720 ( )  
30.2290.4.119 void s1730 ( )  
30.2290.4.120 void s1731 ( )  
30.2290.4.121 void s1732 ( )  
30.2290.4.122 void s1733 ( )  
30.2290.4.123 void s1750 ( )  
30.2290.4.124 void s1774 ( )  
30.2290.4.125 void s1775 ( )  
30.2290.4.126 void s1850 ( )  
30.2290.4.127 void s1851 ( )  
30.2290.4.128 void s1852 ( )  
30.2290.4.129 void s1853 ( )  
30.2290.4.130 void s1854 ( )  
30.2290.4.131 void s1855 ( )  
30.2290.4.132 void s1856 ( )  
30.2290.4.133 void s1857 ( )  
30.2290.4.134 void s1858 ( )  
30.2290.4.135 void s1859 ( )  
30.2290.4.136 void s1860 ( )  
30.2290.4.137 void s1870 ( )

30.2290.4.138 void s1871 ( )

30.2290.4.139 void s1920 ( )

30.2290.4.140 void s1921 ( )

30.2290.4.141 void s1940 ( )

30.2290.4.142 void s1953 ( )

30.2290.4.143 void s1954 ( )

30.2290.4.144 void s1955 ( )

30.2290.4.145 void s1957 ( )

30.2290.4.146 void s1958 ( )

30.2290.4.147 void s1961 ( )

30.2290.4.148 void s1962 ( )

30.2290.4.149 void s1963 ( )

30.2290.4.150 void s1965 ( )

30.2290.4.151 void s1966 ( )

30.2290.4.152 void s1967 ( )

30.2290.4.153 void s1968 ( )

30.2290.4.154 void s1986 ( )

30.2290.4.155 void s1987 ( )

30.2290.4.156 void s1988 ( )

30.2290.4.157 void s1989 ( )

30.2290.4.158 void s2500 ( )

30.2290.4.159 void s2502 ( )

30.2290.4.160 void s2504 ( )

30.2290.4.161 void s2506 ( )

30.2290.4.162 void s2508 ( )

30.2290.4.163 void s2510 ( )

30.2290.4.164 void s2532 ( )

30.2290.4.165 void s2536 ( )

30.2290.4.166 void s2540 ( )

30.2290.4.167 void s2542 ( )

30.2290.4.168 void s2544 ( )

30.2290.4.169 void s2545 ( )

30.2290.4.170 void s2550 ( )

30.2290.4.171 void s2553 ( )

30.2290.4.172 void s2556 ( )

30.2290.4.173 void s2559 ( )

30.2290.4.174 void s2562 ( )

30.2290.4.175 void s6drawseq ( )

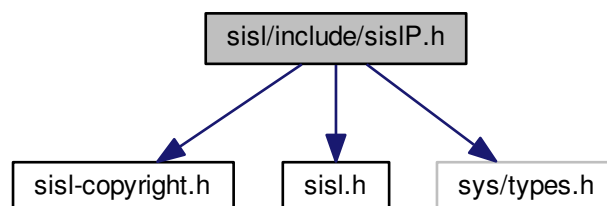
## 30.2291 sisl/include/sislIP.h File Reference

```
#include "sisl-copyright.h"
```

```
#include "sisl.h"
```

```
#include <sys/types.h>
```

Include dependency graph for sislIP.h:





## Classes

- struct [SISLPoint](#)
- struct [SISLObject](#)
- struct [SISLIntsurf](#)
- struct [SISLTrimpar](#)
- struct [SISLIntpt](#)
- struct [SISLTrack](#)
- struct [SISLIntlist](#)
- struct [SISLPtedge](#)
- struct [SISLEdge](#)
- struct [SISLIntdat](#)
- struct [rank\\_info](#)

## Macros

- #define [CONST](#)
- #define [VOIDP](#) (char \*)
- #define [CONSTVOIDP](#) (char \*)
- #define [SISLPOINT](#) 0
- #define [SISLCURVE](#) 1
- #define [SISLSURFACE](#) 2
- #define [AEPSGE](#) (double)1.e-6
- #define [AEPSCO](#) (double)0.0
- #define [REL\\_COMP\\_RES](#) (double)0.0000000000000001
- #define [REL\\_PAR\\_RES](#) (double)0.00000000000001
- #define [ANGULAR\\_TOLERANCE](#) (double)0.01 /\* IN RADIANS \*/
- #define [MAXIMAL\\_RADIUS\\_OF\\_CURVATURE](#) (double)10000.0
- #define [SISL\\_NULL](#) 0
- #define [DZERO](#) (double)0.0

## Typedefs

- typedef struct [SISLPoint](#) [SISLPoint](#)
- typedef struct [SISLObject](#) [SISLObject](#)
- typedef struct [SISLIntsurf](#) [SISLIntsurf](#)
- typedef struct [SISLTrimpar](#) [SISLTrimpar](#)
- typedef struct [SISLIntpt](#) [SISLIntpt](#)
- typedef struct [SISLTrack](#) [SISLTrack](#)
- typedef struct [SISLIntlist](#) [SISLIntlist](#)
- typedef struct [SISLPtedge](#) [SISLPtedge](#)
- typedef struct [SISLEdge](#) [SISLEdge](#)
- typedef struct [SISLIntdat](#) [SISLIntdat](#)
- typedef struct [rank\\_info](#) [rank\\_info](#)

## Enumerations

- enum { [SI\\_ORD](#) = 1, [SI\\_SING](#), [SI\\_TRIM](#), [SI\\_TOUCH](#) }

## Functions

- [SISLIntpt \\* copyIntpt \(\)](#)
- [SISLEdge \\* newEdge \(\)](#)
- [SISLIntdat \\* newIntdat \(\)](#)
- [SISLIntlist \\* newIntlist \(\)](#)
- [SISLIntpt \\* newIntpt \(\)](#)
- [SISLObject \\* newObject \(\)](#)
- [SISLPoint \\* newPoint \(\)](#)
- [SISLPtedge \\* newPtedge \(\)](#)
- [SISLIntsurf \\* newIntsurf \(\)](#)
- [SISLTrack \\* newTrack \(\)](#)
- [SISLTrimpar \\* newTrimpar \(\)](#)
- [SISLIntpt \\* hp\\_newIntpt \(\)](#)
- [SISLIntpt \\* hp\\_copyIntpt \(\)](#)
- [void freeEdge \(\)](#)
- [void freeIntdat \(\)](#)
- [void freeIntlist \(\)](#)
- [void freeIntpt \(\)](#)
- [void freeObject \(\)](#)
- [void freePoint \(\)](#)
- [void freePtedge \(\)](#)
- [void freeIntsurf \(\)](#)
- [void freeTrimpar \(\)](#)
- [void freeTrack \(\)](#)
- [void make3D \(\)](#)
- [void s1119 \(\)](#)
- [void s1161 \(\)](#)
- [void s1162 \(\)](#)
- [void s1172 \(\)](#)
- [void s1173 \(\)](#)
- [void s1174 \(\)](#)
- [void s1190 \(\)](#)
- [void s1192 \(\)](#)
- [void s1219 \(\)](#)
- [void s1220 \(\)](#)
- [void s1222 \(\)](#)
- [void s1223 \(\)](#)
- [void s1224 \(\)](#)
- [void s1231 \(\)](#)
- [void s1232 \(\)](#)
- [void s1235 \(\)](#)
- [void s1236 \(\)](#)
- [void s1239 \(\)](#)
- [void s1244 \(\)](#)
- [void s1245 \(\)](#)
- [void s1251 \(\)](#)
- [void s1252 \(\)](#)
- [void s1301 \(\)](#)
- [void s1304 \(\)](#)
- [void s1305 \(\)](#)
- [void s1306 \(\)](#)
- [void s1307 \(\)](#)
- [void s1308 \(\)](#)
- [double s1309 \(\)](#)

- [double s1311](#) ()
- [void s1312](#) ()
- [void s1313](#) ()
- [void s1320](#) ()
- [void s1321](#) ()
- [void s1322](#) ()
- [void s1323](#) ()
- [void s1324](#) ()
- [double s1325](#) ()
- [void s1329](#) ()
- [void s1330](#) ()
- [void s1331](#) ()
- [void s1333](#) ()
- [void s1333\\_count](#) ()
- [void s1333\\_cyclic](#) ()
- [void s1334](#) ()
- [void s1340](#) ()
- [void s1341](#) ()
- [void s1342](#) ()
- [void s1343](#) ()
- [void s1345](#) ()
- [void s1346](#) ()
- [void s1347](#) ()
- [void s1348](#) ()
- [void s1349](#) ()
- [void s1350](#) ()
- [void s1351](#) ()
- [void s1352](#) ()
- [void s1353](#) ()
- [void s1354](#) ()
- [void s1355](#) ()
- [void s1358](#) ()
- [void s1359](#) ()
- [void s1361](#) ()
- [void s1362](#) ()
- [void s1366](#) ()
- [void s1367](#) ()
- [void s1370](#) ()
- [void s1376](#) ()
- [void s1377](#) ()
- [void s1378](#) ()
- [void s1381](#) ()
- [void s1382](#) ()
- [void s1384](#) ()
- [void s1385](#) ()
- [void s1393](#) ()
- [void s1399](#) ()
- [void s1435](#) ()
- [void s1436](#) ()
- [void s1437](#) ()
- [void s1438](#) ()
- [void s1452](#) ()
- [void s1500](#) ()
- [void s1504](#) ()
- [void s1505](#) ()

- void [s1507](#) ()
- void [s1512](#) ()
- void [s1513](#) ()
- void [s1516](#) ()
- void [s1517](#) ()
- void [s1520](#) ()
- [SISLCurve](#) \* [s1521](#) ()
- void [s1528](#) ()
- void [s1531](#) ()
- void [s1540](#) ()
- void [s1541](#) ()
- void [s1604](#) ()
- void [s1605](#) ()
- void [s1612](#) ()
- void [s1613bez](#) ()
- void [s1614](#) ()
- void [s1615](#) ()
- void [s1616](#) ()
- void [s1617](#) ()
- void [s1618](#) ()
- void [s1619](#) ()
- void [s1700](#) ()
- void [s1701](#) ()
- void [s1705](#) ()
- void [s1707](#) ()
- void [s1708](#) ()
- void [s1741](#) ()
- void [s1753](#) ()
- void [s1754](#) ()
- void [s1755](#) ()
- void [s1770](#) ()
- void [s1770\\_2D](#) ()
- void [s1771](#) ()
- void [s1772](#) ()
- void [s1773](#) ()
- void [s1780](#) ()
- void [s1785](#) ()
- void [s1786](#) ()
- void [s1787](#) ()
- void [s1788](#) ()
- void [s1789](#) ()
- void [s1790](#) ()
- int [s1791](#) ()
- double [s1792](#) ()
- void [s1795](#) ()
- void [s1796](#) ()
- void [s1797](#) ()
- void [s1830](#) ()
- void [s1834](#) ()
- void [s1839](#) ()
- void [s1840](#) ()
- void [s1880](#) ()
- void [s1890](#) ()
- void [s1891](#) ()
- void [s1893](#) ()

- void [s1894](#) ()
- void [s1896](#) ()
- void [s1897](#) ()
- void [s1900](#) ()
- void [s1901](#) ()
- void [s1902](#) ()
- void [s1903](#) ()
- void [s1904](#) ()
- void [s1905](#) ()
- void [s1906](#) ()
- void [s1907](#) ()
- void [s1908](#) ()
- void [s1909](#) ()
- void [s1910](#) ()
- void [s1911](#) ()
- void [s1912](#) ()
- void [s1916](#) ()
- void [s1917](#) ()
- void [s1918](#) ()
- void [s1919](#) ()
- void [s1924](#) ()
- void [s1925](#) ()
- void [s1926](#) ()
- void [s1927](#) ()
- void [s1931](#) ()
- void [s1931unit](#) ()
- void [s1932](#) ()
- void [s1933](#) ()
- void [s1934](#) ()
- void [s1935](#) ()
- void [s1936](#) ()
- void [s1937](#) ()
- void [s1938](#) ()
- void [s1940](#) ()
- void [s1941](#) ()
- void [s1942](#) ()
- void [s1943](#) ()
- void [s1944](#) ()
- void [s1945](#) ()
- void [s1946](#) ()
- void [s1947](#) ()
- void [s1948](#) ()
- void [s1949](#) ()
- void [s1950](#) ()
- void [s1951](#) ()
- void [s1956](#) ()
- void [s1959](#) ()
- void [s1960](#) ()
- void [s1990](#) ()
- void [s1991](#) ()
- void [s1992](#) ()
- void [s1992cu](#) ()
- void [s1992su](#) ()
- void [s1993](#) ()
- void [s1994](#) ()

- void [s2501](#) ()
- void [s2503](#) ()
- void [s2505](#) ()
- void [s2507](#) ()
- void [s2509](#) ()
- void [s2511](#) ()
- void [s2512](#) ()
- void [s2513](#) ()
- void [s2514](#) ()
- void [s2515](#) ()
- void [s2516](#) ()
- void [s2533](#) ()
- void [s2534](#) ()
- void [s2535](#) ()
- void [s2541](#) ()
- void [s2543](#) ()
- void [s2551](#) ()
- void [s2554](#) ()
- void [s2555](#) ()
- void [s2557](#) ()
- void [s2558](#) ()
- void [s2560](#) ()
- void [s2561](#) ()
- void [s6addcurve](#) ()
- [double s6affdist](#) ()
- [double s6ang](#) ()
- [double s6angle](#) ()
- void [s6chpar](#) ()
- void [s6crss](#) ()
- void [s6crvcheck](#) ()
- void [s6curvature](#) ()
- void [s6curvrad](#) ()
- void [s6deCasteljau](#) ()
- void [s6decomp](#) ()
- void [s6degnorm](#) ()
- void [s6dertopt](#) ()
- void [s6diff](#) ()
- [double s6dist](#) ()
- [double s6dline](#) ()
- [double s6dplane](#) ()
- int [s6equal](#) ()
- void [s6err](#) ()
- int [s6existbox](#) ()
- void [s6findfac](#) ()
- void [s6fndintvl](#) ()
- void [s6herm](#) ()
- void [s6hermite\\_bezier](#) ()
- void [s6idcon](#) ()
- void [s6idcpt](#) ()
- void [s6idedg](#) ()
- void [s6identify](#) ()
- void [s6idget](#) ()
- void [s6idint](#) ()
- void [s6idklist](#) ()
- void [s6idkpt](#) ()

- void [s6idlis](#) ()
- void [s6idnpt](#) ()
- void [s6idput](#) ()
- void [s6inv4](#) ()
- void [s6invert](#) ()
- int [s6knotmult](#) ()
- double [s6length](#) ()
- void [s6line](#) ()
- double [s6lprj](#) ()
- void [s6lufacp](#) ()
- void [s6lusolp](#) ()
- void [s6metric](#) ()
- void [s6move](#) ()
- void [s6mulvec](#) ()
- void [s6mvec](#) ()
- void [s6newbox](#) ()
- double [s6norm](#) ()
- void [s6ratder](#) ()
- void [s6rotax](#) ()
- double [s6scpr](#) ()
- void [s6rotmat](#) ()
- double [s6schoen](#) ()
- void [s6sortpar](#) ()
- void [s6sratder](#) ()
- void [s6strider](#) ()
- void [s6takeunion](#) ()
- void [s6twonorm](#) ()
- double [s9adsimp](#) ()
- double [s9adstep](#) ()
- void [s9boundimp](#) ()
- void [s9boundit](#) ()
- void [s9clipimp](#) ()
- void [s9clipit](#) ()
- void [s9conmarch](#) ()
- void [s9iterate](#) ()
- void [s9iterimp](#) ()
- void [s9simple\\_knot](#) ()
- void [s9surmarch](#) ()
- void [sh1260](#) ()
- void [sh1261](#) ()
- void [sh1262](#) ()
- void [sh1263](#) ()
- void [sh1365](#) ()
- void [sh1369](#) ()
- void [sh1371](#) ()
- void [sh1372](#) ()
- void [sh1373](#) ()
- void [sh1374](#) ()
- void [sh1375](#) ()
- void [sh1460](#) ()
- void [sh1461](#) ()
- void [sh1462](#) ()
- void [sh1463](#) ()
- void [sh1464](#) ()
- void [sh1465](#) ()

- void [sh1466](#) ()
- void [sh1467](#) ()
- void [sh1502](#) ()
- void [sh1503](#) ()
- void [sh1510](#) ()
- void [sh1511](#) ()
- void [sh1761](#) ()
- void [sh1762](#) ()
- void [sh1779](#) ()
- void [sh1779\\_at](#) ()
- void [sh1780](#) ()
- void [sh1780\\_at](#) ()
- void [sh1781](#) ()
- void [sh1781\\_at](#) ()
- void [sh1782](#) ()
- void [sh1783](#) ()
- void [sh1784](#) ()
- void [sh1786](#) ()
- void [sh1787](#) ()
- void [sh1790](#) ()
- void [sh1830](#) ()
- void [sh1831](#) ()
- void [sh1834](#) ()
- void [sh1839](#) ()
- void [sh1850](#) ()
- void [sh1851](#) ()
- void [sh1852](#) ()
- void [sh1853](#) ()
- void [sh1854](#) ()
- void [sh1855](#) ()
- void [sh1856](#) ()
- void [sh1857](#) ()
- void [sh1858](#) ()
- void [sh1859](#) ()
- void [sh1860](#) ()
- void [sh1870](#) ()
- void [sh1871](#) ()
- void [sh1922](#) ()
- void [sh1923](#) ()
- void [sh1924](#) ()
- void [sh1925](#) ()
- void [sh1926](#) ()
- void [sh1927](#) ()
- void [sh1928](#) ()
- void [sh1929](#) ()
- void [sh1930](#) ()
- void [sh1992](#) ()
- void [sh1992cu](#) ()
- void [sh1992su](#) ()
- void [sh1993](#) ()
- void [sh1994](#) ()
- void [sh6closevert](#) ()
- void [sh6cvvert](#) ()
- void [sh6comedg](#) ()
- void [sh6connect](#) ()



- int [sh6count](#) ()
- void [sh6degen](#) ()
- void [sh6disconnect](#) ()
- void [sh6edgpoint](#) ()
- void [sh6edgred](#) ()
- void [sh6evalint](#) ()
- void [sh6findsplit](#) ()
- void [sh6floop](#) ()
- void [sh6getgeom](#) ()
- void [sh6getlist](#) ()
- [SISLIntpt](#) \* [sh6getmain](#) ()
- [SISLIntpt](#) \* [sh6getnext](#) ()
- void [sh6getnhbrs](#) ()
- void [sh6getother](#) ()
- int [sh6getprev](#) ()
- void [sh6gettop](#) ()
- void [sh6gettophlp](#) ()
- void [sh6idalledg](#) ()
- void [sh6idcon](#) ()
- void [sh6idfcross](#) ()
- void [sh6idget](#) ()
- void [sh6idkpt](#) ()
- void [sh6idlis](#) ()
- void [sh6idnpt](#) ()
- void [sh6idnewunite](#) ()
- void [sh6idput](#) ()
- void [sh6idrmcross](#) ()
- void [sh6idsplit](#) ()
- void [sh6idunite](#) ()
- void [sh6insert](#) ()
- void [sh6insertpt](#) ()
- int [sh6isconnect](#) ()
- int [sh6ishelp](#) ()
- void [sh6isinside](#) ()
- int [sh6ismain](#) ()
- int [sh6nmbhelp](#) ()
- int [sh6nmbmain](#) ()
- void [sh6ptobj](#) ()
- void [sh6putsing](#) ()
- void [sh6puttouch](#) ()
- void [sh6red](#) ()
- void [sh6remcon](#) ()
- void [sh6removept](#) ()
- void [sh6sepcrv](#) ()
- void [sh6setcnsdir](#) ()
- void [sh6setdir](#) ()
- void [sh6settop](#) ()
- void [sh6splitgeom](#) ()
- void [sh6tohelp](#) ()
- void [sh6tomain](#) ()
- void [sh6trimlist](#) ()
- void [crv\\_arc\\_tang](#) ()
- void [crv\\_crv\\_tang](#) ()
- void [crv\\_lin\\_tang](#) ()
- void [ev\\_cv\\_off](#) ()

- void [eval\\_2\\_crv](#) ()
- void [eval\\_crv\\_arc](#) ()
- void [hp\\_s1880](#) ()
- void [int\\_join\\_per](#) ()
- void [make\\_cv\\_kreg](#) ()
- void [make\\_sf\\_kreg](#) ()
- void [make\\_tracks](#) ()
- void [newknots](#) ()
- void [pick\\_crv\\_sf](#) ()
- void [po\\_crv\\_tang](#) ()
- void [refine\\_all](#) ()
- void [sh\\_1d\\_div](#) ()
- void [sh\\_div\\_crv](#) ()
- void [sh\\_div\\_surf](#) ()
- void [sh\\_set\\_at](#) ()
- void [shape](#) ()
- void [shcheckpoint](#) ()
- int [shchecktype](#) ()
- void [shcsfsing](#) ()
- void [shevalc](#) ()
- void [shmklppts](#) ()
- void [shsing](#) ()
- void [spli\\_silh](#) ()
- void [test\\_cyclic\\_knots](#) ()

### 30.2291.1 Macro Definition Documentation

#### 30.2291.1.1 `#define AEPSCO (double)0.0`

Definition at line 380 of file sisIP.h.

#### 30.2291.1.2 `#define AEPSTGE (double)1.e-6`

Definition at line 379 of file sisIP.h.

#### 30.2291.1.3 `#define ANGULAR_TOLERANCE (double)0.01 /* IN RADIANS */`

Definition at line 384 of file sisIP.h.

#### 30.2291.1.4 `#define CONST`

Definition at line 114 of file sisIP.h.

#### 30.2291.1.5 `#define CONSTVOIDP (char *)`

Definition at line 117 of file sisIP.h.

30.2291.1.6 `#define DZERO (double)0.0`

Definition at line 399 of file sislP.h.

30.2291.1.7 `#define MAXIMAL_RADIUS_OF_CURVATURE (double)10000.0`

Definition at line 385 of file sislP.h.

30.2291.1.8 `#define REL_COMP_RES (double)0.0000000000000001`

Definition at line 382 of file sislP.h.

30.2291.1.9 `#define REL_PAR_RES (double)0.000000000001`

Definition at line 383 of file sislP.h.

30.2291.1.10 `#define SISL_NULL 0`

Definition at line 393 of file sislP.h.

30.2291.1.11 `#define SISLCURVE 1`

Definition at line 375 of file sislP.h.

30.2291.1.12 `#define SISLPOINT 0`

Definition at line 374 of file sislP.h.

30.2291.1.13 `#define SISLSURFACE 2`

Definition at line 376 of file sislP.h.

30.2291.1.14 `#define VOIDP (char *)`

Definition at line 116 of file sislP.h.

## 30.2291.2 Typedef Documentation

30.2291.2.1 typedef struct rank\_info rank\_info

30.2291.2.2 typedef struct SISLEdge SISLEdge

30.2291.2.3 typedef struct SISLIntdat SISLIntdat

30.2291.2.4 typedef struct SISLIntlist SISLIntlist

30.2291.2.5 typedef struct SISLIntpt SISLIntpt

30.2291.2.6 typedef struct SISLIntsurf SISLIntsurf

30.2291.2.7 typedef struct SISLObject SISLObject

30.2291.2.8 typedef struct SISLPoint SISLPoint

30.2291.2.9 typedef struct SISLPtedge SISLPtedge

30.2291.2.10 typedef struct SISLTrack SISLTrack

30.2291.2.11 typedef struct SISLTrimpar SISLTrimpar

## 30.2291.3 Enumeration Type Documentation

30.2291.3.1 anonymous enum

Enumerator

***SI\_ORD***

***SI\_SING***

***SI\_TRIM***

***SI\_TOUCH***

Definition at line 356 of file sisIP.h.

## 30.2291.4 Function Documentation

30.2291.4.1 **SISLIntpt\*** copyIntpt ( )

30.2291.4.2 void crv\_arc\_tang ( )

30.2291.4.3 void crv\_crv\_tang ( )

30.2291.4.4 void crv\_lin\_tang ( )

30.2291.4.5 void ev\_cv\_off ( )

30.2291.4.6 void eval\_2\_crv ( )

30.2291.4.7 void eval\_crv\_arc ( )

30.2291.4.8 void freeEdge ( )

30.2291.4.9 void freeIntdat ( )

30.2291.4.10 void freeIntlist ( )

30.2291.4.11 void freeIntpt ( )

30.2291.4.12 void freeIntsurf ( )

30.2291.4.13 void freeObject ( )

30.2291.4.14 void freePoint ( )

30.2291.4.15 void freePtedge ( )

30.2291.4.16 void freeTrack ( )

30.2291.4.17 void freeTrimpar ( )

30.2291.4.18 **SISLIntpt\*** hp\_copyIntpt ( )

30.2291.4.19 **SISLIntpt\*** hp\_newIntpt ( )

30.2291.4.20 void hp\_s1880 ( )

30.2291.4.21 void int\_join\_per ( )

30.2291.4.22 void make3D ( )

30.2291.4.23 void make\_cv\_kreg ( )

30.2291.4.24 void make\_sf\_kreg ( )

30.2291.4.25 void make\_tracks ( )

30.2291.4.26 **SISLEdge\*** newEdge ( )

30.2291.4.27 **SISLIntdat\*** newIntdat ( )

Definition at line 681 of file construct.c.

- 30.2291.4.28 **SISLIntlist\*** newIntlist ( )
- 30.2291.4.29 **SISLIntpt\*** newIntpt ( )
- 30.2291.4.30 **SISLIntsurf\*** newIntsurf ( )
- 30.2291.4.31 void newknots ( )
- 30.2291.4.32 **SISLObject\*** newObject ( )
- 30.2291.4.33 **SISLPoint\*** newPoint ( )
- 30.2291.4.34 **SISLPtedge\*** newPtedge ( )
- 30.2291.4.35 **SISLTrack\*** newTrack ( )
- 30.2291.4.36 **SISLTrimpar\*** newTrimpar ( )
- 30.2291.4.37 void pick\_crv\_sf ( )
- 30.2291.4.38 void po\_crv\_tang ( )
- 30.2291.4.39 void refine\_all ( )
- 30.2291.4.40 void s1119 ( )
- 30.2291.4.41 void s1161 ( )
- 30.2291.4.42 void s1162 ( )
- 30.2291.4.43 void s1172 ( )
- 30.2291.4.44 void s1173 ( )
- 30.2291.4.45 void s1174 ( )
- 30.2291.4.46 void s1190 ( )
- 30.2291.4.47 void s1192 ( )
- 30.2291.4.48 void s1219 ( )
- 30.2291.4.49 void s1220 ( )
- 30.2291.4.50 void s1222 ( )

30.2291.4.51 void s1223 ( )

30.2291.4.52 void s1224 ( )

30.2291.4.53 void s1231 ( )

30.2291.4.54 void s1232 ( )

30.2291.4.55 void s1235 ( )

30.2291.4.56 void s1236 ( )

30.2291.4.57 void s1239 ( )

30.2291.4.58 void s1244 ( )

30.2291.4.59 void s1245 ( )

30.2291.4.60 void s1251 ( )

30.2291.4.61 void s1252 ( )

30.2291.4.62 void s1301 ( )

30.2291.4.63 void s1304 ( )

30.2291.4.64 void s1305 ( )

30.2291.4.65 void s1306 ( )

30.2291.4.66 void s1307 ( )

30.2291.4.67 void s1308 ( )

30.2291.4.68 double s1309 ( )

30.2291.4.69 double s1311 ( )

30.2291.4.70 void s1312 ( )

30.2291.4.71 void s1313 ( )

30.2291.4.72 void s1320 ( )

30.2291.4.73 void s1321 ( )

30.2291.4.74 void s1322 ( )

30.2291.4.75 void s1323 ( )

30.2291.4.76 void s1324 ( )

30.2291.4.77 double s1325 ( )

30.2291.4.78 void s1329 ( )

30.2291.4.79 void s1330 ( )

30.2291.4.80 void s1331 ( )

30.2291.4.81 void s1333 ( )

30.2291.4.82 void s1333\_count ( )

30.2291.4.83 void s1333\_cyclic ( )

30.2291.4.84 void s1334 ( )

30.2291.4.85 void s1340 ( )

30.2291.4.86 void s1341 ( )

30.2291.4.87 void s1342 ( )

30.2291.4.88 void s1343 ( )

30.2291.4.89 void s1345 ( )

30.2291.4.90 void s1346 ( )

30.2291.4.91 void s1347 ( )

30.2291.4.92 void s1348 ( )

30.2291.4.93 void s1349 ( )

30.2291.4.94 void s1350 ( )

30.2291.4.95 void s1351 ( )

30.2291.4.96 void s1352 ( )



30.2291.4.97 void s1353 ( )

30.2291.4.98 void s1354 ( )

30.2291.4.99 void s1355 ( )

30.2291.4.100 void s1358 ( )

30.2291.4.101 void s1359 ( )

30.2291.4.102 void s1361 ( )

30.2291.4.103 void s1362 ( )

30.2291.4.104 void s1366 ( )

30.2291.4.105 void s1367 ( )

30.2291.4.106 void s1370 ( )

30.2291.4.107 void s1376 ( )

30.2291.4.108 void s1377 ( )

30.2291.4.109 void s1378 ( )

30.2291.4.110 void s1381 ( )

30.2291.4.111 void s1382 ( )

30.2291.4.112 void s1384 ( )

30.2291.4.113 void s1385 ( )

30.2291.4.114 void s1393 ( )

30.2291.4.115 void s1399 ( )

30.2291.4.116 void s1435 ( )

30.2291.4.117 void s1436 ( )

30.2291.4.118 void s1437 ( )

30.2291.4.119 void s1438 ( )

30.2291.4.120 void s1452 ( )

30.2291.4.121 void s1500 ( )

30.2291.4.122 void s1504 ( )

30.2291.4.123 void s1505 ( )

30.2291.4.124 void s1507 ( )

30.2291.4.125 void s1512 ( )

30.2291.4.126 void s1513 ( )

30.2291.4.127 void s1516 ( )

30.2291.4.128 void s1517 ( )

30.2291.4.129 void s1520 ( )

30.2291.4.130 **SISLCurve\*** s1521 ( )

30.2291.4.131 void s1528 ( )

30.2291.4.132 void s1531 ( )

30.2291.4.133 void s1540 ( )

30.2291.4.134 void s1541 ( )

30.2291.4.135 void s1604 ( )

30.2291.4.136 void s1605 ( )

30.2291.4.137 void s1612 ( )

30.2291.4.138 void s1613bez ( )

30.2291.4.139 void s1614 ( )

30.2291.4.140 void s1615 ( )

30.2291.4.141 void s1616 ( )

30.2291.4.142 void s1617 ( )

30.2291.4.143 void s1618 ( )

30.2291.4.144 void s1619 ( )

30.2291.4.145 void s1700 ( )

30.2291.4.146 void s1701 ( )

30.2291.4.147 void s1705 ( )

30.2291.4.148 void s1707 ( )

30.2291.4.149 void s1708 ( )

30.2291.4.150 void s1741 ( )

30.2291.4.151 void s1753 ( )

30.2291.4.152 void s1754 ( )

30.2291.4.153 void s1755 ( )

30.2291.4.154 void s1770 ( )

30.2291.4.155 void s1770\_2D ( )

30.2291.4.156 void s1771 ( )

30.2291.4.157 void s1772 ( )

30.2291.4.158 void s1773 ( )

30.2291.4.159 void s1780 ( )

30.2291.4.160 void s1785 ( )

30.2291.4.161 void s1786 ( )

30.2291.4.162 void s1787 ( )

30.2291.4.163 void s1788 ( )

30.2291.4.164 void s1789 ( )

30.2291.4.165 void s1790 ( )

30.2291.4.166 int s1791 ( )

30.2291.4.167 double s1792 ( )

30.2291.4.168 void s1795 ( )

30.2291.4.169 void s1796 ( )

30.2291.4.170 void s1797 ( )

30.2291.4.171 void s1830 ( )

30.2291.4.172 void s1834 ( )

30.2291.4.173 void s1839 ( )

30.2291.4.174 void s1840 ( )

30.2291.4.175 void s1880 ( )

30.2291.4.176 void s1890 ( )

30.2291.4.177 void s1891 ( )

30.2291.4.178 void s1893 ( )

30.2291.4.179 void s1894 ( )

30.2291.4.180 void s1896 ( )

30.2291.4.181 void s1897 ( )

30.2291.4.182 void s1900 ( )

30.2291.4.183 void s1901 ( )

30.2291.4.184 void s1902 ( )

30.2291.4.185 void s1903 ( )

30.2291.4.186 void s1904 ( )

30.2291.4.187 void s1905 ( )

30.2291.4.188 void s1906 ( )

30.2291.4.189 void s1907 ( )

30.2291.4.190 void s1908 ( )

30.2291.4.191 void s1909 ( )

30.2291.4.192 void s1910 ( )

30.2291.4.193 void s1911 ( )

30.2291.4.194 void s1912 ( )

30.2291.4.195 void s1916 ( )

30.2291.4.196 void s1917 ( )

30.2291.4.197 void s1918 ( )

30.2291.4.198 void s1919 ( )

30.2291.4.199 void s1924 ( )

30.2291.4.200 void s1925 ( )

30.2291.4.201 void s1926 ( )

30.2291.4.202 void s1927 ( )

30.2291.4.203 void s1931 ( )

30.2291.4.204 void s1931unit ( )

30.2291.4.205 void s1932 ( )

30.2291.4.206 void s1933 ( )

30.2291.4.207 void s1934 ( )

30.2291.4.208 void s1935 ( )

30.2291.4.209 void s1936 ( )

30.2291.4.210 void s1937 ( )

30.2291.4.211 void s1938 ( )

30.2291.4.212 void s1940 ( )

30.2291.4.213 void s1941 ( )

30.2291.4.214 void s1942 ( )

30.2291.4.215 void s1943 ( )

30.2291.4.216 void s1944 ( )

30.2291.4.217 void s1945 ( )

30.2291.4.218 void s1946 ( )

30.2291.4.219 void s1947 ( )

30.2291.4.220 void s1948 ( )

30.2291.4.221 void s1949 ( )

30.2291.4.222 void s1950 ( )

30.2291.4.223 void s1951 ( )

30.2291.4.224 void s1956 ( )

30.2291.4.225 void s1959 ( )

30.2291.4.226 void s1960 ( )

30.2291.4.227 void s1990 ( )

30.2291.4.228 void s1991 ( )

30.2291.4.229 void s1992 ( )

30.2291.4.230 void s1992cu ( )

30.2291.4.231 void s1992su ( )

30.2291.4.232 void s1993 ( )

30.2291.4.233 void s1994 ( )

30.2291.4.234 void s2501 ( )

30.2291.4.235 void s2503 ( )

30.2291.4.236 void s2505 ( )

30.2291.4.237 void s2507 ( )

30.2291.4.238 void s2509 ( )

30.2291.4.239 void s2511 ( )

30.2291.4.240 void s2512 ( )

30.2291.4.241 void s2513 ( )

30.2291.4.242 void s2514 ( )

30.2291.4.243 void s2515 ( )

30.2291.4.244 void s2516 ( )

30.2291.4.245 void s2533 ( )

30.2291.4.246 void s2534 ( )

30.2291.4.247 void s2535 ( )

30.2291.4.248 void s2541 ( )

30.2291.4.249 void s2543 ( )

30.2291.4.250 void s2551 ( )

30.2291.4.251 void s2554 ( )

30.2291.4.252 void s2555 ( )

30.2291.4.253 void s2557 ( )

30.2291.4.254 void s2558 ( )

30.2291.4.255 void s2560 ( )

30.2291.4.256 void s2561 ( )

30.2291.4.257 void s6addcurve ( )

30.2291.4.258 **double** s6affdist ( )

30.2291.4.259 **double** s6ang ( )

30.2291.4.260 **double** s6angle ( )

30.2291.4.261 **void** s6chpar ( )

30.2291.4.262 **void** s6crss ( )

30.2291.4.263 **void** s6crvcheck ( )

30.2291.4.264 **void** s6curvature ( )

30.2291.4.265 **void** s6curvrad ( )

30.2291.4.266 **void** s6deCasteljau ( )

30.2291.4.267 **void** s6decomp ( )

30.2291.4.268 **void** s6degnorm ( )

30.2291.4.269 **void** s6dertopt ( )

30.2291.4.270 **void** s6diff ( )

30.2291.4.271 **double** s6dist ( )

30.2291.4.272 **double** s6dline ( )

30.2291.4.273 **double** s6dplane ( )

30.2291.4.274 **int** s6equal ( )

30.2291.4.275 **void** s6err ( )

30.2291.4.276 **int** s6existbox ( )

30.2291.4.277 **void** s6findfac ( )

30.2291.4.278 **void** s6fndintvl ( )

30.2291.4.279 **void** s6herm ( )

30.2291.4.280 **void** s6hermite\_bezier ( )



30.2291.4.281 void s6idcon ( )

30.2291.4.282 void s6idcpt ( )

30.2291.4.283 void s6idedg ( )

30.2291.4.284 void s6identify ( )

30.2291.4.285 void s6idget ( )

30.2291.4.286 void s6idint ( )

30.2291.4.287 void s6idklist ( )

30.2291.4.288 void s6idkpt ( )

30.2291.4.289 void s6idlis ( )

30.2291.4.290 void s6idnpt ( )

30.2291.4.291 void s6idput ( )

30.2291.4.292 void s6inv4 ( )

30.2291.4.293 void s6invert ( )

30.2291.4.294 int s6knotmult ( )

30.2291.4.295 double s6length ( )

30.2291.4.296 void s6line ( )

30.2291.4.297 double s6lprj ( )

30.2291.4.298 void s6lufacp ( )

30.2291.4.299 void s6lusolp ( )

30.2291.4.300 void s6metric ( )

30.2291.4.301 void s6move ( )

30.2291.4.302 void s6mulvec ( )

30.2291.4.303 void s6mvec ( )

30.2291.4.304 void s6newbox ( )

30.2291.4.305 double s6norm ( )

30.2291.4.306 void s6ratder ( )

30.2291.4.307 void s6rotax ( )

30.2291.4.308 void s6rotmat ( )

30.2291.4.309 double s6schoen ( )

30.2291.4.310 double s6scpr ( )

30.2291.4.311 void s6sortpar ( )

30.2291.4.312 void s6sratder ( )

30.2291.4.313 void s6strider ( )

30.2291.4.314 void s6takeunion ( )

30.2291.4.315 void s6twonorm ( )

30.2291.4.316 double s9adsimp ( )

30.2291.4.317 double s9adstep ( )

30.2291.4.318 void s9boundimp ( )

30.2291.4.319 void s9boundit ( )

30.2291.4.320 void s9clipimp ( )

30.2291.4.321 void s9clipit ( )

30.2291.4.322 void s9conmarch ( )

30.2291.4.323 void s9iterate ( )

30.2291.4.324 void s9iterimp ( )

30.2291.4.325 void s9simple\_knot ( )

30.2291.4.326 void s9surmarch ( )

30.2291.4.327 void sh1260 ( )

30.2291.4.328 void sh1261 ( )

30.2291.4.329 void sh1262 ( )

30.2291.4.330 void sh1263 ( )

30.2291.4.331 void sh1365 ( )

30.2291.4.332 void sh1369 ( )

30.2291.4.333 void sh1371 ( )

30.2291.4.334 void sh1372 ( )

30.2291.4.335 void sh1373 ( )

30.2291.4.336 void sh1374 ( )

30.2291.4.337 void sh1375 ( )

30.2291.4.338 void sh1460 ( )

30.2291.4.339 void sh1461 ( )

30.2291.4.340 void sh1462 ( )

30.2291.4.341 void sh1463 ( )

30.2291.4.342 void sh1464 ( )

30.2291.4.343 void sh1465 ( )

30.2291.4.344 void sh1466 ( )

30.2291.4.345 void sh1467 ( )

30.2291.4.346 void sh1502 ( )

30.2291.4.347 void sh1503 ( )

30.2291.4.348 void sh1510 ( )

30.2291.4.349 void sh1511 ( )

30.2291.4.350 void sh1761 ( )

30.2291.4.351 void sh1762 ( )

30.2291.4.352 void sh1779 ( )

30.2291.4.353 void sh1779\_at ( )

30.2291.4.354 void sh1780 ( )

30.2291.4.355 void sh1780\_at ( )

30.2291.4.356 void sh1781 ( )

30.2291.4.357 void sh1781\_at ( )

30.2291.4.358 void sh1782 ( )

30.2291.4.359 void sh1783 ( )

30.2291.4.360 void sh1784 ( )

30.2291.4.361 void sh1786 ( )

30.2291.4.362 void sh1787 ( )

30.2291.4.363 void sh1790 ( )

30.2291.4.364 void sh1830 ( )

30.2291.4.365 void sh1831 ( )

30.2291.4.366 void sh1834 ( )

30.2291.4.367 void sh1839 ( )

30.2291.4.368 void sh1850 ( )

30.2291.4.369 void sh1851 ( )

30.2291.4.370 void sh1852 ( )

30.2291.4.371 void sh1853 ( )

30.2291.4.372 void sh1854 ( )

30.2291.4.373 void sh1855 ( )

30.2291.4.374 void sh1856 ( )

30.2291.4.375 void sh1857 ( )

30.2291.4.376 void sh1858 ( )

30.2291.4.377 void sh1859 ( )

30.2291.4.378 void sh1860 ( )

30.2291.4.379 void sh1870 ( )

30.2291.4.380 void sh1871 ( )

30.2291.4.381 void sh1922 ( )

30.2291.4.382 void sh1923 ( )

30.2291.4.383 void sh1924 ( )

30.2291.4.384 void sh1925 ( )

30.2291.4.385 void sh1926 ( )

30.2291.4.386 void sh1927 ( )

30.2291.4.387 void sh1928 ( )

30.2291.4.388 void sh1929 ( )

30.2291.4.389 void sh1930 ( )

30.2291.4.390 void sh1992 ( )

30.2291.4.391 void sh1992cu ( )

30.2291.4.392 void sh1992su ( )

30.2291.4.393 void sh1993 ( )

30.2291.4.394 void sh1994 ( )

30.2291.4.395 void sh6closevert ( )

30.2291.4.396 void sh6comedg ( )

30.2291.4.397 void sh6connect ( )

30.2291.4.398 int sh6count ( )

30.2291.4.399 void sh6cvvert ( )

30.2291.4.400 void sh6degen ( )

30.2291.4.401 void sh6disconnect ( )

30.2291.4.402 void sh6edgpoint ( )

30.2291.4.403 void sh6edgred ( )

30.2291.4.404 void sh6evalint ( )

30.2291.4.405 void sh6findsplit ( )

30.2291.4.406 void sh6floop ( )

30.2291.4.407 void sh6getgeom ( )

30.2291.4.408 void sh6getlist ( )

30.2291.4.409 **SISLIntpt\*** sh6getmain ( )

30.2291.4.410 **SISLIntpt\*** sh6getnext ( )

30.2291.4.411 void sh6getnhbrs ( )

30.2291.4.412 void sh6getother ( )

30.2291.4.413 int sh6getprev ( )

30.2291.4.414 void sh6gettop ( )

30.2291.4.415 void sh6gettophlp ( )

30.2291.4.416 void sh6idalledg ( )

30.2291.4.417 void sh6idcon ( )

30.2291.4.418 void sh6idfcross ( )

30.2291.4.419 void sh6idget ( )

30.2291.4.420 void sh6idkpt ( )

30.2291.4.421 void sh6idlis ( )

30.2291.4.422 void sh6idnewunite ( )

30.2291.4.423 void sh6idnpt ( )

30.2291.4.424 void sh6idput ( )

30.2291.4.425 void sh6idrmcross ( )

30.2291.4.426 void sh6idsplit ( )

30.2291.4.427 void sh6idunite ( )

30.2291.4.428 void sh6insert ( )

30.2291.4.429 void sh6insertpt ( )

30.2291.4.430 int sh6isconnect ( )

30.2291.4.431 int sh6ishelp ( )

30.2291.4.432 void sh6isinside ( )

30.2291.4.433 int sh6ismain ( )

30.2291.4.434 int sh6nmbhelp ( )

30.2291.4.435 int sh6nmbmain ( )

30.2291.4.436 void sh6ptobj ( )

30.2291.4.437 void sh6putsing ( )

30.2291.4.438 void sh6puttouch ( )

30.2291.4.439 void sh6red ( )

30.2291.4.440 void sh6remcon ( )

30.2291.4.441 void sh6removept ( )

30.2291.4.442 void sh6sepcrv ( )

30.2291.4.443 void sh6setcnsdir ( )

30.2291.4.444 void sh6setdir ( )

30.2291.4.445 void sh6settop ( )

30.2291.4.446 void sh6splitgeom ( )

30.2291.4.447 void sh6tohelp ( )

30.2291.4.448 void sh6tomain ( )

30.2291.4.449 void sh6trimlist ( )

30.2291.4.450 void sh\_1d\_div ( )

30.2291.4.451 void sh\_div\_crv ( )

30.2291.4.452 void sh\_div\_surf ( )

30.2291.4.453 void sh\_set\_at ( )

30.2291.4.454 void shape ( )

30.2291.4.455 void shcheckput ( )

30.2291.4.456 int shchecktype ( )

30.2291.4.457 void shcsfsing ( )

30.2291.4.458 void shevalc ( )

30.2291.4.459 void shmklppts ( )

30.2291.4.460 void shsing ( )

30.2291.4.461 void spli\_silh ( )

30.2291.4.462 void test\_cyclic\_knots ( )

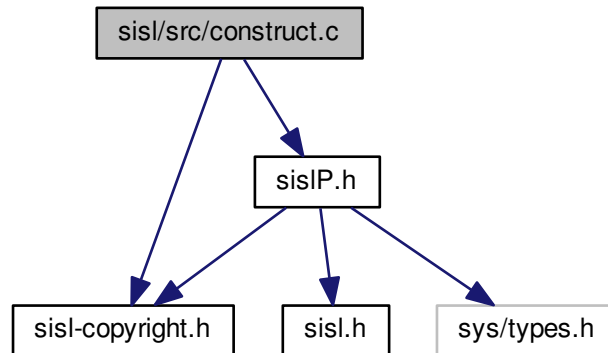


## 30.2292 sisl/src/construct.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for construct.c:



### Macros

- `#define CONSTRUCT`

### Functions

- `SISLIntpt * copyIntpt (SISLIntpt *ppt)`
- `SISLIntpt * hp_copyIntpt (SISLIntpt *ppt)`
- `SISLbox * newbox (int idim)`
- `SISLCurve * newCurve (int in, int ik, double *et, double *ecoef, int ikind, int idim, int icopy)`
- `SISLdir * newdir (int idim)`
- `SISLEdge * newEdge (int iedge)`
- `SISLIntcurve * newIntcurve (int ipoint, int ipar1, int ipar2, double *epar1, double *epar2, int itype)`
- `SISLIntdat * newIntdat ()`
- `SISLIntlist * newIntlist (SISLIntpt *pfirst, SISLIntpt *plast, int itype)`
- `SISLIntpt * newIntpt (int ipar, double *epar, double adist)`
- `SISLIntsurf * newIntsurf (SISLIntlist *pintlist)`
- `SISLTrimpar * newTrimpar (int pt, int par)`
- `SISLIntpt * hp_newIntpt (int ipar, double *epar, double adist, int itype, int ileft1, int irlight1, int ileft2, int irlight2, int size_1, int size_2, double *egeom1, double *egeom2)`
- `SISLTrack * newTrack (SISLSurf *psurf_1, SISLSurf *psurf_2, SISLCurve *pcurve_3d, SISLCurve *pcurve_2d_1, SISLCurve *pcurve_2d_2, int ideg, eimpli, int sing_start, int sing_end, int turned)`
- `SISLObject * newObject (int iobj)`
- `SISLPoint * newPoint (double *ecoef, int idim, int icopy)`
- `SISLPtedge * newPtedge (SISLIntpt *ppt)`
- `SISLSurf * newSurf (int in1, int in2, int ik1, int ik2, double *et1, double *et2, double *ecoef, int ikind, int idim, int icopy)`
- `SISLCurve * copyCurve (SISLCurve *pcurve)`
- `SISLSurf * copySurface (SISLSurf *psurf)`

## 30.2292.1 Macro Definition Documentation

### 30.2292.1.1 #define CONSTRUCT

Definition at line 49 of file construct.c.

## 30.2292.2 Function Documentation

### 30.2292.2.1 SISLCurve\* copyCurve ( SISLCurve \* pcurve )

Definition at line 1847 of file construct.c.

### 30.2292.2.2 SISLIntpt\* copyIntpt ( SISLIntpt \* ppt )

Definition at line 57 of file construct.c.

### 30.2292.2.3 SISLSurf\* copySurface ( SISLSurf \* psurf )

Definition at line 1982 of file construct.c.

### 30.2292.2.4 SISLIntpt\* hp\_copyIntpt ( SISLIntpt \* ppt )

Definition at line 118 of file construct.c.

### 30.2292.2.5 SISLIntpt\* hp\_newIntpt ( int ipar, double \* epar, double adist, int itype, int ileft1, int iright1, int ileft2, int iright2, int size\_1, int size\_2, double \* egeom1, double \* egeom2 )

Definition at line 1123 of file construct.c.

### 30.2292.2.6 SISLbox\* newbox ( int idim )

Definition at line 180 of file construct.c.

### 30.2292.2.7 SISLCurve\* newCurve ( int in, int ik, double \* et, double \* ecoef, int ikind, int idim, int icopy )

Definition at line 268 of file construct.c.

### 30.2292.2.8 SISLdir\* newdir ( int idim )

Definition at line 455 of file construct.c.

30.2292.2.9 **SISLEdge\*** newEdge ( int *iedge* )

Definition at line 509 of file construct.c.

30.2292.2.10 **SISLIntcurve\*** newIntcurve ( int *ipoint*, int *ipar1*, int *ipar2*, double \* *epar1*, double \* *epar2*, int *itype* )

Definition at line 589 of file construct.c.

30.2292.2.11 **SISLIntdat\*** newIntdat ( )

Definition at line 681 of file construct.c.

30.2292.2.12 **SISLIntlist\*** newList ( **SISLIntpt** \* *pfirst*, **SISLIntpt** \* *plast*, int *itype* )

Definition at line 753 of file construct.c.

30.2292.2.13 **SISLIntpt\*** newIntpt ( int *ipar*, double \* *epar*, double *adist* )

Definition at line 823 of file construct.c.

30.2292.2.14 **SISLIntsurf\*** newIntsurf ( **SISLIntlist** \* *pintlist* )

Definition at line 926 of file construct.c.

30.2292.2.15 **SISLObject\*** newObject ( int *iobj* )

Definition at line 1381 of file construct.c.

30.2292.2.16 **SISLPoint\*** newPoint ( double \* *ecoef*, int *idim*, int *icopy* )

Definition at line 1449 of file construct.c.

30.2292.2.17 **SISLPtedge\*** newPtedge ( **SISLIntpt** \* *ppt* )

Definition at line 1541 of file construct.c.

30.2292.2.18 **SISLSurf\*** newSurf ( int *in1*, int *in2*, int *ik1*, int *ik2*, double \* *et1*, double \* *et2*, double \* *ecoef*, int *ikind*, int *idim*, int *icopy* )

Definition at line 1607 of file construct.c.

30.2292.2.19 `SISLTrack* newTrack ( SISLSurf * psurf_1, SISLSurf * psurf_2, SISLCurve * pcurve_3d, SISLCurve * pcurve_2d_1, SISLCurve * pcurve_2d_2, int ideg, eimpli, int sing_start, int sing_end, int turned )`

Definition at line 1297 of file construct.c.

30.2292.2.20 `SISLTrimpar* newTrimpar ( int pt, int par )`

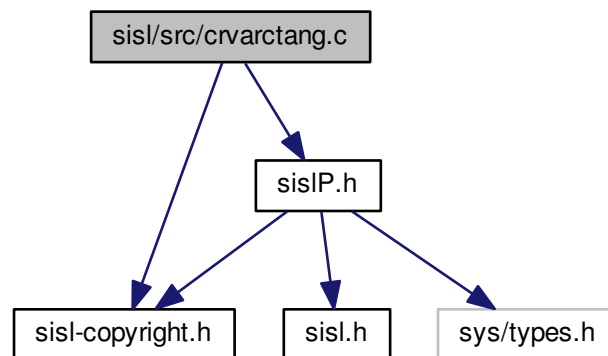
Definition at line 1053 of file construct.c.

## 30.2293 sisl/src/crvarctang.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for crvarctang.c:



### Macros

- `#define CRV_ARC_TANG`

### Functions

- `void crv_arc_tang (SISLCurve *pc1, center, double radius, double aepsge, guess_par, iter_par, int *jstat)`

## 30.2293.1 Macro Definition Documentation

30.2293.1.1 `#define CRV_ARC_TANG`

Definition at line 47 of file crvarctang.c.

## 30.2293.2 Function Documentation

30.2293.2.1 void `crv_arc_tang` ( `SISLCurve * pc1`, `center`, `double radius`, `double aepsge`, `guess_par`, `iter_par`, `int * jstat` )

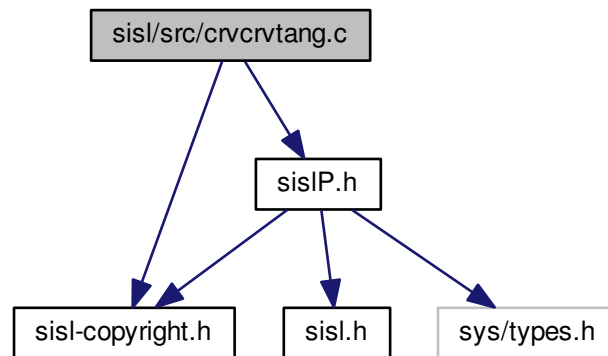
Definition at line 75 of file `crvarctang.c`.

## 30.2294 sisl/src/crvcrvtang.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for `crvcrvtang.c`:



### Macros

- `#define CRV_CRV_TANG`

### Functions

- void `crv_crv_tang` (`SISLCurve *pc1`, `SISLCurve *pc2`, `double aepsge`, `guess_par`, `iter_par`, `int *jstat`)

## 30.2294.1 Macro Definition Documentation

30.2294.1.1 `#define CRV_CRV_TANG`

Definition at line 47 of file `crvcrvtang.c`.

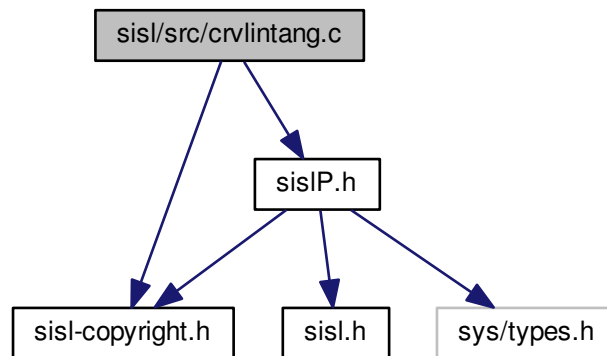
## 30.2294.2 Function Documentation

30.2294.2.1 `void crv_crv_tang ( SISLCurve * pc1, SISLCurve * pc2, double aeprge, guess_par , iter_par , int * jstat )`

Definition at line 74 of file crvcrtang.c.

## 30.2295 sisl/src/crvlintang.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for crvlintang.c:
```



### Macros

- `#define CRV_LIN_TANG`

### Functions

- `void crv_lin_tang (SISLCurve *pc1, point, normal, double ang_tol, double guess_par, double *iter_par, int *jstat)`

## 30.2295.1 Macro Definition Documentation

30.2295.1.1 `#define CRV_LIN_TANG`

Definition at line 49 of file crvlintang.c.

### 30.2295.2 Function Documentation

30.2295.2.1 void `crv_lin_tang` ( `SISLCurve * pc1`, `point`, `normal`, `double ang_tol`, `double guess_par`, `double * iter_par`, `int * jstat` )

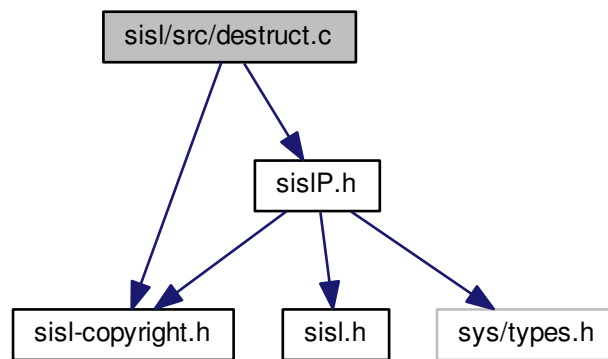
Definition at line 60 of file `crvlintang.c`.

### 30.2296 sisl/src/destruct.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for `destruct.c`:



#### Macros

- `#define DESTRUCT`

#### Functions

- void `freeCurve` (`SISLCurve * pcurve`)
- void `freeEdge` (`SISLEdge * pedge`)
- void `freeIntcrvlist` (`SISLIntcurve ** viclist`, `int icrv`)
- void `freeIntcurve` (`SISLIntcurve * pintc`)
- void `freeIntdat` (`SISLIntdat * pintdat`)
- void `freeIntlist` (`SISLIntlist * plist`)
- void `freeIntpt` (`SISLIntpt * ppt`)
- void `freeIntsurf` (`SISLIntsurf * intsurf`)
- void `freeTrimpar` (`SISLTrimpar * trimpar`)
- void `freeTrack` (`SISLTrack * ppt`)
- void `freeObject` (`SISLObject * pobj`)
- void `freePoint` (`SISLPoint * ppoint`)
- void `freePtedge` (`SISLPtedge * p1`)
- void `freeSurf` (`SISLSurf * psurf`)

## 30.2296.1 Macro Definition Documentation

### 30.2296.1.1 #define DESTRUCT

Definition at line 49 of file destruct.c.

## 30.2296.2 Function Documentation

### 30.2296.2.1 void freeCurve ( SISLCurve \* *pcurve* )

Definition at line 57 of file destruct.c.

### 30.2296.2.2 void freeEdge ( SISLEdge \* *pedge* )

Definition at line 151 of file destruct.c.

### 30.2296.2.3 void freeIntcrvlist ( SISLIntcurve \*\* *viclist*, int *icrv* )

Definition at line 224 of file destruct.c.

### 30.2296.2.4 void freeIntcurve ( SISLIntcurve \* *pintc* )

Definition at line 282 of file destruct.c.

### 30.2296.2.5 void freeIntdat ( SISLIntdat \* *pintdat* )

Definition at line 329 of file destruct.c.

### 30.2296.2.6 void freeIntlist ( SISLIntlist \* *plist* )

Definition at line 400 of file destruct.c.

### 30.2296.2.7 void freeIntpt ( SISLIntpt \* *ppt* )

Definition at line 445 of file destruct.c.

### 30.2296.2.8 void freeIntsurf ( SISLIntsurf \* *intsurf* )

Definition at line 503 of file destruct.c.



30.2296.2.9 void freeObject ( SISLObject \* *pobj* )

Definition at line 649 of file destruct.c.

30.2296.2.10 void freePoint ( SISLPoint \* *ppoint* )

Definition at line 713 of file destruct.c.

30.2296.2.11 void freePtedge ( SISLPtedge \* *p1* )

Definition at line 777 of file destruct.c.

30.2296.2.12 void freeSurf ( SISLSurf \* *psurf* )

Definition at line 821 of file destruct.c.

30.2296.2.13 void freeTrack ( SISLTrack \* *ppt* )

Definition at line 599 of file destruct.c.

30.2296.2.14 void freeTrimpar ( SISLTrimpar \* *trimpar* )

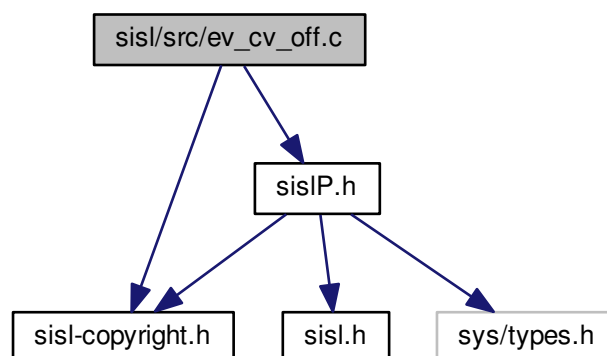
Definition at line 553 of file destruct.c.

## 30.2297 sisl/src/ev\_cv\_off.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for ev\_cv\_off.c:



## Macros

- `#define EV_CV_OFF`

## Functions

- `void ev_cv_off (SISLCurve *pc1, int ider, double ax, int *ileft, double offset, eder, int *jstat)`

### 30.2297.1 Macro Definition Documentation

#### 30.2297.1.1 `#define EV_CV_OFF`

Definition at line 49 of file `ev_cv_off.c`.

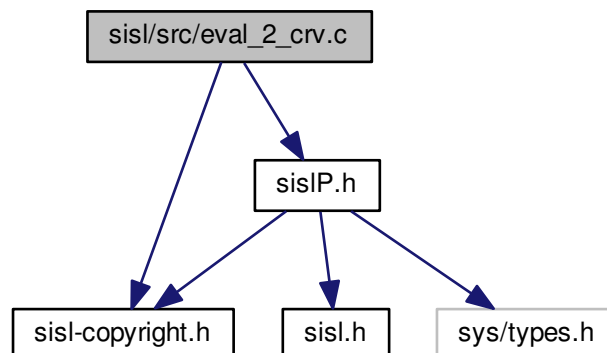
### 30.2297.2 Function Documentation

#### 30.2297.2.1 `void ev_cv_off ( SISLCurve * pc1, int ider, double ax, int * ileft, double offset, eder , int * jstat )`

Definition at line 58 of file `ev_cv_off.c`.

## 30.2298 sisl/src/eval\_2\_crv.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for eval_2_crv.c:
```



## Macros

- `#define EVAL_2_CRV`

## Functions

- void [eval\\_2\\_crv](#) (SISLCurve \*pc1, SISLCurve \*pc2, int ider, epar, int \*ilfs, int \*ilft, eder, int \*jstat)

### 30.2298.1 Macro Definition Documentation

#### 30.2298.1.1 #define EVAL\_2\_CRV

Definition at line 49 of file eval\_2\_crv.c.

### 30.2298.2 Function Documentation

#### 30.2298.2.1 void eval\_2\_crv ( SISLCurve \* pc1, SISLCurve \* pc2, int ider, epar , int \* ilfs, int \* ilft, eder , int \* jstat )

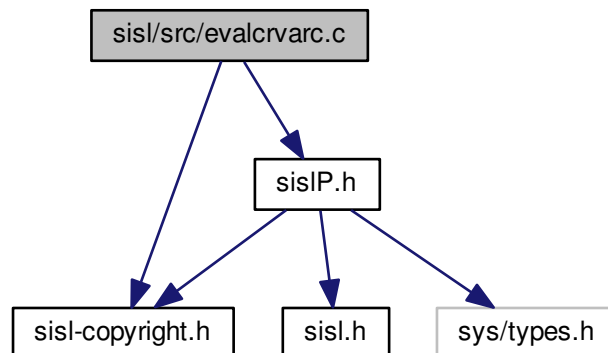
Definition at line 58 of file eval\_2\_crv.c.

## 30.2299 sisl/src/evalcrvarc.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for evalcrvarc.c:



## Macros

- #define [EVAL\\_CRV\\_ARC](#)

## Functions

- void [eval\\_crv\\_arc](#) (SISLCurve \*pc1, center, double radius, int ider, epar, int \*ilfs, eder, int \*jstat)

### 30.2299.1 Macro Definition Documentation

#### 30.2299.1.1 #define EVAL\_CRV\_ARC

Definition at line 49 of file evalcrvarc.c.

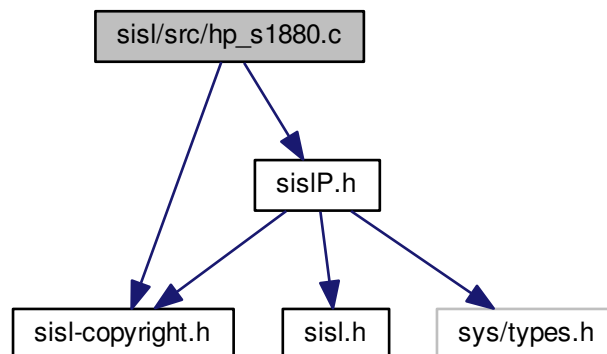
### 30.2299.2 Function Documentation

#### 30.2299.2.1 void eval\_crv\_arc ( SISLCurve \* pc1, center , double radius, int ider, epar , int \* ilfs, eder , int \* jstat )

Definition at line 58 of file evalcrvarc.c.

## 30.2300 sisl/src/hp\_s1880.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for hp_s1880.c:
```



### Macros

- #define [HP\\_S1880](#)

### Functions

- void [hp\\_s1880](#) (SISLObject \*po1, SISLObject \*po2, int ideg, int ipar1, int ipar2, SISLIntdat \*pintdat, int \*jpar, double \*\*gpar1, double \*\*gpar2, int \*\*pretop, int \*jcrv, SISLIntcurve \*\*\*wcrv, int \*jsurf, SISLIntsurf \*\*\*wsurf, int \*jstat)

### 30.2300.1 Macro Definition Documentation

#### 30.2300.1.1 #define HP\_S1880

Definition at line 49 of file hp\_s1880.c.

### 30.2300.2 Function Documentation

#### 30.2300.2.1 void hp\_s1880 ( SISLObject \* po1, SISLObject \* po2, int ideg, int ipar1, int ipar2, SISLIntdat \* pintdat, int \* jpar, double \*\* gpar1, double \*\* gpar2, int \*\* pretop, int \* jcrv, SISLIntcurve \*\*\* wcrv, int \* jsurf, SISLIntsurf \*\*\* wsurf, int \* jstat )

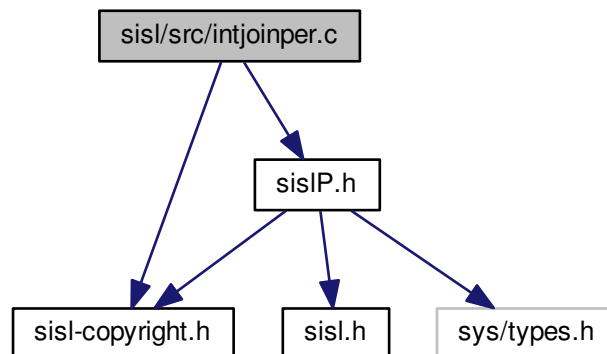
Definition at line 63 of file hp\_s1880.c.

## 30.2301 sisl/src/intjoinper.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for intjoinper.c:



### Macros

- #define [INT\\_JOIN\\_PER](#)

### Functions

- void [int\\_join\\_per](#) (SISLIntdat \*\*pintdat, SISLObject \*po1, SISLObject \*po2, eimpli, int ideg, double aepsge, int \*jstat)

### 30.2301.1 Macro Definition Documentation

#### 30.2301.1.1 #define INT\_JOIN\_PER

Definition at line 49 of file intjoinper.c.

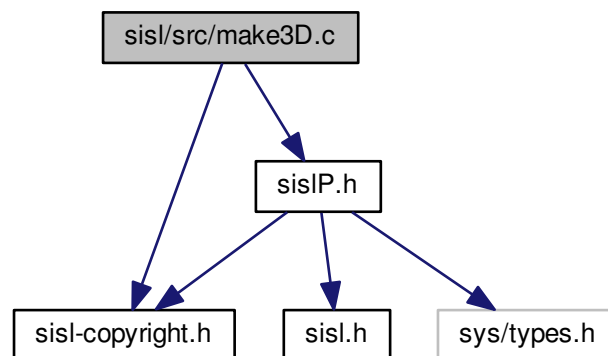
### 30.2301.2 Function Documentation

#### 30.2301.2.1 void int\_join\_per ( SISLIntdat \*\* pintdat, SISLObject \* po1, SISLObject \* po2, eimpli, int ideg, double aepsge, int \* jstat )

Definition at line 65 of file intjoinper.c.

## 30.2302 sisl/src/make3D.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for make3D.c:
```



### Macros

- #define [MAKE3D](#)

### Functions

- void [make3D](#) (SISLSurf \*ps, SISLSurf \*\*rsnew, int \*jstat)

### 30.2302.1 Macro Definition Documentation

#### 30.2302.1.1 #define MAKE3D

Definition at line 49 of file make3D.c.

### 30.2302.2 Function Documentation

#### 30.2302.2.1 void make3D ( SISLSurf \* ps, SISLSurf \*\* rsnew, int \* jstat )

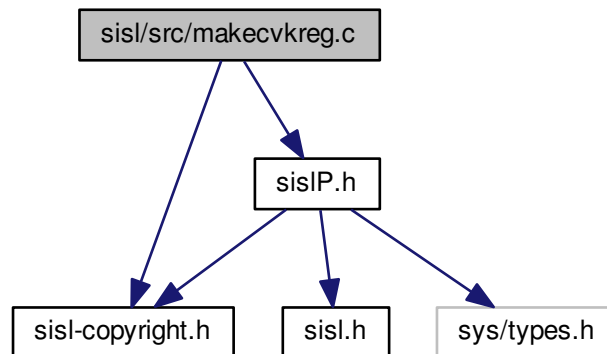
Definition at line 59 of file make3D.c.

## 30.2303 sisl/src/makecvkreg.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for makecvkreg.c:



### Macros

- #define [MAKE\\_CV\\_KREG](#)

### Functions

- void [make\\_cv\\_kreg](#) (SISLCurve \*pc, SISLCurve \*\*rcnew, int \*jstat)

### 30.2303.1 Macro Definition Documentation

#### 30.2303.1.1 #define MAKE\_CV\_KREG

Definition at line 49 of file makecvkreg.c.

### 30.2303.2 Function Documentation

#### 30.2303.2.1 void make\_cv\_kreg ( SISLCurve \* pc, SISLCurve \*\* rcnew, int \* jstat )

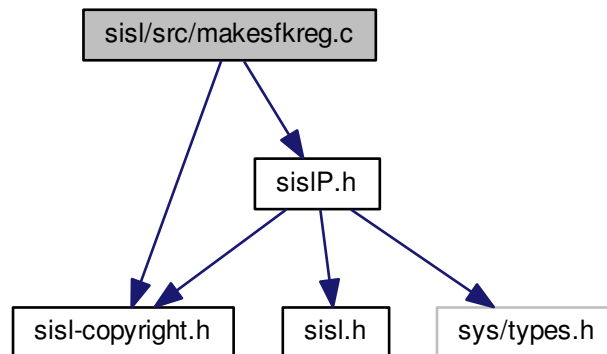
Definition at line 59 of file makecvkreg.c.

## 30.2304 sisl/src/makesfkreg.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for makesfkreg.c:



### Macros

- #define [MAKE\\_SF\\_KREG](#)

### Functions

- void [make\\_sf\\_kreg](#) (SISLSurf \*ps, SISLSurf \*\*rsnew, int \*jstat)



### 30.2304.1 Macro Definition Documentation

#### 30.2304.1.1 #define MAKE\_SF\_KREG

Definition at line 49 of file makesfkreg.c.

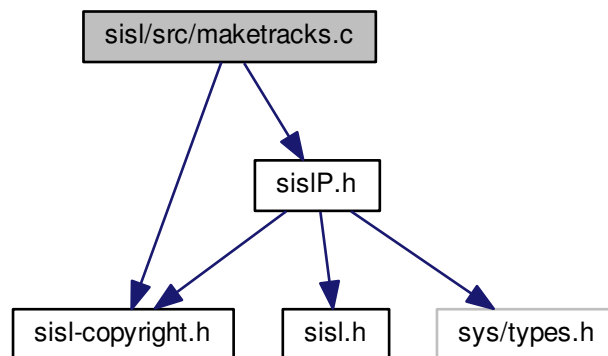
### 30.2304.2 Function Documentation

#### 30.2304.2.1 void make\_sf\_kreg ( SISLSurf \* ps, SISLSurf \*\* rsnew, int \* jstat )

Definition at line 58 of file makesfkreg.c.

## 30.2305 sisl/src/maketracks.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for maketracks.c:
```



### Macros

- #define [MAKE\\_TRACKS](#)

### Functions

- void [make\\_tracks](#) (SISLObject \*po1, SISLObject \*po2, int ideg, eimpli, int icrv, SISLIntlist \*\*vlist, int \*jtrack, SISLTrack \*\*\*wcrv, double aepsge, int \*jstat)

### 30.2305.1 Macro Definition Documentation

#### 30.2305.1.1 #define MAKE\_TRACKS

Definition at line 49 of file maketracks.c.

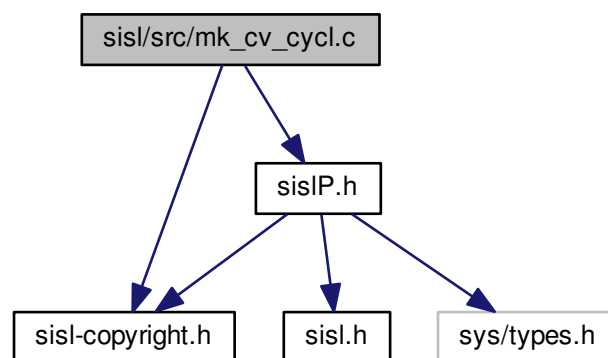
### 30.2305.2 Function Documentation

#### 30.2305.2.1 void make\_tracks ( SISLObject \* po1, SISLObject \* po2, int ideg, eimpli , int icrv, SISLIntlist \*\* vlist, int \* jtrack, SISLTrack \*\*\* wcrv, double aeptsge, int \* jstat )

Definition at line 57 of file maketracks.c.

## 30.2306 sisl/src/mk\_cv\_cycl.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for mk_cv_cycl.c:
```



### Macros

- #define [MAKE\\_CV\\_CYCLIC](#)

### Functions

- void [make\\_cv\\_cyclic](#) (SISLCurve \*pcurve, int icont, int \*jstat)

## 30.2306.1 Macro Definition Documentation

### 30.2306.1.1 #define MAKE\_CV\_CYCLIC

Definition at line 49 of file mk\_cv\_cycl.c.

## 30.2306.2 Function Documentation

### 30.2306.2.1 void make\_cv\_cyclic ( SISLCurve \* pcurve, int icont, int \* jstat )

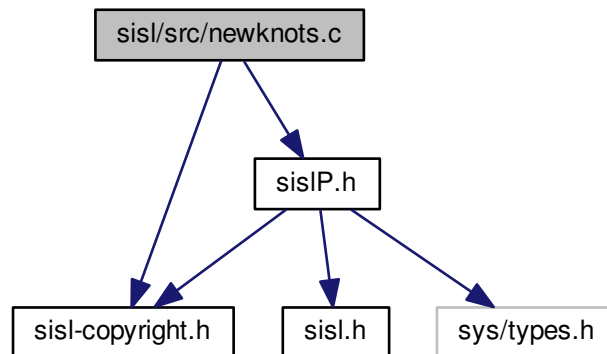
Definition at line 57 of file mk\_cv\_cycl.c.

## 30.2307 sisl/src/newknots.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for newknots.c:



## Macros

- #define [NEWKNOTS](#)

## Functions

- void [newknots](#) (et, int in, int ik, epar, int inpar, double aeaps, double \*\*ginsert, int \*jinsert, int \*jstat)

### 30.2307.1 Macro Definition Documentation

#### 30.2307.1.1 #define NEWKNOTS

Definition at line 49 of file newknots.c.

### 30.2307.2 Function Documentation

#### 30.2307.2.1 void newknots ( et , int in, int ik, epar , int inpar, double aeps, double \*\* ginsert, int \* jinsert, int \* jstat )

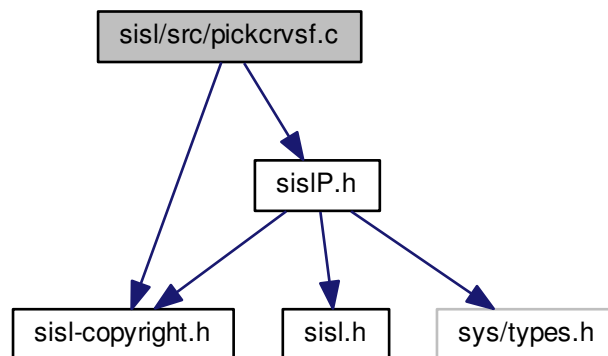
Definition at line 59 of file newknots.c.

## 30.2308 sisl/src/pickcrvsf.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for pickcrvsf.c:



### Macros

- #define [PICK\\_CRV\\_SF](#)

### Functions

- void [pick\\_crv\\_sf](#) ([SISLObject](#) \*po1, [SISLObject](#) \*po2, int ipar, [SISLIntpt](#) \*pt1, [SISLIntpt](#) \*pt2, [SISLCurve](#) \*\*rcrv, int \*jstat)

## 30.2308.1 Macro Definition Documentation

### 30.2308.1.1 #define PICK\_CRV\_SF

Definition at line 49 of file pickcrvsf.c.

## 30.2308.2 Function Documentation

### 30.2308.2.1 void pick\_crv\_sf ( SISLObject \* po1, SISLObject \* po2, int ipar, SISLIntpt \* pt1, SISLIntpt \* pt2, SISLCurve \*\* rcrv, int \* jstat )

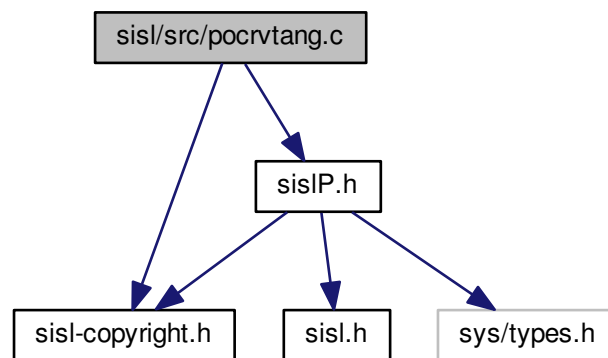
Definition at line 59 of file pickcrvsf.c.

## 30.2309 sisl/src/pocrvtang.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for pocrvtang.c:



## Macros

- #define [PO\\_CRV\\_TANG](#)

## Functions

- void [po\\_crv\\_tang](#) (SISLCurve \*pcurve, point, double ang\_tol, double guess\_par, double \*iter\_par, int \*jstat)

### 30.2309.1 Macro Definition Documentation

#### 30.2309.1.1 #define PO\_CRV\_TANG

Definition at line 49 of file pocrvtang.c.

### 30.2309.2 Function Documentation

#### 30.2309.2.1 void po\_crv\_tang ( SISLCurve \* pcurve, point , double ang\_tol, double guess\_par, double \* iter\_par, int \* jstat )

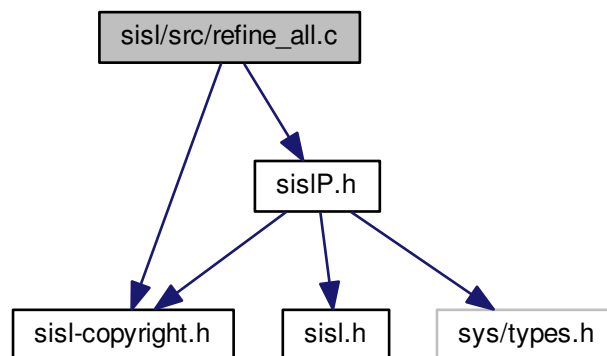
Definition at line 58 of file pocrvtang.c.

## 30.2310 sisl/src/refine\_all.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for refine\_all.c:



### Macros

- #define [REFINE\\_ALL](#)

### Functions

- void [refine\\_all](#) (SISLIntdat \*\*pintdat, SISLObject \*po1, SISLObject \*po2, eimpli, int ideg, double aepsge, int \*jstat)

### 30.2310.1 Macro Definition Documentation

#### 30.2310.1.1 #define REFINE\_ALL

Definition at line 49 of file refine\_all.c.

### 30.2310.2 Function Documentation

#### 30.2310.2.1 void refine\_all ( SISLIntdat \*\* pintdat, SISLObject \* po1, SISLObject \* po2, eimpli , int ideg, double aepsge, int \* jstat )

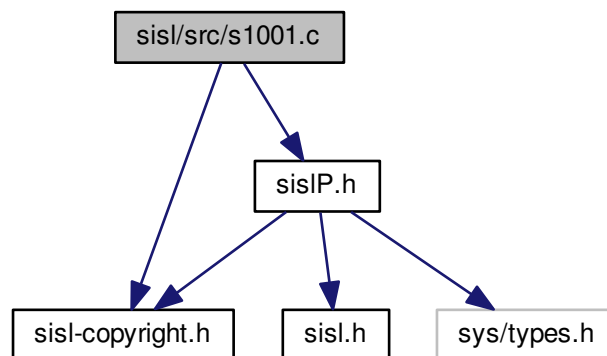
Definition at line 65 of file refine\_all.c.

## 30.2311 sisl/src/s1001.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1001.c:



### Macros

- #define [S1001](#)

### Functions

- void [s1001](#) (SISLSurf \*ps, double min1, double min2, double max1, double max2, SISLSurf \*\*rsnew, int \*jstat)

### 30.2311.1 Macro Definition Documentation

#### 30.2311.1.1 #define S1001

Definition at line 49 of file s1001.c.

### 30.2311.2 Function Documentation

#### 30.2311.2.1 void s1001 ( SISLSurf \* ps, double min1, double min2, double max1, double max2, SISLSurf \*\* rsnew, int \* jstat )

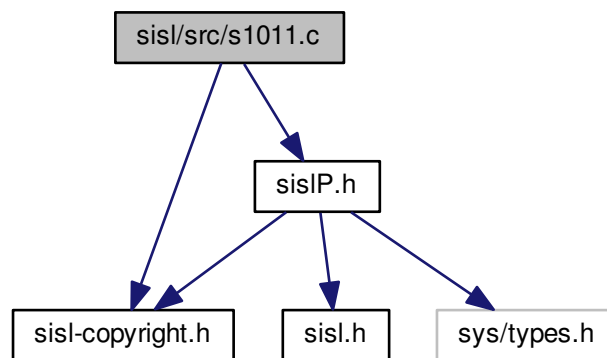
Definition at line 61 of file s1001.c.

## 30.2312 sisl/src/s1011.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1011.c:



### Macros

- #define [S1011](#)

### Functions

- void [s1011](#) (start\_pos, top\_pos, end\_pos, [double shape](#), int dim, [SISLCurve \\*\\*arc\\_seg](#), int \*stat)



### 30.2312.1 Macro Definition Documentation

#### 30.2312.1.1 #define S1011

Definition at line 49 of file s1011.c.

### 30.2312.2 Function Documentation

#### 30.2312.2.1 void s1011 ( start\_pos , top\_pos , end\_pos , double shape, int dim, SISLCurve \*\* arc\_seg, int \* stat )

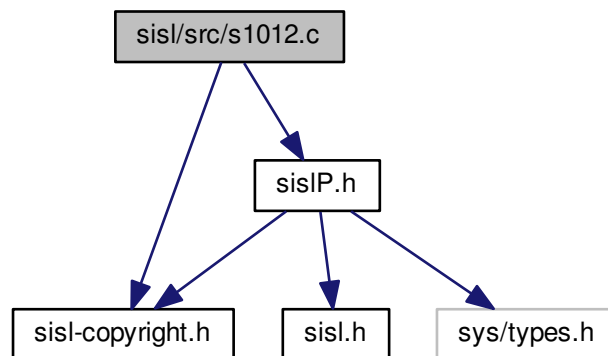
Definition at line 58 of file s1011.c.

## 30.2313 sisl/src/s1012.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1012.c:



### Macros

- #define [S1012](#)

### Functions

- void [s1012](#) (start\_pos, axis\_pos, axis\_dir, double frequency, int numb\_quad, int counter\_clock, [SISLCurve](#) \*\*helix, int \*stat)

### 30.2313.1 Macro Definition Documentation

#### 30.2313.1.1 #define S1012

Definition at line 49 of file s1012.c.

### 30.2313.2 Function Documentation

#### 30.2313.2.1 void s1012 ( start\_pos , axis\_pos , axis\_dir , double frequency, int numb\_quad, int counter\_clock, SISLCurve \*\* helix, int \* stat )

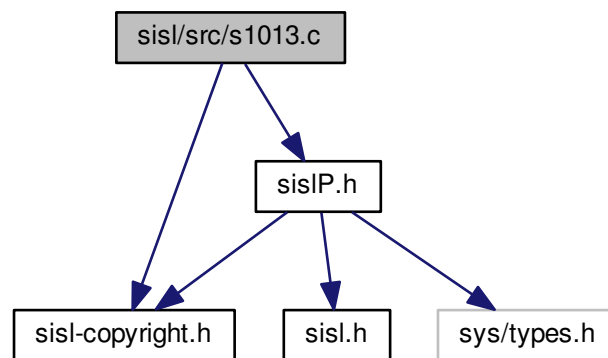
Definition at line 59 of file s1012.c.

## 30.2314 sisl/src/s1013.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1013.c:



### Macros

- #define [S1013](#)

### Functions

- void [s1013](#) ([SISLCurve](#) \*pcurve, [double](#) ang, [double](#) ang\_tol, [double](#) guess\_par, [double](#) \*iter\_par, int \*jstat)

### 30.2314.1 Macro Definition Documentation

#### 30.2314.1.1 #define S1013

Definition at line 49 of file s1013.c.

### 30.2314.2 Function Documentation

#### 30.2314.2.1 void s1013 ( SISLCurve \* pcurve, double ang, double ang\_tol, double guess\_par, double \* iter\_par, int \* jstat )

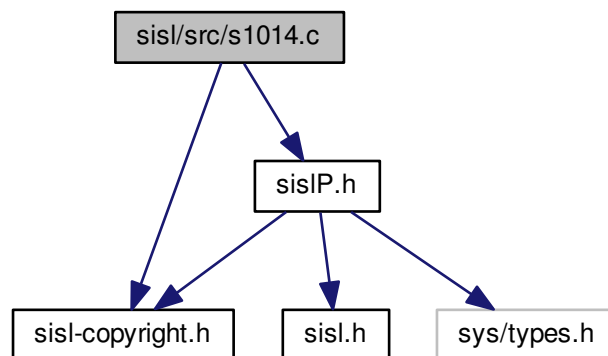
Definition at line 58 of file s1013.c.

## 30.2315 sisl/src/s1014.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1014.c:



### Macros

- #define [S1014](#)

### Functions

- void [s1014](#) (SISLCurve \*pc1, circ\_cen, double circ\_rad, double aepsge, eps1, eps2, double aradius, double \*parpt1, double \*parpt2, center, int \*jstat)

### 30.2315.1 Macro Definition Documentation

#### 30.2315.1.1 #define S1014

Definition at line 47 of file s1014.c.

### 30.2315.2 Function Documentation

30.2315.2.1 void s1014 ( SISLCurve \* pc1, circ\_cen , double circ\_rad, double aepsge, eps1 , eps2 , double aradius, double \* parpt1, double \* parpt2, center , int \* jstat )

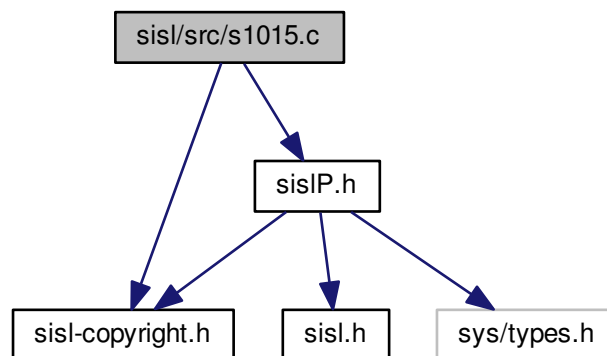
Definition at line 74 of file s1014.c.

## 30.2316 sisl/src/s1015.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1015.c:



### Macros

- #define [S1015](#)

### Functions

- void [s1015](#) (SISLCurve \*pc1, SISLCurve \*pc2, double aepsge, eps1, eps2, double aradius, double \*parpt1, double \*parpt2, center, int \*jstat)

### 30.2316.1 Macro Definition Documentation

#### 30.2316.1.1 #define S1015

Definition at line 47 of file s1015.c.

### 30.2316.2 Function Documentation

#### 30.2316.2.1 void s1015 ( SISLCurve \* pc1, SISLCurve \* pc2, double aepsge, eps1 , eps2 , double aradius, double \* parpt1, double \* parpt2, center , int \* jstat )

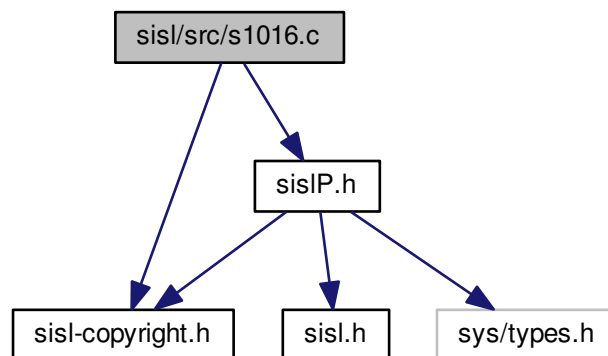
Definition at line 74 of file s1015.c.

## 30.2317 sisl/src/s1016.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1016.c:



### Macros

- #define [S1016](#)

### Functions

- void [s1016](#) (SISLCurve \*pc1, point, [normal](#), [double aepsge](#), eps1, eps2, [double aradius](#), [double](#) \*parpt1, [double](#) \*parpt2, center, int \*jstat)

### 30.2317.1 Macro Definition Documentation

#### 30.2317.1.1 #define S1016

Definition at line 47 of file s1016.c.

### 30.2317.2 Function Documentation

#### 30.2317.2.1 void s1016 ( SISLCurve \* pc1, point , normal , double aepsge, eps1 , eps2 , double aradius, double \* parpt1, double \* parpt2, center , int \* jstat )

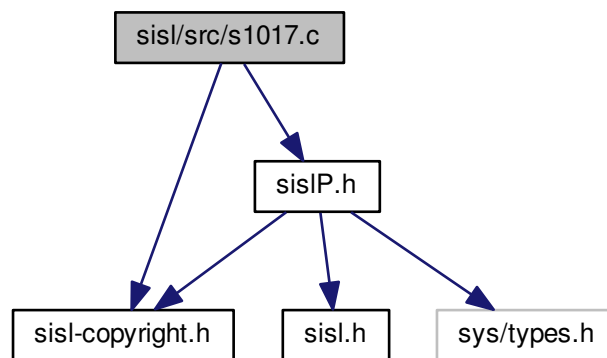
Definition at line 74 of file s1016.c.

## 30.2318 sisl/src/s1017.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1017.c:



### Macros

- #define [S1017](#)

### Functions

- void [s1017](#) (SISLCurve \*pc, SISLCurve \*\*rc, double apar, int \*jstat)

## 30.2318.1 Macro Definition Documentation

### 30.2318.1.1 #define S1017

Definition at line 49 of file s1017.c.

## 30.2318.2 Function Documentation

### 30.2318.2.1 void s1017 ( SISLCurve \* pc, SISLCurve \*\* rc, double apar, int \* jstat )

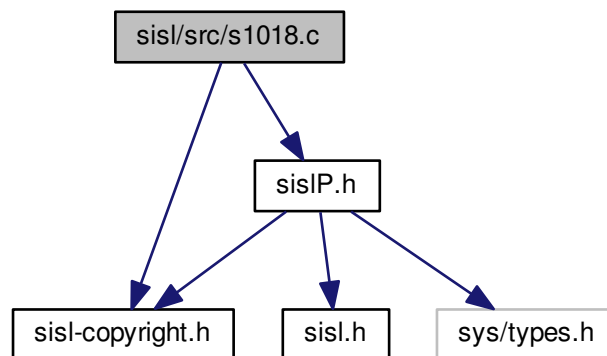
Definition at line 58 of file s1017.c.

## 30.2319 sisl/src/s1018.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1018.c:



## Macros

- #define [S1018](#)

## Functions

- void [s1018](#) (SISLCurve \*pc, epar, int inpar, SISLCurve \*\*rcnew, int \*jstat)

### 30.2319.1 Macro Definition Documentation

#### 30.2319.1.1 #define S1018

Definition at line 49 of file s1018.c.

### 30.2319.2 Function Documentation

#### 30.2319.2.1 void s1018 ( SISLCurve \* pc, epar , int inpar, SISLCurve \*\* rcnew, int \* jstat )

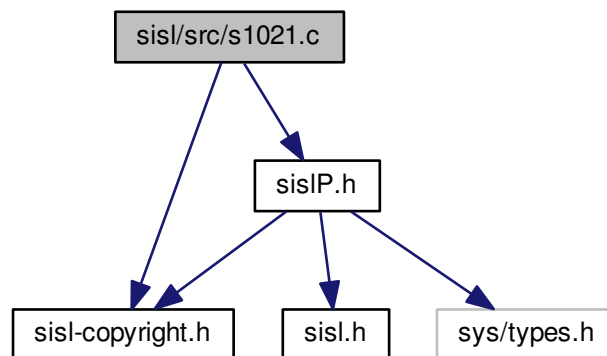
Definition at line 58 of file s1018.c.

## 30.2320 sisl/src/s1021.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1021.c:



### Macros

- #define [S1021](#)

### Functions

- void [s1021](#) (bottom\_pos, bottom\_axis, double ellipse\_ratio, axis\_dir, double height, SISLSurf \*\*cyl, int \*stat)



### 30.2320.1 Macro Definition Documentation

#### 30.2320.1.1 #define S1021

Definition at line 49 of file s1021.c.

### 30.2320.2 Function Documentation

#### 30.2320.2.1 void s1021 ( bottom\_pos , bottom\_axis , double ellipse\_ratio, axis\_dir , double height, SISLSurf \*\* cyl, int \* stat )

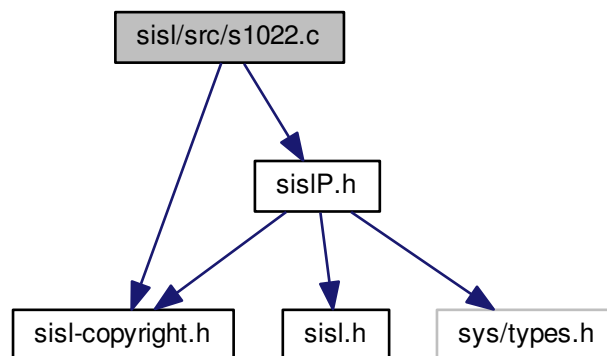
Definition at line 58 of file s1021.c.

## 30.2321 sisl/src/s1022.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1022.c:



### Macros

- #define [S1022](#)

### Functions

- void [s1022](#) (bottom\_pos, bottom\_axis, double ellipse\_ratio, axis\_dir, double cone\_angle, double height, SISLSurf \*\*cone, int \*stat)

### 30.2321.1 Macro Definition Documentation

#### 30.2321.1.1 #define S1022

Definition at line 49 of file s1022.c.

### 30.2321.2 Function Documentation

#### 30.2321.2.1 void s1022 ( bottom\_pos , bottom\_axis , double ellipse\_ratio , axis\_dir , double cone\_angle , double height , SISLSurf \*\* cone , int \* stat )

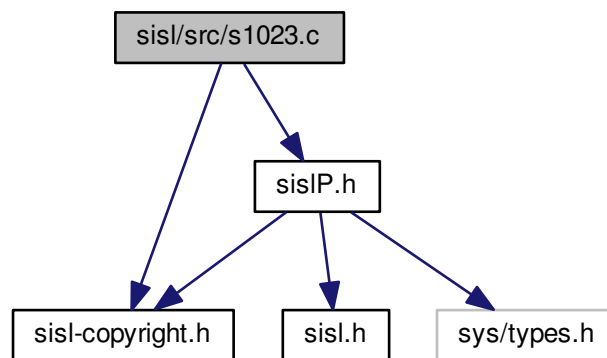
Definition at line 59 of file s1022.c.

## 30.2322 sisl/src/s1023.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1023.c:



### Macros

- #define [S1023](#)

### Functions

- void [s1023](#) (center, axis, equator, int latitude, int longitude, [SISLSurf](#) \*\*sphere, int \*stat)

### 30.2322.1 Macro Definition Documentation

#### 30.2322.1.1 #define S1023

Definition at line 49 of file s1023.c.

### 30.2322.2 Function Documentation

#### 30.2322.2.1 void s1023 ( center , axis , equator , int latitude, int longitude, SISLSurf \*\* sphere, int \* stat )

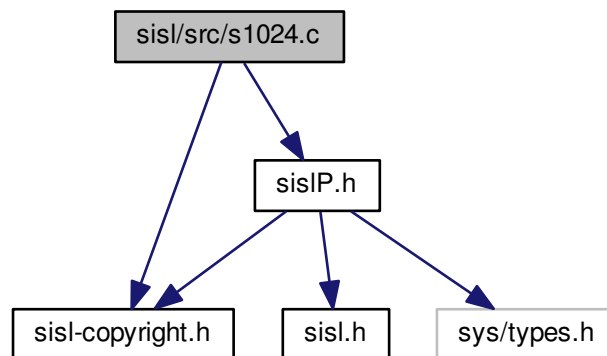
Definition at line 58 of file s1023.c.

## 30.2323 sisl/src/s1024.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1024.c:



### Macros

- #define [S1024](#)

### Functions

- void [s1024](#) (center, axis, equator, [double](#) minor\_radius, int start\_minor, int end\_minor, int numb\_major, [SISL](#)LSurf \*\*torus, int \*stat)

### 30.2323.1 Macro Definition Documentation

#### 30.2323.1.1 #define S1024

Definition at line 49 of file s1024.c.

### 30.2323.2 Function Documentation

#### 30.2323.2.1 void s1024 ( center , axis , equator , double minor\_radius, int start\_minor, int end\_minor, int numb\_major, SISLSurf \*\* torus, int \* stat )

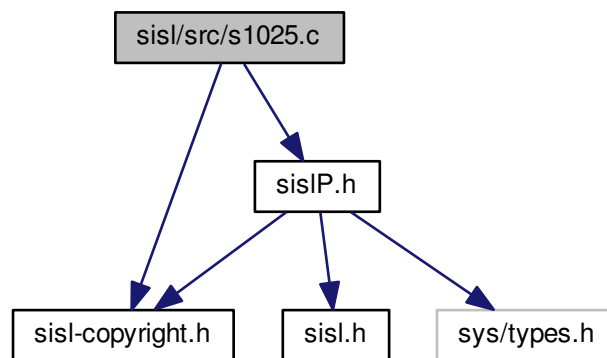
Definition at line 59 of file s1024.c.

## 30.2324 sisl/src/s1025.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1025.c:



### Macros

- #define [S1025](#)

### Functions

- void [s1025](#) (SISLSurf \*ps, epar1, int inpar1, epar2, int inpar2, SISLSurf \*\*rsnew, int \*jstat)

### 30.2324.1 Macro Definition Documentation

#### 30.2324.1.1 #define S1025

Definition at line 49 of file s1025.c.

### 30.2324.2 Function Documentation

#### 30.2324.2.1 void s1025 ( SISLSurf \* ps, epar1 , int inpar1, epar2 , int inpar2, SISLSurf \*\* rsnew, int \* jstat )

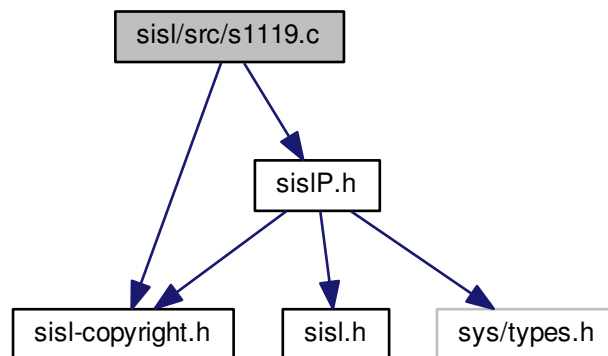
Definition at line 60 of file s1025.c.

## 30.2325 sisl/src/s1119.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1119.c:



### Macros

- #define [S1119](#)

### Functions

- void [s1119](#) (double \*coef, double \*et1, double \*et2, int ik1, int in1, int ik2, int in2, int \*jsimple, int \*jind1, int \*jind2, int \*jstat)

### 30.2325.1 Macro Definition Documentation

#### 30.2325.1.1 #define S1119

Definition at line 49 of file s1119.c.

### 30.2325.2 Function Documentation

#### 30.2325.2.1 void s1119 ( double \* *ecoef*, double \* *et1*, double \* *et2*, int *ik1*, int *in1*, int *ik2*, int *in2*, int \* *jsimple*, int \* *jind1*, int \* *jind2*, int \* *jstat* )

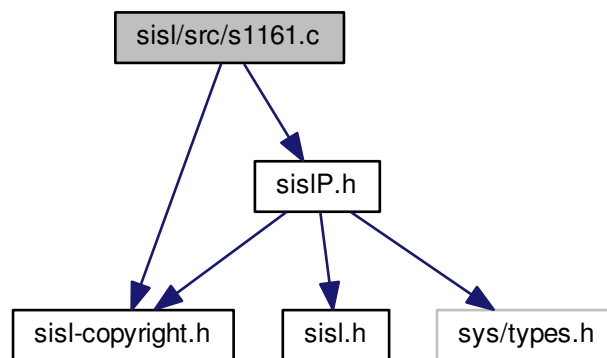
Definition at line 58 of file s1119.c.

### 30.2326 sisl/src/s1161.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1161.c:



#### Macros

- #define [S1161](#)

#### Functions

- void [s1161](#) (SISLObject \*po1, double \*cmax, double aepsge, SISLIntdat \*\*pintdat, int \*jstat)

### 30.2326.1 Macro Definition Documentation

#### 30.2326.1.1 #define S1161

Definition at line 49 of file s1161.c.

### 30.2326.2 Function Documentation

#### 30.2326.2.1 void s1161 ( SISLObject \* po1, double \* cmax, double aepsge, SISLIntdat \*\* pintdat, int \* jstat )

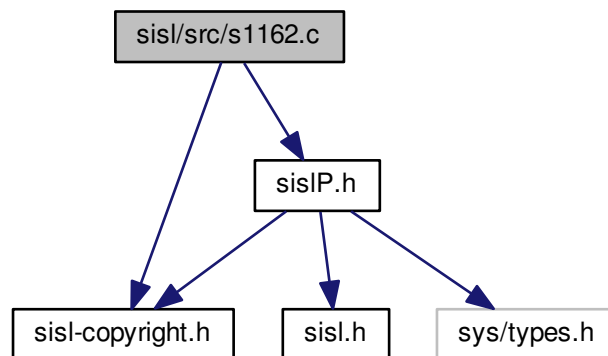
Definition at line 57 of file s1161.c.

## 30.2327 sisl/src/s1162.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1162.c:



### Macros

- #define [S1162](#)

### Functions

- void [s1162](#) (SISLObject \*po1, double \*cmax, double aepsge, SISLIntdat \*\*pintdat, vedge, int ilevel, int inum, int \*jstat)

### 30.2327.1 Macro Definition Documentation

#### 30.2327.1.1 #define S1162

Definition at line 49 of file s1162.c.

### 30.2327.2 Function Documentation

#### 30.2327.2.1 void s1162 ( SISLObject \* po1, double \* cmax, double aepsge, SISLIntdat \*\* pintdat, vedge , int ilevel, int inum, int \* jstat )

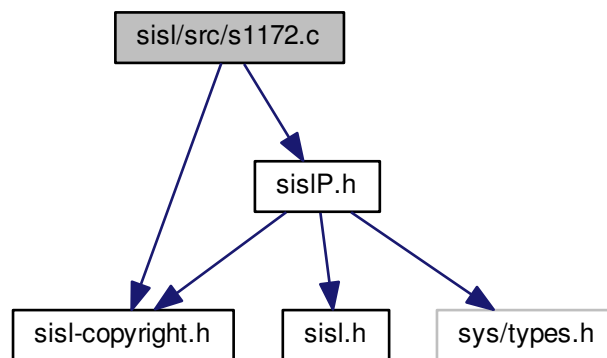
Definition at line 85 of file s1162.c.

### 30.2328 sisl/src/s1172.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1172.c:



#### Macros

- #define [S1172](#)

#### Functions

- void [s1172](#) ([SISLCurve](#) \*pcurve, [double](#) astart, [double](#) aend, [double](#) anext, [double](#) \*cpos, int \*jstat)



## 30.2328.1 Macro Definition Documentation

### 30.2328.1.1 #define S1172

Definition at line 49 of file s1172.c.

## 30.2328.2 Function Documentation

### 30.2328.2.1 void s1172 ( SISLCurve \* pcurve, double astart, double aend, double anext, double \* cpos, int \* jstat )

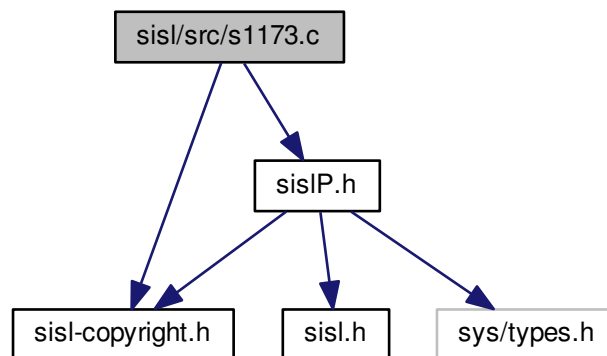
Definition at line 70 of file s1172.c.

## 30.2329 sisl/src/s1173.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1173.c:



## Macros

- #define [S1173](#)

## Functions

- void [s1173](#) (SISLPoint \*ppoint, SISLSurf \*psurf, double aepsge, estart, eend, enext, gpos, int \*jstat)

## 30.2329.1 Macro Definition Documentation

### 30.2329.1.1 #define S1173

Definition at line 49 of file s1173.c.

## 30.2329.2 Function Documentation

### 30.2329.2.1 void s1173 ( SISLPoint \* ppoint, SISLSurf \* psurf, double aepsge, estart , eend , enext , gpos , int \* jstat )

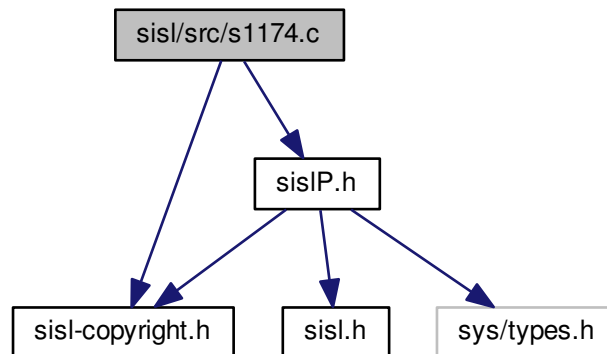
Definition at line 73 of file s1173.c.

## 30.2330 sisl/src/s1174.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1174.c:



## Macros

- #define [S1174](#)

## Functions

- void [s1174](#) (SISLSurf \*psurf, estart, eend, enext, gpos, int \*jstat)

## 30.2330.1 Macro Definition Documentation

### 30.2330.1.1 #define S1174

Definition at line 49 of file s1174.c.

## 30.2330.2 Function Documentation

### 30.2330.2.1 void s1174 ( SISLSurf \* *psurf*, *estart*, *eend*, *enext*, *gpos*, int \* *jstat* )

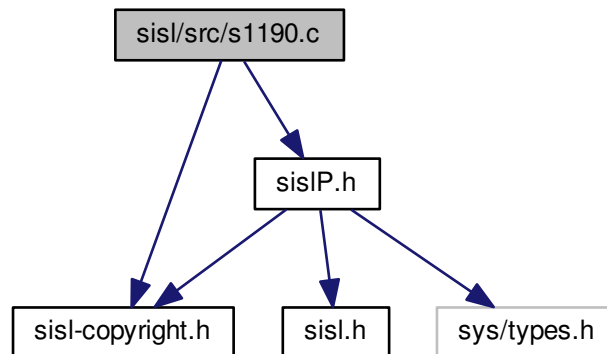
Definition at line 70 of file s1174.c.

## 30.2331 sisl/src/s1190.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1190.c:



## Macros

- #define [S1190](#)

## Functions

- void [s1190](#) (SISLObject \**po1*, double \**cmax*, double *ae<sub>ps</sub>ge*, int \**jstat*)

### 30.2331.1 Macro Definition Documentation

#### 30.2331.1.1 #define S1190

Definition at line 49 of file s1190.c.

### 30.2331.2 Function Documentation

#### 30.2331.2.1 void s1190 ( SISLObject \* *po1*, double \* *cmax*, double *aepsge*, int \* *jstat* )

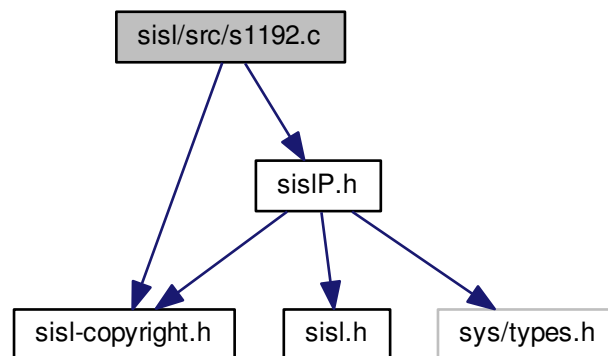
Definition at line 57 of file s1190.c.

## 30.2332 sisl/src/s1192.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1192.c:



### Macros

- #define [S1192](#)

### Functions

- void [s1192](#) (SISLObject \**po*, double *aepsge*, int \**jstat*)

### 30.2332.1 Macro Definition Documentation

#### 30.2332.1.1 #define S1192

Definition at line 49 of file s1192.c.

### 30.2332.2 Function Documentation

#### 30.2332.2.1 void s1192 ( SISLObject \* *po*, double *aepsge*, int \* *jstat* )

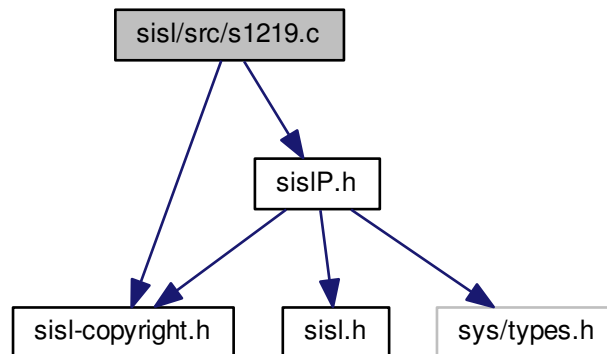
Definition at line 68 of file s1192.c.

## 30.2333 sisl/src/s1219.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1219.c:



### Macros

- #define [S1219](#)

### Functions

- void [s1219](#) (double \**et*, int *ik*, int *in*, int \**ileft*, double *ax*, int \**jstat*)

### 30.2333.1 Macro Definition Documentation

#### 30.2333.1.1 #define S1219

Definition at line 49 of file s1219.c.

### 30.2333.2 Function Documentation

#### 30.2333.2.1 void s1219 ( double \* et, int ik, int in, int \* ileft, double ax, int \* jstat )

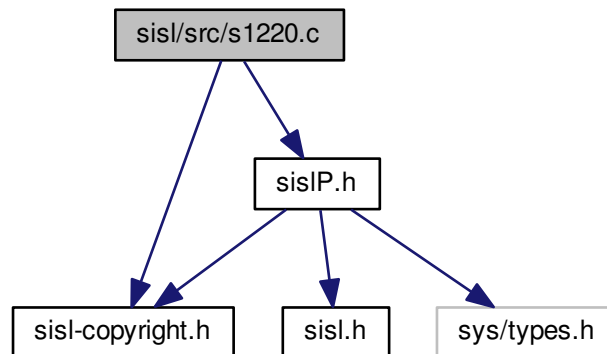
Definition at line 57 of file s1219.c.

## 30.2334 sisl/src/s1220.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1220.c:



### Macros

- #define [S1220](#)

### Functions

- void [s1220](#) (double \*et, int ik, int in, int \*ileft, double ax, int ider, ebder, int \*jstat)

### 30.2334.1 Macro Definition Documentation

#### 30.2334.1.1 #define S1220

Definition at line 49 of file s1220.c.

### 30.2334.2 Function Documentation

#### 30.2334.2.1 void s1220 ( double \* et, int ik, int in, int \* ileft, double ax, int ider, eber , int \* jstat )

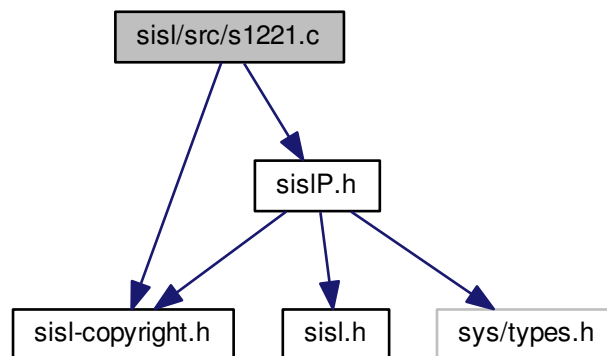
Definition at line 59 of file s1220.c.

## 30.2335 sisl/src/s1221.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1221.c:



### Macros

- #define [S1221](#)

### Functions

- void [s1221](#) ([SISLCurve](#) \*pc1, int ider, [double](#) ax, int \*ileft, eber, int \*jstat)

### 30.2335.1 Macro Definition Documentation

#### 30.2335.1.1 #define S1221

Definition at line 49 of file s1221.c.

### 30.2335.2 Function Documentation

#### 30.2335.2.1 void s1221 ( SISLCurve \* pc1, int ider, double ax, int \* ileft, eder , int \* jstat )

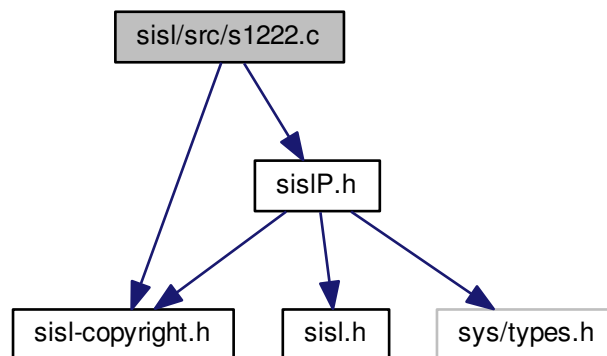
Definition at line 57 of file s1221.c.

## 30.2336 sisl/src/s1222.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1222.c:



### Macros

- #define [S1222](#)

### Functions

- void [s1222](#) (et, int ik, int in, int ibase, [double](#) ax, int ider, ebder, int \*jstat)



### 30.2336.1 Macro Definition Documentation

#### 30.2336.1.1 #define S1222

Definition at line 49 of file s1222.c.

### 30.2336.2 Function Documentation

#### 30.2336.2.1 void s1222 ( et , int ik, int in, int ibase, double ax, int ider, ebder , int \* jstat )

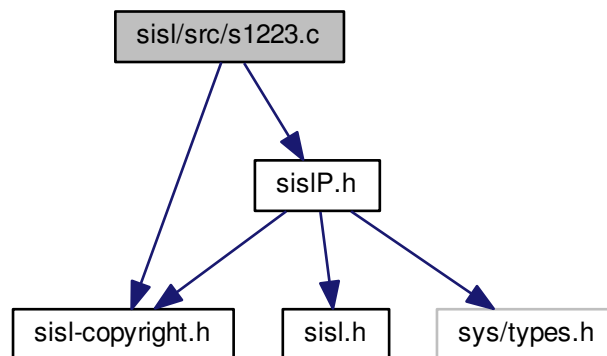
Definition at line 60 of file s1222.c.

## 30.2337 sisl/src/s1223.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1223.c:



### Macros

- #define [S1223](#)

### Functions

- void [s1223](#) (et1, et2, int ik1, int ik2, int in1, int in2, int ibase1, int ibase2, par, int ider1, int ider2, ebder, int \*jstat)

### 30.2337.1 Macro Definition Documentation

#### 30.2337.1.1 #define S1223

Definition at line 49 of file s1223.c.

### 30.2337.2 Function Documentation

#### 30.2337.2.1 void s1223 ( et1 , et2 , int ik1, int ik2, int in1, int in2, int ibase1, int ibase2, par , int ider1, int ider2, ebder , int \* jstat )

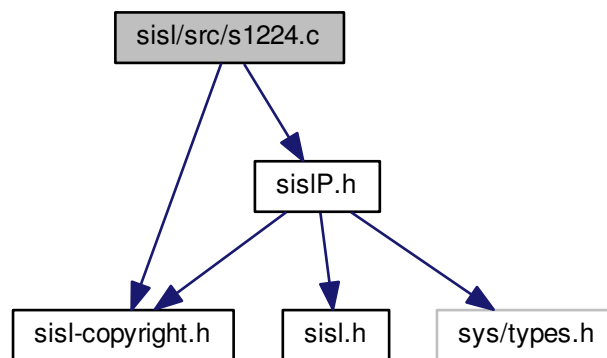
Definition at line 61 of file s1223.c.

## 30.2338 sisl/src/s1224.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1224.c:



### Macros

- #define [S1224](#)

### Functions

- void [s1224](#) (et1, et2, int ik1, int ik2, int in1, int in2, int ibase1, int ibase2, par, int ider, ebder, int \*jstat)

### 30.2338.1 Macro Definition Documentation

#### 30.2338.1.1 #define S1224

Definition at line 49 of file s1224.c.

### 30.2338.2 Function Documentation

#### 30.2338.2.1 void s1224 ( et1 , et2 , int ik1, int ik2, int in1, int in2, int ibase1, int ibase2, par , int nder, ebder , int \* jstat )

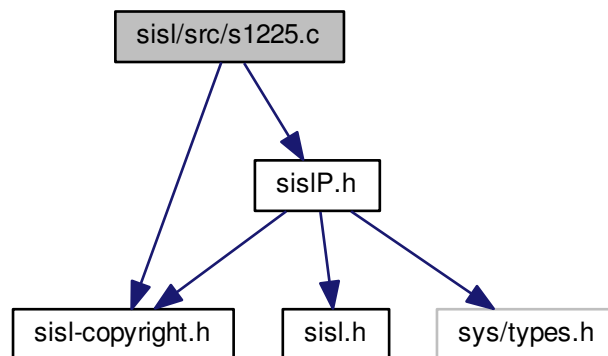
Definition at line 61 of file s1224.c.

## 30.2339 sisl/src/s1225.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1225.c:



### Macros

- #define [S1225](#)

### Functions

- void [s1225](#) ([SISLCurve](#) \*curve, int der, [double](#) parvalue, int \*leftknot, derive, curvature, [double](#) \*radius\_of←\_curvature, int \*jstat)

### 30.2339.1 Macro Definition Documentation

#### 30.2339.1.1 #define S1225

Definition at line 49 of file s1225.c.

### 30.2339.2 Function Documentation

#### 30.2339.2.1 void s1225 ( SISLCurve \* curve, int der, double parvalue, int \* leftknot, derive , curvature , double \* radius\_of\_curvature, int \* jstat )

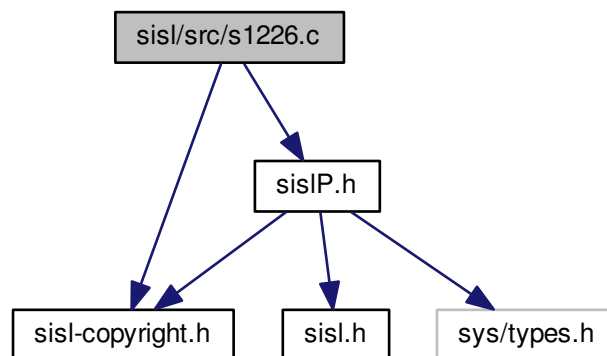
Definition at line 59 of file s1225.c.

## 30.2340 sisl/src/s1226.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1226.c:



### Macros

- #define [S1226](#)

### Functions

- void [s1226](#) (SISLCurve \*curve, int der, double parvalue, int \*leftknot, derive, curvature, double \*radius\_of\_←\_curvature, int \*jstat)

### 30.2340.1 Macro Definition Documentation

#### 30.2340.1.1 #define S1226

Definition at line 49 of file s1226.c.

### 30.2340.2 Function Documentation

#### 30.2340.2.1 void s1226 ( SISLCurve \* curve, int der, double parvalue, int \* leftknot, derive , curvature , double \* radius\_of\_curvature, int \* jstat )

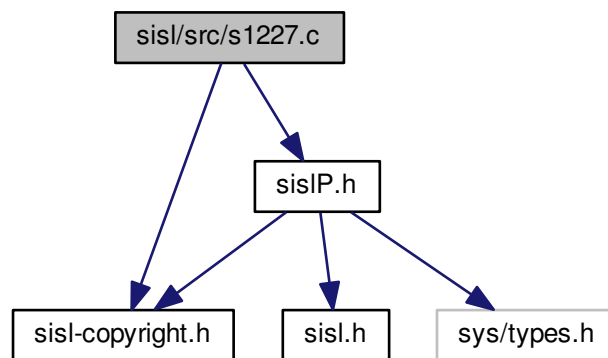
Definition at line 59 of file s1226.c.

## 30.2341 sisl/src/s1227.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1227.c:



### Macros

- #define [S1227](#)

### Functions

- void [s1227](#) (SISLCurve \*pc1, int ider, double ax, int \*ileft, eder, int \*jstat)

### 30.2341.1 Macro Definition Documentation

#### 30.2341.1.1 #define S1227

Definition at line 49 of file s1227.c.

### 30.2341.2 Function Documentation

#### 30.2341.2.1 void s1227 ( SISLCurve \* pc1, int iber, double ax, int \* ileft, iber, int \* jstat )

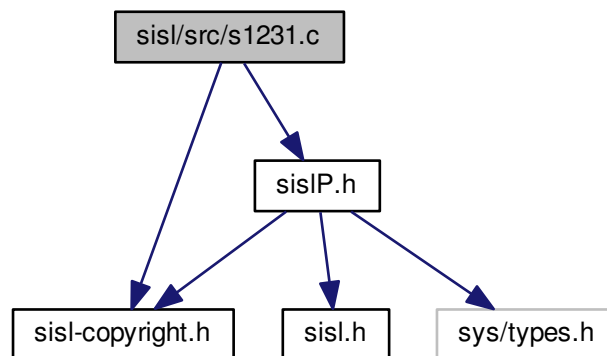
Definition at line 57 of file s1227.c.

## 30.2342 sisl/src/s1231.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1231.c:



### Macros

- #define [S1231](#)

### Functions

- void [s1231](#) (SISLCurve \*pc1, double apar, SISLCurve \*\*rcnew1, SISLCurve \*\*rcnew2, int \*jstat)

### 30.2342.1 Macro Definition Documentation

#### 30.2342.1.1 #define S1231

Definition at line 49 of file s1231.c.

### 30.2342.2 Function Documentation

#### 30.2342.2.1 void s1231 ( SISLCurve \* *pc1*, double *apar*, SISLCurve \*\* *rcnew1*, SISLCurve \*\* *rcnew2*, int \* *jstat* )

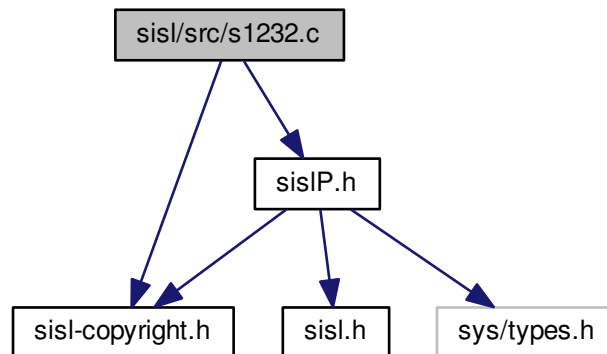
Definition at line 56 of file s1231.c.

## 30.2343 sisl/src/s1232.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1232.c:



### Macros

- #define [S1232](#)

### Functions

- void [s1232](#) (et1, int in, int ik, double afak1, double afak2, et2, int \*jstat)

### 30.2343.1 Macro Definition Documentation

#### 30.2343.1.1 #define S1232

Definition at line 49 of file s1232.c.

### 30.2343.2 Function Documentation

#### 30.2343.2.1 void s1232 ( et1 , int in, int ik, double afak1, double afak2, et2 , int \* jstat )

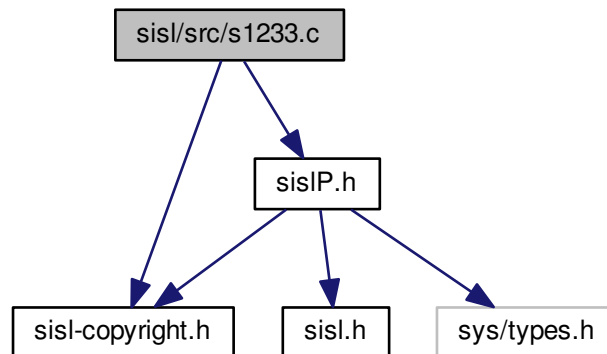
Definition at line 58 of file s1232.c.

## 30.2344 sisl/src/s1233.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1233.c:



### Macros

- #define [S1233](#)

### Functions

- void [s1233](#) ([SISLCurve](#) \*pc, double afak1, double afak2, [SISLCurve](#) \*\*rc, int \*jstat)



### 30.2344.1 Macro Definition Documentation

#### 30.2344.1.1 #define S1233

Definition at line 49 of file s1233.c.

### 30.2344.2 Function Documentation

#### 30.2344.2.1 void s1233 ( SISLCurve \* pc, double afak1, double afak2, SISLCurve \*\* rc, int \* jstat )

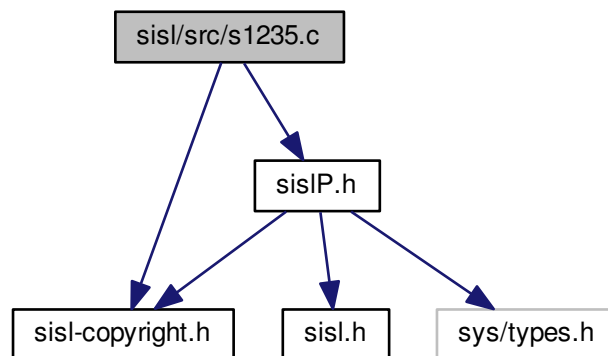
Definition at line 57 of file s1233.c.

## 30.2345 sisl/src/s1235.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1235.c:



### Macros

- #define [S1235](#)

### Functions

- void [s1235](#) (et, int in, int ik, int \*jnbreak, double \*\*gbreak, int \*jstat)

## 30.2345.1 Macro Definition Documentation

### 30.2345.1.1 #define S1235

Definition at line 49 of file s1235.c.

## 30.2345.2 Function Documentation

### 30.2345.2.1 void s1235 ( et , int in, int ik, int \* jnbreak, double \*\* gbreak, int \* jstat )

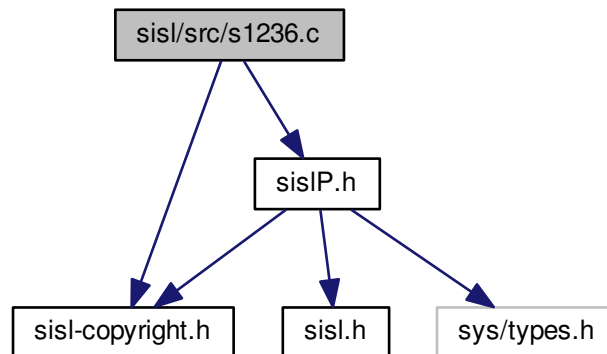
Definition at line 57 of file s1235.c.

## 30.2346 sisl/src/s1236.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1236.c:



## Macros

- #define [S1236](#)

## Functions

- void [s1236](#) (et, int in, int ik, int inpar, epar, int \*jstat)

## 30.2346.1 Macro Definition Documentation

### 30.2346.1.1 #define S1236

Definition at line 49 of file s1236.c.

## 30.2346.2 Function Documentation

### 30.2346.2.1 void s1236 ( et , int in, int ik, int inpar, epar , int \* jstat )

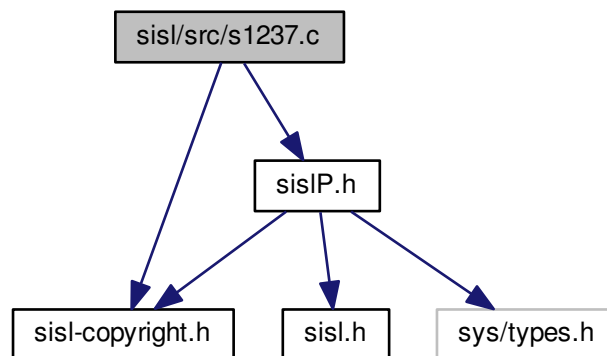
Definition at line 57 of file s1236.c.

## 30.2347 sisl/src/s1237.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1237.c:



## Macros

- #define [S1237](#)

## Functions

- void [s1237](#) ([SISLSurf](#) \*psurf, int inmb1, int inmb2, [double](#) aepscu, int \*jstat)

### 30.2347.1 Macro Definition Documentation

#### 30.2347.1.1 #define S1237

Definition at line 49 of file s1237.c.

### 30.2347.2 Function Documentation

#### 30.2347.2.1 void s1237 ( SISLSurf \* psurf, int inmb1, int inmb2, double aepscu, int \* jstat )

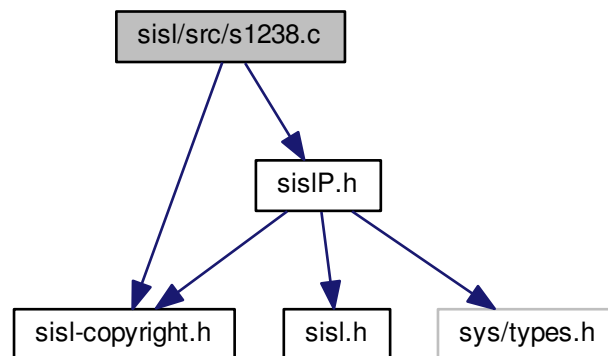
Definition at line 57 of file s1237.c.

## 30.2348 sisl/src/s1238.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1238.c:



### Macros

- #define [S1238](#)

### Functions

- void [s1238](#) (SISLSurf \*psurf, [SISLCurve](#) \*pcurve, int inmb1, int inmb2, double aepscu, double aepscu, int \*jstat)

## 30.2348.1 Macro Definition Documentation

### 30.2348.1.1 #define S1238

Definition at line 49 of file s1238.c.

## 30.2348.2 Function Documentation

### 30.2348.2.1 void s1238 ( SISLSurf \* *psurf*, SISLCurve \* *pcurve*, int *inmb1*, int *inmb2*, double *aepsco*, double *aepsco*, int \* *jstat* )

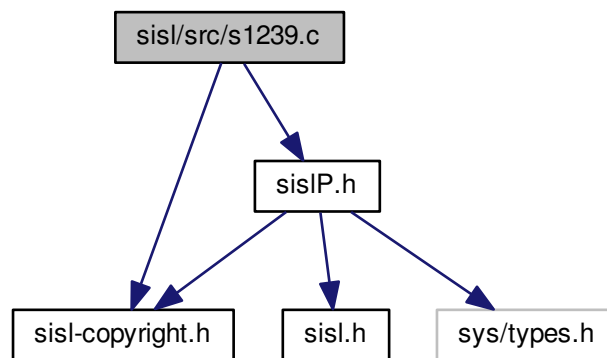
Definition at line 58 of file s1238.c.

## 30.2349 sisl/src/s1239.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1239.c:



## Macros

- #define [S1239](#)

## Functions

- void [s1239](#) (SISLCurve \**pcpar*, int *ipar*, double *apar*, SISLCurve \**pcurve*, double *aepsco*, double *aepsge*, SISLCurve \*\**vpartc*, int *imax*, int \**jpartc*, int \**jstat*)

### 30.2349.1 Macro Definition Documentation

#### 30.2349.1.1 #define S1239

Definition at line 49 of file s1239.c.

### 30.2349.2 Function Documentation

#### 30.2349.2.1 void s1239 ( SISLCurve \* pcp, int ipar, double apar, SISLCurve \* pcurve, double aepsco, double aepsge, SISLCurve \*\* vpartc, int imax, int \* jpartc, int \* jstat )

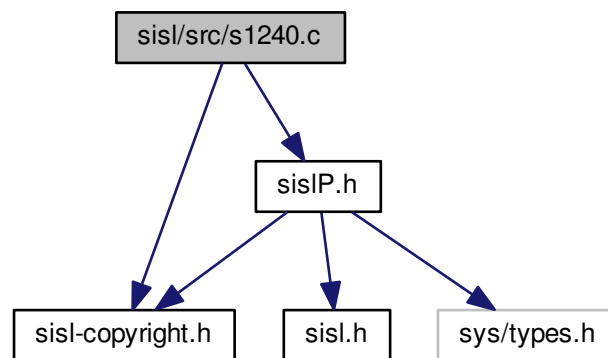
Definition at line 68 of file s1239.c.

### 30.2350 sisl/src/s1240.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1240.c:



#### Macros

- #define [S1240](#)

#### Functions

- void [s1240](#) (SISLCurve \*pcurve, double aepsge, double \*clength, int \*jstat)

## 30.2350.1 Macro Definition Documentation

### 30.2350.1.1 #define S1240

Definition at line 49 of file s1240.c.

## 30.2350.2 Function Documentation

### 30.2350.2.1 void s1240 ( SISLCurve \* pcurve, double epsge, double \* clength, int \* jstat )

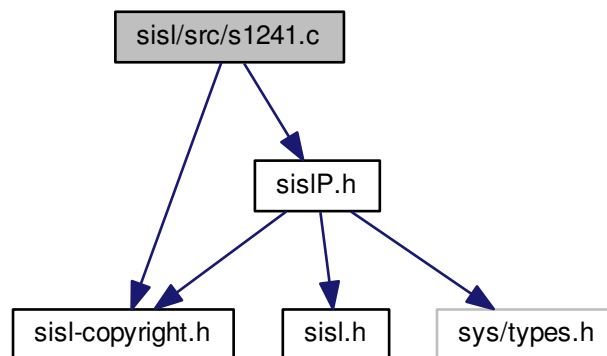
Definition at line 57 of file s1240.c.

## 30.2351 sisl/src/s1241.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1241.c:



## Macros

- #define [S1241](#)

## Functions

- void [s1241](#) (SISLCurve \*pcurve, point, int dim, double epsge, double \*area, int \*stat)

### 30.2351.1 Macro Definition Documentation

#### 30.2351.1.1 #define S1241

Definition at line 49 of file s1241.c.

### 30.2351.2 Function Documentation

#### 30.2351.2.1 void s1241 ( SISLCurve \* pcurve, point , int dim, double epsge, double \* area, int \* stat )

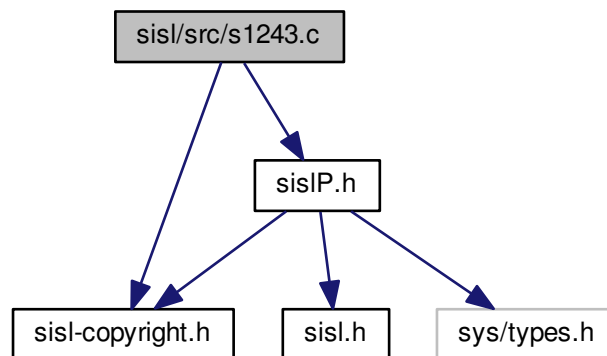
Definition at line 58 of file s1241.c.

## 30.2352 sisl/src/s1243.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1243.c:



### Macros

- #define [S1243](#)

### Functions

- void [s1243](#) (SISLCurve \*pcurve, point, int dim, double epsge, weight, double \*area, double \*moment, int \*stat)



### 30.2352.1 Macro Definition Documentation

#### 30.2352.1.1 #define S1243

Definition at line 49 of file s1243.c.

### 30.2352.2 Function Documentation

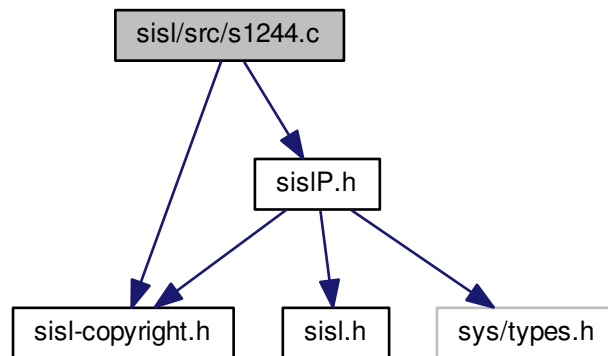
#### 30.2352.2.1 void s1243 ( SISLCurve \* pcurve, point , int dim, double epsge, weight , double \* area, double \* moment, int \* stat )

Definition at line 58 of file s1243.c.

## 30.2353 sisl/src/s1244.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
```

Include dependency graph for s1244.c:



### Macros

- #define [S1244](#)

### Functions

- void [s1244](#) (knots, int knot\_reg, int first\_order, int [second\\_order](#), int in, int first\_index, int second\_index, double \*integral, int \*stat)

### 30.2353.1 Macro Definition Documentation

#### 30.2353.1.1 #define S1244

Definition at line 49 of file s1244.c.

### 30.2353.2 Function Documentation

#### 30.2353.2.1 void s1244 ( knots , int knot\_reg, int first\_order, int second\_order, int in, int first\_index, int second\_index, double \* integral, int \* stat )

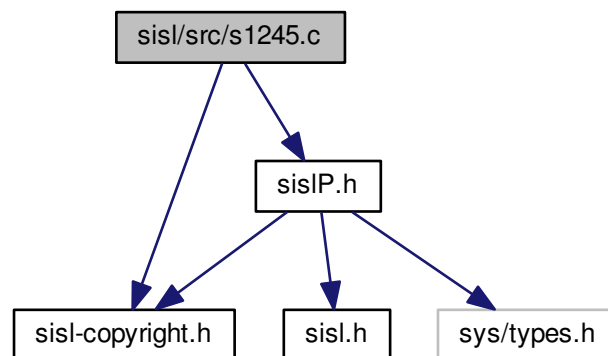
Definition at line 58 of file s1244.c.

## 30.2354 sisl/src/s1245.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1245.c:



### Macros

- #define [S1245](#)

### Functions

- void [s1245](#) (coef, int ik, int dim, point, double local\_tol, int depth, weight, double \*area, double \*moment, int \*stat)

### 30.2354.1 Macro Definition Documentation

#### 30.2354.1.1 #define S1245

Definition at line 49 of file s1245.c.

### 30.2354.2 Function Documentation

#### 30.2354.2.1 void s1245 ( coef , int ik, int dim, point , double local\_tol, int depth, weight , double \* area, double \* moment, int \* stat )

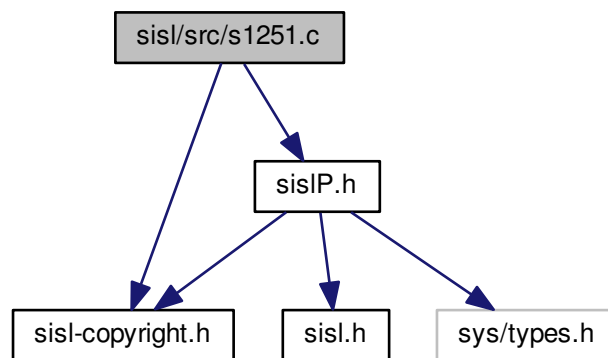
Definition at line 59 of file s1245.c.

## 30.2355 sisl/src/s1251.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1251.c:



### Macros

- #define [S1251](#)

### Functions

- void [s1251](#) ([SISLCurve](#) \*pcurve, [double](#) aepsco, [double](#) \*clength, int \*jstat)

## 30.2355.1 Macro Definition Documentation

### 30.2355.1.1 #define S1251

Definition at line 48 of file s1251.c.

## 30.2355.2 Function Documentation

### 30.2355.2.1 void s1251 ( SISLCurve \* pcurve, double aepsco, double \* clength, int \* jstat )

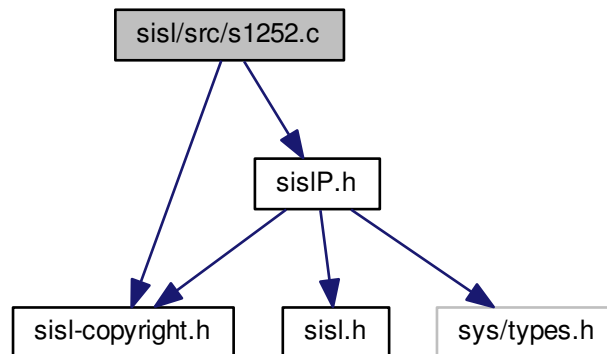
Definition at line 56 of file s1251.c.

## 30.2356 sisl/src/s1252.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1252.c:



## Macros

- #define [S1252](#)

## Functions

- void [s1252](#) (SISLCurve \*pcurve, double aepsge, double astart, double \*cpos, int \*jstat)

## 30.2356.1 Macro Definition Documentation

### 30.2356.1.1 #define S1252

Definition at line 49 of file s1252.c.

## 30.2356.2 Function Documentation

### 30.2356.2.1 void s1252 ( SISLCurve \* pcurve, double aepsge, double astart, double \* cpos, int \* jstat )

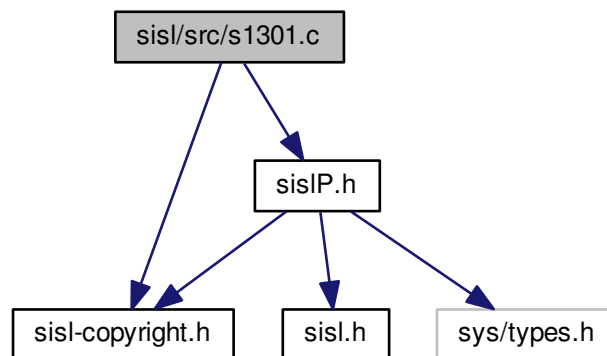
Definition at line 70 of file s1252.c.

## 30.2357 sisl/src/s1301.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1301.c:



## Macros

- #define [S1301](#)

## Functions

- void [s1301](#) (double areler, double angle, int idim, SISLCurve \*\*pc, int \*jstat)

## 30.2357.1 Macro Definition Documentation

### 30.2357.1.1 #define S1301

Definition at line 49 of file s1301.c.

## 30.2357.2 Function Documentation

### 30.2357.2.1 void s1301 ( double *areler*, double *angle*, int *idim*, SISLCurve \*\* *pc*, int \* *jstat* )

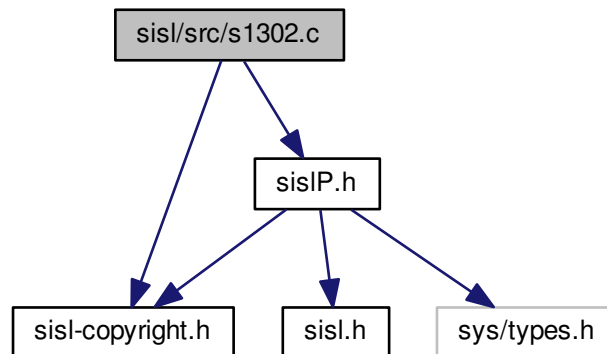
Definition at line 57 of file s1301.c.

## 30.2358 sisl/src/s1302.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1302.c:



## Macros

- #define [S1302](#)

## Functions

- void [s1302](#) (SISLCurve \*pc, double [aepsge](#), double [angle](#), ep, eaxis, SISLSurf \*\*rs, int \*jstat)

## 30.2358.1 Macro Definition Documentation

### 30.2358.1.1 #define S1302

Definition at line 49 of file s1302.c.

## 30.2358.2 Function Documentation

### 30.2358.2.1 void s1302 ( SISLCurve \* pc, double aepsge, double angle, ep, eaxis, SISLSurf \*\* rs, int \* jstat )

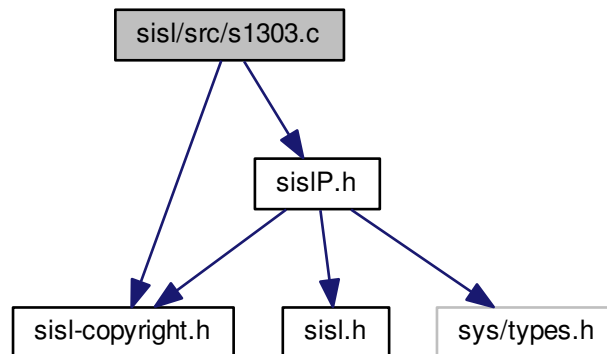
Definition at line 58 of file s1302.c.

## 30.2359 sisl/src/s1303.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1303.c:



## Macros

- #define [S1303](#)

## Functions

- void [s1303](#) (epstrt, [double aepsge](#), [double angle](#), epcnt, eaxis, int idim, [SISLCurve \\*\\*rc](#), int \*jstat)

## 30.2359.1 Macro Definition Documentation

### 30.2359.1.1 #define S1303

Definition at line 47 of file s1303.c.

## 30.2359.2 Function Documentation

### 30.2359.2.1 void s1303 ( epstrt , double aeprge, double angle, epcent , eaxis , int idim, SISLCurve \*\* rc, int \* jstat )

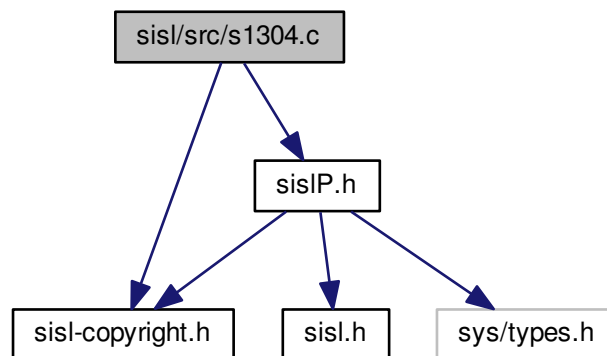
Definition at line 56 of file s1303.c.

## 30.2360 sisl/src/s1304.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1304.c:



## Macros

- #define [S1304](#)

## Functions

- void [s1304](#) (ep, eq, eparp, eparq, egeo3d, egeop, egeopq, int \*jstat)



## 30.2360.1 Macro Definition Documentation

### 30.2360.1.1 #define S1304

Definition at line 49 of file s1304.c.

## 30.2360.2 Function Documentation

### 30.2360.2.1 void s1304 ( ep , eq , eparp , eparq , egeo3d , egeop , egeoq , int \* jstat )

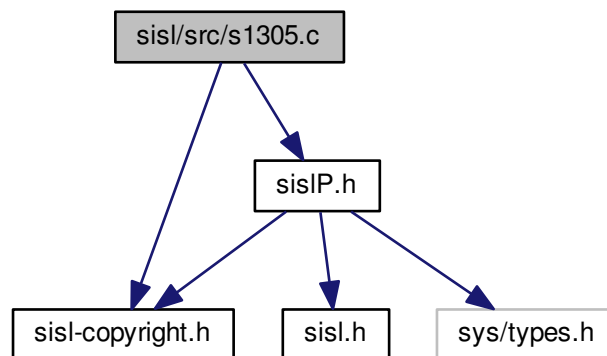
Definition at line 58 of file s1304.c.

## 30.2361 sisl/src/s1305.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1305.c:



## Macros

- #define [S1305](#)

## Functions

- void [s1305](#) (epar1, epar2, eval1, eval2, int \*jbound, gpar, int \*jstat)

## 30.2361.1 Macro Definition Documentation

### 30.2361.1.1 #define S1305

Definition at line 49 of file s1305.c.

## 30.2361.2 Function Documentation

### 30.2361.2.1 void s1305 ( epar1 , epar2 , eval1 , eval2 , int \* jbound , gpar , int \* jstat )

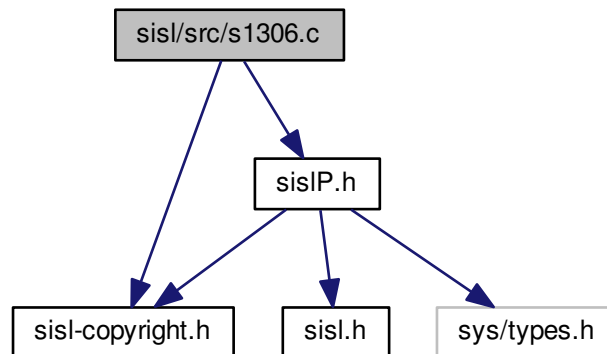
Definition at line 58 of file s1305.c.

## 30.2362 sisl/src/s1306.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1306.c:



## Macros

- #define [S1306](#)

## Functions

- void [s1306](#) (ep, eparp, eimpli, int ideg, egeo3d, egeop, int \*jstat)

## 30.2362.1 Macro Definition Documentation

### 30.2362.1.1 #define S1306

Definition at line 49 of file s1306.c.

## 30.2362.2 Function Documentation

### 30.2362.2.1 void s1306 ( ep , eparp , eimpli , int ideg, egeo3d , egeop , int \* jstat )

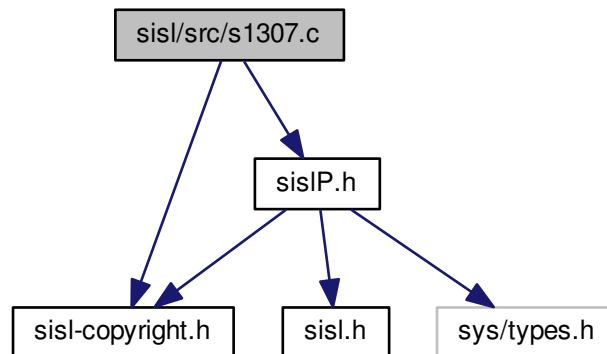
Definition at line 58 of file s1306.c.

## 30.2363 sisl/src/s1307.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1307.c:



## Macros

- #define [S1307](#)

## Functions

- void [s1307](#) (ep, int idim, egeo, int \*jstat)

### 30.2363.1 Macro Definition Documentation

#### 30.2363.1.1 #define S1307

Definition at line 47 of file s1307.c.

### 30.2363.2 Function Documentation

#### 30.2363.2.1 void s1307 ( ep , int *idim*, egeo , int \* *jstat* )

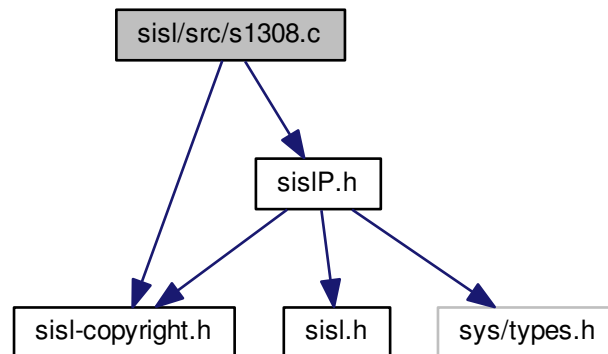
Definition at line 55 of file s1307.c.

## 30.2364 sisl/src/s1308.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1308.c:



### Macros

- #define [S1308](#)

### Functions

- void [s1308](#) (ep, int idim, eimpli, int ideg, enorm, int \*jstat)

## 30.2364.1 Macro Definition Documentation

### 30.2364.1.1 #define S1308

Definition at line 49 of file s1308.c.

## 30.2364.2 Function Documentation

### 30.2364.2.1 void s1308 ( ep , int *idim*, eimpli , int *ideg*, enorm , int \* *jstat* )

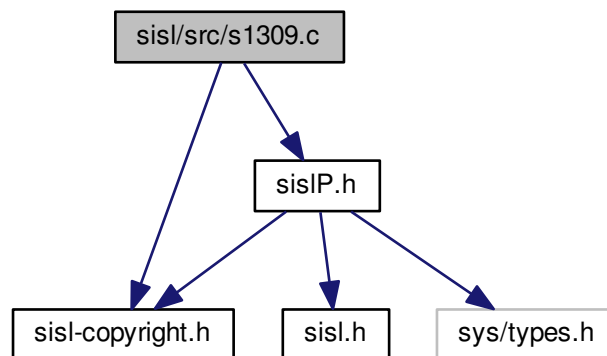
Definition at line 57 of file s1308.c.

## 30.2365 sisl/src/s1309.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1309.c:



## Macros

- `#define S1309`

## Functions

- `double s1309` (epnt, edir, eimpli, int ideg, int \*jstat)

### 30.2365.1 Macro Definition Documentation

#### 30.2365.1.1 #define S1309

Definition at line 47 of file s1309.c.

### 30.2365.2 Function Documentation

#### 30.2365.2.1 double s1309 ( epnt , edir , eimpli , int ideg, int \* jstat )

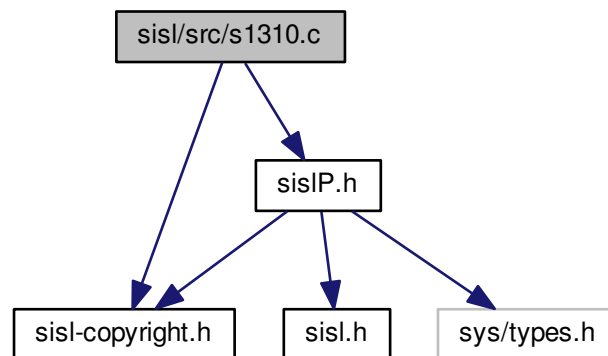
Definition at line 55 of file s1309.c.

## 30.2366 sisl/src/s1310.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1310.c:



### Macros

- #define [S1310](#)

### Functions

- void [s1310](#) ([SISLSurf](#) \*psurf1, [SISLSurf](#) \*psurf2, [SISLIntcurve](#) \*pinter, [double](#) aepsge, [double](#) amax, int icur, int igrph, int \*jstat)

## 30.2366.1 Macro Definition Documentation

### 30.2366.1.1 #define S1310

Definition at line 49 of file s1310.c.

## 30.2366.2 Function Documentation

### 30.2366.2.1 void s1310 ( SISLSurf \* *psurf1*, SISLSurf \* *psurf2*, SISLIntcurve \* *pinter*, double *aepsge*, double *amax*, int *icur*, int *igraph*, int \* *jstat* )

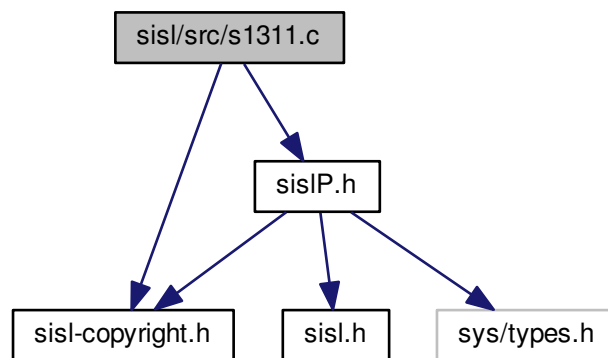
Definition at line 68 of file s1310.c.

## 30.2367 sisl/src/s1311.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1311.c:



## Macros

- #define [S1311](#)

## Functions

- [double s1311](#) (double *arad*, double *aepsge*, double *amax*, int \**jstat*)

## 30.2367.1 Macro Definition Documentation

### 30.2367.1.1 #define S1311

Definition at line 49 of file s1311.c.

## 30.2367.2 Function Documentation

### 30.2367.2.1 double s1311 ( double *arad*, double *aepsge*, double *amax*, int \* *jstat* )

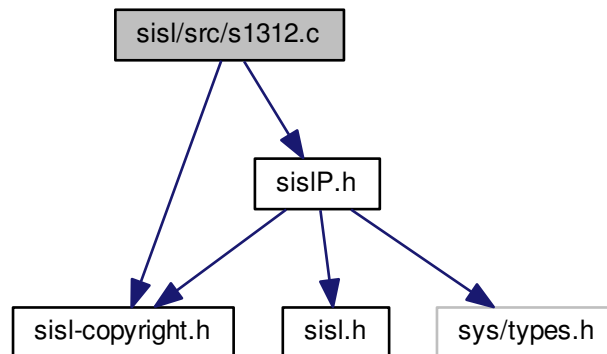
Definition at line 57 of file s1311.c.

## 30.2368 sisl/src/s1312.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1312.c:



## Macros

- #define [S1312](#)

## Functions

- void [s1312](#) (egeo, int idim, int inbinf, int ipar, epar, [SISLCurve](#) \*\*rcurve, int \*jstat)



## 30.2368.1 Macro Definition Documentation

### 30.2368.1.1 #define S1312

Definition at line 49 of file s1312.c.

## 30.2368.2 Function Documentation

### 30.2368.2.1 void s1312 ( egeo , int idim, int inbinf, int ipar, epar , SISLCurve \*\* rcurve, int \* jstat )

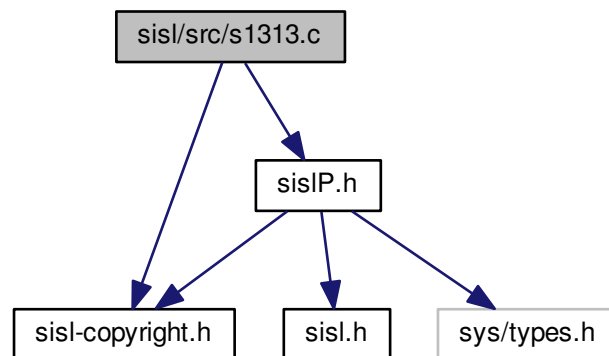
Definition at line 58 of file s1312.c.

## 30.2369 sisl/src/s1313.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1313.c:



## Macros

- #define [S1313](#)

## Functions

- void [s1313](#) (SISLSurf \*ps1, eimpli, int ideg, double aepsco, double aepsge, double amax, SISLIntcurve \*pintcr, int icur, int igrph, int \*jstat)

### 30.2369.1 Macro Definition Documentation

#### 30.2369.1.1 #define S1313

Definition at line 49 of file s1313.c.

### 30.2369.2 Function Documentation

#### 30.2369.2.1 void s1313 ( SISLSurf \* ps1, eimpli , int ideg, double aepsco, double aepsge, double amax, SISLIntcurve \* pintcr, int icur, int igraph, int \* jstat )

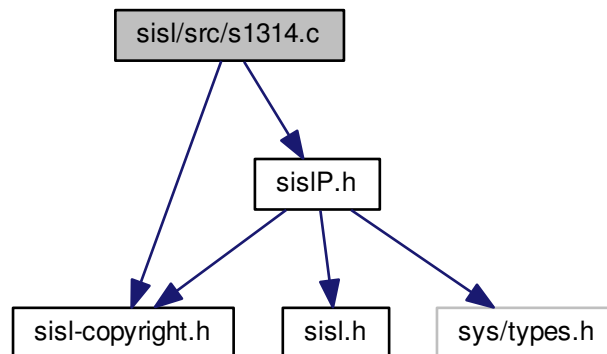
Definition at line 69 of file s1313.c.

## 30.2370 sisl/src/s1314.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1314.c:



### Macros

- #define [S1314](#)

### Functions

- void [s1314](#) (SISLSurf \*ps1, double \*epoint, double \*enorm, int idim, double aepsco, double aepsge, double amax, SISLIntcurve \*pintcr, int icur, int igraph, int \*jstat)

## 30.2370.1 Macro Definition Documentation

### 30.2370.1.1 #define S1314

Definition at line 49 of file s1314.c.

## 30.2370.2 Function Documentation

### 30.2370.2.1 void s1314 ( SISLSurf \* ps1, double \* epoint, double \* enorm, int idim, double aepsco, double aepsge, double amax, SISLIntcurve \* pintcr, int icur, int igrph, int \* jstat )

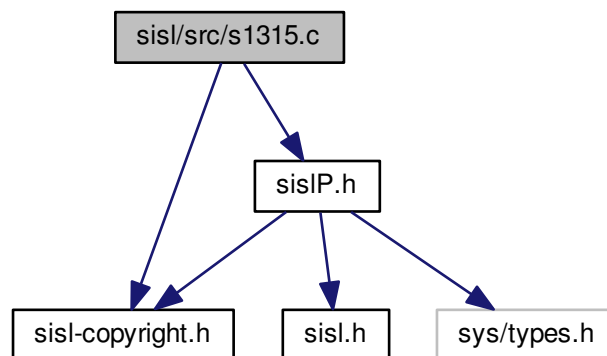
Definition at line 59 of file s1314.c.

## 30.2371 sisl/src/s1315.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1315.c:



## Macros

- #define [S1315](#)

## Functions

- void [s1315](#) (SISLSurf \*ps1, double \*ecentr, double aradiu, int idim, double aepsco, double aepsge, double amax, SISLIntcurve \*pintcr, int icur, int igrph, int \*jstat)

### 30.2371.1 Macro Definition Documentation

#### 30.2371.1.1 #define S1315

Definition at line 49 of file s1315.c.

### 30.2371.2 Function Documentation

#### 30.2371.2.1 void s1315 ( SISLSurf \* ps1, double \* ecentr, double aradiu, int idim, double aepsco, double aepsge, double amax, SISLIntcurve \* pintcr, int icur, int igrph, int \* jstat )

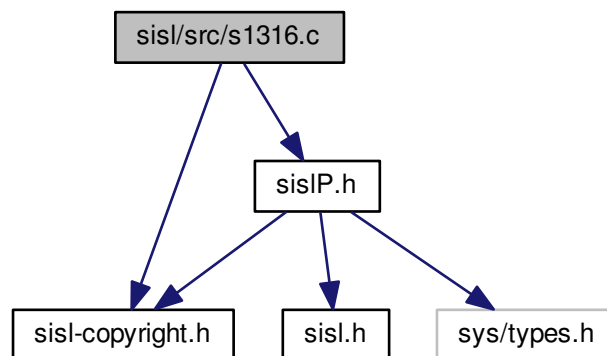
Definition at line 59 of file s1315.c.

## 30.2372 sisl/src/s1316.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1316.c:



### Macros

- #define [S1316](#)

### Functions

- void [s1316](#) (SISLSurf \*ps1, double \*epoint, double \*edirec, double aradiu, int idim, double aepsco, double aepsge, double amax, SISLIntcurve \*pintcr, int icur, int igrph, int \*jstat)

### 30.2372.1 Macro Definition Documentation

#### 30.2372.1.1 #define S1316

Definition at line 49 of file s1316.c.

### 30.2372.2 Function Documentation

30.2372.2.1 `void s1316 ( SISLSurf * ps1, double * epoint, double * edirec, double aradiu, int idim, double aepsco, double aepsge, double amax, SISLIntcurve * pintcr, int icur, int igrph, int * jstat )`

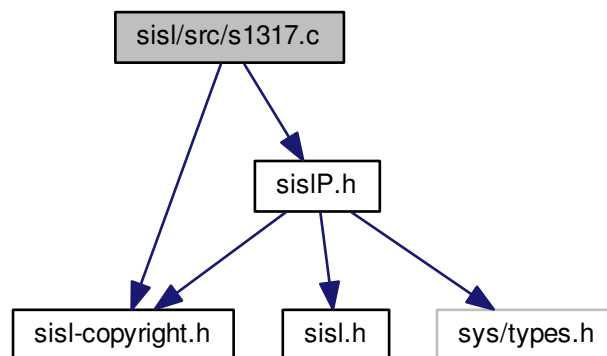
Definition at line 59 of file s1316.c.

## 30.2373 sisl/src/s1317.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1317.c:



### Macros

- #define [S1317](#)

### Functions

- void [s1317](#) (SISLSurf \*ps1, double \*etop, double \*eaxis, double \*econe, int idim, double aepsco, double aepsge, double amax, SISLIntcurve \*pintcr, int icur, int igrph, int \*jstat)

### 30.2373.1 Macro Definition Documentation

#### 30.2373.1.1 #define S1317

Definition at line 49 of file s1317.c.

### 30.2373.2 Function Documentation

30.2373.2.1 void s1317 ( SISLSurf \* ps1, double \* etop, double \* eaxis, double \* econe, int idim, double aepsco, double aepsge, double amax, SISLIntcurve \* pintcr, int icur, int igrph, int \* jstat )

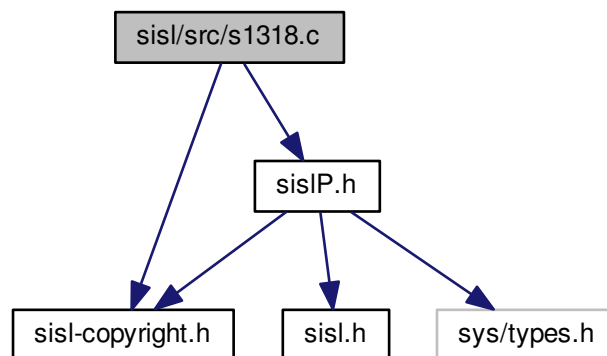
Definition at line 59 of file s1317.c.

## 30.2374 sisl/src/s1318.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1318.c:



### Macros

- #define [S1318](#)

### Functions

- void [s1318](#) (SISLSurf \*ps1, double \*ecentr, double \*enorm, double abigr, double asmalr, int idim, double aepsco, double aepsge, double amax, SISLIntcurve \*pintcr, int icur, int igrph, int \*jstat)

### 30.2374.1 Macro Definition Documentation

#### 30.2374.1.1 #define S1318

Definition at line 49 of file s1318.c.

### 30.2374.2 Function Documentation

#### 30.2374.2.1 void s1318 ( SISLSurf \* ps1, double \* ecentr, double \* enorm, double abigr, double asmalr, int idim, double aepsco, double aepsge, double amax, SISLIntcurve \* pintcr, int icur, int igrph, int \* jstat )

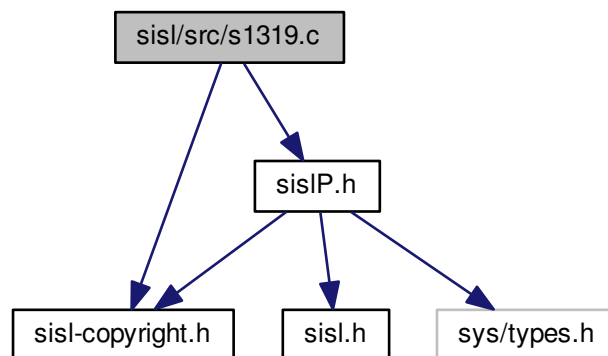
Definition at line 59 of file s1318.c.

## 30.2375 sisl/src/s1319.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1319.c:



### Macros

- #define [S1319](#)

### Functions

- void [s1319](#) (SISLSurf \*ps1, double \*eview, int idim, double aepsco, double aepsge, double amax, SISLIntcurve \*pintcr, int icur, int igrph, int \*jstat)

### 30.2375.1 Macro Definition Documentation

#### 30.2375.1.1 #define S1319

Definition at line 49 of file s1319.c.

### 30.2375.2 Function Documentation

#### 30.2375.2.1 void s1319 ( SISLSurf \* ps1, double \* eview, int idim, double aepsco, double aepsge, double amax, SISLIntcurve \* pintcr, int icur, int igraph, int \* jstat )

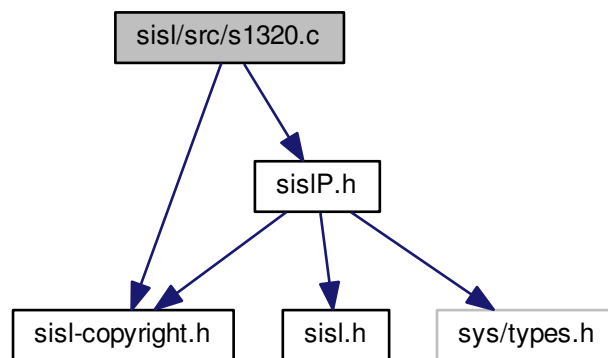
Definition at line 58 of file s1319.c.

## 30.2376 sisl/src/s1320.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1320.c:



### Macros

- #define [S1320](#)

### Functions

- void [s1320](#) (SISLSurf \*psurf, earray, int inarr, int ratflag, SISLSurf \*\*rsurf, int \*jstat)



## 30.2376.1 Macro Definition Documentation

### 30.2376.1.1 #define S1320

Definition at line 49 of file s1320.c.

## 30.2376.2 Function Documentation

### 30.2376.2.1 void s1320 ( SISLSurf \* *psurf*, earray , int *inarr*, int *ratflag*, SISLSurf \*\* *rsurf*, int \* *jstat* )

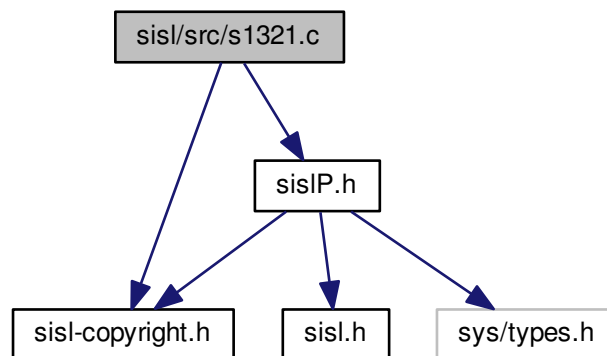
Definition at line 59 of file s1320.c.

## 30.2377 sisl/src/s1321.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1321.c:



## Macros

- #define [S1321](#)

## Functions

- void [s1321](#) (eentr, [double](#) aradiu, int idim, int inumb, carray, int \*jstat)

### 30.2377.1 Macro Definition Documentation

#### 30.2377.1.1 #define S1321

Definition at line 49 of file s1321.c.

### 30.2377.2 Function Documentation

#### 30.2377.2.1 void s1321 ( ecentr , double aradiu, int idim, int inumb, carray , int \* jstat )

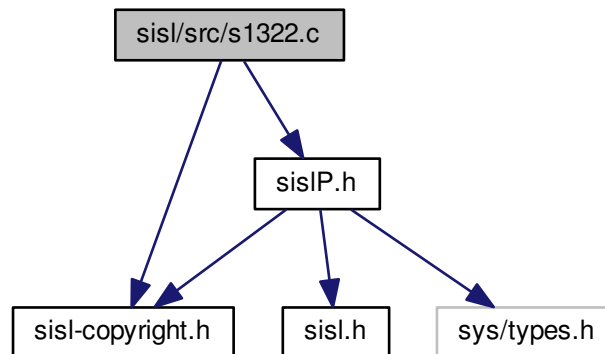
Definition at line 58 of file s1321.c.

## 30.2378 sisl/src/s1322.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1322.c:



### Macros

- #define [S1322](#)

### Functions

- void [s1322](#) (epoint, edirec, [double](#) aradiu, int idim, int inumb, carray, int \*jstat)

## 30.2378.1 Macro Definition Documentation

### 30.2378.1.1 #define S1322

Definition at line 49 of file s1322.c.

## 30.2378.2 Function Documentation

### 30.2378.2.1 void s1322 ( epoint , edirec , double aradiu, int idim, int inumb, carray , int \* jstat )

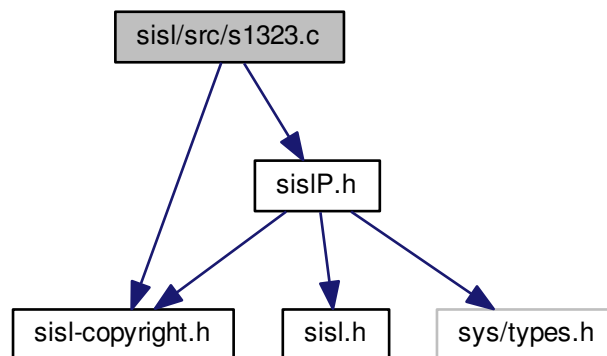
Definition at line 58 of file s1322.c.

## 30.2379 sisl/src/s1323.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1323.c:



## Macros

- #define [S1323](#)

## Functions

- void [s1323](#) (etop, eaxis, econe, int idim, int inumb, carray, int \*jstat)

## 30.2379.1 Macro Definition Documentation

### 30.2379.1.1 #define S1323

Definition at line 49 of file s1323.c.

## 30.2379.2 Function Documentation

### 30.2379.2.1 void s1323 ( etop , eaxis , econe , int *idim*, int *inumb*, carray , int \* *jstat* )

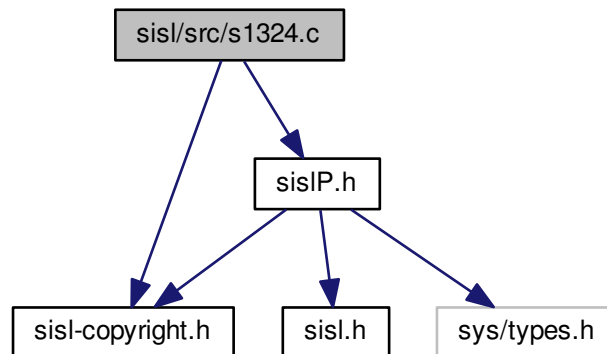
Definition at line 58 of file s1323.c.

## 30.2380 sisl/src/s1324.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1324.c:



## Macros

- #define [S1324](#)

## Functions

- void [s1324](#) (ecentr, [double](#) aradiu, enorm, int idim, carray, int \*jstat)

## 30.2380.1 Macro Definition Documentation

### 30.2380.1.1 #define S1324

Definition at line 49 of file s1324.c.

## 30.2380.2 Function Documentation

### 30.2380.2.1 void s1324 ( ecentr , double aradiu, enorm , int idim, carray , int \* jstat )

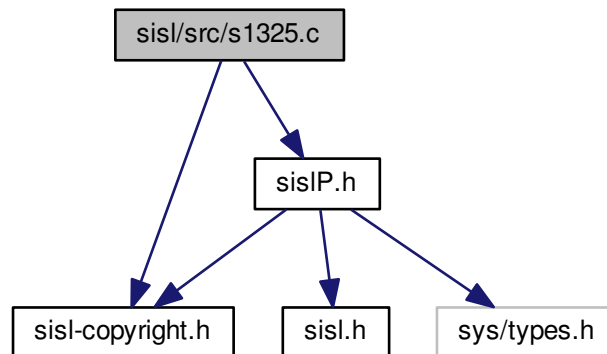
Definition at line 58 of file s1324.c.

## 30.2381 sisl/src/s1325.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1325.c:



## Macros

- #define [S1325](#)

## Functions

- [double s1325](#) (double aradiu, double angle)

## 30.2381.1 Macro Definition Documentation

### 30.2381.1.1 #define S1325

Definition at line 49 of file s1325.c.

## 30.2381.2 Function Documentation

### 30.2381.2.1 double s1325 ( double *aradiu*, double *angle* )

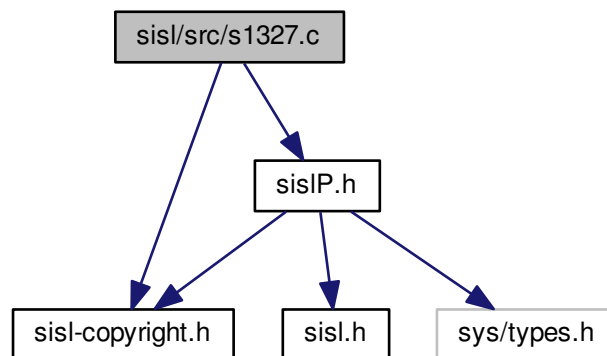
Definition at line 57 of file s1325.c.

## 30.2382 sisl/src/s1327.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1327.c:



## Macros

- #define [S1327](#)

## Functions

- void [s1327](#) ([SISLCurve](#) \*pcold, epoint, enorm1, enorm2, int idim, [SISLCurve](#) \*\*rcnew, int \*jstat)

## 30.2382.1 Macro Definition Documentation

### 30.2382.1.1 #define S1327

Definition at line 49 of file s1327.c.

## 30.2382.2 Function Documentation

### 30.2382.2.1 void s1327 ( SISLCurve \* *pcold*, epoint , enorm1 , enorm2 , int *idim*, SISLCurve \*\* *rcnew*, int \* *jstat* )

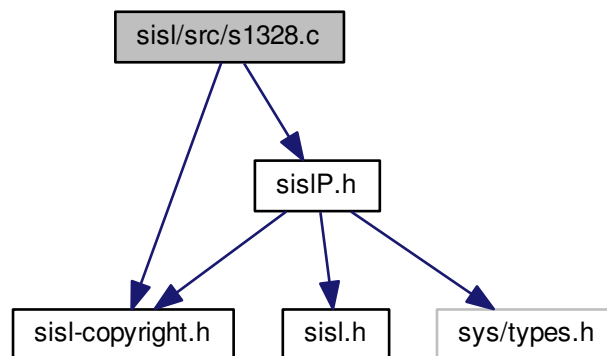
Definition at line 58 of file s1327.c.

## 30.2383 sisl/src/s1328.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1328.c:



## Macros

- #define [S1328](#)

## Functions

- void [s1328](#) (SISLSurf \**psold*, epoint, enorm1, enorm2, int *idim*, SISLSurf \*\**rsnew*, int \**jstat*)

### 30.2383.1 Macro Definition Documentation

#### 30.2383.1.1 #define S1328

Definition at line 49 of file s1328.c.

### 30.2383.2 Function Documentation

#### 30.2383.2.1 void s1328 ( SISLSurf \* *psold*, epoint , enorm1 , enorm2 , int *idim*, SISLSurf \*\* *rsnew*, int \* *jstat* )

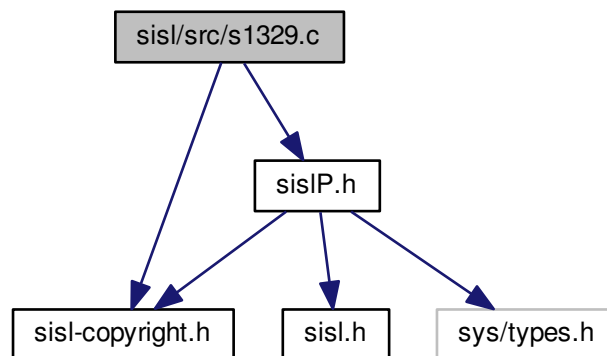
Definition at line 58 of file s1328.c.

## 30.2384 sisl/src/s1329.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1329.c:



### Macros

- #define [S1329](#)

### Functions

- void [s1329](#) (SISLSurf \**psold*, epoint, enorm, int *idim*, SISLSurf \*\**rsnew*, int \**jstat*)



### 30.2384.1 Macro Definition Documentation

#### 30.2384.1.1 #define S1329

Definition at line 49 of file s1329.c.

### 30.2384.2 Function Documentation

#### 30.2384.2.1 void s1329 ( SISLSurf \* *psold*, epoint , enorm , int *idim*, SISLSurf \*\* *rsnew*, int \* *jstat* )

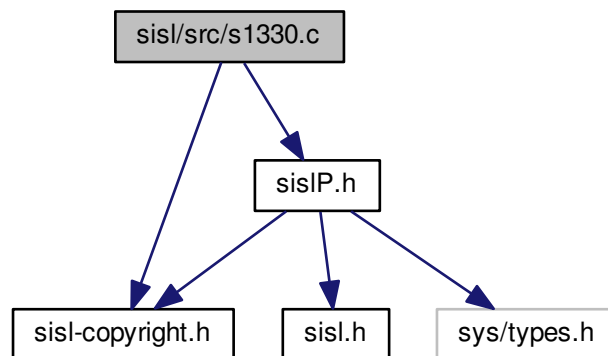
Definition at line 58 of file s1329.c.

## 30.2385 sisl/src/s1330.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1330.c:



### Macros

- #define [S1330](#)

### Functions

- void [s1330](#) (epar11, epar12, epar21, epar22, eval11, eval12, eval21, eval22, int \*jbound, gpar1, gpar2, int \*jstat)

## 30.2385.1 Macro Definition Documentation

### 30.2385.1.1 #define S1330

Definition at line 49 of file s1330.c.

## 30.2385.2 Function Documentation

### 30.2385.2.1 void s1330 ( epar11 , epar12 , epar21 , epar22 , eval11 , eval12 , eval21 , eval22 , int \* jbound , gpar1 , gpar2 , int \* jstat )

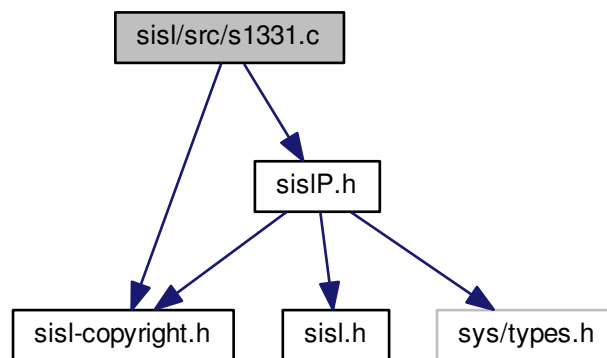
Definition at line 59 of file s1330.c.

## 30.2386 sisl/src/s1331.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1331.c:



### Macros

- #define [S1331](#)

### Functions

- void [s1331](#) (ep, eimpli, int ideg, int ider, gder, gnorm, int \*jstat)

## 30.2386.1 Macro Definition Documentation

### 30.2386.1.1 #define S1331

Definition at line 49 of file s1331.c.

## 30.2386.2 Function Documentation

### 30.2386.2.1 void s1331 ( ep , eimpli , int ideg , int ider , gder , gnorm , int \* jstat )

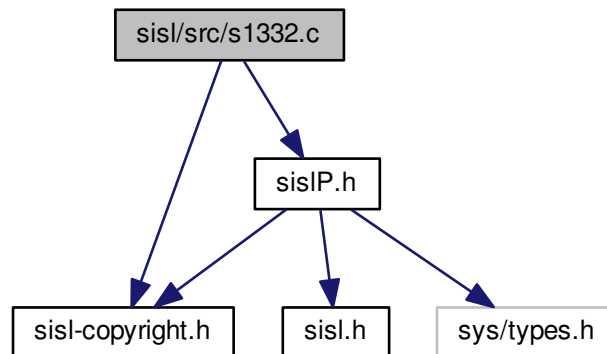
Definition at line 58 of file s1331.c.

## 30.2387 sisl/src/s1332.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1332.c:



## Macros

- #define [S1332](#)

## Functions

- void [s1332](#) ([SISLCurve](#) \*pc1, [SISLCurve](#) \*pc2, [double](#) aepsge, ep, [SISLSurf](#) \*\*rs, int \*jstat)

### 30.2387.1 Macro Definition Documentation

#### 30.2387.1.1 #define S1332

Definition at line 49 of file s1332.c.

### 30.2387.2 Function Documentation

#### 30.2387.2.1 void s1332 ( SISLCurve \* pc1, SISLCurve \* pc2, double aepsge, ep , SISLSurf \*\* rs, int \* jstat )

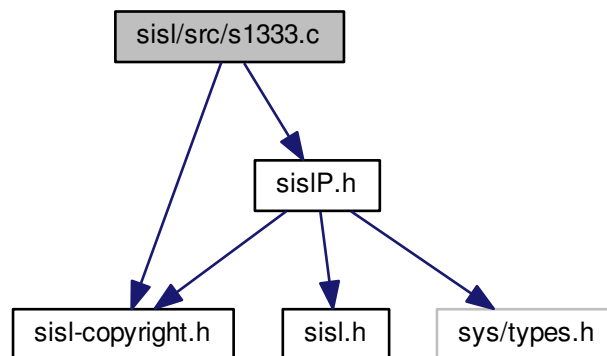
Definition at line 57 of file s1332.c.

## 30.2388 sisl/src/s1333.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1333.c:



### Macros

- #define [S1333](#)

### Functions

- void [s1333](#) (int inbcrv, vpcurv, nctyp, [double](#) astpar, int iopen, int iord2, int iflag, [SISLSurf](#) \*\*rsurf, [double](#) \*\*gpar, int \*jstat)

## 30.2388.1 Macro Definition Documentation

### 30.2388.1.1 #define S1333

Definition at line 49 of file s1333.c.

## 30.2388.2 Function Documentation

### 30.2388.2.1 void s1333 ( int *inbcrv*, vpcurv , nctyp , double *astpar*, int *iopen*, int *iord2*, int *iflag*, SISLSurf \*\* *rsurf*, double \*\* *gpar*, int \* *jstat* )

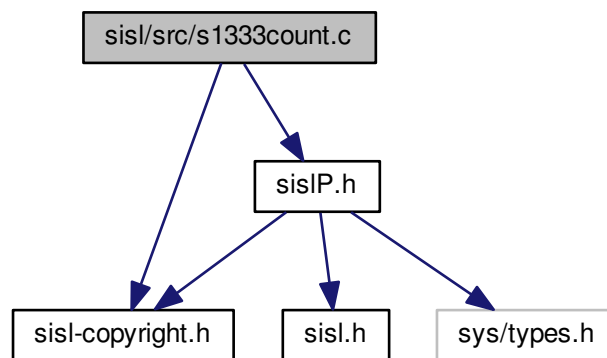
Definition at line 59 of file s1333.c.

## 30.2389 sisl/src/s1333count.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1333count.c:



### Macros

- #define [S1333\\_COUNT](#)

### Functions

- void [s1333\\_count](#) (int *inbcrv*, vpcurv, int \**jcont*, int \**jstat*)

## 30.2389.1 Macro Definition Documentation

### 30.2389.1.1 #define S1333\_COUNT

Definition at line 49 of file s1333count.c.

## 30.2389.2 Function Documentation

### 30.2389.2.1 void s1333\_count ( int *inbcrv*, *vpcurv* , int \* *jcont*, int \* *jstat* )

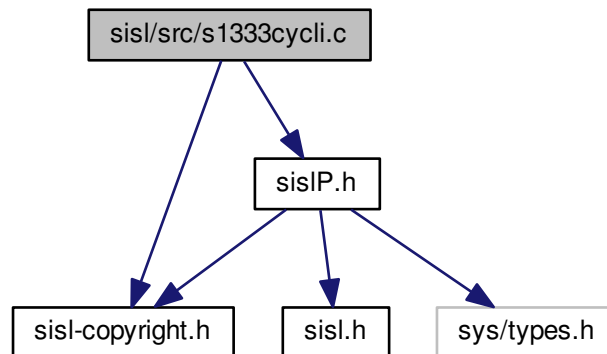
Definition at line 56 of file s1333count.c.

## 30.2390 sisl/src/s1333cycli.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1333cycli.c:



### Macros

- #define [S1333\\_CYCLIC](#)

### Functions

- void [s1333\\_cyclic](#) (SISLSurf \*vsurf, int icont, int \*jstat)

## 30.2390.1 Macro Definition Documentation

### 30.2390.1.1 #define S1333\_CYCLIC

Definition at line 49 of file s1333cycli.c.

## 30.2390.2 Function Documentation

### 30.2390.2.1 void s1333\_cyclic ( SISLSurf \* vsurf, int icont, int \* jstat )

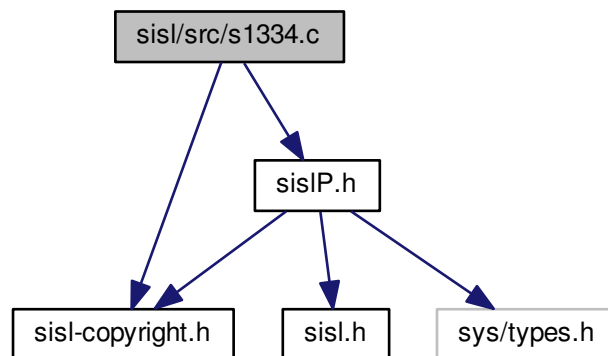
Definition at line 57 of file s1333cycli.c.

## 30.2391 sisl/src/s1334.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1334.c:



## Macros

- #define [S1334](#)

## Functions

- void [s1334](#) (epoint, int inbpnt, int idim, nptyp, int icnsta, int icnend, int iopen, int ik, [double](#) astpar, [double](#) \*cendpar, [SISLCurve](#) \*\*rc, [double](#) \*\*gpar, int \*jnbpar, int \*jstat)

### 30.2391.1 Macro Definition Documentation

#### 30.2391.1.1 #define S1334

Definition at line 49 of file s1334.c.

### 30.2391.2 Function Documentation

#### 30.2391.2.1 void s1334 ( epoint , int inbpnt, int idim, nptyp , int icnsta, int icnend, int iopen, int ik, double astpar, double \* cendpar, SISLCurve \*\* rc, double \*\* gpar, int \* jnbpnr, int \* jstat )

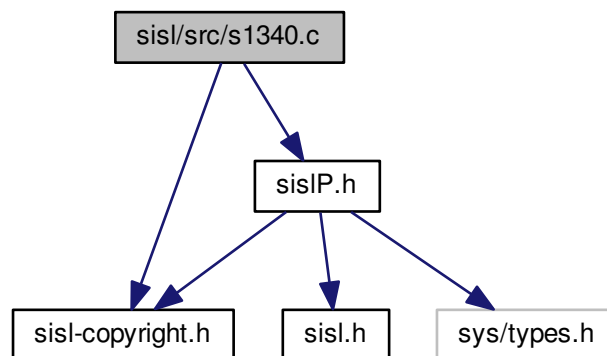
Definition at line 59 of file s1334.c.

## 30.2392 sisl/src/s1340.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1340.c:



### Macros

- #define [S1340](#)

### Functions

- void [s1340](#) ([SISLCurve](#) \*oldcurve, eps, int startfix, int endfix, [double](#) epsco, int itmax, [SISLCurve](#) \*\*newcurve, maxerr, int \*stat)



### 30.2392.1 Macro Definition Documentation

#### 30.2392.1.1 #define S1340

Definition at line 48 of file s1340.c.

### 30.2392.2 Function Documentation

#### 30.2392.2.1 void s1340 ( SISLCurve \* oldcurve, eps , int startfix, int endfix, double epsco, int itmax, SISLCurve \*\* newcurve, maxerr , int \* stat )

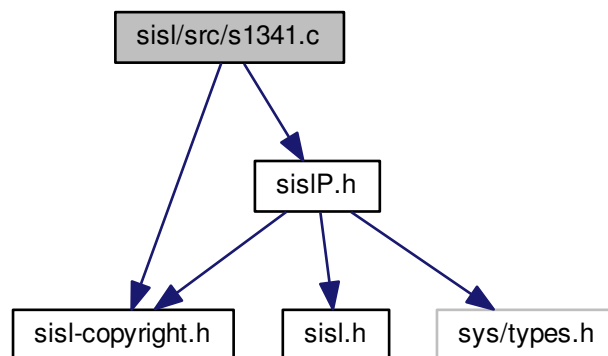
Definition at line 59 of file s1340.c.

## 30.2393 sisl/src/s1341.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1341.c:



### Macros

- #define [S1341](#)

### Functions

- void [s1341](#) (ep, int im, int idim, int ipar, epar, eeps, int ilend, int irend, [double](#) afctol, [double](#) aepsco, int itmax, int ik, [SISLCurve](#) \*\*rc, emxerr, int \*jstat)

### 30.2393.1 Macro Definition Documentation

#### 30.2393.1.1 #define S1341

Definition at line 48 of file s1341.c.

### 30.2393.2 Function Documentation

#### 30.2393.2.1 void s1341 ( ep , int im, int idim, int ipar, epar , eeps , int ilend, int irend, double afctol, double aepsco, int itmax, int ik, SISLCurve \*\* rc, emxerr , int \* jstat )

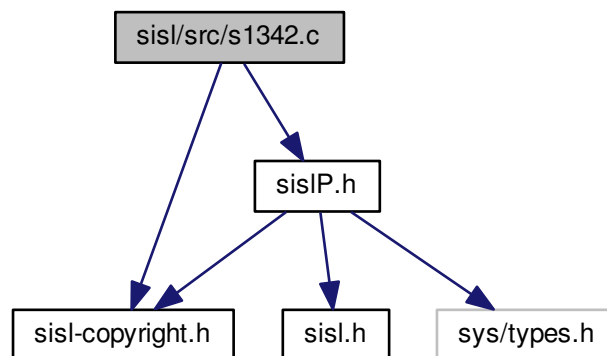
Definition at line 57 of file s1341.c.

## 30.2394 sisl/src/s1342.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1342.c:



### Macros

- #define [S1342](#)

### Functions

- void [s1342](#) (ep, ev, int im, int idim, int ipar, epar, eeps, int ilend, int irend, double aepsco, int itmax, [SISLCurve](#) \*\*rc, emxerr, int \*jstat)

## 30.2394.1 Macro Definition Documentation

### 30.2394.1.1 #define S1342

Definition at line 48 of file s1342.c.

## 30.2394.2 Function Documentation

### 30.2394.2.1 void s1342 ( ep , ev , int im, int idim, int ipar, epar , eeps , int ilend, int irend, double aepsco, int itmax, SISLCurve \*\* rc, emxerr , int \* jstat )

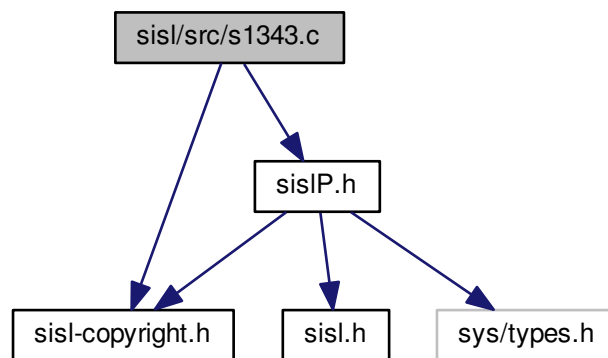
Definition at line 57 of file s1342.c.

## 30.2395 sisl/src/s1343.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1343.c:



## Macros

- #define [S1343](#)

## Functions

- void [s1343](#) ([SISLCurve](#) \*pc, eeps, int ilend, int irend, [double](#) aepsco, int itmax, [SISLCurve](#) \*\*rc, int \*jstat)

### 30.2395.1 Macro Definition Documentation

#### 30.2395.1.1 #define S1343

Definition at line 48 of file s1343.c.

### 30.2395.2 Function Documentation

#### 30.2395.2.1 void s1343 ( SISLCurve \* pc, eeps , int ilend, int irend, double aepsco, int itmax, SISLCurve \*\* rc, int \* jstat )

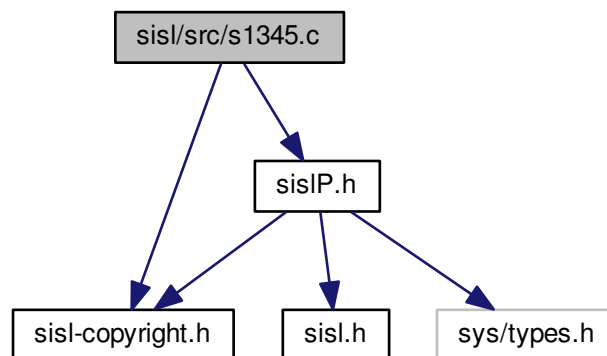
Definition at line 56 of file s1343.c.

## 30.2396 sisl/src/s1345.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1345.c:



### Macros

- #define [S1345](#)

### Functions

- void [s1345](#) (SISLSurf \*oldsurf, eps, edgefix, edgeps, double epsco, int opt, int itmax, SISLSurf \*\*newsurf, maxerr, int \*stat)

### 30.2396.1 Macro Definition Documentation

#### 30.2396.1.1 #define S1345

Definition at line 48 of file s1345.c.

### 30.2396.2 Function Documentation

#### 30.2396.2.1 void s1345 ( SISLSurf \* *oldsurf*, *eps*, *edgefix*, *edgeps*, double *epsco*, int *opt*, int *itmax*, SISLSurf \*\* *newsurf*, *maxerr*, int \* *stat* )

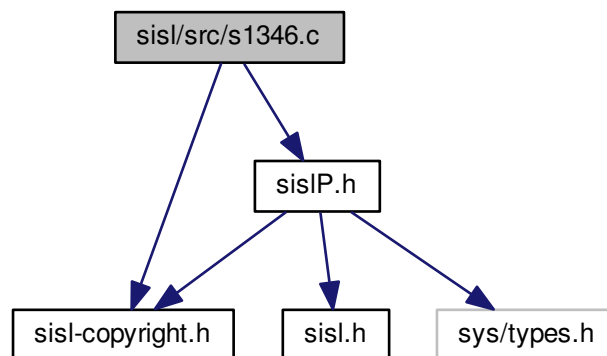
Definition at line 58 of file s1345.c.

## 30.2397 sisl/src/s1346.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1346.c:



### Macros

- #define [S1346](#)

### Functions

- void [s1346](#) (*ep*, int *im1*, int *im2*, int *idim*, int *ipar*, *epar1*, *epar2*, *eeps*, *nend*, *edgeps*, double *afctol*, double *aepsco*, int *iopt*, int *itmax*, int *ik1*, int *ik2*, SISLSurf \*\**rs*, *emxerr*, int \**jstat*)

### 30.2397.1 Macro Definition Documentation

#### 30.2397.1.1 #define S1346

Definition at line 48 of file s1346.c.

### 30.2397.2 Function Documentation

30.2397.2.1 void s1346 ( ep , int im1 , int im2 , int idim , int ipar , epar1 , epar2 , eeps , nend , edgeps , double afctol , double aepsco , int iopt , int itmax , int ik1 , int ik2 , SISLSurf \*\* rs , emxerr , int \* jstat )

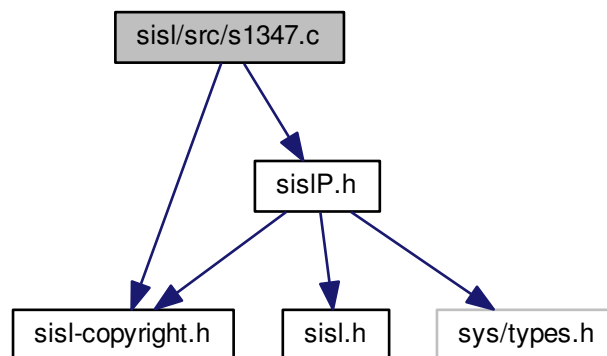
Definition at line 58 of file s1346.c.

## 30.2398 sisl/src/s1347.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1347.c:



### Macros

- #define [S1347](#)

### Functions

- void [s1347](#) (ep, etang1, etang2, eder11, int im1, int im2, int idim, int ipar, epar1, epar2, eeps, nend, edgeps, double aepsco, int iopt, int itmax, [SISLSurf](#) \*\*rs, emxerr, int \*jstat)

## 30.2398.1 Macro Definition Documentation

### 30.2398.1.1 #define S1347

Definition at line 48 of file s1347.c.

## 30.2398.2 Function Documentation

30.2398.2.1 void s1347 ( ep , etang1 , etang2 , eder11 , int im1 , int im2 , int idim , int ipar , epar1 , epar2 , eeps , nend , edgeps , double aepsco , int iopt , int itmax , SISLSurf \*\* rs , emxerr , int \* jstat )

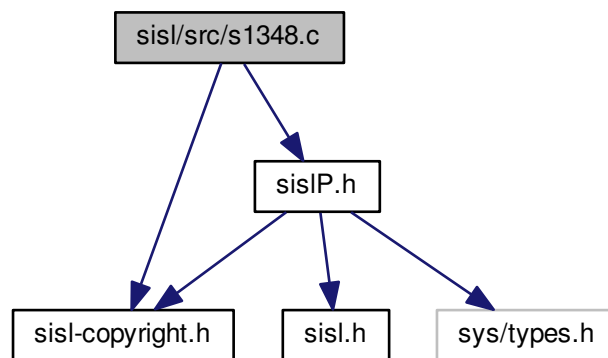
Definition at line 58 of file s1347.c.

## 30.2399 sisl/src/s1348.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1348.c:



### Macros

- #define [S1348](#)

### Functions

- void [s1348](#) (SISLSurf \*ps, eeps, nend, edgeps, double aepsco, int iopt, int itmax, SISLSurf \*\*rs, int \*jstat)

## 30.2399.1 Macro Definition Documentation

### 30.2399.1.1 #define S1348

Definition at line 48 of file s1348.c.

## 30.2399.2 Function Documentation

### 30.2399.2.1 void s1348 ( SISLSurf \* ps, eeps , nend , edgeps , double aepsco, int iopt, int itmax, SISLSurf \*\* rs, int \* jstat )

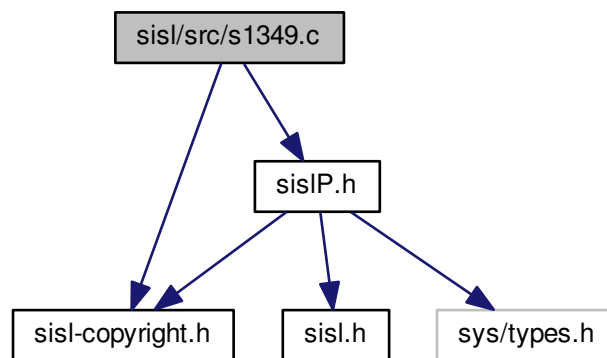
Definition at line 56 of file s1348.c.

## 30.2400 sisl/src/s1349.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1349.c:



### Macros

- #define [S1349](#)

### Functions

- void [s1349](#) (int inbcrv, vpcrv, int \*jstat)



## 30.2400.1 Macro Definition Documentation

### 30.2400.1.1 #define S1349

Definition at line 49 of file s1349.c.

## 30.2400.2 Function Documentation

### 30.2400.2.1 void s1349 ( int *inbcrv*, *vpcrv*, int \* *jstat* )

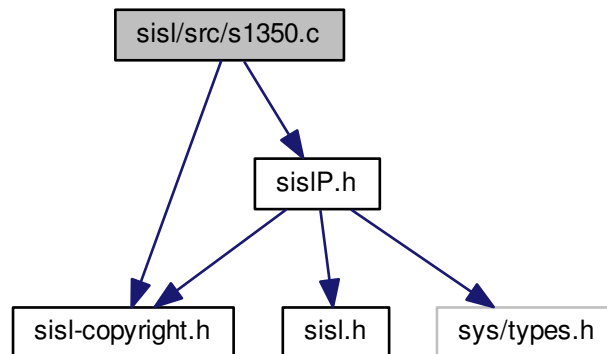
Definition at line 57 of file s1349.c.

## 30.2401 sisl/src/s1350.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1350.c:



## Macros

- #define [S1350](#)

## Functions

- void [s1350](#) (ep, epar, int im, int idim, int ik, [SISLCurve](#) \*\*rc, int \*jstat)

### 30.2401.1 Macro Definition Documentation

#### 30.2401.1.1 #define S1350

Definition at line 48 of file s1350.c.

### 30.2401.2 Function Documentation

#### 30.2401.2.1 void s1350 ( ep , epar , int im, int idim, int ik, SISLCurve \*\* rc, int \* jstat )

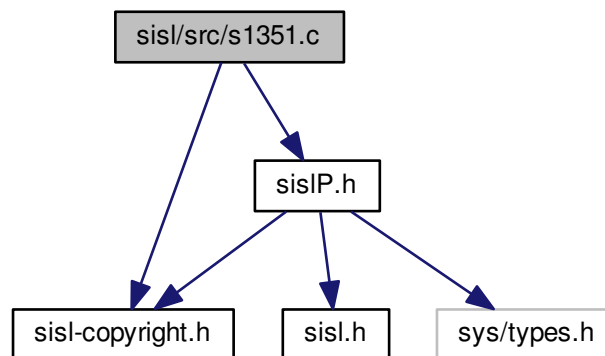
Definition at line 57 of file s1350.c.

## 30.2402 sisl/src/s1351.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1351.c:



### Macros

- #define [S1351](#)

### Functions

- void [s1351](#) (ep, int ipar, int im, int idim, int ik, [SISLCurve](#) \*\*rc, int \*jstat)

## 30.2402.1 Macro Definition Documentation

### 30.2402.1.1 #define S1351

Definition at line 48 of file s1351.c.

## 30.2402.2 Function Documentation

### 30.2402.2.1 void s1351 ( ep , int ipar, int im, int idim, int ik, SISLCurve \*\* rc, int \* jstat )

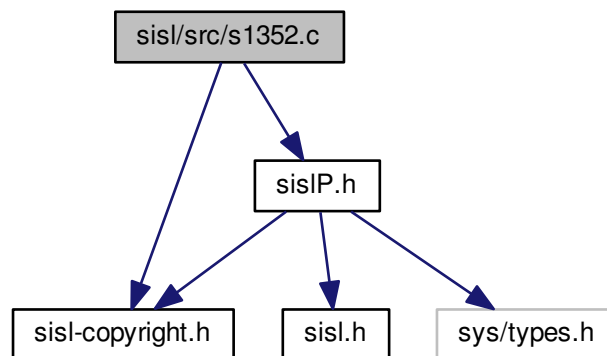
Definition at line 57 of file s1351.c.

## 30.2403 sisl/src/s1352.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1352.c:



## Macros

- #define [S1352](#)

## Functions

- void [s1352](#) (t, int n, int k, inteps, lefteps, righteps, int dim, int leftfix, int rightfix, eps, int \*stat)

### 30.2403.1 Macro Definition Documentation

#### 30.2403.1.1 #define S1352

Definition at line 48 of file s1352.c.

### 30.2403.2 Function Documentation

#### 30.2403.2.1 void s1352 ( t , int n , int k , inteps , lefteps , righteps , int dim , int leftfix , int rightfix , eps , int \* stat )

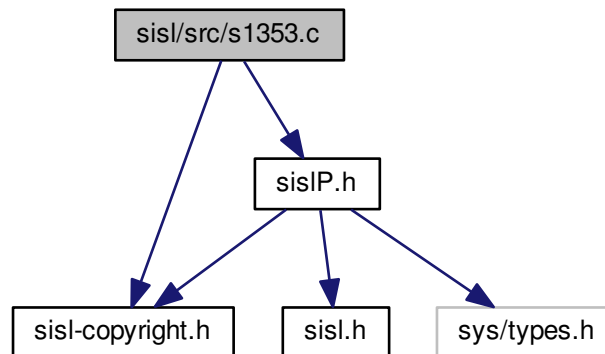
Definition at line 57 of file s1352.c.

## 30.2404 sisl/src/s1353.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1353.c:



### Macros

- #define [S1353](#)

### Functions

- void [s1353](#) ([SISLCurve](#) \*curve, eps, [rank\\_info](#) \*ranking, int \*stat)

### 30.2404.1 Macro Definition Documentation

#### 30.2404.1.1 #define S1353

Definition at line 48 of file s1353.c.

### 30.2404.2 Function Documentation

#### 30.2404.2.1 void s1353 ( SISLCurve \* curve, eps , rank\_info \* ranking, int \* stat )

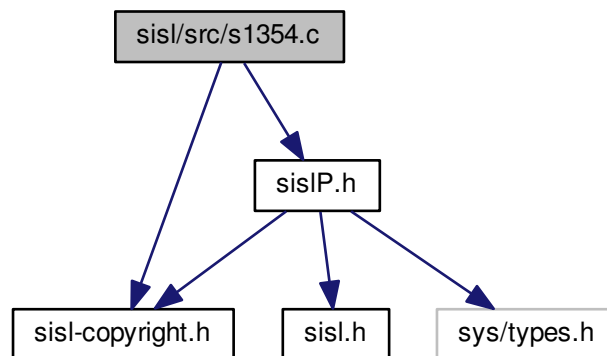
Definition at line 55 of file s1353.c.

## 30.2405 sisl/src/s1354.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1354.c:



### Macros

- #define [S1354](#)

### Functions

- void [s1354](#) (SISLCurve \*oldcurve, SISLCurve \*rankcurve, rank\_info \*ranking, eps, epsco, int startfix, int endfix, int mini, int maxi, SISLCurve \*\*newcurve, maxerr, int \*stat)

### 30.2405.1 Macro Definition Documentation

#### 30.2405.1.1 #define S1354

Definition at line 48 of file s1354.c.

### 30.2405.2 Function Documentation

#### 30.2405.2.1 void s1354 ( SISLCurve \* oldcurve, SISLCurve \* rankcurve, rank\_info \* ranking, eps , epsco , int startfix, int endfix, int mini, int maxi, SISLCurve \*\* newcurve, maxerr , int \* stat )

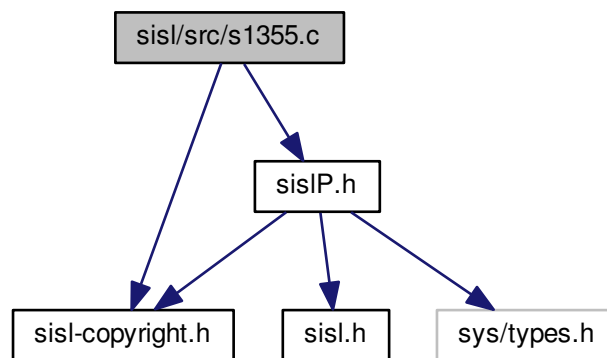
Definition at line 57 of file s1354.c.

## 30.2406 sisl/src/s1355.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1355.c:



### Macros

- #define [S1355](#)

### Functions

- void [s1355](#) (SISLCurve \*pc, eeps, double \*\*epar, int \*im, int \*jstat)

### 30.2406.1 Macro Definition Documentation

#### 30.2406.1.1 #define S1355

Definition at line 48 of file s1355.c.

### 30.2406.2 Function Documentation

#### 30.2406.2.1 void s1355 ( SISLCurve \* pc, eeps , double \*\* epar, int \* im, int \* jstat )

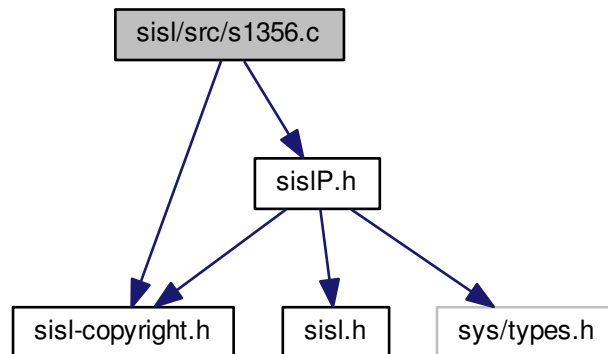
Definition at line 56 of file s1355.c.

## 30.2407 sisl/src/s1356.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1356.c:



### Macros

- #define [S1356](#)

### Functions

- void [s1356](#) (epoint, int inbpnt, int idim, nptyp, int icnsta, int icnend, int iopen, int ik, [double](#) astpar, [double](#) \*cendpar, [SISLCurve](#) \*\*rc, [double](#) \*\*gpar, int \*jnbpar, int \*jstat)

### 30.2407.1 Macro Definition Documentation

#### 30.2407.1.1 #define S1356

Definition at line 42 of file s1356.c.

### 30.2407.2 Function Documentation

#### 30.2407.2.1 void s1356 ( epoint , int inbpnt, int idim, nptyp , int icnsta, int icnend, int iopen, int ik, double astpar, double \* cendpar, SISLCurve \*\*rc, double \*\*gpar, int \*jnbpar, int \*jstat )

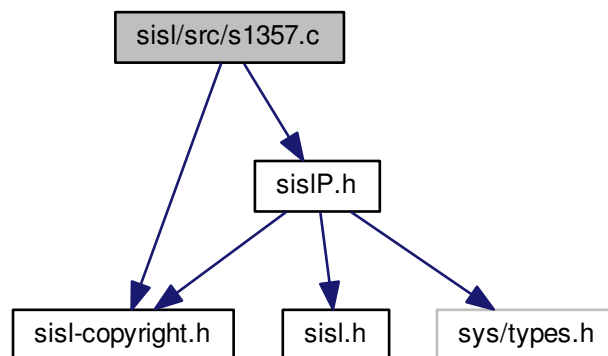
Definition at line 52 of file s1356.c.

## 30.2408 sisl/src/s1357.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1357.c:



### Macros

- #define [S1357](#)

### Functions

- void [s1357](#) (epoint, int inbpnt, int idim, ntype, epar, int icnsta, int icnend, int iopen, int ik, [double](#) astpar, [double](#) \*cendpar, [SISLCurve](#) \*\*rc, [double](#) \*\*gpar, int \*jnbpar, int \*jstat)



## 30.2408.1 Macro Definition Documentation

### 30.2408.1.1 #define S1357

Definition at line 43 of file s1357.c.

## 30.2408.2 Function Documentation

### 30.2408.2.1 void s1357 ( epoint , int inbpnt, int idim, ntype , epar , int icnsta, int icnend, int iopen, int ik, double astpar, double \* cendpar, SISLCurve \*\* rc, double \*\* gpar, int \* jnbpar, int \* jstat )

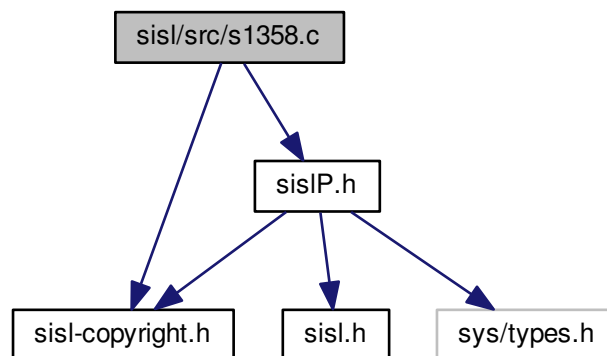
Definition at line 53 of file s1357.c.

## 30.2409 sisl/src/s1358.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1358.c:



## Macros

- #define [S1358](#)

## Functions

- void [s1358](#) (epoint, int inbpnt, int idim, ntype, epar, int icnsta, int icnend, int iopen, int ik, [double](#) astpar, [double](#) \*cendpar, [SISLCurve](#) \*\*rc, [double](#) \*\*gpar, int \*jnbpar, int \*jstat)

### 30.2409.1 Macro Definition Documentation

#### 30.2409.1.1 #define S1358

Definition at line 49 of file s1358.c.

### 30.2409.2 Function Documentation

#### 30.2409.2.1 void s1358 ( epoint , int inbpnt, int idim, ntype , epar , int icnsta, int icnend, int iopen, int ik, double astpar, double \* cendpar, SISLCurve \*\* rc, double \*\* gpar, int \* jnbpar, int \* jstat )

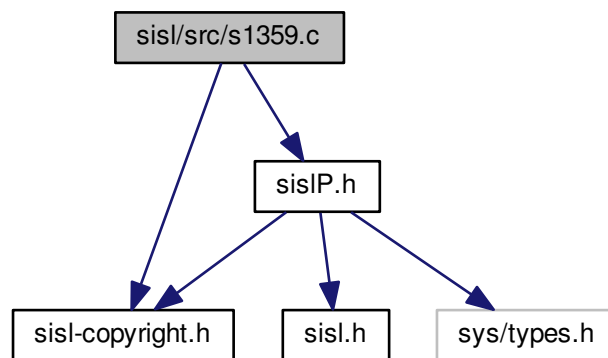
Definition at line 59 of file s1358.c.

## 30.2410 sisl/src/s1359.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1359.c:



### Macros

- #define [S1359](#)

### Functions

- void [s1359](#) (egeo, [double aepsge](#), int idim, int inbinf, int ipar, epar, [SISLCurve \\*\\*rcurve](#), int \*jstat)

### 30.2410.1 Macro Definition Documentation

#### 30.2410.1.1 #define S1359

Definition at line 47 of file s1359.c.

### 30.2410.2 Function Documentation

#### 30.2410.2.1 void s1359 ( egeo , double aepsge, int idim, int inbinf, int ipar, epar , SISLCurve \*\* rcurve, int \* jstat )

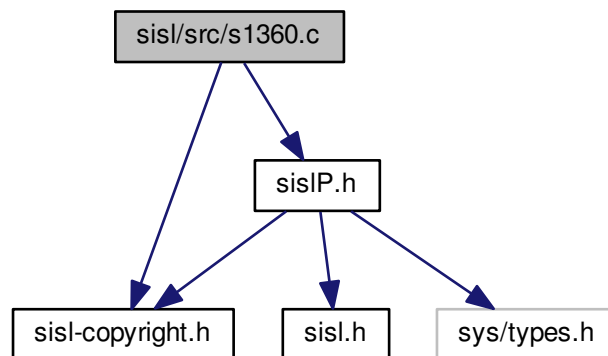
Definition at line 56 of file s1359.c.

## 30.2411 sisl/src/s1360.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1360.c:



### Macros

- #define [S1360](#)

### Functions

- void [s1360](#) (SISLCurve \*pc, double aoffset, double aepsge, enorm, double amax, int idim, SISLCurve \*\*rc, int \*jstat)

### 30.2411.1 Macro Definition Documentation

#### 30.2411.1.1 #define S1360

Definition at line 49 of file s1360.c.

### 30.2411.2 Function Documentation

#### 30.2411.2.1 void s1360 ( SISLCurve \* pc, double aoffset, double aepsge, enorm , double amax, int idim, SISLCurve \*\* rc, int \* jstat )

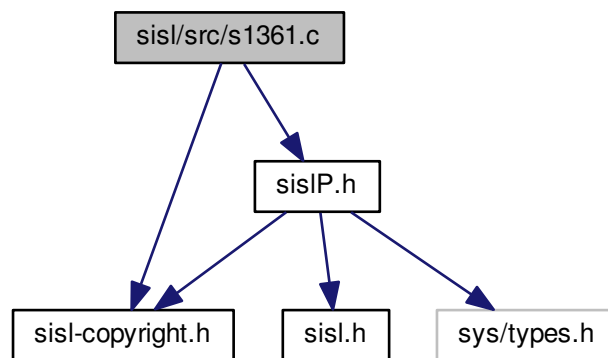
Definition at line 58 of file s1360.c.

## 30.2412 sisl/src/s1361.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1361.c:



### Macros

- #define [S1361](#)

### Functions

- void [s1361](#) (epnt1, epnt2, int idim, gmidd, gmtang, int \*jstat)

### 30.2412.1 Macro Definition Documentation

#### 30.2412.1.1 #define S1361

Definition at line 47 of file s1361.c.

### 30.2412.2 Function Documentation

#### 30.2412.2.1 void s1361 ( epnt1 , epnt2 , int *idim* , gmidd , gmtang , int \* *jstat* )

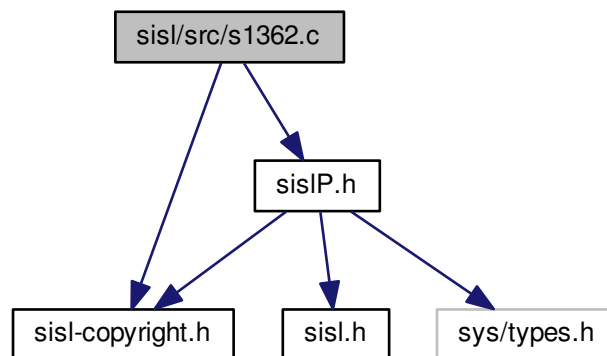
Definition at line 56 of file s1361.c.

## 30.2413 sisl/src/s1362.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1362.c:



### Macros

- #define [S1362](#)

### Functions

- void [s1362](#) ([SISLCurve](#) \*pc1, [double](#) aoffset, [enorm](#), int idim, int ider, [double](#) ax, int \*ileft, [eder](#), int \*jstat)

### 30.2413.1 Macro Definition Documentation

#### 30.2413.1.1 #define S1362

Definition at line 49 of file s1362.c.

### 30.2413.2 Function Documentation

#### 30.2413.2.1 void s1362 ( SISLCurve \* pc1, double aoffset, enorm , int idim, int ider, double ax, int \* ileft, eder , int \* jstat )

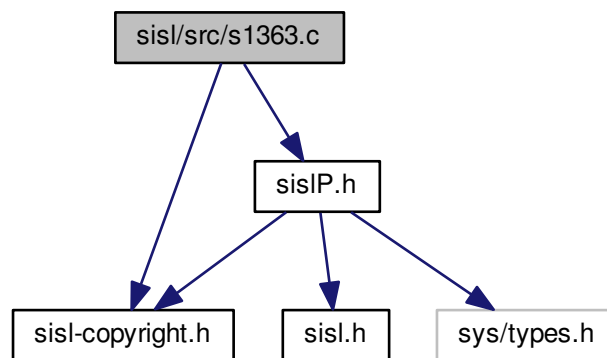
Definition at line 58 of file s1362.c.

## 30.2414 sisl/src/s1363.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1363.c:



### Macros

- #define [S1363](#)

### Functions

- void [s1363](#) (SISLCurve \*pc, double \*cmin, double \*cmax, int \*jstat)

### 30.2414.1 Macro Definition Documentation

#### 30.2414.1.1 #define S1363

Definition at line 49 of file s1363.c.

### 30.2414.2 Function Documentation

#### 30.2414.2.1 void s1363 ( SISLCurve \* pc, double \* cmin, double \* cmax, int \* jstat )

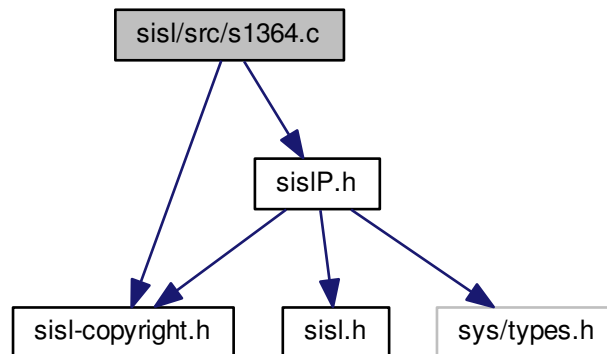
Definition at line 57 of file s1363.c.

## 30.2415 sisl/src/s1364.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1364.c:



### Macros

- #define [S1364](#)

### Functions

- void [s1364](#) (SISLCurve \*pc, double aepsge, int \*jstat)

### 30.2415.1 Macro Definition Documentation

#### 30.2415.1.1 #define S1364

Definition at line 49 of file s1364.c.

### 30.2415.2 Function Documentation

#### 30.2415.2.1 void s1364 ( SISLCurve \* pc, double aepsge, int \* jstat )

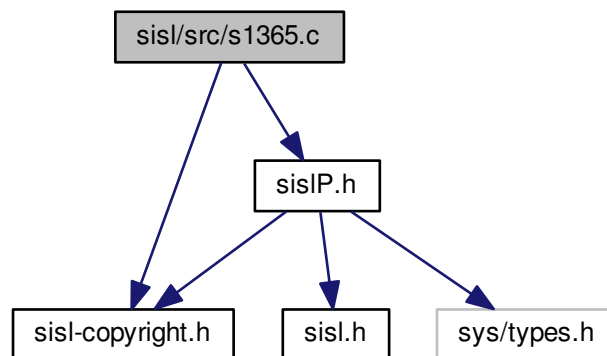
Definition at line 57 of file s1364.c.

## 30.2416 sisl/src/s1365.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1365.c:



### Macros

- #define [S1365](#)

### Functions

- void [s1365](#) (SISLSurf \*ps, double aoffset, double aepsge, double amax, int idim, SISLSurf \*\*rs, int \*jstat)



### 30.2416.1 Macro Definition Documentation

#### 30.2416.1.1 #define S1365

Definition at line 49 of file s1365.c.

### 30.2416.2 Function Documentation

#### 30.2416.2.1 void s1365 ( SISLSurf \* ps, double aoffset, double aeapsge, double amax, int idim, SISLSurf \*\* rs, int \* jstat )

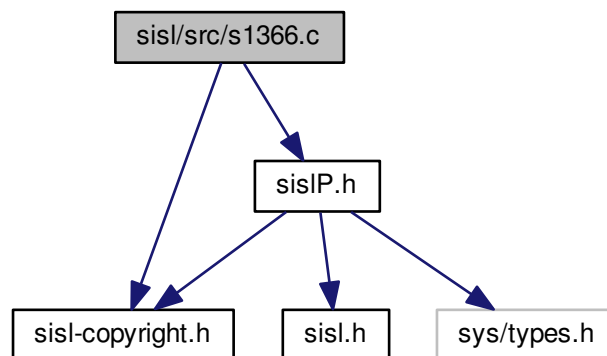
Definition at line 57 of file s1365.c.

## 30.2417 sisl/src/s1366.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1366.c:



### Macros

- #define [S1366](#)

### Functions

- void [s1366](#) (SISLSurf \*ps, double aoffset, double aeapsge, double amax, int idim, double \*eknot13, int in13, int ik13, double \*eknot24, int in24, int ik24, SISLSurf \*\*rs, int \*jstat)

### 30.2417.1 Macro Definition Documentation

#### 30.2417.1.1 #define S1366

Definition at line 49 of file s1366.c.

### 30.2417.2 Function Documentation

#### 30.2417.2.1 void s1366 ( SISLSurf \* ps, double aoffset, double aepsge, double amax, int idim, double \* eknot13, int in13, int ik13, double \* eknot24, int in24, int ik24, SISLSurf \*\* rs, int \* jstat )

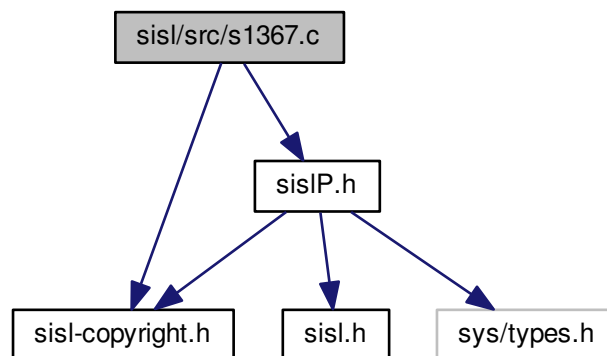
Definition at line 59 of file s1366.c.

## 30.2418 sisl/src/s1367.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1367.c:



### Macros

- #define [S1366](#)

### Functions

- void [s1367](#) (SISLSurf \*ps, double aoffset, double aepsge, int idim, epar, int ider, int \*ilfs, int \*ilft, eder, int \*jstat)

### 30.2418.1 Macro Definition Documentation

#### 30.2418.1.1 #define S1366

Definition at line 49 of file s1367.c.

### 30.2418.2 Function Documentation

#### 30.2418.2.1 void s1367 ( SISLSurf \* ps, double aoffset, double aepsge, int idim, epar, int nder, int \* ilfs, int \* ilft, eder, int \* jstat )

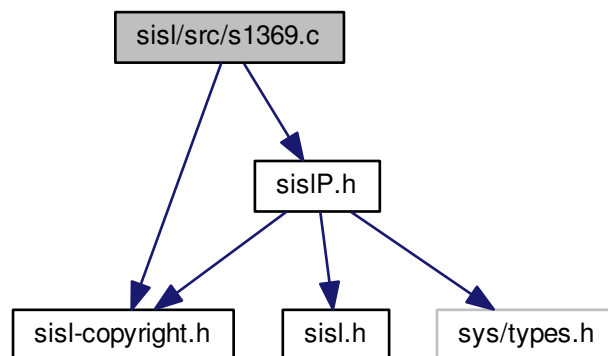
Definition at line 58 of file s1367.c.

## 30.2419 sisl/src/s1369.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1369.c:



### Macros

- #define [S1369](#)

### Functions

- void [s1369](#) (SISLSurf \*ps, ecentr, enorm, [double](#) abigr, [double](#) asmalr, int idim, [double](#) aepsco, [double](#) aepsge, int \*jpt, [double](#) \*\*gpar, int \*jcrv, [SISLIntcurve](#) \*\*\*wcurve, int \*jstat)

### 30.2419.1 Macro Definition Documentation

#### 30.2419.1.1 #define S1369

Definition at line 49 of file s1369.c.

### 30.2419.2 Function Documentation

#### 30.2419.2.1 void s1369 ( SISLSurf \* ps, ecentr, enorm, double abigr, double asmalr, int idim, double aepsco, double aepsge, int \* jpt, double \*\* gpar, int \* jcrv, SISLIntcurve \*\*\* wcurve, int \* jstat )

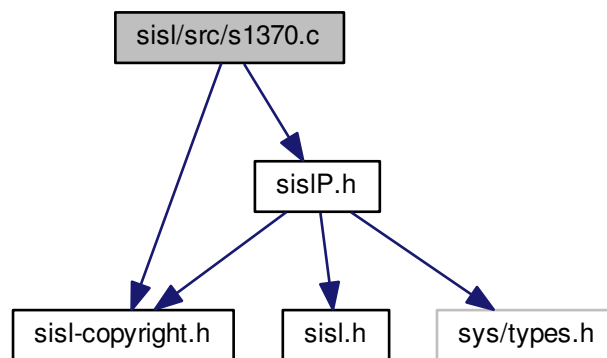
Definition at line 58 of file s1369.c.

## 30.2420 sisl/src/s1370.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1370.c:



### Macros

- #define [S1370](#)

### Functions

- void [s1370](#) ([SISLCurve](#) \*pcurv, earray, int idim, int inarr, int ratflag, [SISLCurve](#) \*\*rcurv, int \*jstat)

### 30.2420.1 Macro Definition Documentation

#### 30.2420.1.1 #define S1370

Definition at line 49 of file s1370.c.

### 30.2420.2 Function Documentation

#### 30.2420.2.1 void s1370 ( SISLCurve \* pcurv, earray , int idim, int inarr, int ratflag, SISLCurve \*\* rcurv, int \* jstat )

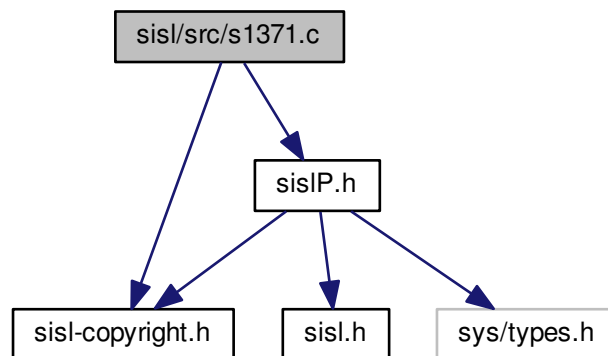
Definition at line 59 of file s1370.c.

## 30.2421 sisl/src/s1371.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1371.c:



### Macros

- #define [S1371](#)

### Functions

- void [s1371](#) (SISLCurve \*pc1, ecentr, double aradiu, int idim, double aepsco, double aepsge, int \*jpt, double \*\*gpar, int \*jcrv, SISLIntcurve \*\*\*wcurve, int \*jstat)

### 30.2421.1 Macro Definition Documentation

#### 30.2421.1.1 #define S1371

Definition at line 49 of file s1371.c.

### 30.2421.2 Function Documentation

#### 30.2421.2.1 void s1371 ( SISLCurve \* pc1, ecentr, double aradiu, int idim, double aepsco, double aepsge, int \* jpt, double \*\* gpar, int \* jcrv, SISLIntcurve \*\*\* wcurve, int \* jstat )

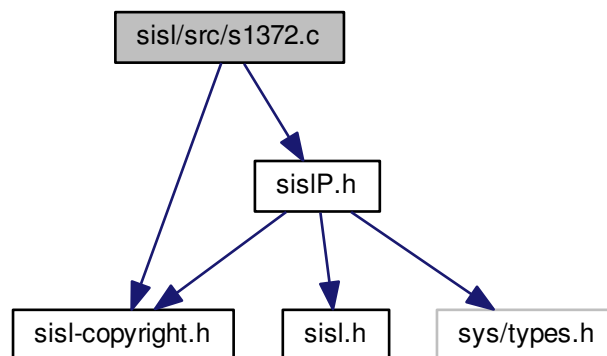
Definition at line 58 of file s1371.c.

## 30.2422 sisl/src/s1372.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1372.c:



### Macros

- #define [S1372](#)

### Functions

- void [s1372](#) (SISLCurve \*pc1, epoint, edirec, double aradiu, int idim, double aepsco, double aepsge, int \*jpt, double \*\*gpar, int \*jcrv, SISLIntcurve \*\*\*wcurve, int \*jstat)

### 30.2422.1 Macro Definition Documentation

#### 30.2422.1.1 #define S1372

Definition at line 49 of file s1372.c.

### 30.2422.2 Function Documentation

#### 30.2422.2.1 void s1372 ( SISLCurve \* pc1, epoint, edirec, double aradiu, int idim, double aepsco, double aepsge, int \* jpt, double \*\* gpar, int \* jcrv, SISLIntcurve \*\*\* wcurve, int \* jstat )

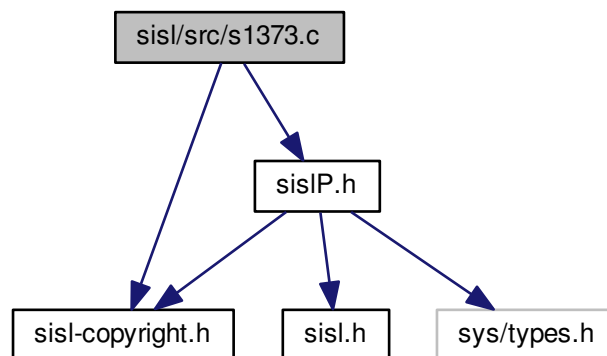
Definition at line 58 of file s1372.c.

## 30.2423 sisl/src/s1373.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1373.c:



### Macros

- #define [S1373](#)

### Functions

- void [s1373](#) (SISLCurve \*pc1, etop, eaxis, econ, int idim, double aepsco, double aepsge, int \*jpt, double \*\*gpar, int \*jcrv, SISLIntcurve \*\*\*wcurve, int \*jstat)

### 30.2423.1 Macro Definition Documentation

#### 30.2423.1.1 #define S1373

Definition at line 49 of file s1373.c.

### 30.2423.2 Function Documentation

#### 30.2423.2.1 void s1373 ( SISLCurve \* pc1, etop, eaxis, econe, int idim, double aepsco, double aepsge, int \* jpt, double \*\* gpar, int \* jcrv, SISLIntcurve \*\*\* wcurve, int \* jstat )

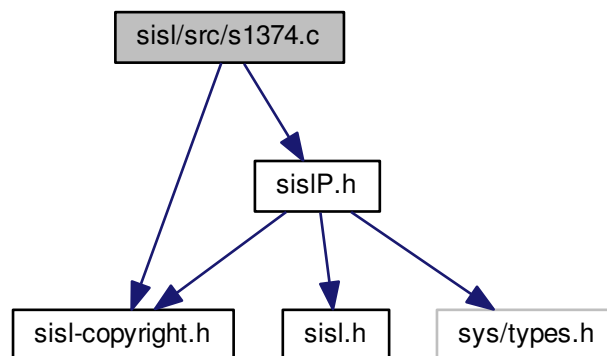
Definition at line 58 of file s1373.c.

## 30.2424 sisl/src/s1374.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1374.c:



### Macros

- #define [S1374](#)

### Functions

- void [s1374](#) (SISLCurve \*pc1, earray, int idim, [double](#) aepsco, [double](#) aepsge, int \*jpt, [double](#) \*\*gpar, int \*jcrv, SISLIntcurve \*\*\*wcurve, int \*jstat)



### 30.2424.1 Macro Definition Documentation

#### 30.2424.1.1 #define S1374

Definition at line 49 of file s1374.c.

### 30.2424.2 Function Documentation

#### 30.2424.2.1 void s1374 ( SISLCurve \* pc1, earray , int idim, double aepsco, double aepsge, int \* jpt, double \*\* gpar, int \* jcrv, SISLIntcurve \*\*\* wcurve, int \* jstat )

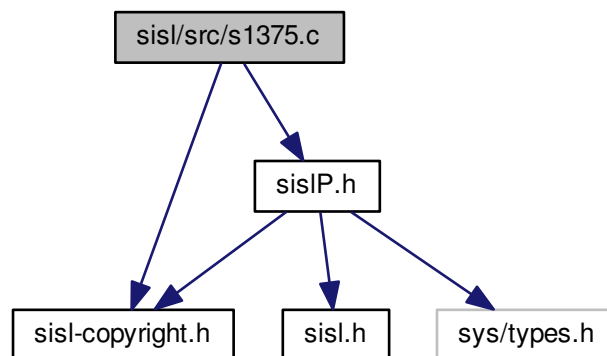
Definition at line 57 of file s1374.c.

## 30.2425 sisl/src/s1375.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1375.c:



### Macros

- #define [S1375](#)

### Functions

- void [s1375](#) (SISLCurve \*pc1, ecentr, enorm, double abigr, double asmalr, int idim, double aepsco, double aepsge, int \*jpt, double \*\*gpar, int \*jcrv, SISLIntcurve \*\*\*wcurve, int \*jstat)

### 30.2425.1 Macro Definition Documentation

#### 30.2425.1.1 #define S1375

Definition at line 49 of file s1375.c.

### 30.2425.2 Function Documentation

#### 30.2425.2.1 void s1375 ( SISLCurve \* *pc1*, *ecentr*, *enorm*, double *abigr*, double *asmalr*, int *idim*, double *aepsco*, double *aepsge*, int \* *jpt*, double \*\* *gpar*, int \* *jcrv*, SISLIntcurve \*\*\* *wcurve*, int \* *jstat* )

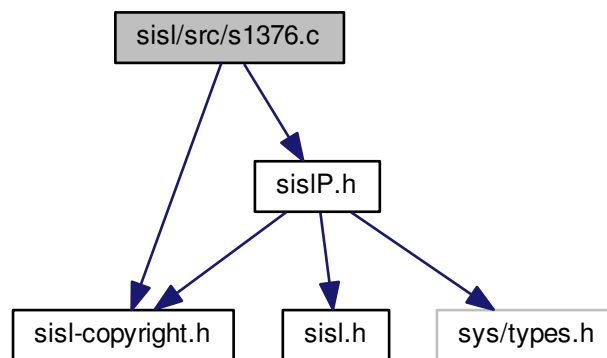
Definition at line 58 of file s1375.c.

## 30.2426 sisl/src/s1376.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1376.c:



### Macros

- #define [S1376](#)

### Functions

- void [s1376](#) (et, int in, int ik, double \*\*gt, int \*jkn, int \*jkk, int \*jstat)

## 30.2426.1 Macro Definition Documentation

### 30.2426.1.1 #define S1376

Definition at line 49 of file s1376.c.

## 30.2426.2 Function Documentation

### 30.2426.2.1 void s1376 ( et , int in, int ik, double \*\* gt, int \* jkn, int \* jkk, int \* jstat )

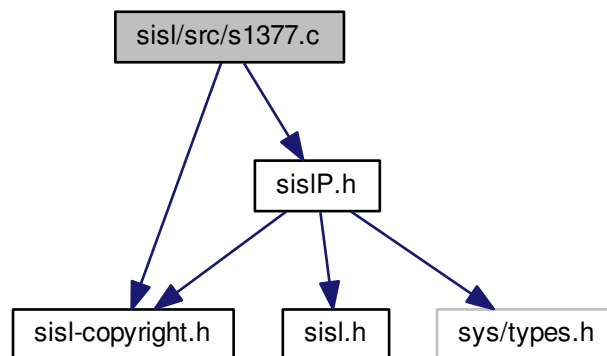
Definition at line 57 of file s1376.c.

## 30.2427 sisl/src/s1377.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1377.c:



## Macros

- #define [S1377](#)

## Functions

- void [s1377](#) ([SISLCurve](#) \*pcurv, econic, int ideg, int idim, [SISLCurve](#) \*\*rcurv, int \*jstat)

### 30.2427.1 Macro Definition Documentation

#### 30.2427.1.1 #define S1377

Definition at line 49 of file s1377.c.

### 30.2427.2 Function Documentation

#### 30.2427.2.1 void s1377 ( SISLCurve \* pcurv, econic , int ideg, int idim, SISLCurve \*\* rcurv, int \* jstat )

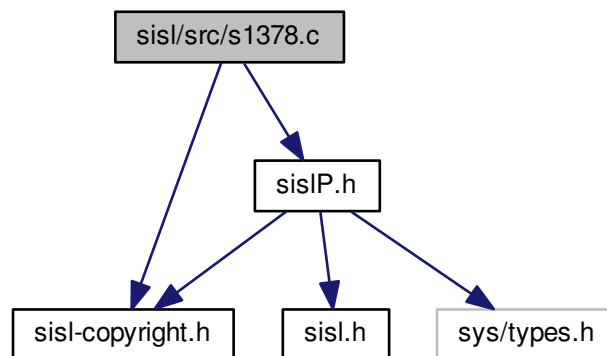
Definition at line 59 of file s1377.c.

## 30.2428 sisl/src/s1378.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1378.c:



### Macros

- #define [S1378](#)

### Functions

- void [s1378](#) (SISLSurf \*psurf, econic, int ideg, int idim, SISLSurf \*\*rsurf, int \*jstat)

## 30.2428.1 Macro Definition Documentation

### 30.2428.1.1 #define S1378

Definition at line 49 of file s1378.c.

## 30.2428.2 Function Documentation

### 30.2428.2.1 void s1378 ( SISLSurf \* *psurf*, *econic* , int *ideg*, int *idim*, SISLSurf \*\* *rsurf*, int \* *jstat* )

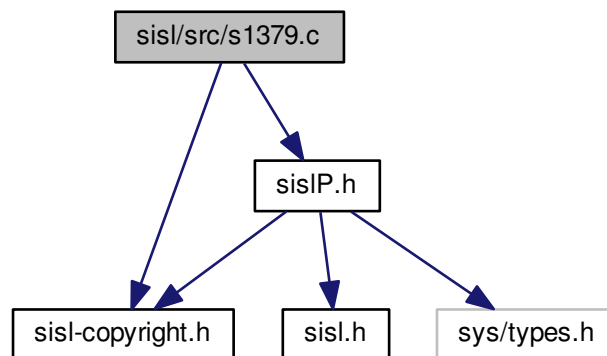
Definition at line 59 of file s1378.c.

## 30.2429 sisl/src/s1379.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1379.c:



## Macros

- #define [S1379](#)

## Functions

- void [s1379](#) (ep, ev, epar, int im, int idim, [SISLCurve](#) \*\*rcurve, int \*jstat)

## 30.2429.1 Macro Definition Documentation

### 30.2429.1.1 #define S1379

Definition at line 49 of file s1379.c.

## 30.2429.2 Function Documentation

### 30.2429.2.1 void s1379 ( ep , ev , epar , int im, int idim, SISLCurve \*\* rcurve, int \* jstat )

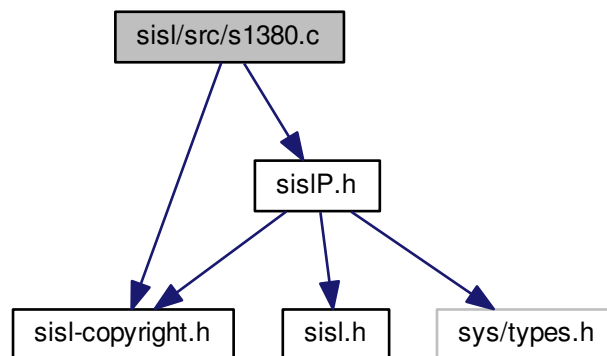
Definition at line 58 of file s1379.c.

## 30.2430 sisl/src/s1380.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1380.c:



## Macros

- #define [S1380](#)

## Functions

- void [s1380](#) (ep, ev, int im, int idim, int ipar, [SISLCurve](#) \*\*rcurve, int \*jstat)

## 30.2430.1 Macro Definition Documentation

### 30.2430.1.1 #define S1380

Definition at line 49 of file s1380.c.

## 30.2430.2 Function Documentation

### 30.2430.2.1 void s1380 ( ep , ev , int im , int idim , int ipar , SISLCurve \*\* rcurve , int \* jstat )

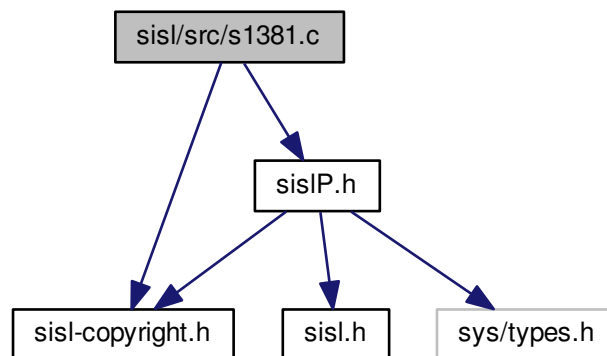
Definition at line 58 of file s1380.c.

## 30.2431 sisl/src/s1381.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1381.c:



## Macros

- #define [S1381](#)

## Functions

- void [s1381](#) (et, int in, int ik, [double](#) \*\*gt, int \*jkn, int jkk, int \*jstat)

### 30.2431.1 Macro Definition Documentation

#### 30.2431.1.1 #define S1381

Definition at line 49 of file s1381.c.

### 30.2431.2 Function Documentation

#### 30.2431.2.1 void s1381 ( et , int in, int ik, double \*\* gt, int \* jkn, int jkk, int \* jstat )

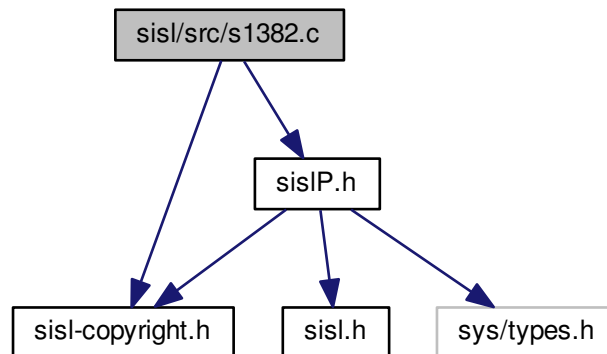
Definition at line 57 of file s1381.c.

## 30.2432 sisl/src/s1382.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1382.c:



### Macros

- #define [S1382](#)

### Functions

- void [s1382](#) ([SISLSurf](#) \*psurf, eview, int idim, [SISLSurf](#) \*\*rsurf, int \*jstat)



### 30.2432.1 Macro Definition Documentation

#### 30.2432.1.1 #define S1382

Definition at line 49 of file s1382.c.

### 30.2432.2 Function Documentation

#### 30.2432.2.1 void s1382 ( SISLSurf \* psurf, eview , int idim, SISLSurf \*\* rsurf, int \* jstat )

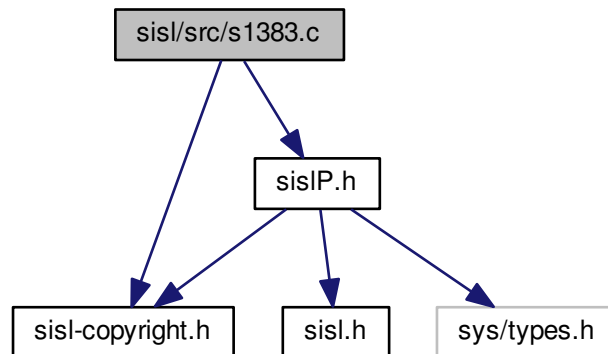
Definition at line 59 of file s1382.c.

## 30.2433 sisl/src/s1383.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1383.c:



### Macros

- #define [S1383](#)

### Functions

- void [s1383](#) (SISLSurf \*psurf, SISLCurve \*pcurv, double aepsge, double amax, int ider, SISLCurve \*\*rcpos, SISLCurve \*\*rcder1, SISLCurve \*\*rcder2, int \*jstat)

### 30.2433.1 Macro Definition Documentation

#### 30.2433.1.1 #define S1383

Definition at line 49 of file s1383.c.

### 30.2433.2 Function Documentation

#### 30.2433.2.1 void s1383 ( SISLSurf \* *psurf*, SISLCurve \* *pcurv*, double *aepsge*, double *amax*, int *ider*, SISLCurve \*\* *rcpos*, SISLCurve \*\* *rcder1*, SISLCurve \*\* *rcder2*, int \* *jstat* )

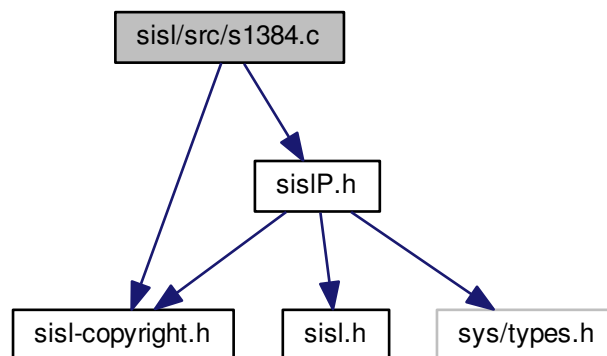
Definition at line 58 of file s1383.c.

## 30.2434 sisl/src/s1384.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1384.c:



### Macros

- #define [S1384](#)

### Functions

- void [s1384](#) (SISLCurve \**pcurve*, SISLSurf \**psurf*, int *idim*, int *iside*, double *ax*, int \**ileftc*, int \**ilefts1*, int \**ilefts2*, *eder*, *ederu*, *ederv*, *edern*, int \**jstat*)

### 30.2434.1 Macro Definition Documentation

#### 30.2434.1.1 #define S1384

Definition at line 49 of file s1384.c.

### 30.2434.2 Function Documentation

#### 30.2434.2.1 void s1384 ( SISLCurve \* pcurve, SISLSurf \* psurf, int idim, int iside, double ax, int \* ileftc, int \* ilefts1, int \* ilefts2, eder, ederu, ederv, edern, int \* jstat )

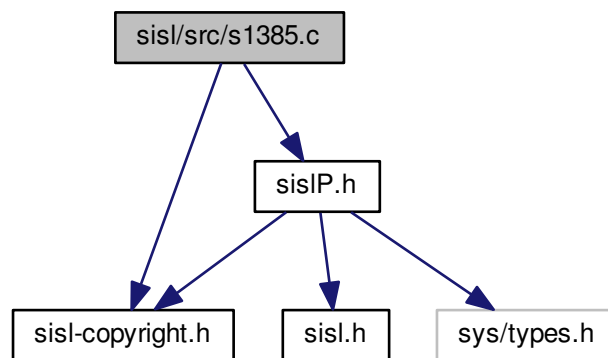
Definition at line 59 of file s1384.c.

## 30.2435 sisl/src/s1385.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1385.c:



### Macros

- #define [S1385](#)

### Functions

- void [s1385](#) (ep0, ept, ep1, double as, int idim, double aepsge, SISLCurve \*\*rc, int \*jstat)

### 30.2435.1 Macro Definition Documentation

#### 30.2435.1.1 #define S1385

Definition at line 49 of file s1385.c.

### 30.2435.2 Function Documentation

#### 30.2435.2.1 void s1385 ( ep0 , ept , ep1 , double as, int idim, double aepsge, SISLCurve \*\* rc, int \* jstat )

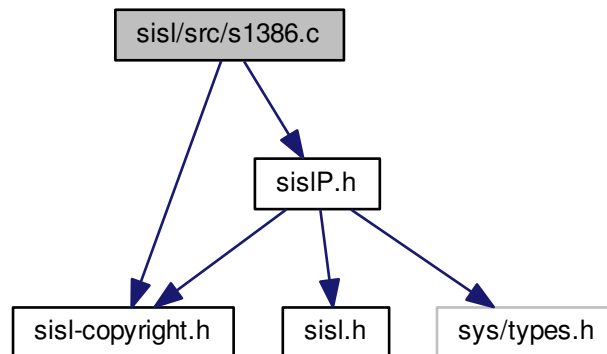
Definition at line 58 of file s1385.c.

## 30.2436 sisl/src/s1386.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1386.c:



### Macros

- #define [S1386](#)

### Functions

- void [s1386](#) ([SISLSurf](#) \*ps, int ider1, int ider2, [SISLSurf](#) \*\*rsnew, int \*jstat)

## 30.2436.1 Macro Definition Documentation

### 30.2436.1.1 #define S1386

Definition at line 49 of file s1386.c.

## 30.2436.2 Function Documentation

### 30.2436.2.1 void s1386 ( SISLSurf \* ps, int nder1, int nder2, SISLSurf \*\* rsnew, int \* jstat )

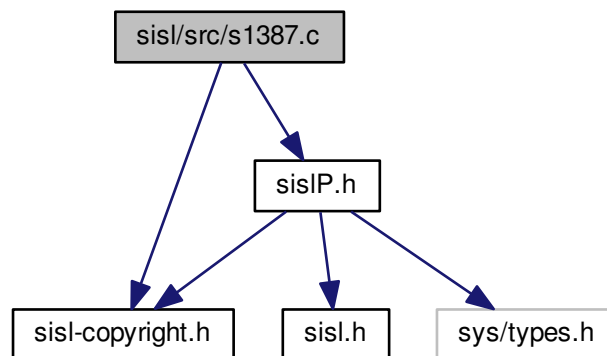
Definition at line 57 of file s1386.c.

## 30.2437 sisl/src/s1387.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1387.c:



## Macros

- #define [S1387](#)

## Functions

- void [s1387](#) (SISLSurf \*ps, int ik1, int ik2, SISLSurf \*\*rsnew, int \*jstat)

### 30.2437.1 Macro Definition Documentation

#### 30.2437.1.1 #define S1387

Definition at line 49 of file s1387.c.

### 30.2437.2 Function Documentation

#### 30.2437.2.1 void s1387 ( SISLSurf \* ps, int ik1, int ik2, SISLSurf \*\* rsnew, int \* jstat )

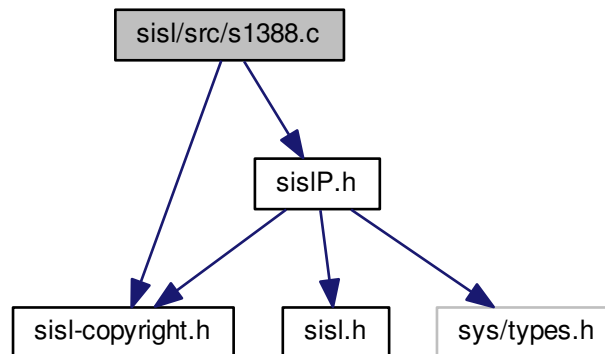
Definition at line 57 of file s1387.c.

## 30.2438 sisl/src/s1388.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1388.c:



### Macros

- #define [S1388](#)

### Functions

- void [s1388](#) (SISLSurf \*ps1, gcoons, int \*jnumb1, int \*jnumb2, int \*jdim, int \*jstat)

## 30.2438.1 Macro Definition Documentation

### 30.2438.1.1 #define S1388

Definition at line 49 of file s1388.c.

## 30.2438.2 Function Documentation

### 30.2438.2.1 void s1388 ( SISLSurf \* ps1, gcoons , int \* jnumb1, int \* jnumb2, int \* jdim, int \* jstat )

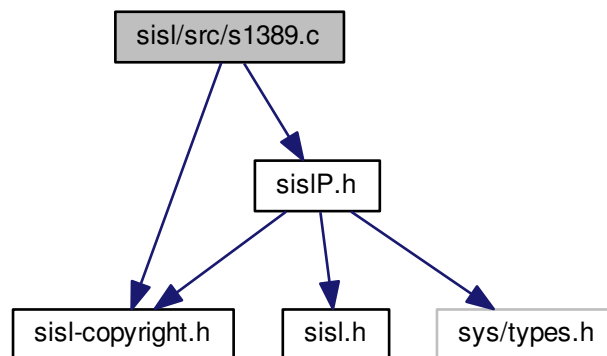
Definition at line 57 of file s1388.c.

## 30.2439 sisl/src/s1389.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1389.c:



## Macros

- #define [S1389](#)

## Functions

- void [s1389](#) ([SISLCurve](#) \*pc1, gcubic, int \*jnumb, int \*jdim, int \*jstat)

## 30.2439.1 Macro Definition Documentation

### 30.2439.1.1 #define S1389

Definition at line 49 of file s1389.c.

## 30.2439.2 Function Documentation

### 30.2439.2.1 void s1389 ( SISLCurve \* pc1, gcubic , int \* jnumb, int \* jdim, int \* jstat )

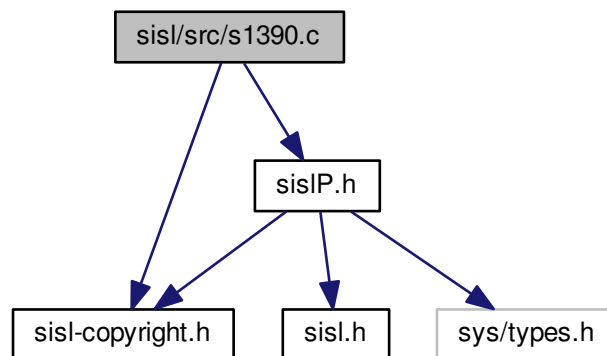
Definition at line 57 of file s1389.c.

## 30.2440 sisl/src/s1390.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1390.c:



## Macros

- #define [S1390](#)

## Functions

- void [s1390](#) (pc1, [SISLSurf](#) \*\*ps1, [nder](#), int \*jstat)



### 30.2440.1 Macro Definition Documentation

#### 30.2440.1.1 #define S1390

Definition at line 49 of file s1390.c.

### 30.2440.2 Function Documentation

#### 30.2440.2.1 void s1390 ( pc1 , SISLSurf \*\* ps1, nder , int \* jstat )

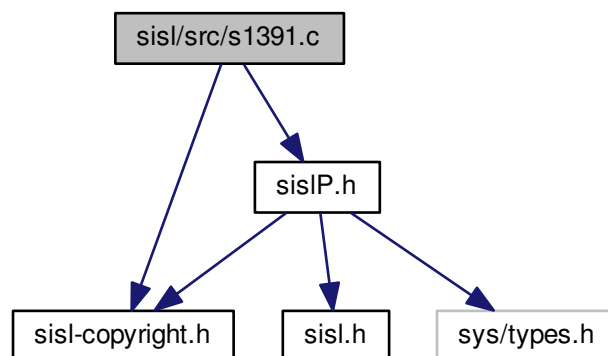
Definition at line 58 of file s1390.c.

## 30.2441 sisl/src/s1391.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1391.c:



### Macros

- #define [S1391](#)

### Typedefs

- typedef void(\* [fshapeProc](#)) ()

### Functions

- void [s1391](#) ([SISLCurve](#) \*\*pc, [SISLSurf](#) \*\*\*ws, int icurv, nder, int \*jstat)

### 30.2441.1 Macro Definition Documentation

#### 30.2441.1.1 #define S1391

Definition at line 49 of file s1391.c.

### 30.2441.2 Typedef Documentation

#### 30.2441.2.1 typedef void(\* fshapeProc) ()

Definition at line 53 of file s1391.c.

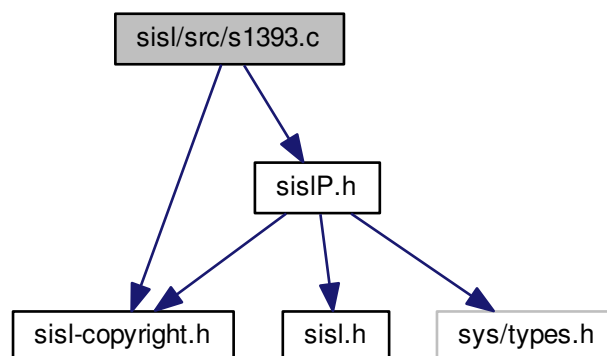
### 30.2441.3 Function Documentation

#### 30.2441.3.1 void s1391 ( SISLCurve \*\* pc, SISLSurf \*\*\* ws, int icurv, nder , int \* jstat )

Definition at line 68 of file s1391.c.

## 30.2442 sisl/src/s1393.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for s1393.c:
```



## Macros

- #define [S1393](#)

## Functions

- void [s1393](#) (int n1, pc1, sc1, ec1, int \*jstat)

### 30.2442.1 Macro Definition Documentation

#### 30.2442.1.1 #define S1393

Definition at line 49 of file s1393.c.

### 30.2442.2 Function Documentation

#### 30.2442.2.1 void s1393 ( int n1, pc1 , sc1 , ec1 , int \* jstat )

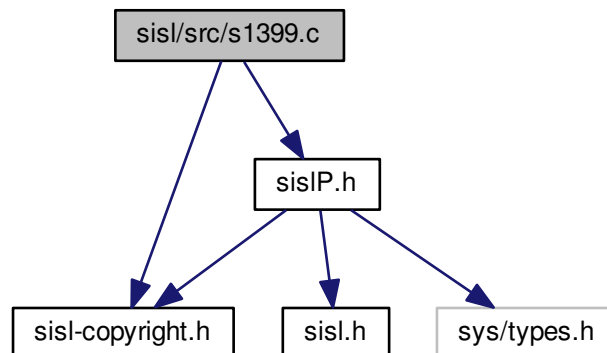
Definition at line 57 of file s1393.c.

## 30.2443 sisl/src/s1399.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1399.c:



## Macros

- #define [S1399](#)

## Functions

- void [s1399](#) (SISLCurve \*pc, double astart, double astop)

### 30.2443.1 Macro Definition Documentation

#### 30.2443.1.1 #define S1399

Definition at line 49 of file s1399.c.

### 30.2443.2 Function Documentation

#### 30.2443.2.1 void s1399 ( SISLCurve \* pc, double astart, double astop )

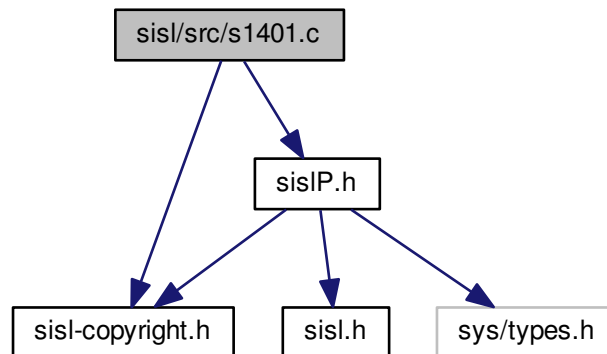
Definition at line 57 of file s1399.c.

## 30.2444 sisl/src/s1401.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1401.c:



### Macros

- #define [S1401](#)

### Functions

- void [s1401](#) (vcurve, etwist, [SISLSurf](#) \*\*rsurf, int \*jstat)

## 30.2444.1 Macro Definition Documentation

### 30.2444.1.1 #define S1401

Definition at line 49 of file s1401.c.

## 30.2444.2 Function Documentation

### 30.2444.2.1 void s1401 ( vcurve , etwist , SISLSurf \*\* rsurf, int \* jstat )

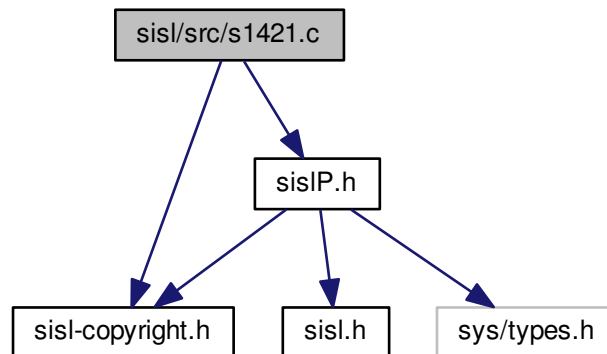
Definition at line 70 of file s1401.c.

## 30.2445 sisl/src/s1421.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1421.c:



## Macros

- #define [S1421](#)

## Functions

- void [s1421](#) ([SISLSurf](#) \*ps1, int ider, epar, int \*ilfs, int \*ilft, eder, enorm, int \*jstat)

## 30.2445.1 Macro Definition Documentation

### 30.2445.1.1 #define S1421

Definition at line 47 of file s1421.c.

## 30.2445.2 Function Documentation

### 30.2445.2.1 void s1421 ( SISLSurf \*ps1, int ider, epar , int \*ilfs, int \*ilft, eder , enorm , int \*jstat )

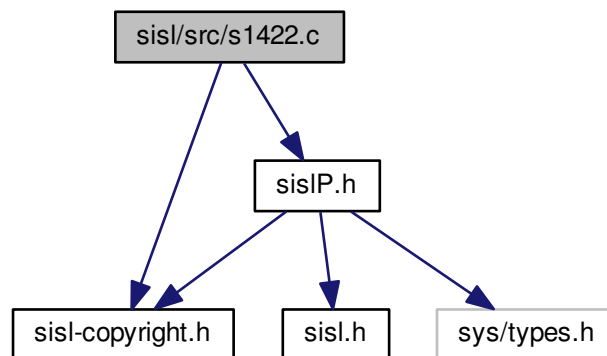
Definition at line 56 of file s1421.c.

## 30.2446 sisl/src/s1422.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1422.c:



## Macros

- #define [S1422](#)

## Functions

- void [s1422](#) (SISLSurf \*ps1, int ider, int iside1, int iside2, epar, int \*ilfs, int \*ilft, eder, enorm, int \*jstat)

## 30.2446.1 Macro Definition Documentation

### 30.2446.1.1 #define S1422

Definition at line 49 of file s1422.c.

## 30.2446.2 Function Documentation

### 30.2446.2.1 void s1422 ( SISLSurf \*ps1, int ider, int iside1, int iside2, epar , int \*ilfs, int \*ilft, eder , enorm , int \*jstat )

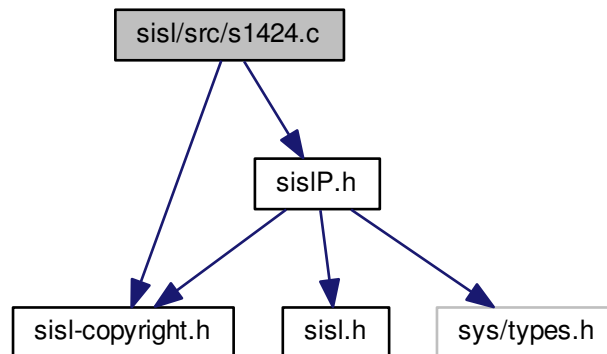
Definition at line 58 of file s1422.c.

## 30.2447 sisl/src/s1424.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1424.c:



## Macros

- #define [S1424](#)

## Functions

- void [s1424](#) (SISLSurf \*ps1, int ider1, int ider2, epar, int \*ileft1, int \*ileft2, eder, int \*jstat)

### 30.2447.1 Macro Definition Documentation

#### 30.2447.1.1 #define S1424

Definition at line 49 of file s1424.c.

### 30.2447.2 Function Documentation

#### 30.2447.2.1 void s1424 ( SISLSurf \*ps1, int ider1, int ider2, epar , int \*ileft1, int \*ileft2, eder , int \*jstat )

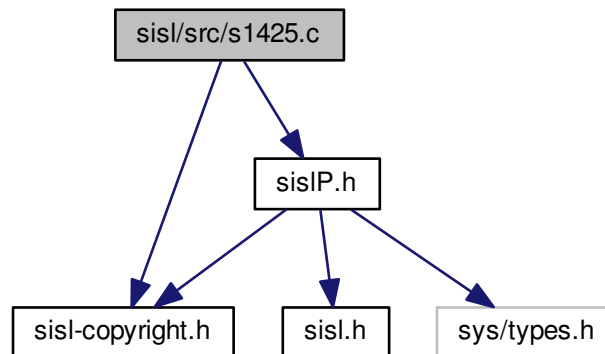
Definition at line 58 of file s1424.c.

## 30.2448 sisl/src/s1425.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1425.c:



### Macros

- #define [S1425](#)

### Functions

- void [s1425](#) (SISLSurf \*ps1, int ider1, int ider2, int iside1, int iside2, epar, int \*ileft1, int \*ileft2, eder, int \*jstat)



## 30.2448.1 Macro Definition Documentation

### 30.2448.1.1 #define S1425

Definition at line 49 of file s1425.c.

## 30.2448.2 Function Documentation

### 30.2448.2.1 void s1425 ( SISLSurf \* ps1, int nder1, int nder2, int iside1, int iside2, epar , int \* ileft1, int \* ileft2, nder , int \* jstat )

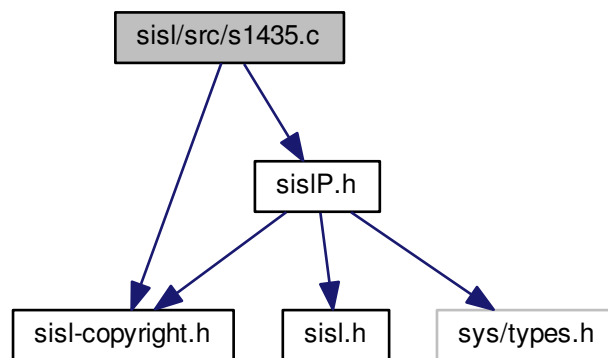
Definition at line 58 of file s1425.c.

## 30.2449 sisl/src/s1435.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1435.c:



## Macros

- #define [S1435](#)

## Functions

- void [s1435](#) (SISLSurf \*ps1, int nder1, SISLCurve \*\*rledge, double \*cpar, int \*jstat)

## 30.2449.1 Macro Definition Documentation

### 30.2449.1.1 #define S1435

Definition at line 49 of file s1435.c.

## 30.2449.2 Function Documentation

### 30.2449.2.1 void s1435 ( SISLSurf \* ps1, int iedge, SISLCurve \*\* rcedge, double \* cpar, int \* jstat )

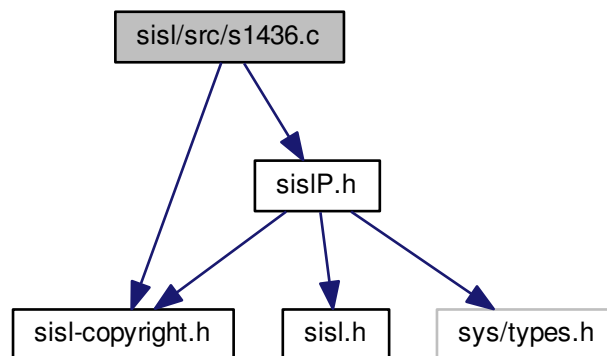
Definition at line 57 of file s1435.c.

## 30.2450 sisl/src/s1436.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1436.c:



## Macros

- #define [S1436](#)

## Functions

- void [s1436](#) (SISLSurf \*ps1, double apar, SISLCurve \*\*rcurve, int \*jstat)

## 30.2450.1 Macro Definition Documentation

### 30.2450.1.1 #define S1436

Definition at line 49 of file s1436.c.

## 30.2450.2 Function Documentation

### 30.2450.2.1 void s1436 ( SISLSurf \* ps1, double apar, SISLCurve \*\* rcurve, int \* jstat )

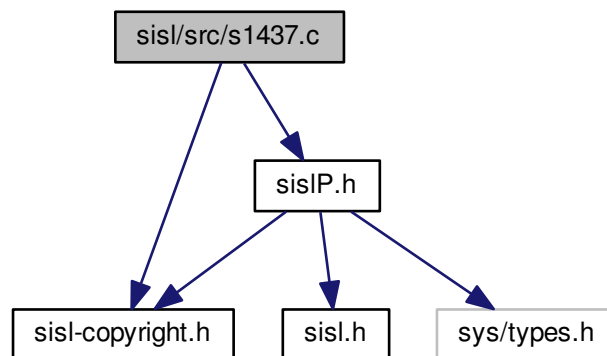
Definition at line 57 of file s1436.c.

## 30.2451 sisl/src/s1437.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1437.c:



## Macros

- #define [S1437](#)

## Functions

- void [s1437](#) (SISLSurf \*ps1, double apar, SISLCurve \*\*rcurve, int \*jstat)

### 30.2451.1 Macro Definition Documentation

#### 30.2451.1.1 #define S1437

Definition at line 49 of file s1437.c.

### 30.2451.2 Function Documentation

#### 30.2451.2.1 void s1437 ( SISLSurf \* ps1, double apar, SISLCurve \*\* rcurve, int \* jstat )

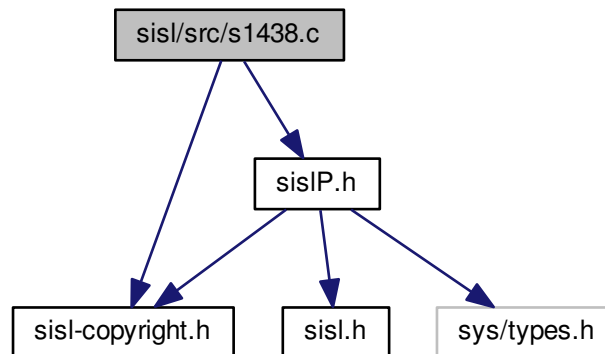
Definition at line 57 of file s1437.c.

## 30.2452 sisl/src/s1438.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1438.c:



### Macros

- #define [S1438](#)

### Functions

- void [s1438](#) (SISLCurve \*pc, int iedge, SISLPoint \*\*rpedge, double \*cpar, int \*jstat)

### 30.2452.1 Macro Definition Documentation

#### 30.2452.1.1 #define S1438

Definition at line 49 of file s1438.c.

### 30.2452.2 Function Documentation

#### 30.2452.2.1 void s1438 ( SISLCurve \* pc, int iedge, SISLPoint \*\* rpedge, double \* cpar, int \* jstat )

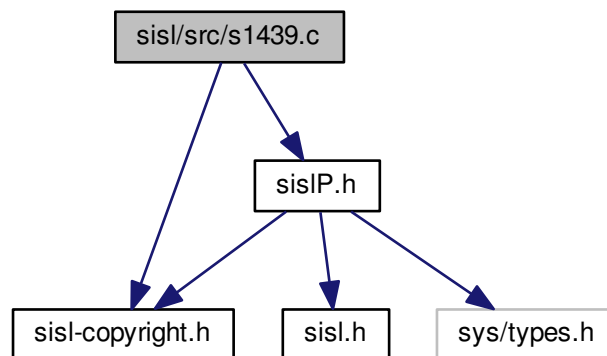
Definition at line 57 of file s1438.c.

## 30.2453 sisl/src/s1439.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1439.c:



### Macros

- #define [S1439](#)

### Functions

- void [s1439](#) (SISLSurf \*ps1, double apar, int idirec, SISLCurve \*\*rcurve, int \*jstat)

### 30.2453.1 Macro Definition Documentation

#### 30.2453.1.1 #define S1439

Definition at line 49 of file s1439.c.

### 30.2453.2 Function Documentation

#### 30.2453.2.1 void s1439 ( SISLSurf \* ps1, double apar, int idirec, SISLCurve \*\* rcurve, int \* jstat )

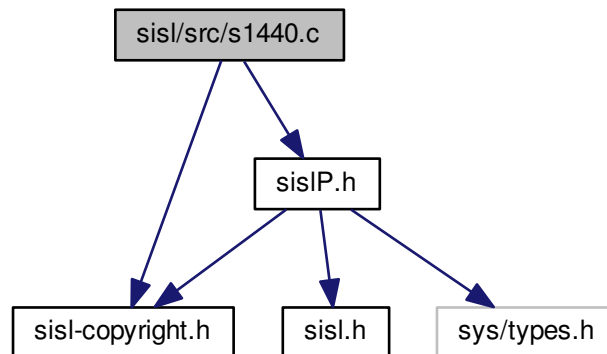
Definition at line 56 of file s1439.c.

## 30.2454 sisl/src/s1440.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1440.c:



### Macros

- #define [S1440](#)

### Functions

- void [s1440](#) (SISLSurf \*ps1, SISLSurf \*\*rs2, int \*jstat)

### 30.2454.1 Macro Definition Documentation

#### 30.2454.1.1 #define S1440

Definition at line 49 of file s1440.c.

### 30.2454.2 Function Documentation

#### 30.2454.2.1 void s1440 ( SISLSurf \* ps1, SISLSurf \*\* rs2, int \* jstat )

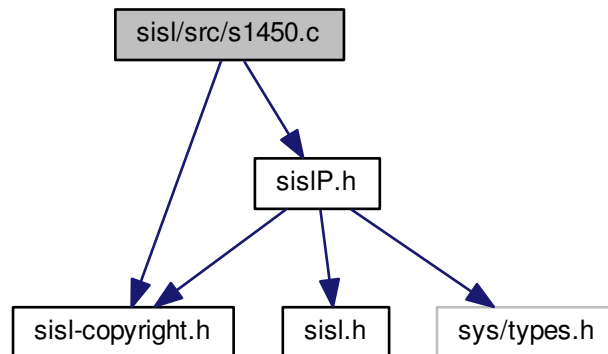
Definition at line 57 of file s1440.c.

## 30.2455 sisl/src/s1450.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1450.c:



### Macros

- #define [S1450](#)

### Functions

- void [s1450](#) (SISLSurf \*ps1, double aepsge, int \*jclos1, int \*jclos2, int \*jdgen1, int \*jdgen2, int \*jdgen3, int \*jdgen4, int \*jstat)

### 30.2455.1 Macro Definition Documentation

#### 30.2455.1.1 #define S1450

Definition at line 49 of file s1450.c.

### 30.2455.2 Function Documentation

#### 30.2455.2.1 void s1450 ( SISLSurf \* ps1, double aepsge, int \* jclos1, int \* jclos2, int \* jdgen1, int \* jdgen2, int \* jdgen3, int \* jdgen4, int \* jstat )

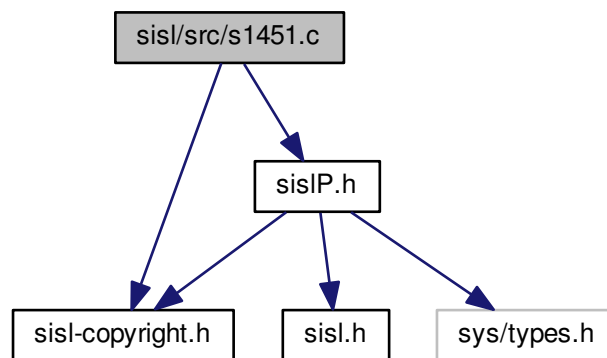
Definition at line 58 of file s1450.c.

## 30.2456 sisl/src/s1451.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1451.c:



### Macros

- #define [S1451](#)

### Functions

- void [s1451](#) (SISLCurve \*pc1, double aepsge, int \*jdgen, int \*jstat)



## 30.2456.1 Macro Definition Documentation

### 30.2456.1.1 #define S1451

Definition at line 49 of file s1451.c.

## 30.2456.2 Function Documentation

### 30.2456.2.1 void s1451 ( SISLCurve \* pc1, double aepsge, int \* jdgen, int \* jstat )

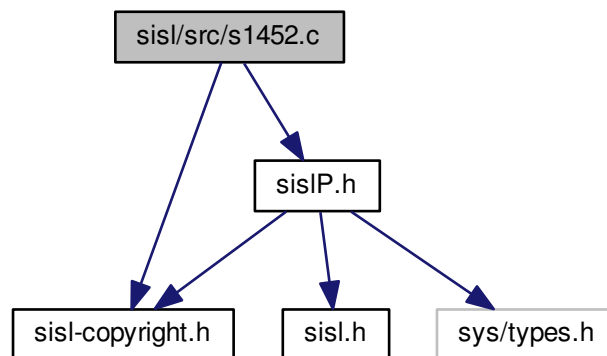
Definition at line 57 of file s1451.c.

## 30.2457 sisl/src/s1452.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1452.c:



## Macros

- #define [S1452](#)

## Functions

- void [s1452](#) (SISLSurf \*ps, double aepsge, double aoffset, SISLSurf \*\*rs, int \*jstat)

### 30.2457.1 Macro Definition Documentation

#### 30.2457.1.1 #define S1452

Definition at line 42 of file s1452.c.

### 30.2457.2 Function Documentation

#### 30.2457.2.1 void s1452 ( SISLSurf \* ps, double aepsge, double aoffset, SISLSurf \*\* rs, int \* jstat )

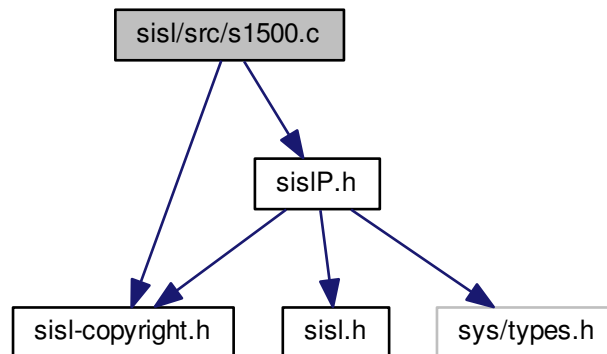
Definition at line 50 of file s1452.c.

## 30.2458 sisl/src/s1500.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1500.c:



### Macros

- #define [S1500](#)

### Functions

- void [s1500](#) (base, norm, axisA, double alpha, double ratio, int idim, int inumb, carray, int \*jstat)

### 30.2458.1 Macro Definition Documentation

#### 30.2458.1.1 #define S1500

Definition at line 49 of file s1500.c.

### 30.2458.2 Function Documentation

#### 30.2458.2.1 void s1500 ( base , norm , axisA , double alpha, double ratio, int idim, int inumb, carray , int \* jstat )

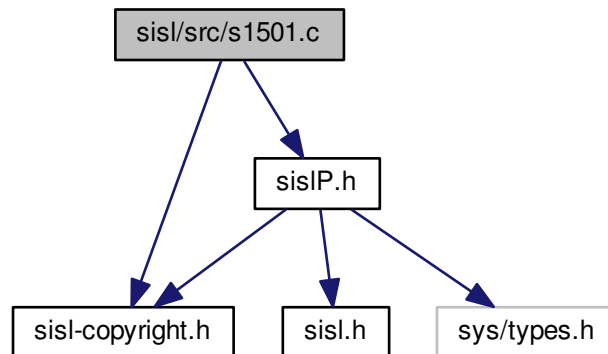
Definition at line 58 of file s1500.c.

## 30.2459 sisl/src/s1501.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1501.c:



### Macros

- #define [S1501](#)

### Functions

- void [s1501](#) ([SISLSurf](#) \*ps1, [double](#) \*base, [double](#) \*norm, [double](#) \*axisA, [double](#) alpha, [double](#) ratio, int idim, [double](#) aeprco, [double](#) aeprge, [double](#) amax, [SISLIntcurve](#) \*pinctr, int icur, int igrph, int \*jstat)

### 30.2459.1 Macro Definition Documentation

#### 30.2459.1.1 #define S1501

Definition at line 49 of file s1501.c.

### 30.2459.2 Function Documentation

30.2459.2.1 void s1501 ( SISLSurf \* ps1, double \* base, double \* norm, double \* axisA, double alpha, double ratio, int idim, double aepsco, double aepsge, double amax, SISLIntcurve \* pintcr, int icur, int igrph, int \* jstat )

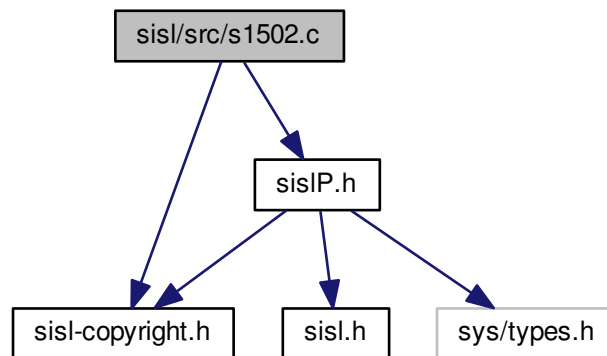
Definition at line 59 of file s1501.c.

## 30.2460 sisl/src/s1502.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1502.c:



### Macros

- #define [S1502](#)

### Functions

- void [s1502](#) (SISLCurve \*pc1, base, norm, axisA, alpha, ratio, int idim, double aepsco, double aepsge, int \*jpt, double \*\*gpar, int \*jcrv, SISLIntcurve \*\*\*wcurve, int \*jstat)

## 30.2460.1 Macro Definition Documentation

### 30.2460.1.1 #define S1502

Definition at line 49 of file s1502.c.

## 30.2460.2 Function Documentation

30.2460.2.1 void s1502 ( SISLCurve \* *pc1*, base, norm, axisA, alpha, ratio, int *idim*, double *aepsco*, double *aepsge*, int \* *jpt*, double \*\* *gpar*, int \* *jcrv*, SISLIntcurve \*\*\* *wcurve*, int \* *jstat* )

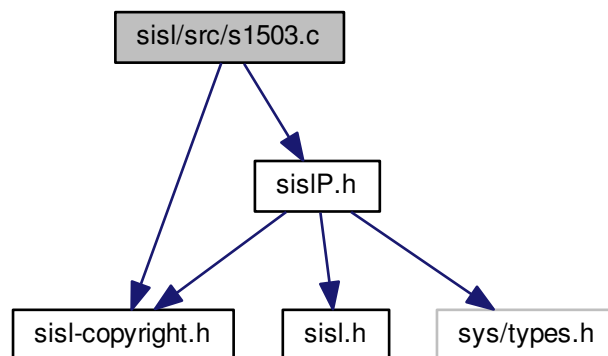
Definition at line 58 of file s1502.c.

## 30.2461 sisl/src/s1503.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1503.c:



## Macros

- #define [S1503](#)

## Functions

- void [s1503](#) (SISLSurf \**ps1*, base, norm, axisA, double *alpha*, double *ratio*, int *idim*, double *aepsco*, double *aepsge*, int \**jpt*, double \*\**gpar*, int \**jcrv*, SISLIntcurve \*\*\**wcurve*, int \**jstat*)

### 30.2461.1 Macro Definition Documentation

#### 30.2461.1.1 #define S1503

Definition at line 49 of file s1503.c.

### 30.2461.2 Function Documentation

#### 30.2461.2.1 void s1503 ( SISLSurf \* ps1, base , norm , axisA , double alpha, double ratio, int idim, double aepsco, double aepsge, int \* jpt, double \*\* gpar, int \* jcrv, SISLIntcurve \*\*\* wcurve, int \* jstat )

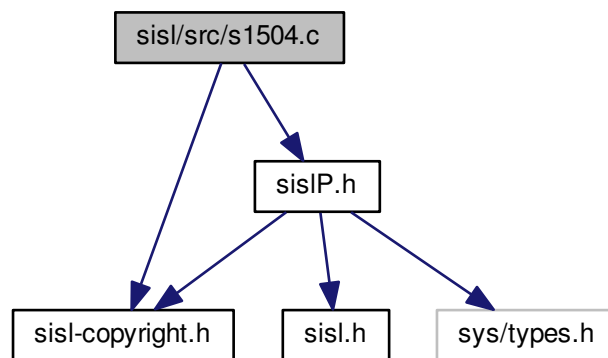
Definition at line 58 of file s1503.c.

## 30.2462 sisl/src/s1504.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1504.c:



### Macros

- #define [S1504](#)

### Functions

- void [s1504](#) (double \*et, int ik, int in, double \*ax, int im, int ider, ebder, ileft, int \*jstat)

### 30.2462.1 Macro Definition Documentation

#### 30.2462.1.1 #define S1504

Definition at line 49 of file s1504.c.

### 30.2462.2 Function Documentation

#### 30.2462.2.1 void s1504 ( double \* et, int ik, int in, double \* ax, int im, int ider, ebder, ileft, int \* jstat )

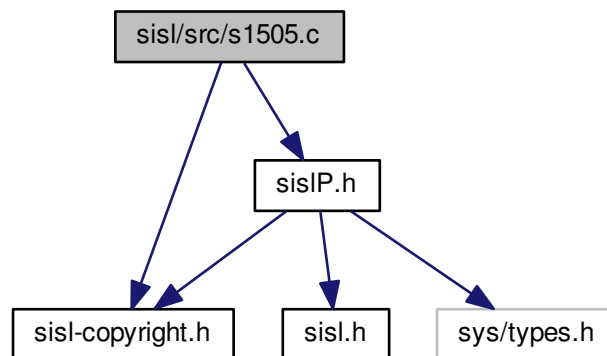
Definition at line 59 of file s1504.c.

## 30.2463 sisl/src/s1505.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1505.c:



### Macros

- #define [S1505](#)

### Functions

- void [s1505](#) (SISLSurf \*ps1, int ider, int m1, int m2, double \*ebder1, double \*ebder2, int \*ileft1, int \*ileft2, eder, norm, int \*jstat)

### 30.2463.1 Macro Definition Documentation

#### 30.2463.1.1 #define S1505

Definition at line 47 of file s1505.c.

### 30.2463.2 Function Documentation

#### 30.2463.2.1 void s1505 ( SISLSurf \* ps1, int ider, int m1, int m2, double \* ebder1, double \* ebder2, int \* ileft1, int \* ileft2, eder, norm, int \* jstat )

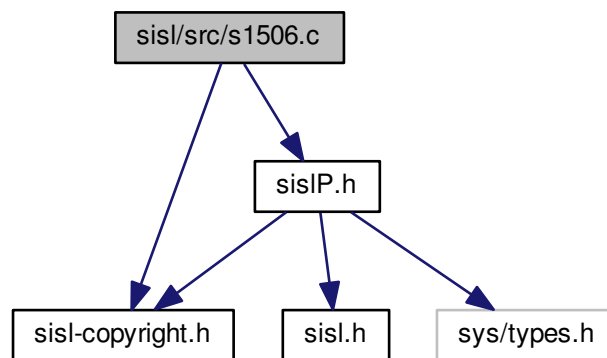
Definition at line 56 of file s1505.c.

## 30.2464 sisl/src/s1506.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1506.c:



### Macros

- #define [S1506](#)

### Functions

- void [s1506](#) (SISLSurf \*ps1, int ider, int m1, double \*x, int m2, double \*y, eder, norm, int \*jstat)



## 30.2464.1 Macro Definition Documentation

### 30.2464.1.1 #define S1506

Definition at line 47 of file s1506.c.

## 30.2464.2 Function Documentation

### 30.2464.2.1 void s1506 ( SISLSurf \* ps1, int nder, int m1, double \* x, int m2, double \* y, eder , norm , int \* jstat )

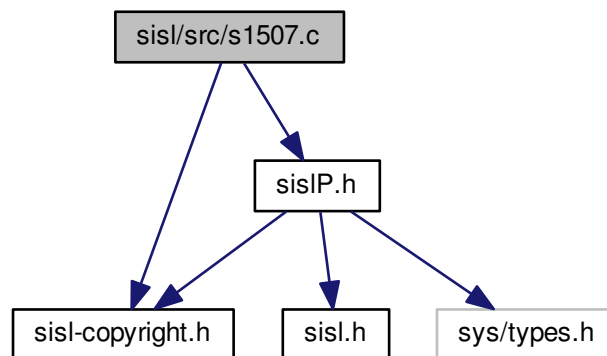
Definition at line 56 of file s1506.c.

## 30.2465 sisl/src/s1507.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1507.c:



## Macros

- #define [S1507](#)

## Functions

- void [s1507](#) ([SISLCurve](#) \*\*curves, int nc, int periodic, [SISLCurve](#) \*\*\*newcurves, int \*jstat)

## 30.2465.1 Macro Definition Documentation

### 30.2465.1.1 #define S1507

Definition at line 49 of file s1507.c.

## 30.2465.2 Function Documentation

### 30.2465.2.1 void s1507 ( SISLCurve \*\* curves, int nc, int periodic, SISLCurve \*\*\* newcurves, int \* jstat )

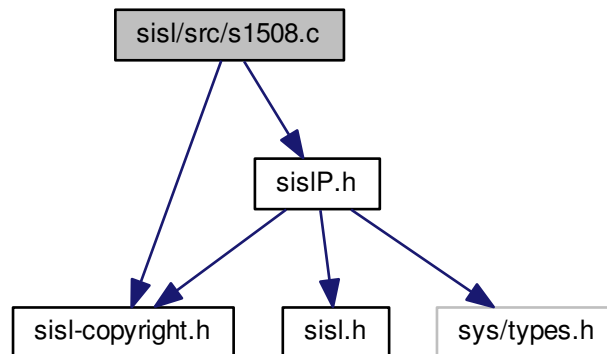
Definition at line 57 of file s1507.c.

## 30.2466 sisl/src/s1508.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1508.c:



## Macros

- #define [S1508](#)

## Functions

- void [s1508](#) (int inbcrv, [SISLCurve](#) \*\*vpcurv, par\_arr, [SISLSurf](#) \*\*rsurf, int \*jstat)

### 30.2466.1 Macro Definition Documentation

#### 30.2466.1.1 #define S1508

Definition at line 49 of file s1508.c.

### 30.2466.2 Function Documentation

#### 30.2466.2.1 void s1508 ( int *inbcrv*, SISLCurve \*\* *vpcurv*, par\_arr , SISLSurf \*\* *rsurf*, int \* *jstat* )

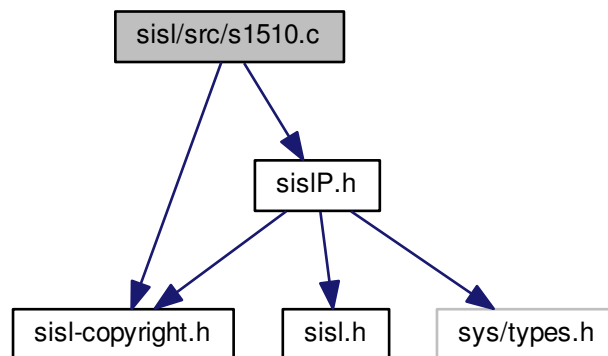
Definition at line 59 of file s1508.c.

## 30.2467 sisl/src/s1510.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1510.c:



### Macros

- #define [S1510](#)

### Functions

- void [s1510](#) (SISLSurf \*ps, eyepoint, int idim, double aepsco, double aepsge, int \*jpt, double \*\*gpar, int \*jcrv, SISLIntcurve \*\*\*wcurve, int \*jstat)

### 30.2467.1 Macro Definition Documentation

#### 30.2467.1.1 #define S1510

Definition at line 49 of file s1510.c.

### 30.2467.2 Function Documentation

#### 30.2467.2.1 void s1510 ( SISLSurf \* ps, eyepoint , int idim, double aepsco, double aepsge, int \* jpt, double \*\* gpar, int \* jcrv, SISLIntcurve \*\*\* wcurve, int \* jstat )

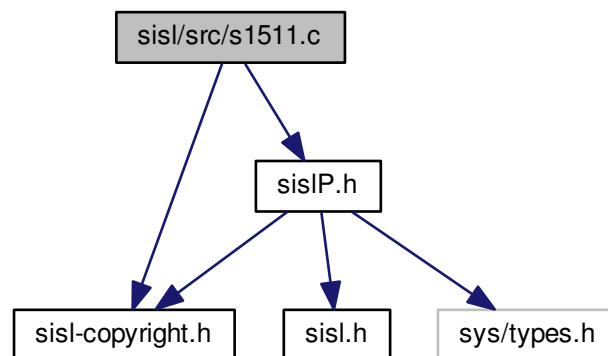
Definition at line 58 of file s1510.c.

## 30.2468 sisl/src/s1511.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1511.c:



### Macros

- #define [S1511](#)

### Functions

- void [s1511](#) (SISLSurf \*ps, qpoint, bvec, int idim, [double](#) aepsco, [double](#) aepsge, int \*jpt, [double](#) \*\*gpar, int \*jcrv, [SISLIntcurve](#) \*\*\*wcurve, int \*jstat)

## 30.2468.1 Macro Definition Documentation

### 30.2468.1.1 #define S1511

Definition at line 49 of file s1511.c.

## 30.2468.2 Function Documentation

### 30.2468.2.1 void s1511 ( SISLSurf \* ps, qpoint , bvec , int idim, double aepsco, double aepsge, int \* jpt, double \*\* gpar, int \* jcrv, SISLIntcurve \*\*\* wcurve, int \* jstat )

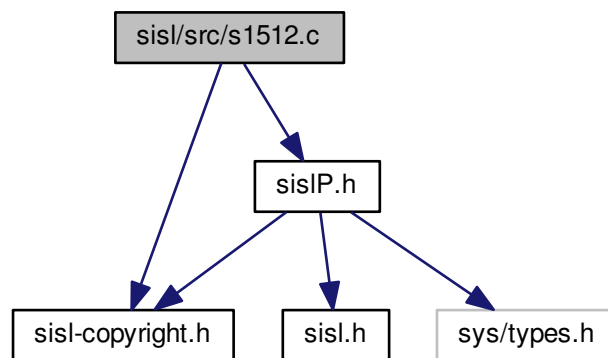
Definition at line 58 of file s1511.c.

## 30.2469 sisl/src/s1512.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1512.c:



## Macros

- #define [S1512](#)

## Functions

- void [s1512](#) (SISLSurf \*psurf, eyepoint, int idim, SISLSurf \*\*rsurf, int \*jstat)

## 30.2469.1 Macro Definition Documentation

### 30.2469.1.1 #define S1512

Definition at line 49 of file s1512.c.

## 30.2469.2 Function Documentation

### 30.2469.2.1 void s1512 ( SISLSurf \* psurf, eyepoint , int idim, SISLSurf \*\* rsurf, int \* jstat )

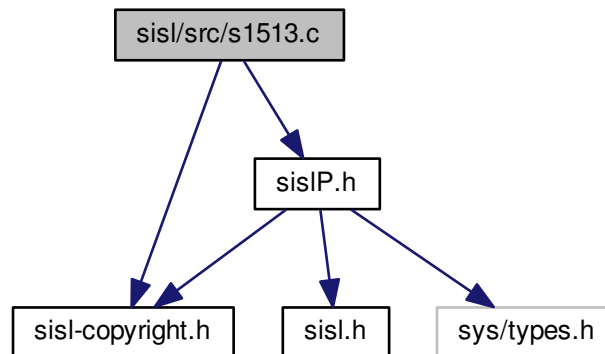
Definition at line 59 of file s1512.c.

## 30.2470 sisl/src/s1513.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1513.c:



## Macros

- #define [S1513](#)

## Functions

- void [s1513](#) (SISLSurf \*psurf, qpoint, bvec, int idim, SISLSurf \*\*rsurf, int \*jstat)

### 30.2470.1 Macro Definition Documentation

#### 30.2470.1.1 #define S1513

Definition at line 49 of file s1513.c.

### 30.2470.2 Function Documentation

#### 30.2470.2.1 void s1513 ( SISLSurf \* psurf, qpoint, bvec , int idim, SISLSurf \*\* rsurf, int \* jstat )

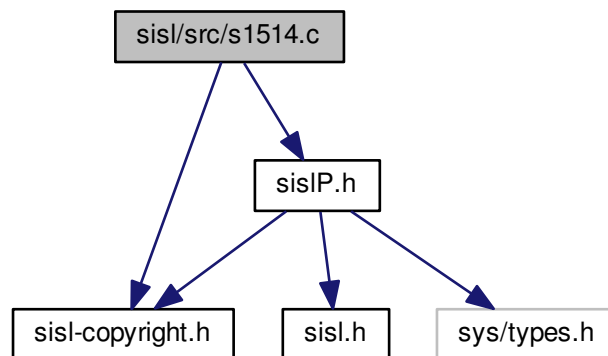
Definition at line 59 of file s1513.c.

## 30.2471 sisl/src/s1514.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1514.c:



### Macros

- #define [S1514](#)

### Functions

- void [s1514](#) (SISLSurf \*ps1, eyepoint, int idim, double aepsco, double aepsge, double amax, SISLIntcurve \*pintcr, int icur, int igrph, int \*jstat)

### 30.2471.1 Macro Definition Documentation

#### 30.2471.1.1 #define S1514

Definition at line 49 of file s1514.c.

### 30.2471.2 Function Documentation

#### 30.2471.2.1 void s1514 ( SISLSurf \* ps1, eyepoint , int idim, double aepsco, double aepsge, double amax, SISLIntcurve \* pintcr, int icur, int igrph, int \* jstat )

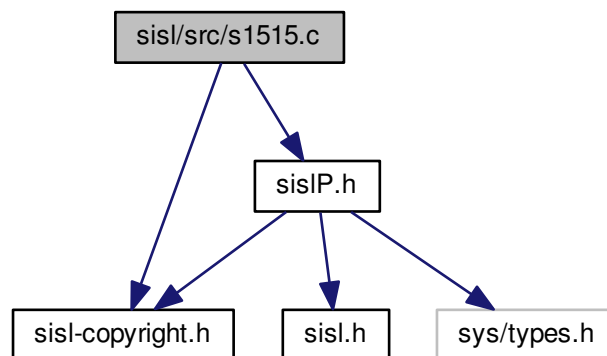
Definition at line 59 of file s1514.c.

## 30.2472 sisl/src/s1515.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1515.c:



### Macros

- #define [S1515](#)

### Functions

- void [s1515](#) (SISLSurf \*ps1, qpoint, bvec, int idim, [double](#) aepsco, [double](#) aepsge, [double](#) amax, SISLIntcurve \*pintcr, int icur, int igrph, int \*jstat)



### 30.2472.1 Macro Definition Documentation

#### 30.2472.1.1 #define S1515

Definition at line 49 of file s1515.c.

### 30.2472.2 Function Documentation

#### 30.2472.2.1 void s1515 ( SISLSurf \* ps1, qpoint , bvec , int idim, double aepsco, double aepsge, double amax, SISLIntcurve \* pintcr, int icur, int igraph, int \* jstat )

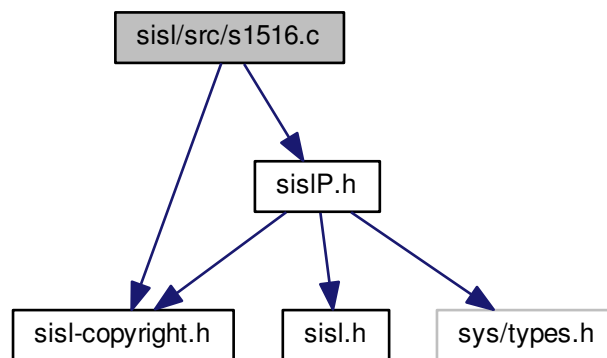
Definition at line 59 of file s1515.c.

## 30.2473 sisl/src/s1516.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1516.c:



### Macros

- #define [S1516](#)

### Functions

- void [s1516](#) (ep, epar, int im, int idim, double \*\*ev, int \*jstat)

### 30.2473.1 Macro Definition Documentation

#### 30.2473.1.1 #define S1516

Definition at line 49 of file s1516.c.

### 30.2473.2 Function Documentation

#### 30.2473.2.1 void s1516 ( ep , epar , int im, int idim, double \*\* ev, int \* jstat )

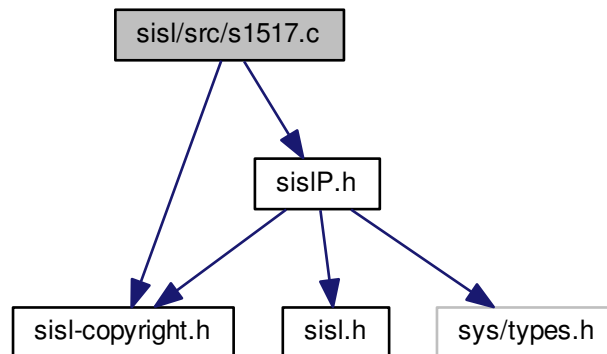
Definition at line 57 of file s1516.c.

## 30.2474 sisl/src/s1517.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1517.c:



### Macros

- #define [S1517](#)

### Functions

- void [s1517](#) (ep, ev, epar, int im, double mu, double \*\*evnew, int \*jstat)

### 30.2474.1 Macro Definition Documentation

#### 30.2474.1.1 #define S1517

Definition at line 49 of file s1517.c.

### 30.2474.2 Function Documentation

#### 30.2474.2.1 void s1517 ( ep , ev , epar , int im, double mu, double \*\* evnew, int \* jstat )

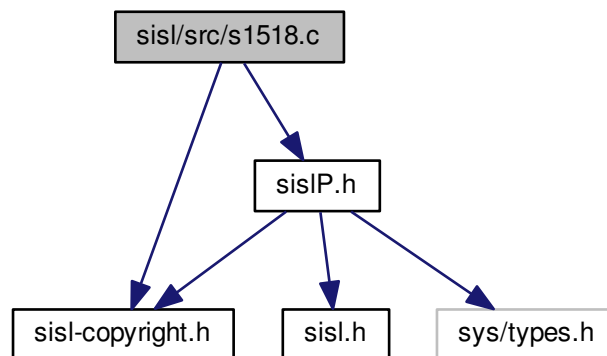
Definition at line 57 of file s1517.c.

## 30.2475 sisl/src/s1518.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1518.c:



### Macros

- #define [S1518](#)

### Functions

- void [s1518](#) ([SISLSurf](#) \*surf, point, dir, [double epsge](#), start, end, parin, parout, int \*stat)

### 30.2475.1 Macro Definition Documentation

#### 30.2475.1.1 #define S1518

Definition at line 48 of file s1518.c.

### 30.2475.2 Function Documentation

#### 30.2475.2.1 void s1518 ( SISLSurf \* surf, point , dir , double epsge, start , end , parin , parout , int \* stat )

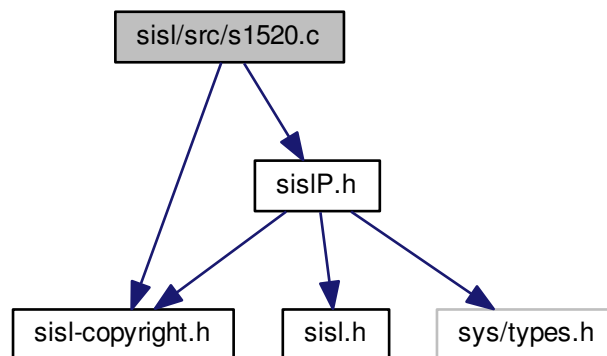
Definition at line 59 of file s1518.c.

## 30.2476 sisl/src/s1520.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1520.c:



### Macros

- #define [S1520](#)

### Functions

- void [s1520](#) (SISLCurve \*pc, double angle, ep, eaxis, SISLSurf \*\*rs, int \*jstat)

## 30.2476.1 Macro Definition Documentation

### 30.2476.1.1 #define S1520

Definition at line 49 of file s1520.c.

## 30.2476.2 Function Documentation

### 30.2476.2.1 void s1520 ( SISLCurve \* pc, double angle, ep, eaxis, SISLSurf \*\* rs, int \* jstat )

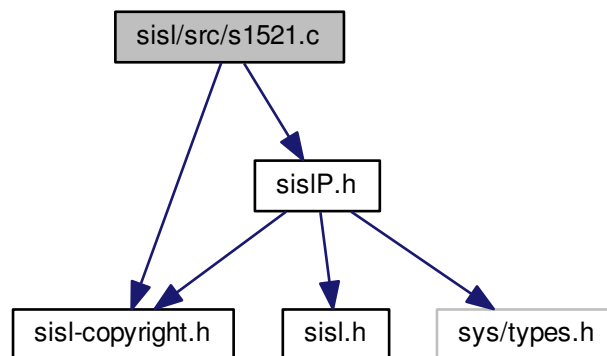
Definition at line 59 of file s1520.c.

## 30.2477 sisl/src/s1521.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1521.c:



## Macros

- #define [S1521](#)

## Functions

- [SISLCurve \\* s1521](#) (SISLCurve \*pc, int \*jstat)

## 30.2477.1 Macro Definition Documentation

### 30.2477.1.1 #define S1521

Definition at line 49 of file s1521.c.

## 30.2477.2 Function Documentation

### 30.2477.2.1 SISLCurve\* s1521 ( SISLCurve \* pc, int \* jstat )

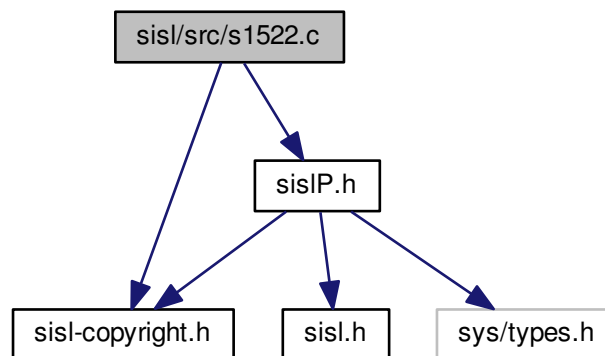
Definition at line 57 of file s1521.c.

## 30.2478 sisl/src/s1522.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1522.c:



## Macros

- #define [S1522](#)

## Functions

- void [s1522](#) ([normal](#), centre, ellipaxis, [double](#) ratio, int [dim](#), [SISLCurve](#) \*\*ellipse, int \*jstat)

### 30.2478.1 Macro Definition Documentation

#### 30.2478.1.1 #define S1522

Definition at line 42 of file s1522.c.

### 30.2478.2 Function Documentation

#### 30.2478.2.1 void s1522 ( normal , centre , ellipaxis , double ratio, int dim, SISLCurve\*\* ellipse, int \* jstat )

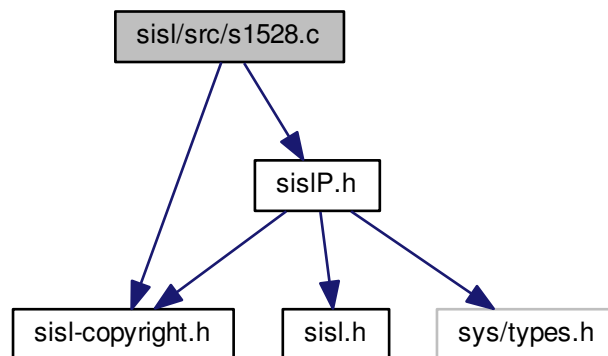
Definition at line 50 of file s1522.c.

## 30.2479 sisl/src/s1528.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1528.c:



### Macros

- #define [S1528](#)

### Functions

- void [s1528](#) (int idim, int m1, int m2, points, int ipar, int iopen1, int iopen2, double \*\*par1, double \*\*par2, int \*jstat)

### 30.2479.1 Macro Definition Documentation

#### 30.2479.1.1 #define S1528

Definition at line 42 of file s1528.c.

### 30.2479.2 Function Documentation

#### 30.2479.2.1 void s1528 ( int *idim*, int *m1*, int *m2*, points , int *ipar*, int *iopen1*, int *iopen2*, double \*\* *par1*, double \*\* *par2*, int \* *jstat* )

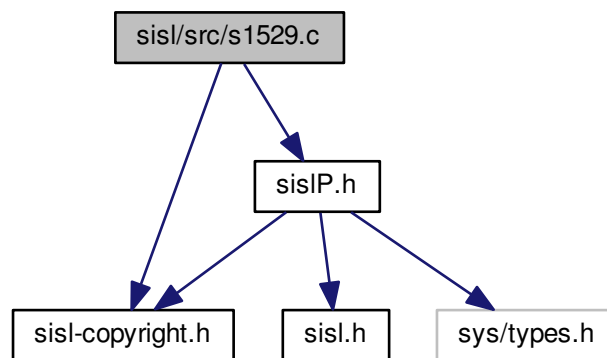
Definition at line 50 of file s1528.c.

## 30.2480 sisl/src/s1529.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1529.c:



### Macros

- #define [S1529](#)

### Functions

- void [s1529](#) (ep, eder10, eder01, eder11, int im1, int im2, int idim, int ipar, [SISLSurf](#) \*\*rsurf, int \*jstat)



## 30.2480.1 Macro Definition Documentation

### 30.2480.1.1 #define S1529

Definition at line 42 of file s1529.c.

## 30.2480.2 Function Documentation

### 30.2480.2.1 void s1529 ( ep , eder10 , eder01 , eder11 , int im1, int im2, int idim, int ipar, SISLSurf \*\* rsurf, int \* jstat )

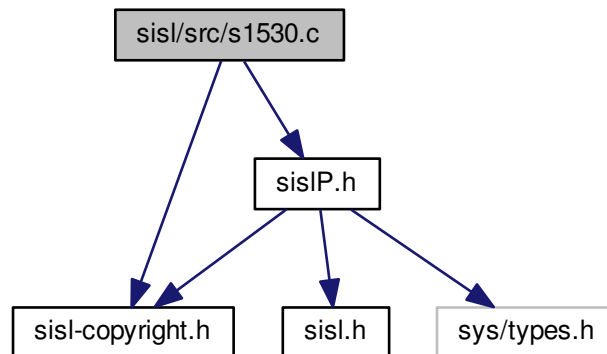
Definition at line 51 of file s1529.c.

## 30.2481 sisl/src/s1530.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1530.c:



## Macros

- #define [S1530](#)

## Functions

- void [s1530](#) (ep, eder10, eder01, eder11, epar1, epar2, int im1, int im2, int idim, [SISLSurf](#) \*\*rsurf, int \*jstat)

### 30.2481.1 Macro Definition Documentation

#### 30.2481.1.1 #define S1530

Definition at line 49 of file s1530.c.

### 30.2481.2 Function Documentation

#### 30.2481.2.1 void s1530 ( ep , eder10 , eder01 , eder11 , epar1 , epar2 , int im1 , int im2 , int idim , SISLSurf \*\* rsurf , int \* jstat )

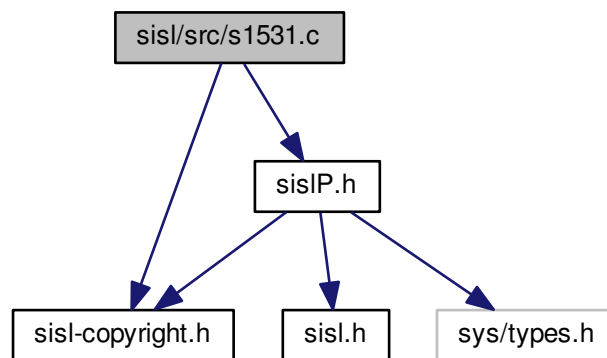
Definition at line 59 of file s1530.c.

## 30.2482 sisl/src/s1531.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1531.c:



### Macros

- #define [S1531](#)

### Functions

- void [s1531](#) (ea, int idim, int in1, int in2, double \*\*eb, int \*jstat)

### 30.2482.1 Macro Definition Documentation

#### 30.2482.1.1 #define S1531

Definition at line 49 of file s1531.c.

### 30.2482.2 Function Documentation

#### 30.2482.2.1 void s1531 ( ea , int *idim*, int *in1*, int *in2*, double \*\* *eb*, int \* *jstat* )

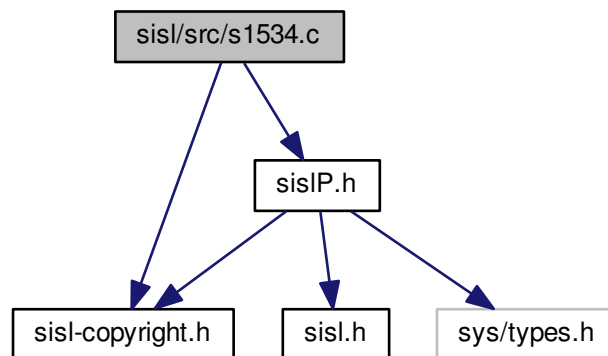
Definition at line 56 of file s1531.c.

## 30.2483 sisl/src/s1534.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1534.c:



### Macros

- #define [S1534](#)

### Functions

- void [s1534](#) (points, der10, der01, der11, int im1, int im2, int idim, int ipar, int con1, int con2, int con3, int con4, int order1, int order2, int iopen1, int iopen2, [SISLSurf](#) \*\*rsurf, int \*jstat)

### 30.2483.1 Macro Definition Documentation

#### 30.2483.1.1 #define S1534

Definition at line 42 of file s1534.c.

### 30.2483.2 Function Documentation

#### 30.2483.2.1 void s1534 ( points , der10 , der01 , der11 , int im1, int im2, int idim, int ipar, int con1, int con2, int con3, int con4, int order1, int order2, int iopen1, int iopen2, SISLSurf \*\*rsurf, int \*jstat )

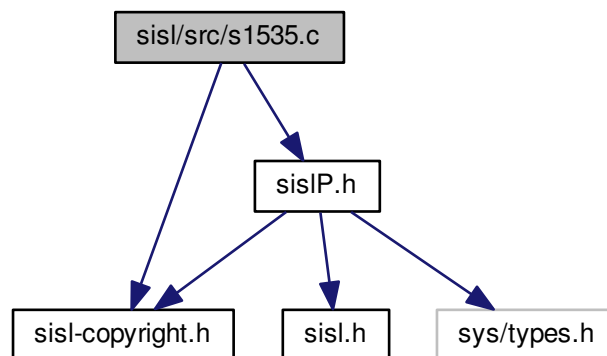
Definition at line 52 of file s1534.c.

## 30.2484 sisl/src/s1535.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1535.c:



### Macros

- #define [S1535](#)

### Functions

- void [s1535](#) (points, der10, der01, der11, int im1, int im2, int idim, par1, par2, int con1, int con2, int con3, int con4, int order1, int order2, int iopen1, int iopen2, [SISLSurf](#) \*\*rsurf, int \*jstat)

### 30.2484.1 Macro Definition Documentation

#### 30.2484.1.1 #define S1535

Definition at line 42 of file s1535.c.

### 30.2484.2 Function Documentation

#### 30.2484.2.1 void s1535 ( points , der10 , der01 , der11 , int im1, int im2, int idim, par1 , par2 , int con1, int con2, int con3, int con4, int order1, int order2, int iopen1, int iopen2, SISLSurf \*\*rsurf, int \*jstat )

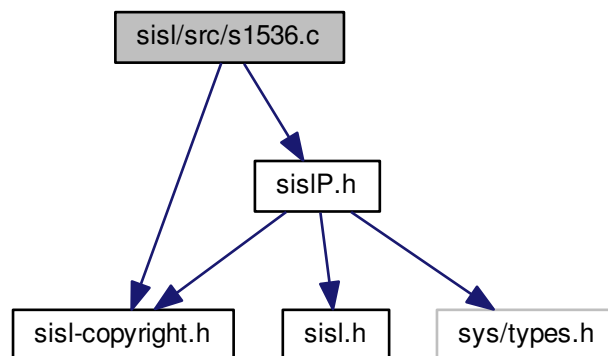
Definition at line 53 of file s1535.c.

## 30.2485 sisl/src/s1536.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1536.c:



### Macros

- #define [S1536](#)

### Functions

- void [s1536](#) (points, int im1, int im2, int idim, int ipar, int con1, int con2, int con3, int con4, int order1, int order2, int iopen1, int iopen2, [SISLSurf](#) \*\*rsurf, int \*jstat)

### 30.2485.1 Macro Definition Documentation

#### 30.2485.1.1 #define S1536

Definition at line 42 of file s1536.c.

### 30.2485.2 Function Documentation

#### 30.2485.2.1 void s1536 ( points , int im1, int im2, int idim, int ipar, int con1, int con2, int con3, int con4, int order1, int order2, int iopen1, int iopen2, SISLSurf \*\*rsurf, int \*jstat )

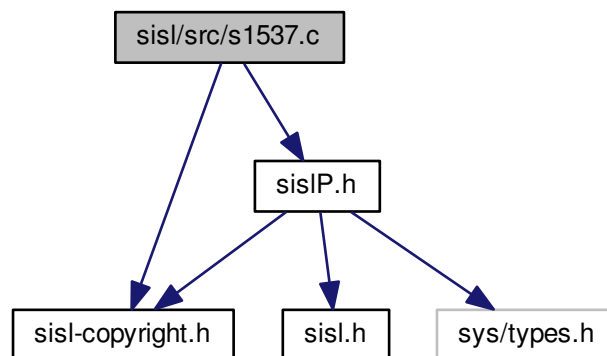
Definition at line 51 of file s1536.c.

## 30.2486 sisl/src/s1537.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1537.c:



### Macros

- #define [S1537](#)

### Functions

- void [s1537](#) (points, int im1, int im2, int idim, par1, par2, int con1, int con2, int con3, int con4, int order1, int order2, int iopen1, int iopen2, [SISLSurf](#) \*\*rsurf, int \*jstat)

### 30.2486.1 Macro Definition Documentation

#### 30.2486.1.1 #define S1537

Definition at line 42 of file s1537.c.

### 30.2486.2 Function Documentation

#### 30.2486.2.1 void s1537 ( points , int im1, int im2, int idim, par1 , par2 , int con1, int con2, int con3, int con4, int order1, int order2, int iopen1, int iopen2, SISLSurf \*\* rsurf, int \* jstat )

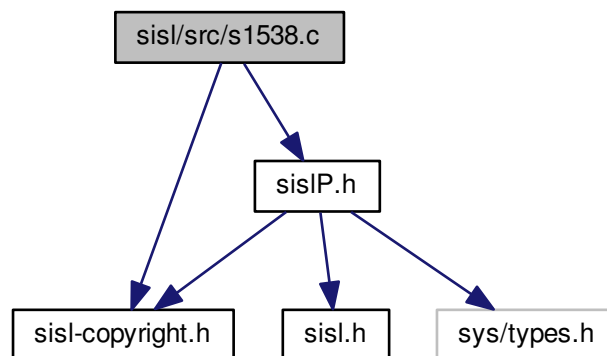
Definition at line 52 of file s1537.c.

## 30.2487 sisl/src/s1538.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1538.c:



### Macros

- #define [S1538](#)

### Functions

- void [s1538](#) (int inbcrv, vpcurv, nctyp, [double](#) astpar, int iopen, int iord2, int iflag, [SISLSurf](#) \*\*rsurf, [double](#) \*\*gpar, int \*jstat)

### 30.2487.1 Macro Definition Documentation

#### 30.2487.1.1 #define S1538

Definition at line 42 of file s1538.c.

### 30.2487.2 Function Documentation

#### 30.2487.2.1 void s1538 ( int *inbcrv*, *vpcurv* , *nctyp* , double *astpar*, int *iopen*, int *iord2*, int *iflag*, SISLSurf \*\**rsurf*, double \*\**gpar*, int \**jstat* )

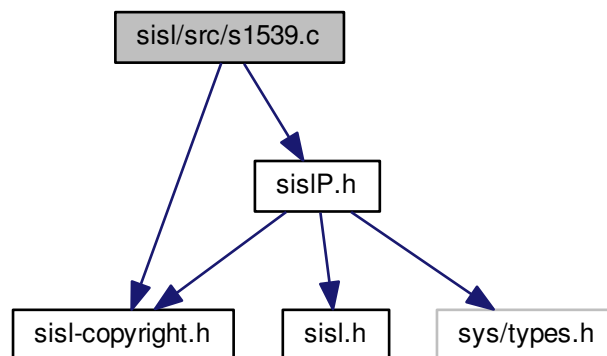
Definition at line 52 of file s1538.c.

## 30.2488 sisl/src/s1539.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1539.c:



### Macros

- #define [S1539](#)

### Functions

- void [s1539](#) (int *inbcrv*, *vpcurv*, *nctyp*, *epar*, double *astpar*, int *iopen*, int *iord2*, int *iflag*, SISLSurf \*\**rsurf*, double \*\**gpar*, int \**jstat*)



## 30.2488.1 Macro Definition Documentation

### 30.2488.1.1 #define S1539

Definition at line 42 of file s1539.c.

## 30.2488.2 Function Documentation

### 30.2488.2.1 void s1539 ( int *inbcrv*, vpcurv , nctyp , epar , double *astpar*, int *iopen*, int *iord2*, int *iflag*, SISLSurf \*\* *rsurf*, double \*\* *gpar*, int \* *jstat* )

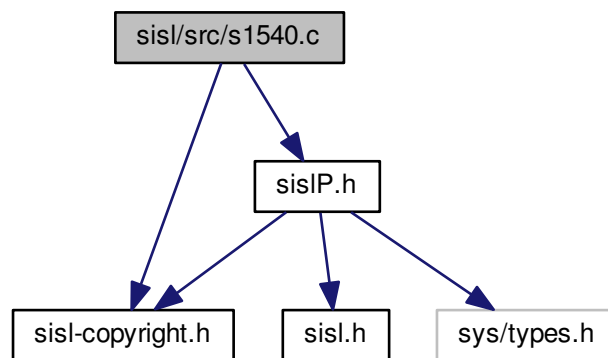
Definition at line 52 of file s1539.c.

## 30.2489 sisl/src/s1540.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1540.c:



## Macros

- #define [S1540](#)

## Functions

- void [s1540](#) (et, int ik, int in, ax, int im, int ider, ebder, ileft, int \*jstat)

## 30.2489.1 Macro Definition Documentation

### 30.2489.1.1 #define S1540

Definition at line 47 of file s1540.c.

## 30.2489.2 Function Documentation

### 30.2489.2.1 void s1540 ( et , int ik, int in, ax , int im, int iber, ebder , ileft , int \* jstat )

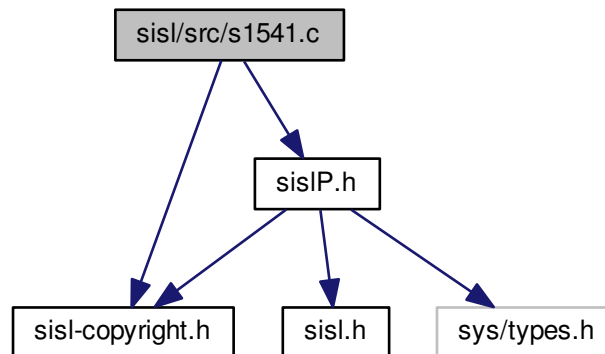
Definition at line 65 of file s1540.c.

## 30.2490 sisl/src/s1541.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1541.c:



## Macros

- #define [S1541](#)

## Functions

- void [s1541](#) ([SISLCurve](#) \*pc1, int npol, ebder, ileft, eder, int \*jstat)

## 30.2490.1 Macro Definition Documentation

### 30.2490.1.1 #define S1541

Definition at line 47 of file s1541.c.

## 30.2490.2 Function Documentation

### 30.2490.2.1 void s1541 ( SISLCurve\* *pc1*, int *npol*, *ebder*, *ileft*, *eder*, int \* *jstat* )

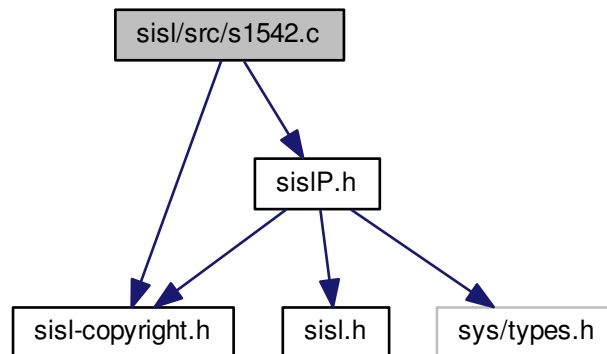
Definition at line 62 of file s1541.c.

## 30.2491 sisl/src/s1542.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1542.c:



## Macros

- #define [S1542](#)

## Functions

- void [s1542](#) ([SISLCurve](#) \**pc1*, int *m*, *x*, *eder*, int \**jstat*)

### 30.2491.1 Macro Definition Documentation

#### 30.2491.1.1 #define S1542

Definition at line 47 of file s1542.c.

### 30.2491.2 Function Documentation

#### 30.2491.2.1 void s1542 ( SISLCurve \* pc1, int m, x , eder , int \* jstat )

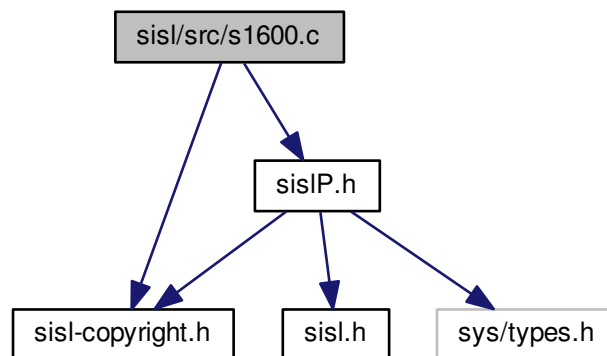
Definition at line 55 of file s1542.c.

## 30.2492 sisl/src/s1600.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1600.c:



### Macros

- #define [S1600](#)

### Functions

- void [s1600](#) (SISLCurve \*pc, epoint, enorm, int idim, SISLCurve \*\*rc, int \*jstat)

## 30.2492.1 Macro Definition Documentation

### 30.2492.1.1 #define S1600

Definition at line 49 of file s1600.c.

## 30.2492.2 Function Documentation

### 30.2492.2.1 void s1600 ( SISLCurve \* pc, epoint , enorm , int idim, SISLCurve \*\* rc, int \* jstat )

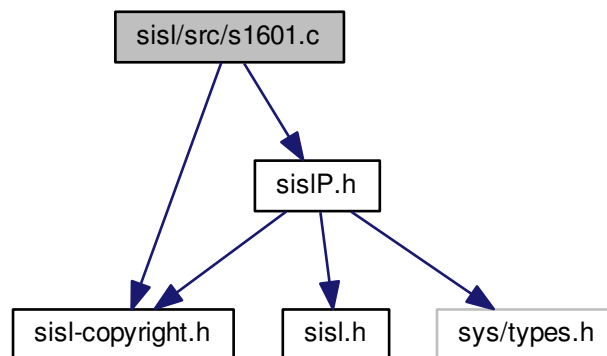
Definition at line 57 of file s1600.c.

## 30.2493 sisl/src/s1601.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1601.c:



## Macros

- #define [S1601](#)

## Functions

- void [s1601](#) ([SISLSurf](#) \*psurf, epoint, enorm, int idim, [SISLSurf](#) \*\*rsurf, int \*jstat)

### 30.2493.1 Macro Definition Documentation

#### 30.2493.1.1 #define S1601

Definition at line 49 of file s1601.c.

### 30.2493.2 Function Documentation

#### 30.2493.2.1 void s1601 ( SISLSurf \* *psurf*, epoint , enorm , int *idim*, SISLSurf \*\* *rsurf*, int \* *jstat* )

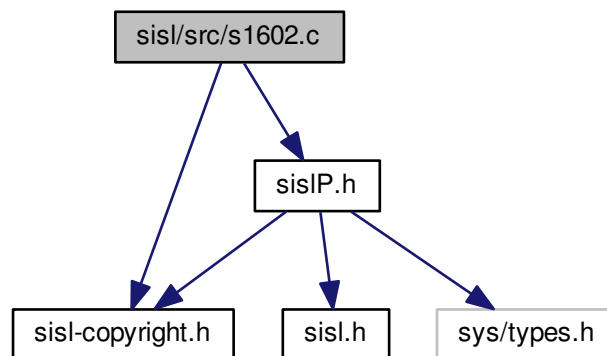
Definition at line 57 of file s1601.c.

## 30.2494 sisl/src/s1602.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1602.c:



### Macros

- #define [S1602](#)

### Functions

- void [s1602](#) (estapt, endpt, int ik, int idim, [double](#) astpar, [double](#) \*cendpar, [SISLCurve](#) \*\*rc, int \*jstat)

## 30.2494.1 Macro Definition Documentation

### 30.2494.1.1 #define S1602

Definition at line 49 of file s1602.c.

## 30.2494.2 Function Documentation

### 30.2494.2.1 void s1602 ( estapt , endpt , int ik, int idim, double astpar, double \* cendpar, SISLCurve \*\* rc, int \* jstat )

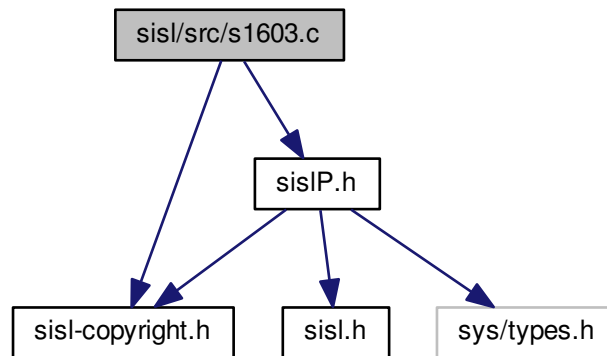
Definition at line 58 of file s1602.c.

## 30.2495 sisl/src/s1603.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1603.c:



## Macros

- #define [S1603](#)

## Functions

- void [s1603](#) ([SISLSurf](#) \*psurf, [double](#) \*cmin1, [double](#) \*cmin2, [double](#) \*cmax1, [double](#) \*cmax2, int \*jstat)

### 30.2495.1 Macro Definition Documentation

#### 30.2495.1.1 #define S1603

Definition at line 49 of file s1603.c.

### 30.2495.2 Function Documentation

#### 30.2495.2.1 void s1603 ( SISLSurf \* psurf, double \* cmin1, double \* cmin2, double \* cmax1, double \* cmax2, int \* jstat )

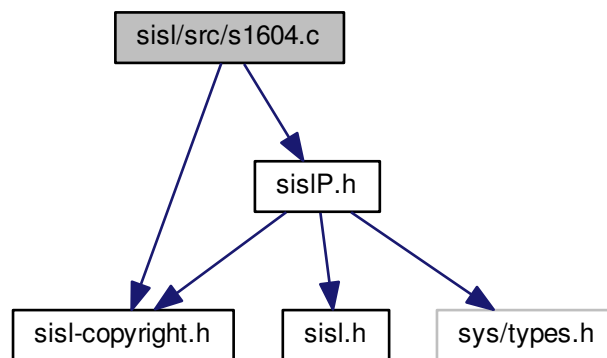
Definition at line 57 of file s1603.c.

## 30.2496 sisl/src/s1604.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1604.c:



### Macros

- #define [S1604](#)

### Functions

- void [s1604](#) (epoint, int inbpnt, double astpar, int iopen, int idim, int ik, [SISLCurve](#) \*\*rc, int \*jstat)



## 30.2496.1 Macro Definition Documentation

### 30.2496.1.1 #define S1604

Definition at line 49 of file s1604.c.

## 30.2496.2 Function Documentation

### 30.2496.2.1 void s1604 ( epoint , int inbpnt, double astpar, int iopen, int idim, int ik, SISLCurve \*\* rc, int \* jstat )

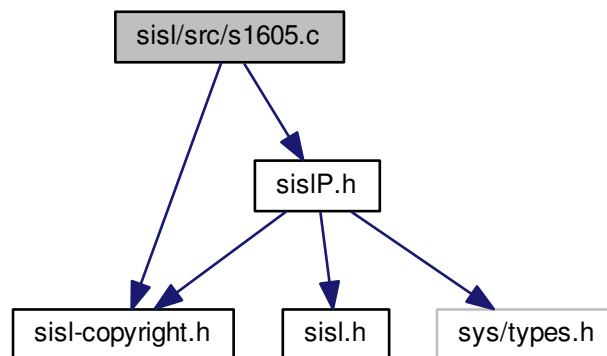
Definition at line 58 of file s1604.c.

## 30.2497 sisl/src/s1605.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1605.c:



## Macros

- #define [S1605](#)

## Functions

- void [s1605](#) (SISLCurve \*pc, double aepsge, double \*\*gpoint, int \*jnbpnt, int \*jstat)

### 30.2497.1 Macro Definition Documentation

#### 30.2497.1.1 #define S1605

Definition at line 49 of file s1605.c.

### 30.2497.2 Function Documentation

#### 30.2497.2.1 void s1605 ( SISLCurve \* pc, double aepsge, double \*\* gpoint, int \* jnpnt, int \* jstat )

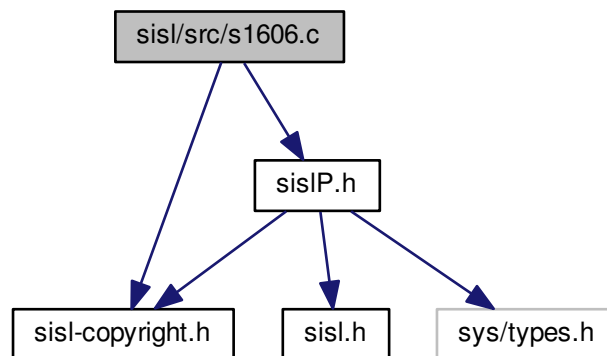
Definition at line 57 of file s1605.c.

## 30.2498 sisl/src/s1606.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1606.c:



### Macros

- #define [S1606](#)

### Functions

- void [s1606](#) (SISLCurve \*pc1, SISLCurve \*pc2, double aepsge, epoint1, epoint2, int itype, int idim, int ik, SISLCurve \*\*rc, int \*jstat)

## 30.2498.1 Macro Definition Documentation

### 30.2498.1.1 #define S1606

Definition at line 49 of file s1606.c.

## 30.2498.2 Function Documentation

### 30.2498.2.1 void s1606 ( SISLCurve \* pc1, SISLCurve \* pc2, double aepsge, epoint1 , epoint2 , int itype, int idim, int ik, SISLCurve \*\* rc, int \* jstat )

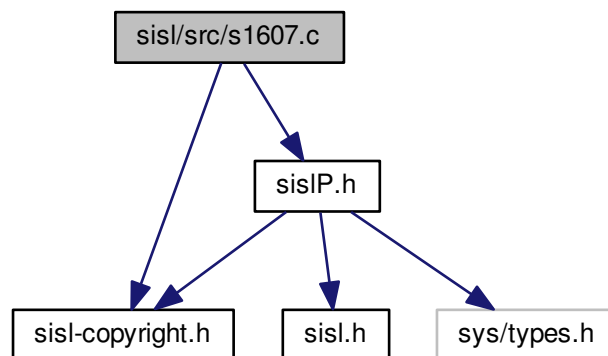
Definition at line 59 of file s1606.c.

## 30.2499 sisl/src/s1607.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1607.c:



## Macros

- #define [S1607](#)

## Functions

- void [s1607](#) (SISLCurve \*pc1, SISLCurve \*pc2, double aepsge, double aend1, double afile1, double aend2, double afile2, int itype, int idim, int ik, SISLCurve \*\*rc, int \*jstat)

### 30.2499.1 Macro Definition Documentation

#### 30.2499.1.1 #define S1607

Definition at line 49 of file s1607.c.

### 30.2499.2 Function Documentation

#### 30.2499.2.1 void s1607 ( SISLCurve \* pc1, SISLCurve \* pc2, double aepsge, double aend1, double afil1, double aend2, double afil2, int itype, int idim, int ik, SISLCurve \*\* rc, int \* jstat )

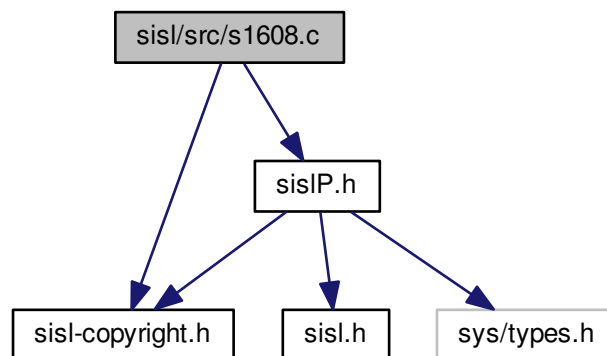
Definition at line 61 of file s1607.c.

## 30.2500 sisl/src/s1608.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1608.c:



### Macros

- #define [S1608](#)

### Functions

- void [s1608](#) (SISLCurve \*pc1, SISLCurve \*pc2, double aepsge, ep11, epf1, ep21, epf2, int itype, int idim, int ik, SISLCurve \*\*rc, double \*ct11, double \*ctf1, double \*ct21, double \*ctf2, int \*jstat)

### 30.2500.1 Macro Definition Documentation

#### 30.2500.1.1 #define S1608

Definition at line 49 of file s1608.c.

### 30.2500.2 Function Documentation

#### 30.2500.2.1 void s1608 ( SISLCurve \* pc1, SISLCurve \* pc2, double aepsge, ep11, epf1, ep21, epf2, int itype, int idim, int ik, SISLCurve \*\* rc, double \* ct11, double \* ctf1, double \* ct21, double \* ctf2, int \* jstat )

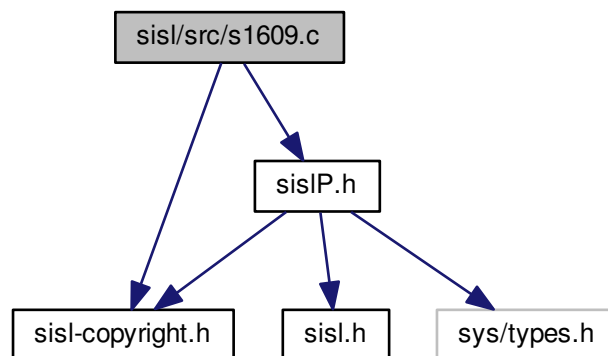
Definition at line 60 of file s1608.c.

## 30.2501 sisl/src/s1609.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1609.c:



### Macros

- #define [S1609](#)

### Functions

- void [s1609](#) (SISLCurve \*pc1, SISLCurve \*pc2, double aepsge, eps1, epf, eps2, double aradius, enorm, int itype, int idim, int ik, SISLCurve \*\*rc, double \*ct11, double \*ctf1, double \*ct21, double \*ctf2, int \*jstat)

### 30.2501.1 Macro Definition Documentation

#### 30.2501.1.1 #define S1609

Definition at line 49 of file s1609.c.

### 30.2501.2 Function Documentation

30.2501.2.1 void s1609 ( SISLCurve \* *pc1*, SISLCurve \* *pc2*, double *aepsge*, eps1 , epf , eps2 , double *aradius*, enorm , int *itype*, int *idim*, int *ik*, SISLCurve \*\* *rc*, double \* *ct11*, double \* *ctf1*, double \* *ct21*, double \* *ctf2*, int \* *jstat* )

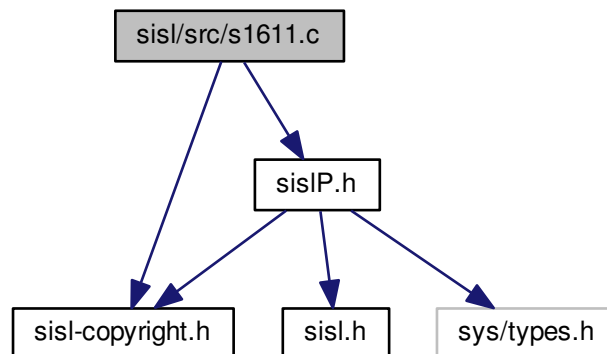
Definition at line 61 of file s1609.c.

## 30.2502 sisl/src/s1611.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1611.c:



### Macros

- #define [S1611](#)

### Functions

- void [s1611](#) (epoint, int inbpnt, int idim, eptyp, int iopen, int ik, [double](#) astpar, [double](#) aepsge, [double](#) \*cendpar, [SISLCurve](#) \*\*rc, int \*jstat)

### 30.2502.1 Macro Definition Documentation

#### 30.2502.1.1 #define S1611

Definition at line 49 of file s1611.c.

### 30.2502.2 Function Documentation

#### 30.2502.2.1 void s1611 ( epoint , int inbpnt, int idim, epty , int iopen, int ik, double astpar, double aepsge, double \* cendpar, SISLCurve \*\* rc, int \* jstat )

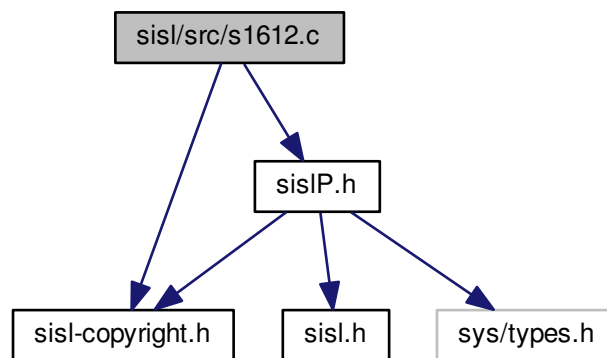
Definition at line 60 of file s1611.c.

## 30.2503 sisl/src/s1612.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1612.c:



### Macros

- #define [S1612](#)

### Functions

- void [s1612](#) (SISLCurve \*pc, double aepsge, double \*\*gpoint, int \*jnbpnt, int \*jleng, int \*jstat)

### 30.2503.1 Macro Definition Documentation

#### 30.2503.1.1 #define S1612

Definition at line 49 of file s1612.c.

### 30.2503.2 Function Documentation

#### 30.2503.2.1 void s1612 ( SISLCurve \* pc, double aepsge, double \*\* gpoint, int \* jnbpnt, int \* jleng, int \* jstat )

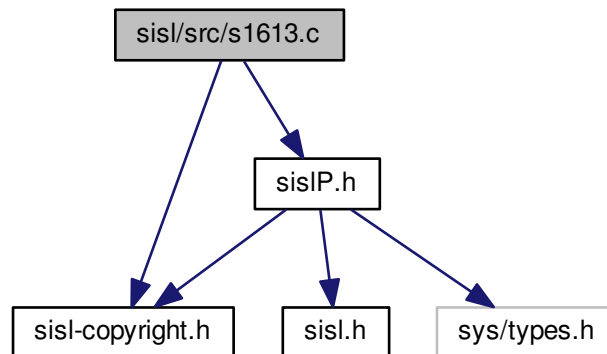
Definition at line 57 of file s1612.c.

## 30.2504 sisl/src/s1613.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1613.c:



### Macros

- #define [S1613](#)

### Functions

- void [s1613](#) (SISLCurve \*pc, double aepsge, double \*\*gpoint, int \*jnbpnt, int \*jstat)



## 30.2504.1 Macro Definition Documentation

### 30.2504.1.1 #define S1613

Definition at line 49 of file s1613.c.

## 30.2504.2 Function Documentation

### 30.2504.2.1 void s1613 ( SISLCurve \* pc, double aepsge, double \*\* gpoint, int \* jnbpnt, int \* jstat )

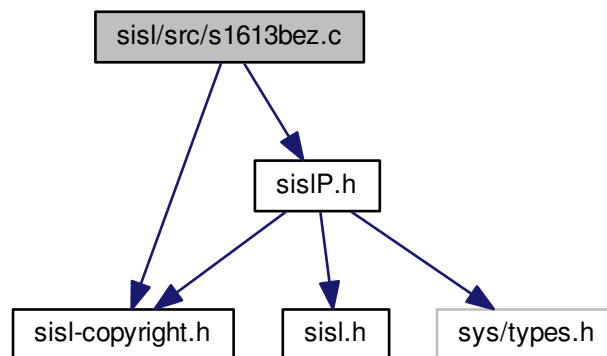
Definition at line 57 of file s1613.c.

## 30.2505 sisl/src/s1613bez.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1613bez.c:



## Macros

- #define [S1613BEZ](#)

## Functions

- void [s1613bez](#) (SISLCurve \*pc, int idiv, double aepsge, double \*\*gpar, int \*jnpar, int \*jstat)

### 30.2505.1 Macro Definition Documentation

#### 30.2505.1.1 #define S1613BEZ

Definition at line 49 of file s1613bez.c.

### 30.2505.2 Function Documentation

#### 30.2505.2.1 void s1613bez ( SISLCurve \* pc, int idiv, double aeprge, double \*\* gpar, int \* jnpar, int \* jstat )

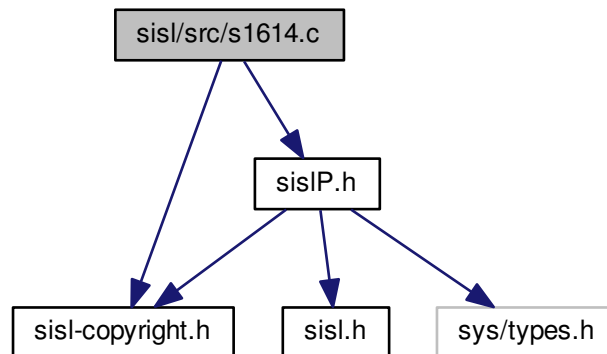
Definition at line 57 of file s1613bez.c.

## 30.2506 sisl/src/s1614.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1614.c:



### Macros

- #define [S1614](#)

### Functions

- void [s1614](#) (epoint, int inbpnt, int idim, eptyp, spoint, int \*jnbpnt, sptyp, int \*jstat)

## 30.2506.1 Macro Definition Documentation

### 30.2506.1.1 #define S1614

Definition at line 49 of file s1614.c.

## 30.2506.2 Function Documentation

### 30.2506.2.1 void s1614 ( epoint , int inbpnt, int idim, eptyp , spoint , int \* jnbpnt, sptyp , int \* jstat )

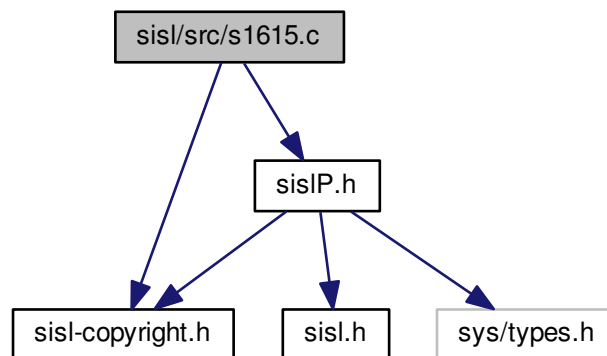
Definition at line 58 of file s1614.c.

## 30.2507 sisl/src/s1615.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1615.c:



## Macros

- #define [S1615](#)

## Functions

- void [s1615](#) (epoint, int inbpnt, int idim, eptyp, int \*jstat)

### 30.2507.1 Macro Definition Documentation

#### 30.2507.1.1 #define S1615

Definition at line 49 of file s1615.c.

### 30.2507.2 Function Documentation

#### 30.2507.2.1 void s1615 ( epoint , int inbpnt, int idim, eptyp , int \* jstat )

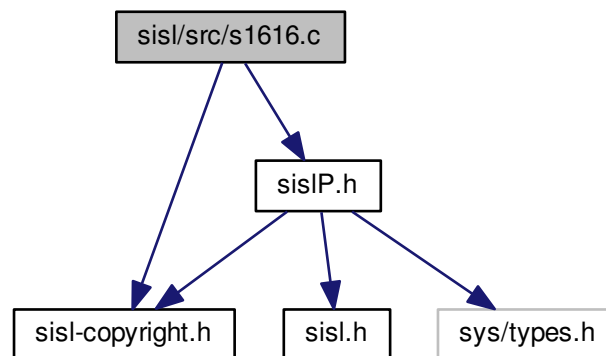
Definition at line 56 of file s1615.c.

## 30.2508 sisl/src/s1616.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1616.c:



### Macros

- #define [S1616](#)

### Functions

- void [s1616](#) (epoint, int inbpnt, int idim, eptyp, econic, int \*jstat)

### 30.2508.1 Macro Definition Documentation

#### 30.2508.1.1 #define S1616

Definition at line 49 of file s1616.c.

### 30.2508.2 Function Documentation

#### 30.2508.2.1 void s1616 ( epoint , int inbpnt , int idim , eptyp , econic , int \* jstat )

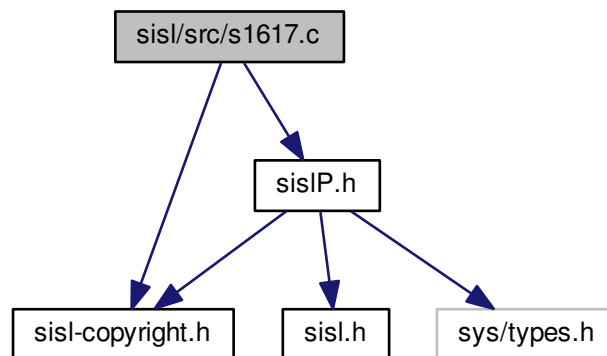
Definition at line 57 of file s1616.c.

## 30.2509 sisl/src/s1617.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1617.c:



### Macros

- #define [S1617](#)

### Functions

- void [s1617](#) (epoint, int inbpnt, int idim, eptyp, [double aepsge](#), econic, estart, etang, estop, [double \\*ashape](#), int \*jstat)

### 30.2509.1 Macro Definition Documentation

#### 30.2509.1.1 #define S1617

Definition at line 49 of file s1617.c.

### 30.2509.2 Function Documentation

#### 30.2509.2.1 void s1617 ( epoint , int inbpnt , int idim , epty , double aepsge , econic , estart , etang , estop , double \* ashape , int \* jstat )

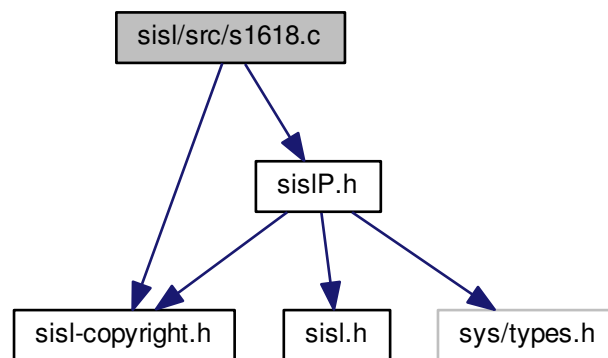
Definition at line 59 of file s1617.c.

## 30.2510 sisl/src/s1618.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1618.c:



### Macros

- #define [S1618](#)

### Functions

- void [s1618](#) (ematrix, eright, esol, int in, [double](#) \*adiff)

## 30.2510.1 Macro Definition Documentation

### 30.2510.1.1 #define S1618

Definition at line 49 of file s1618.c.

## 30.2510.2 Function Documentation

### 30.2510.2.1 void s1618 ( ematrix , eright , esol , int in , double \* adiff )

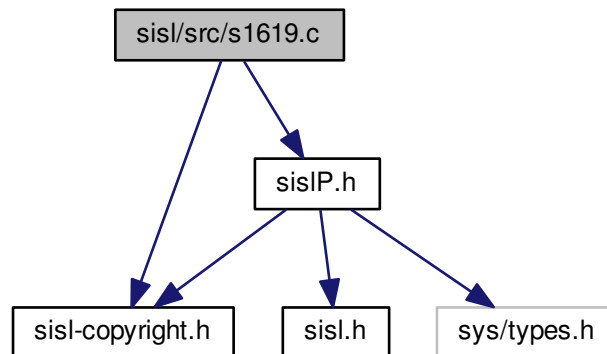
Definition at line 57 of file s1618.c.

## 30.2511 sisl/src/s1619.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1619.c:



## Macros

- #define [S1619](#)

## Functions

- void [s1619](#) (epoint, int inbpnt, int idim, eptyp, econic, int ityp, etang, [double](#) \*ashape, int \*jstat)

### 30.2511.1 Macro Definition Documentation

#### 30.2511.1.1 #define S1619

Definition at line 49 of file s1619.c.

### 30.2511.2 Function Documentation

#### 30.2511.2.1 void s1619 ( epoint , int inbpnt , int idim , epty , econic , int ityp , etang , double \* ashape , int \* jstat )

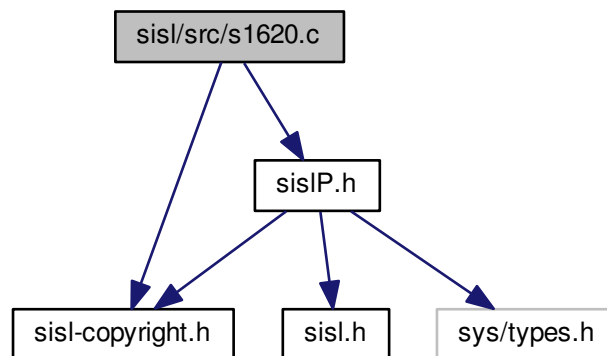
Definition at line 58 of file s1619.c.

## 30.2512 sisl/src/s1620.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1620.c:



### Macros

- #define [S1620](#)

### Functions

- void [s1620](#) (epoint, int inbpnt1, int inbpnt2, int ipar, int iopen1, int iopen2, int ik1, int ik2, int idim, [SISLSurf](#) \*\*rs, int \*jstat)



### 30.2512.1 Macro Definition Documentation

#### 30.2512.1.1 #define S1620

Definition at line 48 of file s1620.c.

### 30.2512.2 Function Documentation

#### 30.2512.2.1 void s1620 ( epoint , int inbpnt1, int inbpnt2, int ipar, int iopen1, int iopen2, int ik1, int ik2, int idim, SISLSurf \*\* rs, int \* jstat )

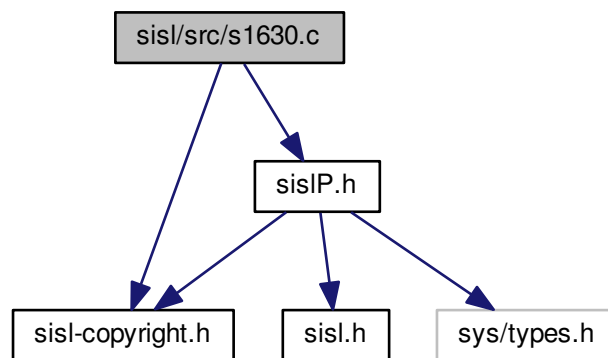
Definition at line 57 of file s1620.c.

## 30.2513 sisl/src/s1630.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1630.c:



### Macros

- #define [S1630](#)

### Functions

- void [s1630](#) (epoint, int inbpnt, [double](#) astpar, int iopen, int idim, int ik, [SISLCurve](#) \*\*rc, int \*jstat)

### 30.2513.1 Macro Definition Documentation

#### 30.2513.1.1 #define S1630

Definition at line 42 of file s1630.c.

### 30.2513.2 Function Documentation

#### 30.2513.2.1 void s1630 ( epoint , int inbpnt, double astpar, int iopen, int idim, int ik, SISLCurve \*\*rc, int \*jstat )

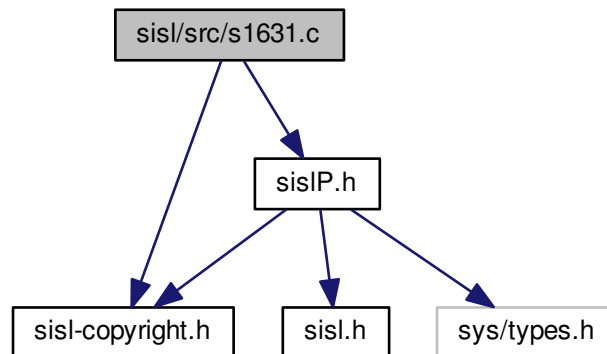
Definition at line 51 of file s1630.c.

## 30.2514 sisl/src/s1631.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1631.c:



### Macros

- #define [S1631](#)

### Functions

- void [s1631](#) ([SISLCurve](#) \*pc, epoint, enorm, projdir, int idim, [SISLCurve](#) \*\*rc, int \*jstat)

### 30.2514.1 Macro Definition Documentation

#### 30.2514.1.1 #define S1631

Definition at line 43 of file s1631.c.

### 30.2514.2 Function Documentation

#### 30.2514.2.1 void s1631 ( SISLCurve \* pc, epoint , enorm , projdir , int idim, SISLCurve \*\* rc, int \* jstat )

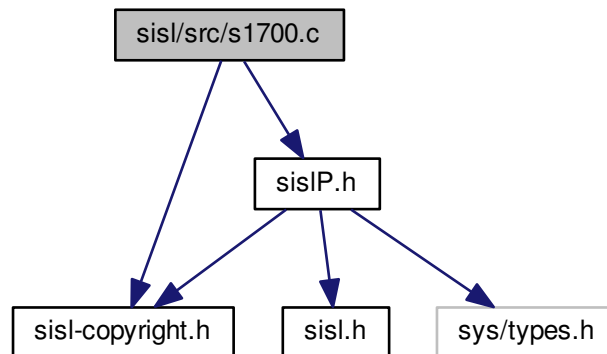
Definition at line 51 of file s1631.c.

## 30.2515 sisl/src/s1700.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1700.c:



### Macros

- #define [S1700](#)

### Functions

- void [s1700](#) (int imy, int ik, int in, int iv, int \*jpl, int \*jfi, int \*jla, double \*et, double apar, double \*galfa, int \*jstat)

### 30.2515.1 Macro Definition Documentation

#### 30.2515.1.1 #define S1700

Definition at line 49 of file s1700.c.

### 30.2515.2 Function Documentation

#### 30.2515.2.1 void s1700 ( int *imy*, int *ik*, int *in*, int *iv*, int \* *jpl*, int \* *jfi*, int \* *jla*, double \* *et*, double *apar*, double \* *galfa*, int \* *jstat* )

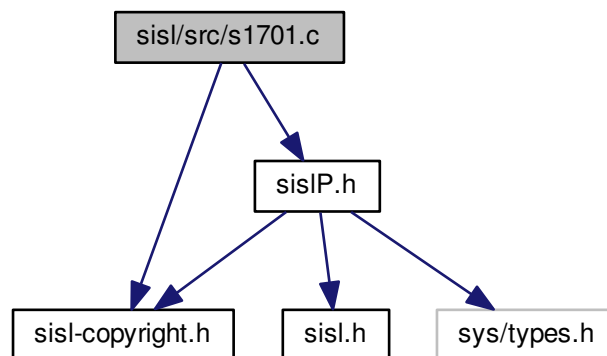
Definition at line 58 of file s1700.c.

## 30.2516 sisl/src/s1701.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1701.c:



### Macros

- #define [S1701](#)

### Functions

- void [s1701](#) (int *ij*, int *imy*, int *ik*, int *in*, int \* *jpl*, int \* *jfi*, int \* *jla*, double \* *et*, double \* *etau*, double \* *ep*, double \* *galfa*, int \* *jstat*)

## 30.2516.1 Macro Definition Documentation

### 30.2516.1.1 #define S1701

Definition at line 49 of file s1701.c.

## 30.2516.2 Function Documentation

### 30.2516.2.1 void s1701 ( int *ij*, int *imy*, int *ik*, int *in*, int \* *jpl*, int \* *jfi*, int \* *jla*, double \* *et*, double \* *etau*, double \* *ep*, double \* *galfa*, int \* *jstat* )

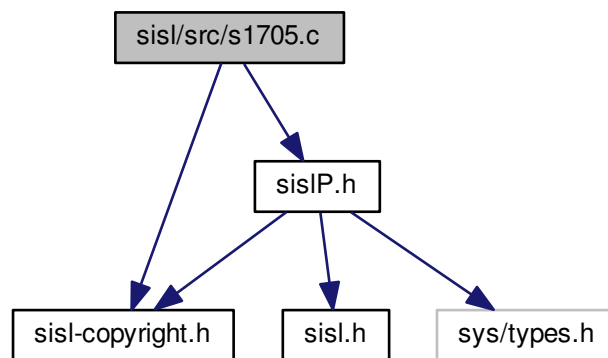
Definition at line 58 of file s1701.c.

## 30.2517 sisl/src/s1705.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1705.c:



## Macros

- #define [S1705](#)

## Functions

- void [s1705](#) ([SISLCurve](#) \*pc, int \*jstat)

### 30.2517.1 Macro Definition Documentation

#### 30.2517.1.1 #define S1705

Definition at line 49 of file s1705.c.

### 30.2517.2 Function Documentation

#### 30.2517.2.1 void s1705 ( SISLCurve \* pc, int \* jstat )

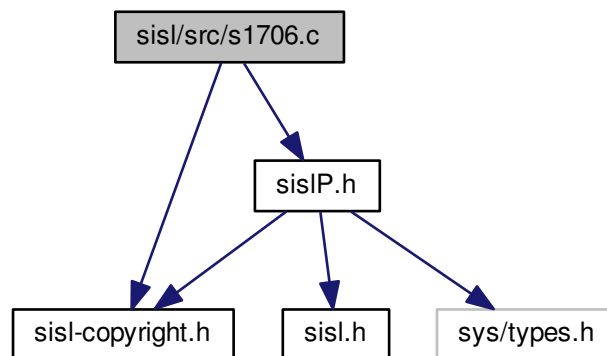
Definition at line 57 of file s1705.c.

## 30.2518 sisl/src/s1706.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1706.c:



### Macros

- #define [S1706](#)

### Functions

- void [s1706](#) (SISLCurve \*pc)

## 30.2518.1 Macro Definition Documentation

### 30.2518.1.1 #define S1706

Definition at line 49 of file s1706.c.

## 30.2518.2 Function Documentation

### 30.2518.2.1 void s1706 ( SISLCurve \* pc )

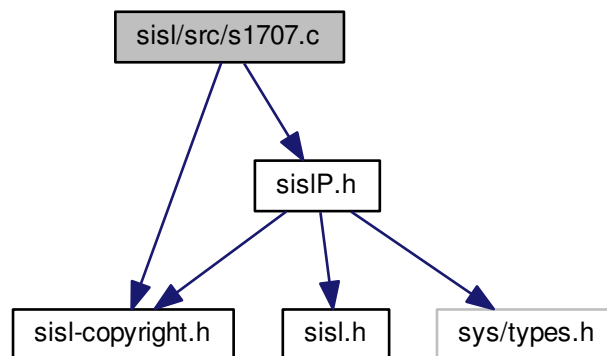
Definition at line 57 of file s1706.c.

## 30.2519 sisl/src/s1707.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1707.c:



## Macros

- #define [S1707](#)

## Functions

- void [s1707](#) (SISLCurve \*pc, int \*jstat)

### 30.2519.1 Macro Definition Documentation

#### 30.2519.1.1 #define S1707

Definition at line 49 of file s1707.c.

### 30.2519.2 Function Documentation

#### 30.2519.2.1 void s1707 ( SISLCurve \* pc, int \* jstat )

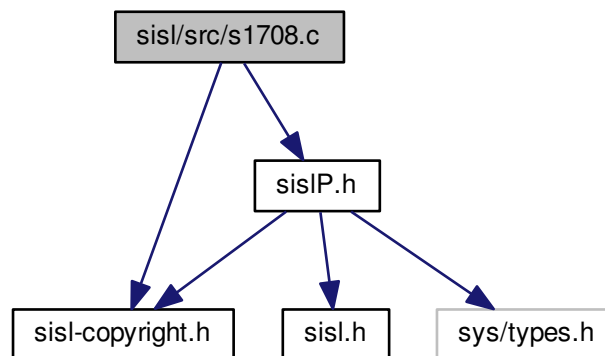
Definition at line 56 of file s1707.c.

## 30.2520 sisl/src/s1708.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1708.c:



### Macros

- #define [S1708](#)

### Functions

- void [s1708](#) (SISLSurf \*ps, int \*jstat)



## 30.2520.1 Macro Definition Documentation

### 30.2520.1.1 #define S1708

Definition at line 42 of file s1708.c.

## 30.2520.2 Function Documentation

### 30.2520.2.1 void s1708 ( SISLSurf \* ps, int \* jstat )

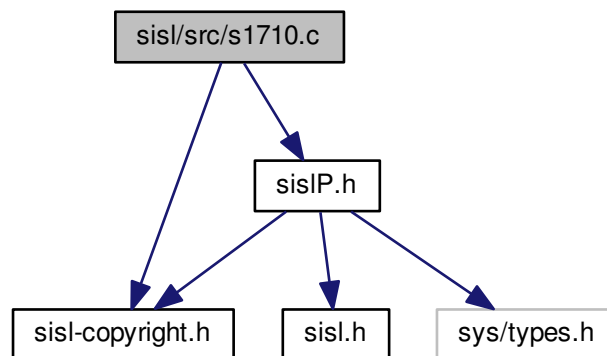
Definition at line 49 of file s1708.c.

## 30.2521 sisl/src/s1710.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1710.c:



## Macros

- #define [S1710](#)

## Functions

- void [s1710](#) ([SISLCurve](#) \*pc1, [double](#) apar, [SISLCurve](#) \*\*rcnew1, [SISLCurve](#) \*\*rcnew2, int \*jstat)

### 30.2521.1 Macro Definition Documentation

#### 30.2521.1.1 #define S1710

Definition at line 49 of file s1710.c.

### 30.2521.2 Function Documentation

#### 30.2521.2.1 void s1710 ( SISLCurve \* pc1, double apar, SISLCurve \*\* rcnew1, SISLCurve \*\* rcnew2, int \* jstat )

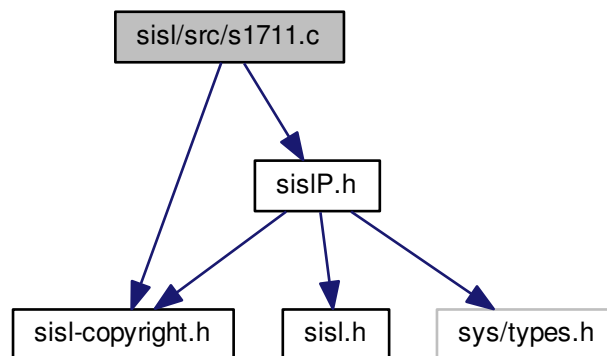
Definition at line 58 of file s1710.c.

## 30.2522 sisl/src/s1711.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1711.c:



### Macros

- #define [S1711](#)

### Functions

- void [s1711](#) (SISLSurf \*ps, int ipar, double apar, SISLSurf \*\*rsnew1, SISLSurf \*\*rsnew2, int \*jstat)

### 30.2522.1 Macro Definition Documentation

#### 30.2522.1.1 #define S1711

Definition at line 49 of file s1711.c.

### 30.2522.2 Function Documentation

#### 30.2522.2.1 void s1711 ( SISLSurf \* ps, int ipar, double apar, SISLSurf \*\* rsnew1, SISLSurf \*\* rsnew2, int \* jstat )

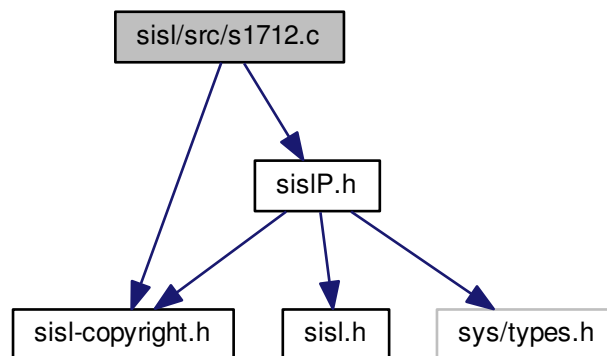
Definition at line 57 of file s1711.c.

## 30.2523 sisl/src/s1712.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1712.c:



### Macros

- #define [S1712](#)

### Functions

- void [s1712](#) ([SISLCurve](#) \*pc, [double](#) abeg, [double](#) aend, [SISLCurve](#) \*\*rcnew, int \*jstat)

### 30.2523.1 Macro Definition Documentation

#### 30.2523.1.1 #define S1712

Definition at line 49 of file s1712.c.

### 30.2523.2 Function Documentation

#### 30.2523.2.1 void s1712 ( SISLCurve \* pc, double abeg, double aend, SISLCurve \*\* rcnew, int \* jstat )

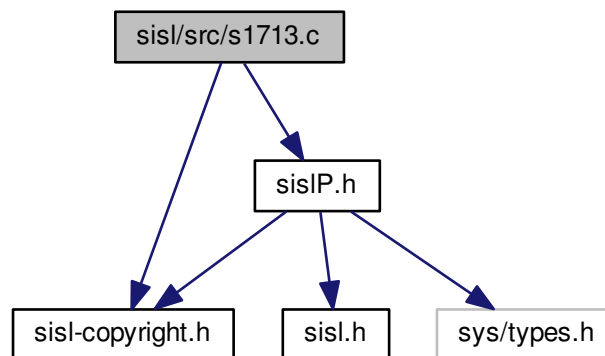
Definition at line 58 of file s1712.c.

## 30.2524 sisl/src/s1713.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1713.c:



### Macros

- #define [S1713](#)

### Functions

- void [s1713](#) (SISLCurve \*pc, double abeg, double aend, SISLCurve \*\*rcnew, int \*jstat)

### 30.2524.1 Macro Definition Documentation

#### 30.2524.1.1 #define S1713

Definition at line 49 of file s1713.c.

### 30.2524.2 Function Documentation

#### 30.2524.2.1 void s1713 ( SISLCurve \* pc, double abeg, double aend, SISLCurve \*\* rcnew, int \* jstat )

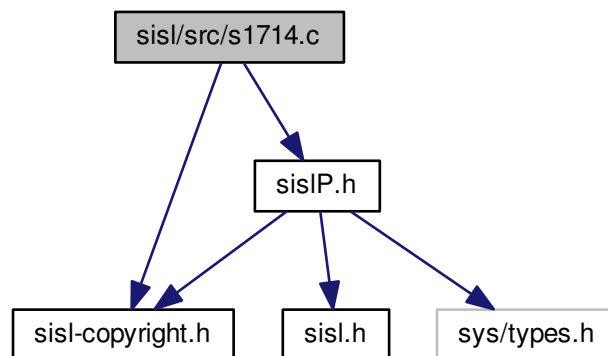
Definition at line 57 of file s1713.c.

## 30.2525 sisl/src/s1714.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1714.c:



### Macros

- #define [S1714](#)
- #define [SISL\\_CRV\\_PERIODIC](#) -1
- #define [SISL\\_CRV\\_OPEN](#) 1
- #define [SISL\\_CRV\\_CLOSED](#) 0

### Functions

- void [s1714](#) (SISLCurve \*pc, double apar1, double apar2, SISLCurve \*\*rcnew1, SISLCurve \*\*rcnew2, int \*jstat)

### 30.2525.1 Macro Definition Documentation

#### 30.2525.1.1 #define S1714

Definition at line 49 of file s1714.c.

#### 30.2525.1.2 #define SISL\_CRV\_CLOSED 0

Definition at line 53 of file s1714.c.

#### 30.2525.1.3 #define SISL\_CRV\_OPEN 1

Definition at line 52 of file s1714.c.

#### 30.2525.1.4 #define SISL\_CRV\_PERIODIC -1

Definition at line 51 of file s1714.c.

### 30.2525.2 Function Documentation

#### 30.2525.2.1 void s1714 ( SISLCurve \* pc, double apar1, double apar2, SISLCurve \*\* rcnew1, SISLCurve \*\* rcnew2, int \* jstat )

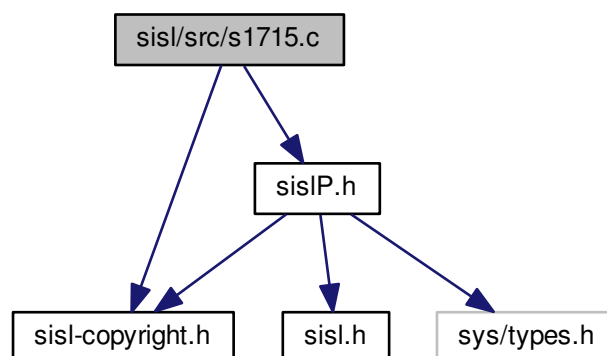
Definition at line 62 of file s1714.c.

### 30.2526 sisl/src/s1715.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1715.c:



## Macros

- `#define S1715`

## Functions

- `void s1715 (SISLCurve *pc1, SISLCurve *pc2, int iend1, int iend2, SISLCurve **rcnew, int *jstat)`

### 30.2526.1 Macro Definition Documentation

#### 30.2526.1.1 `#define S1715`

Definition at line 49 of file s1715.c.

### 30.2526.2 Function Documentation

#### 30.2526.2.1 `void s1715 ( SISLCurve * pc1, SISLCurve * pc2, int iend1, int iend2, SISLCurve ** rcnew, int * jstat )`

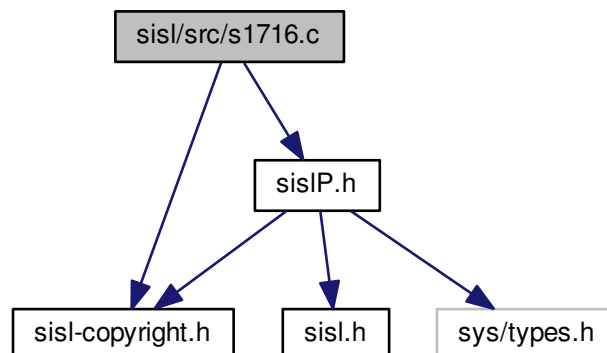
Definition at line 57 of file s1715.c.

## 30.2527 sisl/src/s1716.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1716.c:



## Macros

- `#define S1716`

## Functions

- void [s1716](#) ([SISLCurve](#) \*pc1, [SISLCurve](#) \*pc2, [double](#) aeps, [SISLCurve](#) \*\*rcnew, int \*jstat)

### 30.2527.1 Macro Definition Documentation

#### 30.2527.1.1 #define S1716

Definition at line 49 of file s1716.c.

### 30.2527.2 Function Documentation

#### 30.2527.2.1 void s1716 ( [SISLCurve](#) \* pc1, [SISLCurve](#) \* pc2, [double](#) aeps, [SISLCurve](#) \*\* rcnew, int \* jstat )

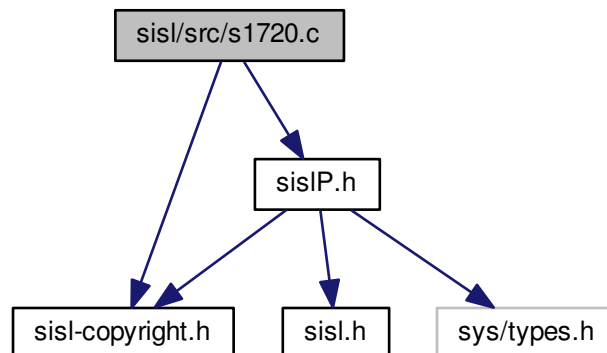
Definition at line 57 of file s1716.c.

## 30.2528 sisl/src/s1720.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1720.c:



## Macros

- #define [S1720](#)

## Functions

- void [s1720](#) ([SISLCurve](#) \*pc, int ider, [SISLCurve](#) \*\*rcnew, int \*jstat)



## 30.2528.1 Macro Definition Documentation

### 30.2528.1.1 #define S1720

Definition at line 49 of file s1720.c.

## 30.2528.2 Function Documentation

### 30.2528.2.1 void s1720 ( SISLCurve \* pc, int iber, SISLCurve \*\* rcnew, int \* jstat )

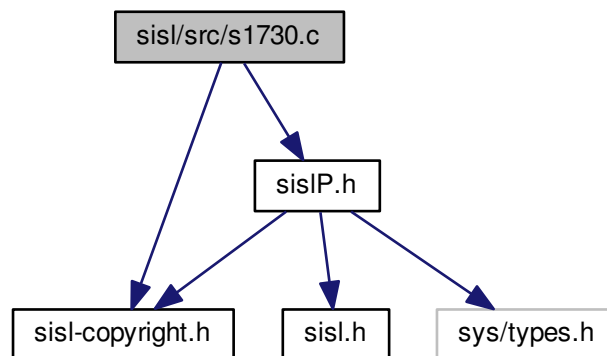
Definition at line 57 of file s1720.c.

## 30.2529 sisl/src/s1730.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1730.c:



## Macros

- #define [S1730](#)

## Functions

- void [s1730](#) ([SISLCurve](#) \*pc, [SISLCurve](#) \*\*rcnew, int \*jstat)

## 30.2529.1 Macro Definition Documentation

### 30.2529.1.1 #define S1730

Definition at line 49 of file s1730.c.

## 30.2529.2 Function Documentation

### 30.2529.2.1 void s1730 ( SISLCurve \* pc, SISLCurve \*\* rcnew, int \* jstat )

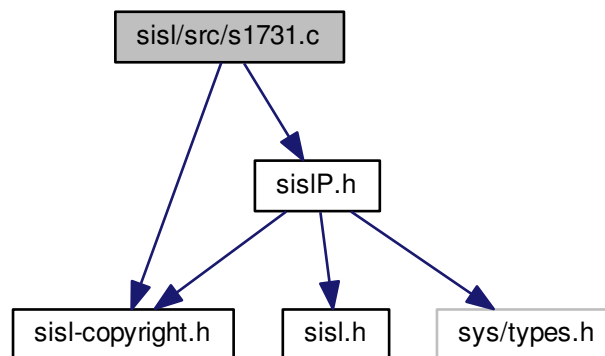
Definition at line 57 of file s1730.c.

## 30.2530 sisl/src/s1731.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1731.c:



### Macros

- #define [S1731](#)

### Functions

- void [s1731](#) ([SISLSurf](#) \*ps, [SISLSurf](#) \*\*rsnew, int \*jstat)

## 30.2530.1 Macro Definition Documentation

### 30.2530.1.1 #define S1731

Definition at line 49 of file s1731.c.

## 30.2530.2 Function Documentation

### 30.2530.2.1 void s1731 ( SISLSurf \* ps, SISLSurf \*\* rsnew, int \* jstat )

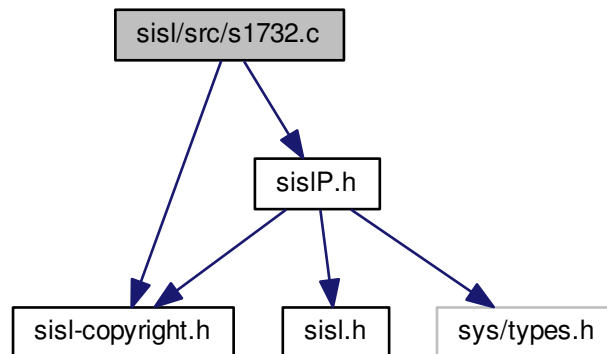
Definition at line 57 of file s1731.c.

## 30.2531 sisl/src/s1732.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1732.c:



## Macros

- #define [S1732](#)

## Functions

- void [s1732](#) (SISLCurve \*pc, int icont, double \*cstart, double \*cend, double \*gcoef, int \*jstat)

### 30.2531.1 Macro Definition Documentation

#### 30.2531.1.1 #define S1732

Definition at line 49 of file s1732.c.

### 30.2531.2 Function Documentation

#### 30.2531.2.1 void s1732 ( SISLCurve \* pc, int icont, double \* cstart, double \* cend, double \* gcoef, int \* jstat )

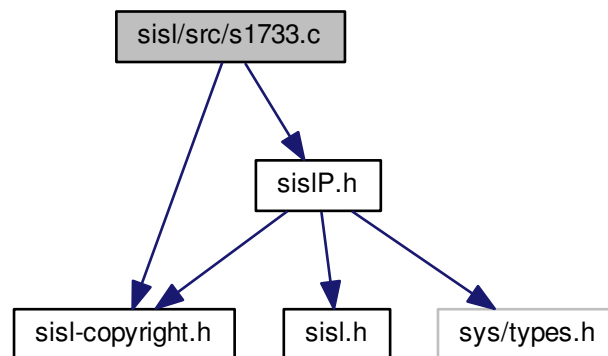
Definition at line 57 of file s1732.c.

## 30.2532 sisl/src/s1733.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1733.c:



### Macros

- #define [S1733](#)

### Functions

- void [s1733](#) (SISLSurf \*ps, int icont1, int icont2, double \*cstart1, double \*cend1, double \*cstart2, double \*cend2, double \*gcoef, int \*jstat)

### 30.2532.1 Macro Definition Documentation

#### 30.2532.1.1 #define S1733

Definition at line 49 of file s1733.c.

### 30.2532.2 Function Documentation

#### 30.2532.2.1 void s1733 ( SISLSurf \* ps, int icont1, int icont2, double \* cstart1, double \* cend1, double \* cstart2, double \* cend2, double \* gcoef, int \* jstat )

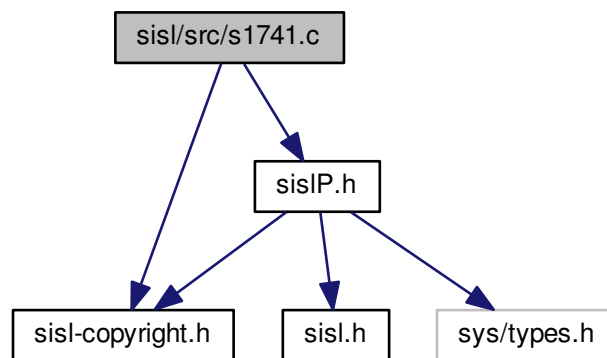
Definition at line 58 of file s1733.c.

## 30.2533 sisl/src/s1741.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1741.c:



### Macros

- #define [S1741](#)

### Functions

- void [s1741](#) (SISLObject \*po1, SISLObject \*po2, double aepsge, int \*jstat)

### 30.2533.1 Macro Definition Documentation

#### 30.2533.1.1 #define S1741

Definition at line 49 of file s1741.c.

### 30.2533.2 Function Documentation

#### 30.2533.2.1 void s1741 ( SISLObject \* po1, SISLObject \* po2, double aepsge, int \* jstat )

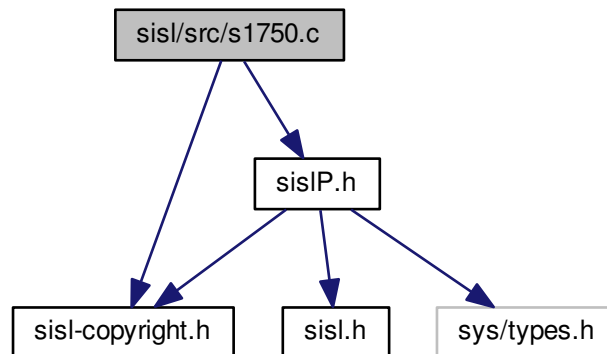
Definition at line 57 of file s1741.c.

## 30.2534 sisl/src/s1750.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1750.c:



### Macros

- #define [S1750](#)

### Functions

- void [s1750](#) (SISLCurve \*pc, int ikh, SISLCurve \*\*rc, int \*jstat)

## 30.2534.1 Macro Definition Documentation

### 30.2534.1.1 #define S1750

Definition at line 49 of file s1750.c.

## 30.2534.2 Function Documentation

### 30.2534.2.1 void s1750 ( SISLCurve \* pc, int ikh, SISLCurve \*\* rc, int \* jstat )

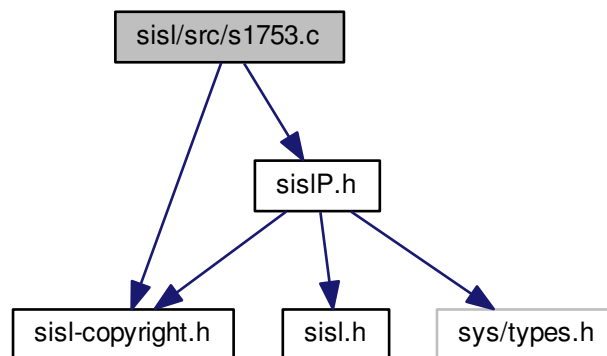
Definition at line 57 of file s1750.c.

## 30.2535 sisl/src/s1753.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1753.c:



## Macros

- #define [S1753](#)

## Functions

- void [s1753](#) (et, ecf, int in, int ik, int idim, etr, ecf, int inr, ecc, ecw, int \*jstat)

## 30.2535.1 Macro Definition Documentation

### 30.2535.1.1 #define S1753

Definition at line 49 of file s1753.c.

## 30.2535.2 Function Documentation

### 30.2535.2.1 void s1753 ( et , ecf , int in, int ik, int idim, etr , ectr , int inr, ecc , ecw , int \* jstat )

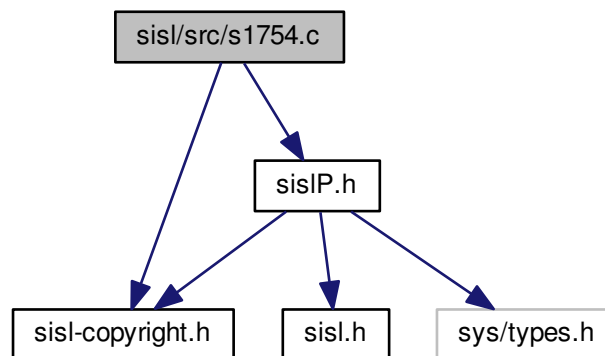
Definition at line 60 of file s1753.c.

## 30.2536 sisl/src/s1754.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1754.c:



## Macros

- #define [S1754](#)

## Functions

- void [s1754](#) (double \*et, int in, int ik, int ikh, double \*\*iknt, int \*inh, int \*jstat)



## 30.2536.1 Macro Definition Documentation

### 30.2536.1.1 #define S1754

Definition at line 49 of file s1754.c.

## 30.2536.2 Function Documentation

### 30.2536.2.1 void s1754 ( double \* et, int in, int ik, int ikh, double \*\* iknt, int \* inh, int \* jstat )

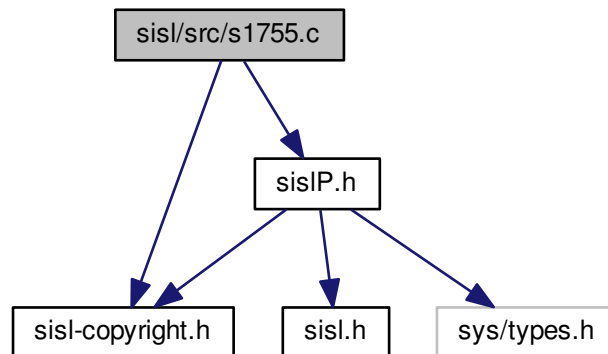
Definition at line 60 of file s1754.c.

## 30.2537 sisl/src/s1755.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1755.c:



## Macros

- #define [S1755](#)

## Functions

- void [s1755](#) (orknt, int in, int ik, extknt, int \*inh, int \*jstat)

### 30.2537.1 Macro Definition Documentation

#### 30.2537.1.1 #define S1755

Definition at line 49 of file s1755.c.

### 30.2537.2 Function Documentation

#### 30.2537.2.1 void s1755 ( orknt , int in, int ik, extknt , int \* inh, int \* jstat )

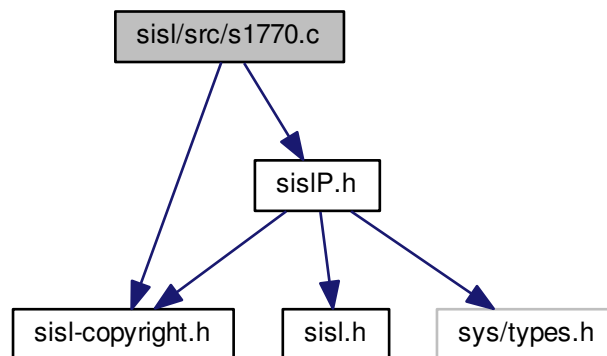
Definition at line 60 of file s1755.c.

## 30.2538 sisl/src/s1770.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1770.c:



### Macros

- #define [S1770](#)

### Functions

- void [s1770](#) ([SISLCurve](#) \*pcurve1, [SISLCurve](#) \*pcurve2, [double](#) aepsge, [double](#) astart1, [double](#) astart2, [double](#) aend1, [double](#) aend2, [double](#) anext1, [double](#) anext2, [double](#) \*cpos1, [double](#) \*cpos2, int \*jstat)

### 30.2538.1 Macro Definition Documentation

#### 30.2538.1.1 #define S1770

Definition at line 47 of file s1770.c.

### 30.2538.2 Function Documentation

30.2538.2.1 `void s1770 ( SISLCurve * pcurve1, SISLCurve * pcurve2, double aepsge, double astart1, double astart2, double aend1, double aend2, double anext1, double anext2, double * cpos1, double * cpos2, int * jstat )`

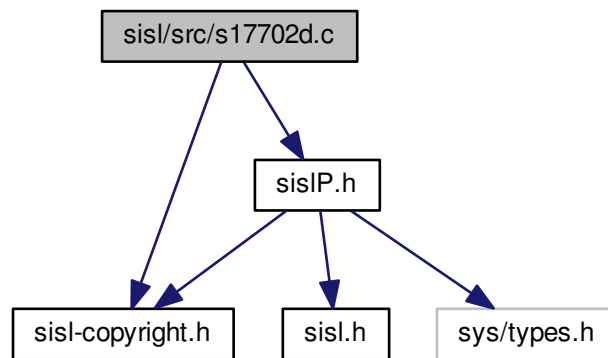
Definition at line 74 of file s1770.c.

## 30.2539 sisl/src/s17702d.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s17702d.c:



## Macros

- #define `S1770_2D`
- #define `SINGULAR` 1.0e-16
- #define `copy2(a, b, c)` for (ki=0;ki<(c);ki++) (a)[ki]=(b)[ki]
- #define `copy3(a, b, c, d)` for (ki=0;ki<(d);ki++) (a)[ki]=(b)[ki]=(c)[ki]
- #define `incr2(a, b, c)` for (ki=0;ki<(c);ki++) (a)[ki]+=(b)[ki]
- #define `decr2(a, b, c)` for (ki=0;ki<(c);ki++) (a)[ki]-=(b)[ki]
- #define `set_order(a)` {if((a)==1) order=0; else order=1;}

## Functions

- void `s1770_2D` (`SISLCurve *pcurve1`, `SISLCurve *pcurve2`, `double aepsge`, `double astart1`, `double astart2`, `double aend1`, `double aend2`, `double anext1`, `double anext2`, `double *cpos1`, `double *cpos2`, `int *jstat`)

### 30.2539.1 Macro Definition Documentation

30.2539.1.1 `#define copy2( a, b, c ) for (ki=0;ki<(c);ki++) (a)[ki]=(b)[ki]`

Definition at line 53 of file s17702d.c.

30.2539.1.2 `#define copy3( a, b, c, d ) for (ki=0;ki<(d);ki++) (a)[ki]=(b)[ki]=(c)[ki]`

Definition at line 54 of file s17702d.c.

30.2539.1.3 `#define decr2( a, b, c ) for (ki=0;ki<(c);ki++) (a)[ki]-=(b)[ki]`

Definition at line 56 of file s17702d.c.

30.2539.1.4 `#define incr2( a, b, c ) for (ki=0;ki<(c);ki++) (a)[ki]+=(b)[ki]`

Definition at line 55 of file s17702d.c.

30.2539.1.5 `#define S1770_2D`

Definition at line 47 of file s17702d.c.

30.2539.1.6 `#define set_order( a ) {if((a)==1) order=0; else order=1;}`

Definition at line 57 of file s17702d.c.

30.2539.1.7 `#define SINGULAR 1.0e-16`

Definition at line 52 of file s17702d.c.

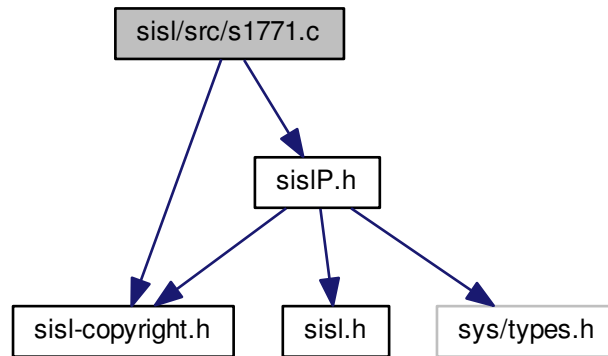
### 30.2539.2 Function Documentation

30.2539.2.1 void `s1770_2D` ( `SISLCurve *pcurve1`, `SISLCurve *pcurve2`, `double aepsge`, `double astart1`, `double astart2`, `double aend1`, `double aend2`, `double anext1`, `double anext2`, `double *cpos1`, `double *cpos2`, `int *jstat` )

Definition at line 100 of file s17702d.c.

## 30.2540 sisl/src/s1771.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for s1771.c:
```



### Macros

- `#define S1771`

### Functions

- void `s1771` (`SISLPoint` \*ppoint, `SISLCurve` \*pcurve, double aepsge, double astart, double aend, double anext, double \*cpos, int \*jstat)

#### 30.2540.1 Macro Definition Documentation

##### 30.2540.1.1 #define S1771

Definition at line 47 of file `s1771.c`.

#### 30.2540.2 Function Documentation

##### 30.2540.2.1 void s1771 ( `SISLPoint` \* ppoint, `SISLCurve` \* pcurve, double aepsge, double astart, double aend, double anext, double \* cpos, int \* jstat )

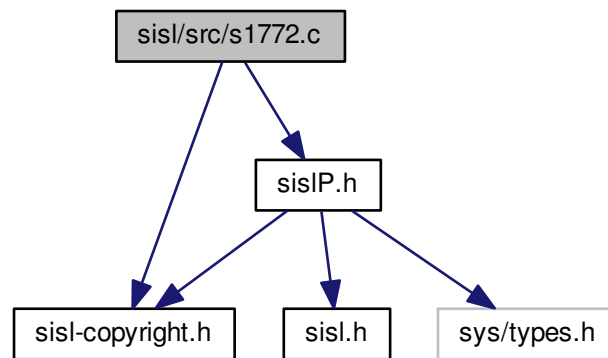
Definition at line 78 of file `s1771.c`.

## 30.2541 sisl/src/s1772.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1772.c:



### Macros

- `#define S1772`
- `#define SINGULAR 1.0e-16`
- `#define copy2(a, b, c) for (ki=0;ki<(c);ki++) (a)[ki]=(b)[ki]`
- `#define copy3(a, b, c, d) for (ki=0;ki<(d);ki++) (a)[ki]=(b)[ki]=(c)[ki]`
- `#define incr2(a, b, c) for (ki=0;ki<(c);ki++) (a)[ki]+=(b)[ki]`
- `#define decr2(a, b, c) for (ki=0;ki<(c);ki++) (a)[ki]-=(b)[ki]`
- `#define set_order(a) if((a)==1) {s_v=s_uu;order=0;} else {s_v=s_v1;order=1;}`

### Functions

- void `s1772` (SISLCurve \*pcurve, SISLSurf \*psurf, double aepsge, double astart1, estart2, double aend1, eend2, double anext1, enext2, double \*cpos1, gpos2, int \*jstat)

### 30.2541.1 Macro Definition Documentation

30.2541.1.1 `#define copy2( a, b, c ) for (ki=0;ki<(c);ki++) (a)[ki]=(b)[ki]`

Definition at line 52 of file s1772.c.

30.2541.1.2 `#define copy3( a, b, c, d ) for (ki=0;ki<(d);ki++) (a)[ki]=(b)[ki]=(c)[ki]`

Definition at line 53 of file s1772.c.

30.2541.1.3 `#define decr2( a, b, c ) for (ki=0;ki<(c);ki++) (a)[ki]-=(b)[ki]`

Definition at line 55 of file s1772.c.

30.2541.1.4 `#define incr2( a, b, c ) for (ki=0;ki<(c);ki++) (a)[ki]+=(b)[ki]`

Definition at line 54 of file s1772.c.

30.2541.1.5 `#define S1772`

Definition at line 47 of file s1772.c.

30.2541.1.6 `#define set_order( a ) if((a)==1) {s_v=s_uu;order=0;} else {s_v=s_v1;order=1;}`

Definition at line 56 of file s1772.c.

30.2541.1.7 `#define SINGULAR 1.0e-16`

Definition at line 51 of file s1772.c.

## 30.2541.2 Function Documentation

30.2541.2.1 `void s1772 ( SISLCurve * pcurve, SISLSurf * psurf, double aepsge, double astart1, estart2, double aend1, eend2, double anext1, enext2, double * cpos1, gpos2, int * jstat )`

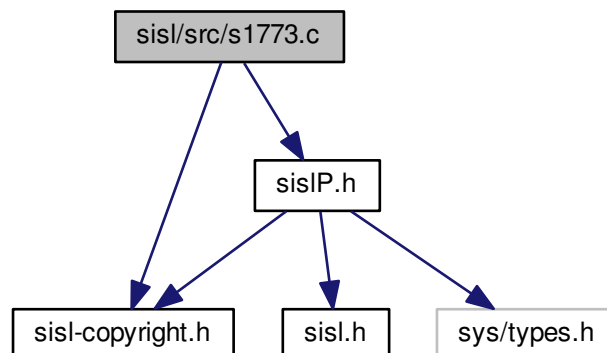
Definition at line 94 of file s1772.c.

## 30.2542 sisl/src/s1773.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1773.c:



## Macros

- #define [S1773](#)

## Functions

- void [s1773](#) ([SISLPoint](#) \*ppoint, [SISLSurf](#) \*psurf, [double](#) [aepsge](#), estart, eend, enext, gpos, int \*jstat)

### 30.2542.1 Macro Definition Documentation

#### 30.2542.1.1 #define S1773

Definition at line 47 of file s1773.c.

### 30.2542.2 Function Documentation

#### 30.2542.2.1 void s1773 ( [SISLPoint](#) \* *ppoint*, [SISLSurf](#) \* *psurf*, [double](#) *aepsge*, *estart* , *eend* , *enext* , *gpos* , *int* \* *jstat* )

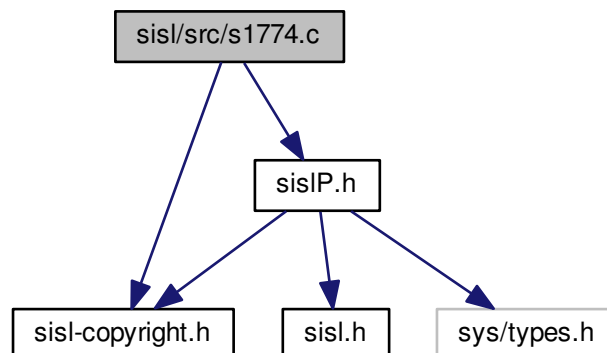
Definition at line 74 of file s1773.c.

## 30.2543 sisl/src/s1774.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1774.c:



## Macros

- #define [S1774](#)



## Functions

- void `s1774` (`SISLCurve *crv`, `point`, `int dim`, `double epsge`, `double start`, `double end`, `double guess`, `double *clpar`, `int *stat`)

### 30.2543.1 Macro Definition Documentation

#### 30.2543.1.1 #define S1774

Definition at line 47 of file s1774.c.

### 30.2543.2 Function Documentation

#### 30.2543.2.1 void s1774 ( SISLCurve \* crv, point , int dim, double epsge, double start, double end, double guess, double \* clpar, int \* stat )

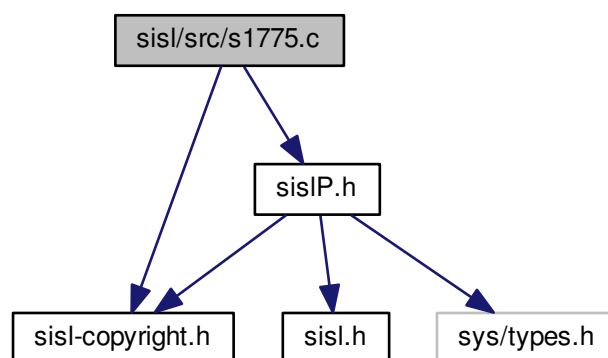
Definition at line 57 of file s1774.c.

## 30.2544 sisl/src/s1775.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1775.c:



## Macros

- #define `S1775`

## Functions

- void [s1775](#) ([SISLSurf](#) \*surf, point, int [dim](#), [double](#) [epsge](#), start, end, guess, cpar, int \*stat)

### 30.2544.1 Macro Definition Documentation

#### 30.2544.1.1 #define S1775

Definition at line 47 of file s1775.c.

### 30.2544.2 Function Documentation

#### 30.2544.2.1 void s1775 ( [SISLSurf](#) \* surf , int dim , [double](#) epsge , start , end , guess , cpar , int \* stat )

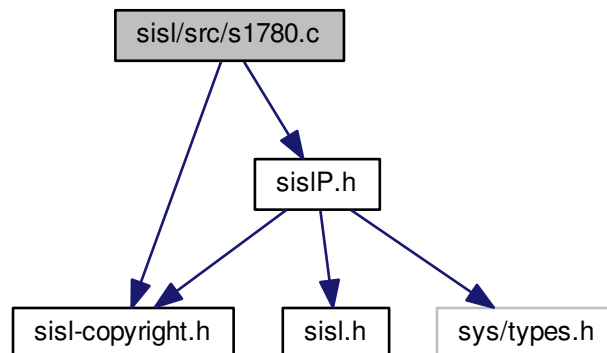
Definition at line 58 of file s1775.c.

## 30.2545 sisl/src/s1780.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1780.c:



## Macros

- #define [S1780](#)

## Functions

- void [s1780](#) ([SISLCurve](#) \*pc1, [SISLCurve](#) \*pc2, vipt, int \*jstat)

## 30.2545.1 Macro Definition Documentation

### 30.2545.1.1 #define S1780

Definition at line 49 of file s1780.c.

## 30.2545.2 Function Documentation

### 30.2545.2.1 void s1780 ( SISLCurve \* pc1, SISLCurve \* pc2, vipt, int \* jstat )

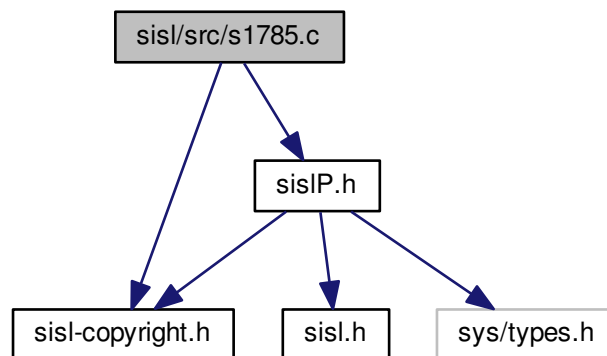
Definition at line 57 of file s1780.c.

## 30.2546 sisl/src/s1785.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1785.c:



## Macros

- #define [S1785](#)

## Functions

- void [s1785](#) (SISLCurve \*pcurve, SISLSurf \*psurf, double aepsge, epar1, epar2, int icur, int \*jstat)

### 30.2546.1 Macro Definition Documentation

#### 30.2546.1.1 #define S1785

Definition at line 47 of file s1785.c.

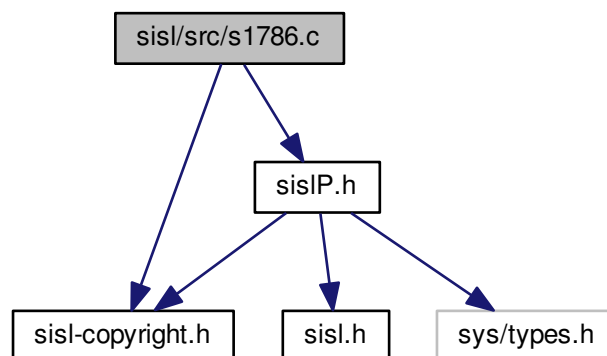
### 30.2546.2 Function Documentation

#### 30.2546.2.1 void s1785 ( SISLCurve \* pcurve, SISLSurf \* psurf, double aepsge, epar1 , epar2 , int icur, int \* jstat )

Definition at line 63 of file s1785.c.

## 30.2547 sisl/src/s1786.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for s1786.c:
```



### Macros

- #define [S1786](#)

### Typedefs

- typedef void(\* [fevalcProc](#)) ()

### Functions

- void [s1786](#) (SISLCurve \*pc1, SISLCurve \*pc2, double aepsge, epar1, epar2, int \*jstat)

### 30.2547.1 Macro Definition Documentation

#### 30.2547.1.1 #define S1786

Definition at line 47 of file s1786.c.

### 30.2547.2 Typedef Documentation

#### 30.2547.2.1 typedef void(\* fevalcProc) ()

Definition at line 51 of file s1786.c.

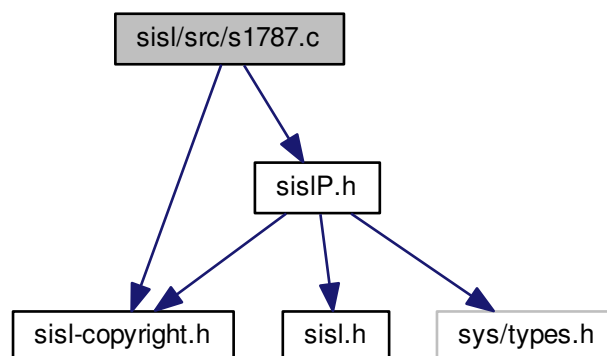
### 30.2547.3 Function Documentation

#### 30.2547.3.1 void s1786 ( SISLCurve \* pc1, SISLCurve \* pc2, double aeprge, epar1 , epar2 , int \* jstat )

Definition at line 77 of file s1786.c.

## 30.2548 sisl/src/s1787.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for s1787.c:
```



### Macros

- #define [S1787](#)

## Functions

- void [s1787](#) ([SISLSurf](#) \*ps, [double](#) alevel, [double](#) aepsge, epar, gpar1, gpar2, int \*jstat)

### 30.2548.1 Macro Definition Documentation

#### 30.2548.1.1 #define S1787

Definition at line 49 of file s1787.c.

### 30.2548.2 Function Documentation

#### 30.2548.2.1 void s1787 ( [SISLSurf](#) \* ps, [double](#) alevel, [double](#) aepsge, epar , gpar1 , gpar2 , int \* jstat )

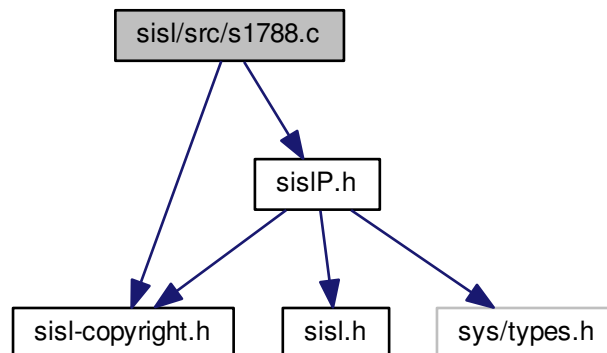
Definition at line 58 of file s1787.c.

## 30.2549 sisl/src/s1788.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1788.c:



## Macros

- #define [S1788](#)

## Functions

- void [s1788](#) ([SISLSurf](#) \*ps1, [SISLSurf](#) \*ps2, [double](#) aepsge, epar, gpar1, gpar2, int \*jstat)

## 30.2549.1 Macro Definition Documentation

### 30.2549.1.1 #define S1788

Definition at line 49 of file s1788.c.

## 30.2549.2 Function Documentation

### 30.2549.2.1 void s1788 ( SISLSurf \* ps1, SISLSurf \* ps2, double aepsge, epar , gpar1 , gpar2 , int \* jstat )

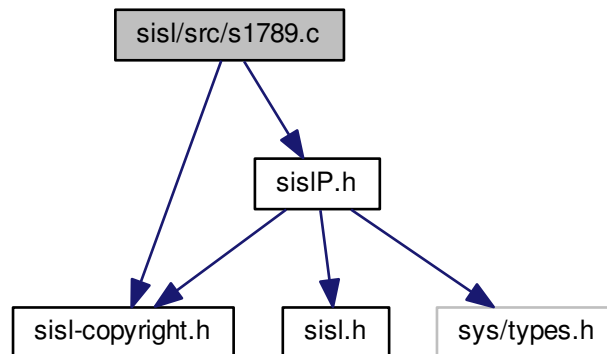
Definition at line 58 of file s1788.c.

## 30.2550 sisl/src/s1789.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1789.c:



## Macros

- #define [S1789](#)

## Functions

- void [s1789](#) (SISLPoint \*ppoint, SISLSurf \*psurf, double aepsge, epar1, epar2, int \*jstat)

## 30.2550.1 Macro Definition Documentation

### 30.2550.1.1 #define S1789

Definition at line 42 of file s1789.c.

## 30.2550.2 Function Documentation

### 30.2550.2.1 void s1789 ( SISLPoint \* ppoint, SISLSurf \* psurf, double aepsge, epar1 , epar2 , int \* jstat )

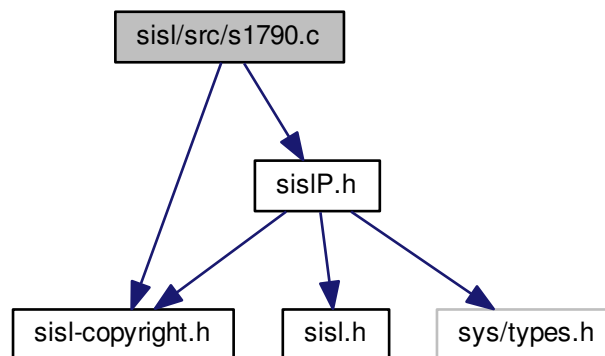
Definition at line 60 of file s1789.c.

## 30.2551 sisl/src/s1790.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1790.c:



## Macros

- #define [S1790](#)

## Functions

- void [s1790](#) (SISLObject \*po1, SISLObject \*po2, double aepsge, int \*jstat)



### 30.2551.1 Macro Definition Documentation

#### 30.2551.1.1 #define S1790

Definition at line 49 of file s1790.c.

### 30.2551.2 Function Documentation

#### 30.2551.2.1 void s1790 ( SISLObject \* *po1*, SISLObject \* *po2*, double *aepsge*, int \* *jstat* )

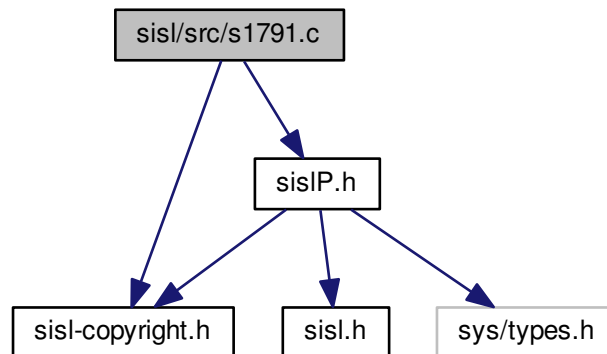
Definition at line 57 of file s1790.c.

## 30.2552 sisl/src/s1791.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1791.c:



### Macros

- #define [S1791](#)

### Functions

- int [s1791](#) (et, int ik, int in)

## 30.2552.1 Macro Definition Documentation

### 30.2552.1.1 #define S1791

Definition at line 49 of file s1791.c.

## 30.2552.2 Function Documentation

### 30.2552.2.1 int s1791 ( et , int ik, int in )

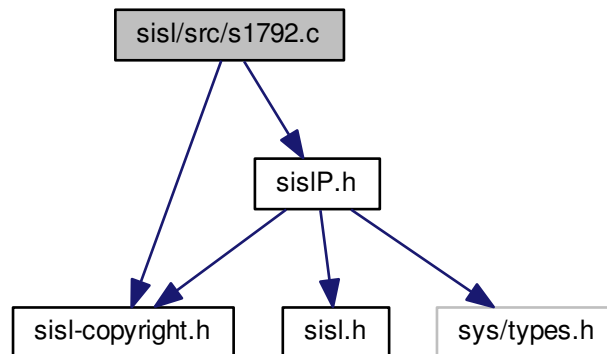
Definition at line 57 of file s1791.c.

## 30.2553 sisl/src/s1792.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1792.c:



## Macros

- `#define S1792`

## Functions

- `double s1792` (et, int ik, int in)

### 30.2553.1 Macro Definition Documentation

#### 30.2553.1.1 #define S1792

Definition at line 49 of file s1792.c.

### 30.2553.2 Function Documentation

#### 30.2553.2.1 double s1792 ( et , int ik, int in )

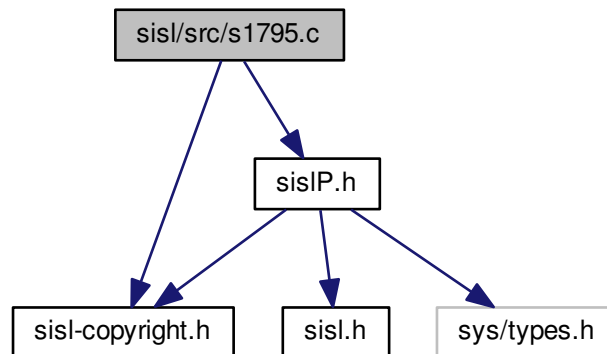
Definition at line 57 of file s1792.c.

## 30.2554 sisl/src/s1795.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1795.c:



### Macros

- #define [S1795](#)

### Functions

- void [s1795](#) ([SISLSurf](#) \*ps1, [SISLSurf](#) \*ps2, [double](#) aepsge, [double](#) aang, int \*jstat)

## 30.2554.1 Macro Definition Documentation

### 30.2554.1.1 #define S1795

Definition at line 49 of file s1795.c.

## 30.2554.2 Function Documentation

### 30.2554.2.1 void s1795 ( SISLSurf \* ps1, SISLSurf \* ps2, double aepsge, double aang, int \* jstat )

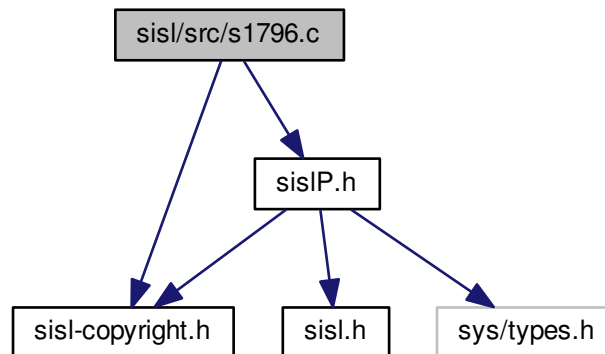
Definition at line 57 of file s1795.c.

## 30.2555 sisl/src/s1796.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1796.c:



## Macros

- #define [S1796](#)

## Functions

- void [s1796](#) ([SISLCurve](#) \*pc1, [SISLCurve](#) \*pc2, [double](#) aepsge, [double](#) aang, int \*jstat)

## 30.2555.1 Macro Definition Documentation

### 30.2555.1.1 #define S1796

Definition at line 47 of file s1796.c.

## 30.2555.2 Function Documentation

### 30.2555.2.1 void s1796 ( SISLCurve \* pc1, SISLCurve \* pc2, double aepsge, double aang, int \* jstat )

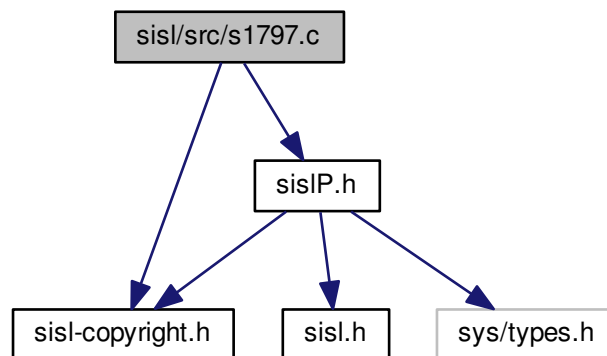
Definition at line 55 of file s1796.c.

## 30.2556 sisl/src/s1797.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1797.c:



## Macros

- #define [S1797](#)

## Functions

- void [s1797](#) (SISLSurf \*ps1, SISLCurve \*pc1, double aepsge, double aang, int \*jstat)

### 30.2556.1 Macro Definition Documentation

#### 30.2556.1.1 #define S1797

Definition at line 47 of file s1797.c.

### 30.2556.2 Function Documentation

#### 30.2556.2.1 void s1797 ( SISLSurf \* ps1, SISLCurve \* pc1, double aepsge, double aang, int \* jstat )

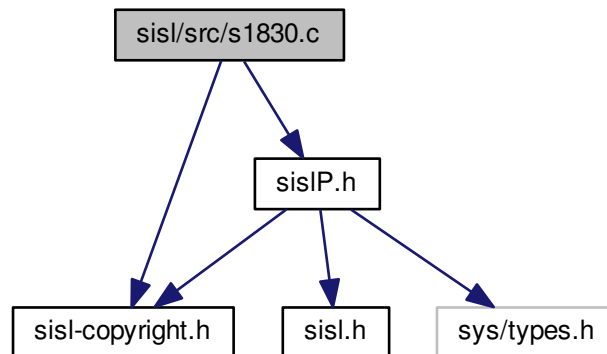
Definition at line 55 of file s1797.c.

## 30.2557 sisl/src/s1830.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1830.c:



### Macros

- #define [S1830](#)

### Functions

- void [s1830](#) (SISLSurf \*psurf, SISLCurve \*pcurve, int \*jstat)

### 30.2557.1 Macro Definition Documentation

#### 30.2557.1.1 #define S1830

Definition at line 49 of file s1830.c.

### 30.2557.2 Function Documentation

#### 30.2557.2.1 void s1830 ( SISLSurf \* *psurf*, SISLCurve \* *pcurve*, int \* *jstat* )

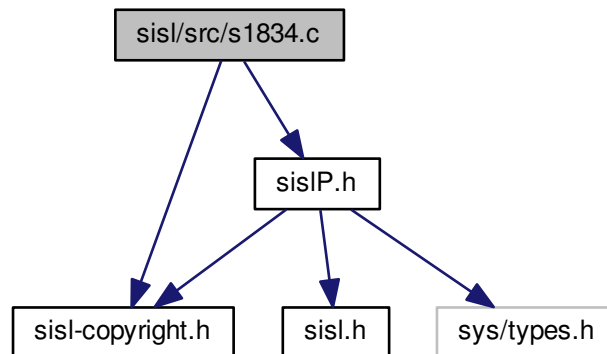
Definition at line 57 of file s1830.c.

## 30.2558 sisl/src/s1834.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1834.c:



### Macros

- #define [S1834](#)

### Functions

- void [s1834](#) (ecoef1, int in1, ecoef2, int in2, int idim, edir1, edir2, int \*jstat)

## 30.2558.1 Macro Definition Documentation

### 30.2558.1.1 #define S1834

Definition at line 49 of file s1834.c.

## 30.2558.2 Function Documentation

### 30.2558.2.1 void s1834 ( ecoef1 , int in1, ecoef2 , int in2, int idim, edir1 , edir2 , int \* jstat )

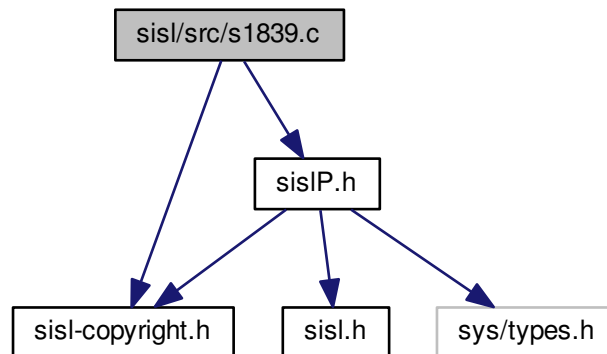
Definition at line 73 of file s1834.c.

## 30.2559 sisl/src/s1839.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1839.c:



### Macros

- #define [S1839](#)

### Functions

- void [s1839](#) ([SISLSurf](#) \*ps1, epol, int in, int idim, int \*jstat)



## 30.2559.1 Macro Definition Documentation

### 30.2559.1.1 #define S1839

Definition at line 49 of file s1839.c.

## 30.2559.2 Function Documentation

### 30.2559.2.1 void s1839 ( SISLSurf \* ps1, epol , int in, int idim, int \* jstat )

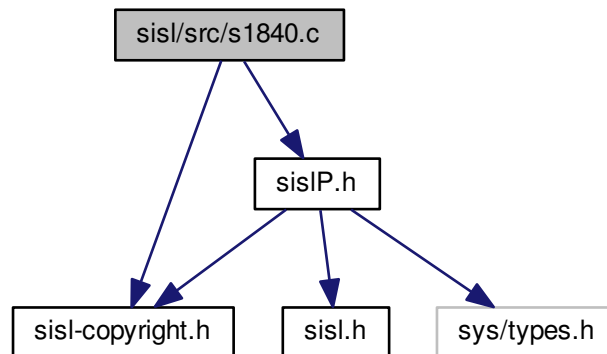
Definition at line 57 of file s1839.c.

## 30.2560 sisl/src/s1840.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1840.c:



## Macros

- #define [S1840](#)

## Functions

- void [s1840](#) (SISLCurve \*pcurve, double \*cdist, int \*jstat)

### 30.2560.1 Macro Definition Documentation

#### 30.2560.1.1 #define S1840

Definition at line 49 of file s1840.c.

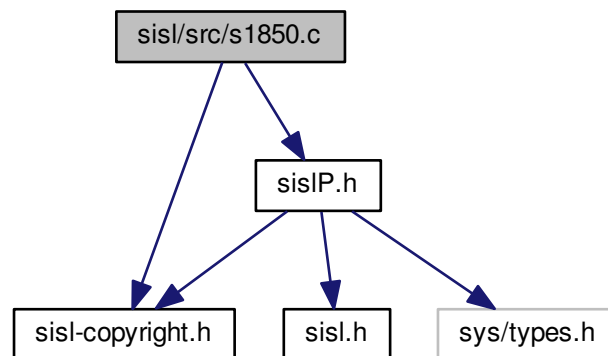
### 30.2560.2 Function Documentation

#### 30.2560.2.1 void s1840 ( SISLCurve \* pcurve, double \* cdist, int \* jstat )

Definition at line 57 of file s1840.c.

## 30.2561 sisl/src/s1850.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for s1850.c:
```



### Macros

- #define [S1850](#)

### Functions

- void [s1850](#) ([SISLCurve](#) \*pc1, epoint, enorm, int idim, [double](#) aepsco, [double](#) aepsge, int \*jpt, [double](#) \*\*gpar, int \*jcrv, [SISLIntcurve](#) \*\*\*wcurve, int \*jstat)

### 30.2561.1 Macro Definition Documentation

#### 30.2561.1.1 #define S1850

Definition at line 49 of file s1850.c.

### 30.2561.2 Function Documentation

#### 30.2561.2.1 void s1850 ( SISLCurve \* *pc1*, epoint, enorm, int *idim*, double *aepsco*, double *aepsge*, int \* *jpt*, double \*\* *gpar*, int \* *jcrv*, SISLIntcurve \*\*\* *wcurve*, int \* *jstat* )

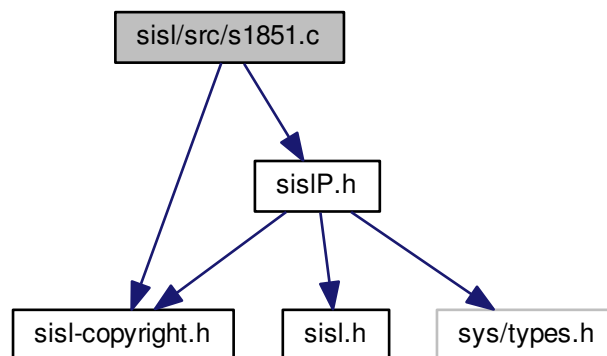
Definition at line 58 of file s1850.c.

## 30.2562 sisl/src/s1851.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1851.c:



### Macros

- #define [S1851](#)

### Functions

- void [s1851](#) (SISLSurf \**ps1*, epoint, enorm, int *idim*, double *aepsco*, double *aepsge*, int \**jpt*, double \*\**gpar*, int \**jcrv*, SISLIntcurve \*\*\**wcurve*, int \**jstat*)

### 30.2562.1 Macro Definition Documentation

#### 30.2562.1.1 #define S1851

Definition at line 49 of file s1851.c.

### 30.2562.2 Function Documentation

#### 30.2562.2.1 void s1851 ( SISLSurf \* ps1, epoint, enorm, int idim, double aepsco, double aepsge, int \* jpt, double \*\* gpar, int \* jcrv, SISLIntcurve \*\*\* wcurve, int \* jstat )

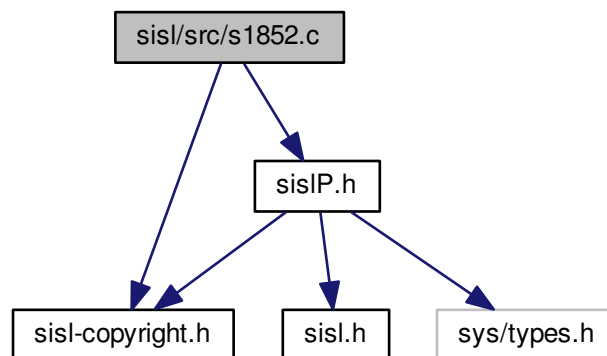
Definition at line 58 of file s1851.c.

## 30.2563 sisl/src/s1852.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1852.c:



### Macros

- #define [S1852](#)

### Functions

- void [s1852](#) (SISLSurf \*ps1, ecenter, double aradius, int idim, double aepsco, double aepsge, int \*jpt, double \*\*gpar, int \*jcrv, SISLIntcurve \*\*\*wcurve, int \*jstat)

### 30.2563.1 Macro Definition Documentation

#### 30.2563.1.1 #define S1852

Definition at line 49 of file s1852.c.

### 30.2563.2 Function Documentation

#### 30.2563.2.1 void s1852 ( SISLSurf \* ps1, ecenter , double aradius, int idim, double aepsco, double aepsge, int \* jpt, double \*\* gpar, int \* jcrv, SISLIntcurve \*\*\* wcurve, int \* jstat )

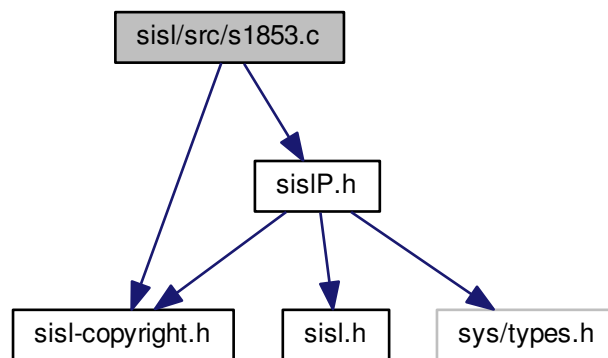
Definition at line 59 of file s1852.c.

## 30.2564 sisl/src/s1853.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1853.c:



### Macros

- #define [S1853](#)

### Functions

- void [s1853](#) (SISLSurf \*ps1, epoint, edirec, double aradius, int idim, double aepsco, double aepsge, int \*jpt, double \*\*gpar, int \*jcrv, SISLIntcurve \*\*\*wcurve, int \*jstat)

### 30.2564.1 Macro Definition Documentation

#### 30.2564.1.1 #define S1853

Definition at line 49 of file s1853.c.

### 30.2564.2 Function Documentation

#### 30.2564.2.1 void s1853 ( SISLSurf \* ps1, epoint, edirec, double aradius, int idim, double aepsco, double aepsge, int \* jpt, double \*\* gpar, int \* jcrv, SISLIntcurve \*\*\* wcurve, int \* jstat )

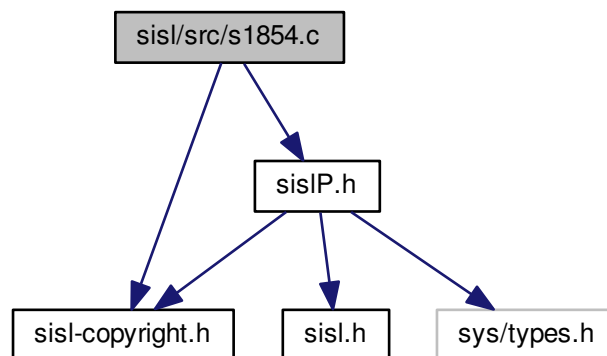
Definition at line 58 of file s1853.c.

## 30.2565 sisl/src/s1854.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1854.c:



### Macros

- #define [S1854](#)

### Functions

- void [s1854](#) (SISLSurf \*ps1, etop, eaxis, econc, int idim, double aepsco, double aepsge, int \*jpt, double \*\*gpar, int \*jcrv, SISLIntcurve \*\*\*wcurve, int \*jstat)

### 30.2565.1 Macro Definition Documentation

#### 30.2565.1.1 #define S1854

Definition at line 49 of file s1854.c.

### 30.2565.2 Function Documentation

#### 30.2565.2.1 void s1854 ( SISLSurf \* ps1, etop , eaxis , econe , int idim, double aepsco, double aepsge, int \* jpt, double \*\* gpar, int \* jcrv, SISLIntcurve \*\*\* wcurve, int \* jstat )

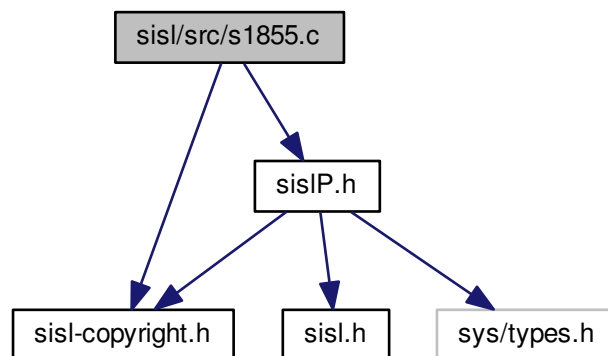
Definition at line 58 of file s1854.c.

## 30.2566 sisl/src/s1855.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1855.c:



### Macros

- #define [S1855](#)

### Functions

- void [s1855](#) (SISLSurf \*ps1, ecentr, [double](#) aradius, enorm, int idim, [double](#) aepsco, [double](#) aepsge, int \*jpt, [double](#) \*\*gpar, int \*jcrv, [SISLIntcurve](#) \*\*\*wcurve, int \*jstat)

### 30.2566.1 Macro Definition Documentation

#### 30.2566.1.1 #define S1855

Definition at line 49 of file s1855.c.

### 30.2566.2 Function Documentation

#### 30.2566.2.1 void s1855 ( SISLSurf \* ps1, ecentr , double aradius, enorm , int idim, double aepsco, double aepsge, int \* jpt, double \*\* gpar, int \* jcrv, SISLIntcurve \*\*\* wcurve, int \* jstat )

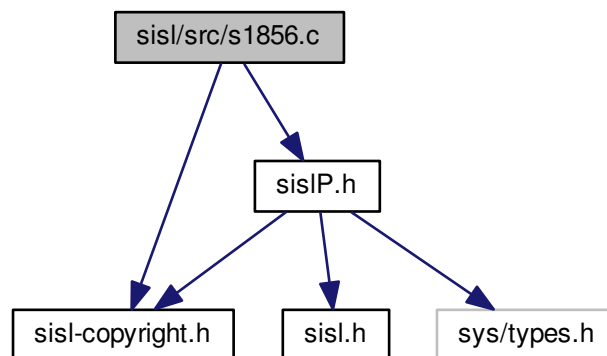
Definition at line 58 of file s1855.c.

## 30.2567 sisl/src/s1856.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1856.c:



### Macros

- #define [S1856](#)

### Functions

- void [s1856](#) (SISLSurf \*ps1, epoint, edir, int idim, [double](#) aepsco, [double](#) aepsge, int \*jpt, [double](#) \*\*gpar, int \*jcrv, [SISLIntcurve](#) \*\*\*wcurve, int \*jstat)



### 30.2567.1 Macro Definition Documentation

#### 30.2567.1.1 #define S1856

Definition at line 49 of file s1856.c.

### 30.2567.2 Function Documentation

#### 30.2567.2.1 void s1856 ( SISLSurf \* ps1, epoint, edir, int idim, double aepsco, double aepsge, int \* jpt, double \*\* gpar, int \* jcrv, SISLIntcurve \*\*\* wcurve, int \* jstat )

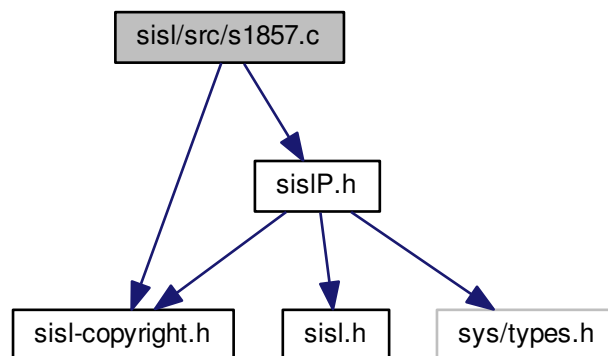
Definition at line 58 of file s1856.c.

## 30.2568 sisl/src/s1857.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1857.c:



### Macros

- #define [S1857](#)

### Functions

- void [s1857](#) (SISLCurve \*pc1, SISLCurve \*pc2, double aepsco, double aepsge, int \*jpt, double \*\*gpar1, double \*\*gpar2, int \*jcrv, SISLIntcurve \*\*\*wcurve, int \*jstat)

### 30.2568.1 Macro Definition Documentation

#### 30.2568.1.1 #define S1857

Definition at line 49 of file s1857.c.

### 30.2568.2 Function Documentation

#### 30.2568.2.1 void s1857 ( SISLCurve \* pc1, SISLCurve \* pc2, double aepsco, double aepsge, int \* jpt, double \*\* gpar1, double \*\* gpar2, int \* jcrv, SISLIntcurve \*\*\* wcurve, int \* jstat )

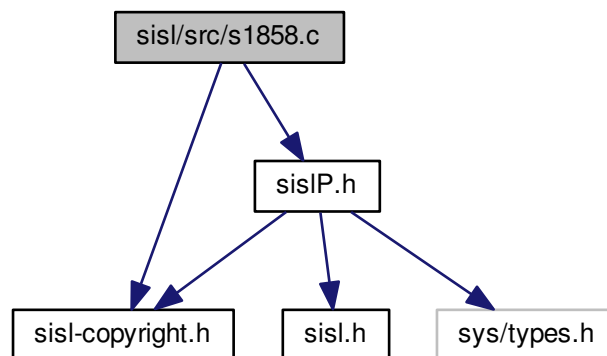
Definition at line 58 of file s1857.c.

## 30.2569 sisl/src/s1858.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1858.c:



### Macros

- #define [S1858](#)

### Functions

- void [s1858](#) (SISLSurf \*ps1, SISLCurve \*pc1, double aepsco, double aepsge, int \*jpt, double \*\*gpar1, double \*\*gpar2, int \*jcrv, SISLIntcurve \*\*\*wcurve, int \*jstat)

### 30.2569.1 Macro Definition Documentation

#### 30.2569.1.1 #define S1858

Definition at line 49 of file s1858.c.

### 30.2569.2 Function Documentation

#### 30.2569.2.1 void s1858 ( SISLSurf \* ps1, SISLCurve \* pc1, double aepsco, double aepsge, int \* jpt, double \*\* gpar1, double \*\* gpar2, int \* jcrv, SISLIntcurve \*\*\* wcurve, int \* jstat )

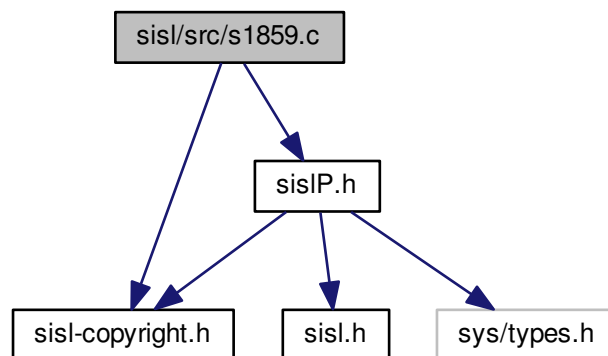
Definition at line 58 of file s1858.c.

## 30.2570 sisl/src/s1859.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1859.c:



### Macros

- #define [S1859](#)

### Functions

- void [s1859](#) (SISLSurf \*ps1, SISLSurf \*ps2, double aepsco, double aepsge, int \*jpt, double \*\*gpar1, double \*\*gpar2, int \*jcrv, SISLIntcurve \*\*\*wcurve, int \*jstat)

### 30.2570.1 Macro Definition Documentation

#### 30.2570.1.1 #define S1859

Definition at line 49 of file s1859.c.

### 30.2570.2 Function Documentation

#### 30.2570.2.1 void s1859 ( SISLSurf \* ps1, SISLSurf \* ps2, double aepsco, double aepsge, int \* jpt, double \*\* gpar1, double \*\* gpar2, int \* jcrv, SISLIntcurve \*\*\* wcurve, int \* jstat )

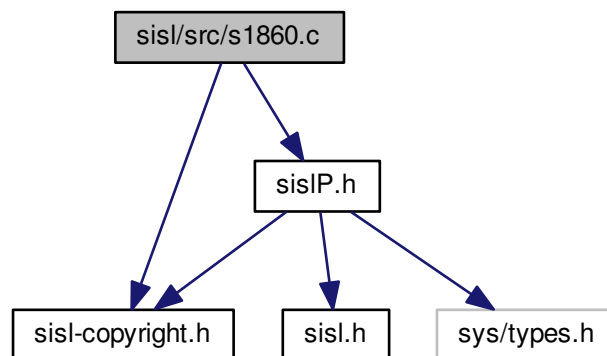
Definition at line 58 of file s1859.c.

## 30.2571 sisl/src/s1860.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1860.c:



### Macros

- #define [S1860](#)

### Functions

- void [s1860](#) (SISLSurf \*ps, eview, int idim, double aepsco, double aepsge, int \*jpt, double \*\*gpar, int \*jcrv, SISLIntcurve \*\*\*wcurve, int \*jstat)

### 30.2571.1 Macro Definition Documentation

#### 30.2571.1.1 #define S1860

Definition at line 49 of file s1860.c.

### 30.2571.2 Function Documentation

#### 30.2571.2.1 void s1860 ( SISLSurf \* ps, eview , int idim, double aepsco, double aepsge, int \* jpt, double \*\* gpar, int \* jcrv, SISLIntcurve \*\*\* wcurve, int \* jstat )

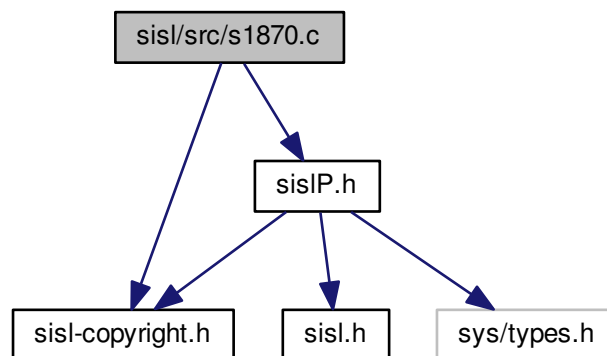
Definition at line 57 of file s1860.c.

## 30.2572 sisl/src/s1870.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1870.c:



### Macros

- #define [S1870](#)

### Functions

- void [s1870](#) (SISLSurf \*ps1, double \*pt1, int idim, double aepsge, int \*jpt, double \*\*gpar1, int \*jcrv, SISLIntcurve \*\*\*wcurve, int \*jstat)

### 30.2572.1 Macro Definition Documentation

#### 30.2572.1.1 #define S1870

Definition at line 42 of file s1870.c.

### 30.2572.2 Function Documentation

#### 30.2572.2.1 void s1870 ( SISLSurf \* ps1, double \* pt1, int idim, double aepsge, int \* jpt, double \*\* gpar1, int \* jcrv, SISLIntcurve \*\*\* wcurve, int \* jstat )

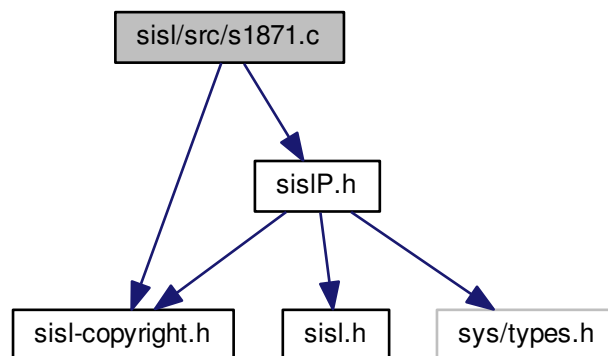
Definition at line 51 of file s1870.c.

## 30.2573 sisl/src/s1871.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1871.c:



### Macros

- #define [S1871](#)

### Functions

- void [s1871](#) ([SISLCurve](#) \*pc1, [double](#) \*pt1, int idim, [double](#) aepsge, int \*jpt, [double](#) \*\*gpar1, int \*jcrv, [SISLIntcurve](#) \*\*\*wcurve, int \*jstat)

### 30.2573.1 Macro Definition Documentation

#### 30.2573.1.1 #define S1871

Definition at line 42 of file s1871.c.

### 30.2573.2 Function Documentation

#### 30.2573.2.1 void s1871 ( SISLCurve \* pc1, double \* pt1, int idim, double aepsge, int \* jpt, double \*\* gpar1, int \* jcrv, SISLIntcurve \*\*\* wcurve, int \* jstat )

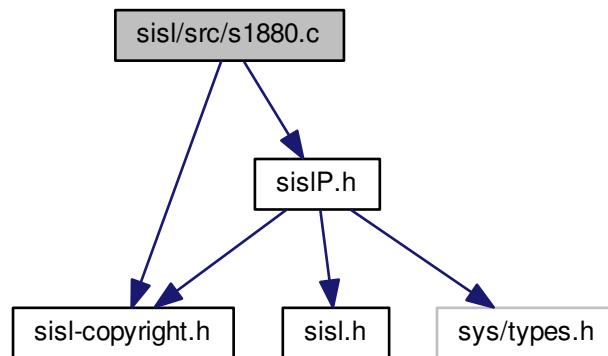
Definition at line 51 of file s1871.c.

## 30.2574 sisl/src/s1880.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1880.c:



### Macros

- #define [S1880](#)

### Functions

- void [s1880](#) (int ipar1, int ipar2, int \*jpt, [SISLIntpt](#) \*\*vpoint, int \*jlist, [SISLIntlist](#) \*\*vlist, int \*jpar, double \*\*gpar1, double \*\*gpar2, int \*jcrv, [SISLIntcurve](#) \*\*\*wcrv, int \*jstat)

### 30.2574.1 Macro Definition Documentation

#### 30.2574.1.1 #define S1880

Definition at line 49 of file s1880.c.

### 30.2574.2 Function Documentation

#### 30.2574.2.1 void s1880 ( int *ipar1*, int *ipar2*, int \* *jpt*, SISLIntpt \*\* *vpoint*, int \* *jlist*, SISLIntlist \*\* *vlist*, int \* *jpar*, double \*\* *gpar1*, double \*\* *gpar2*, int \* *jcrv*, SISLIntcurve \*\*\* *wcrv*, int \* *jstat* )

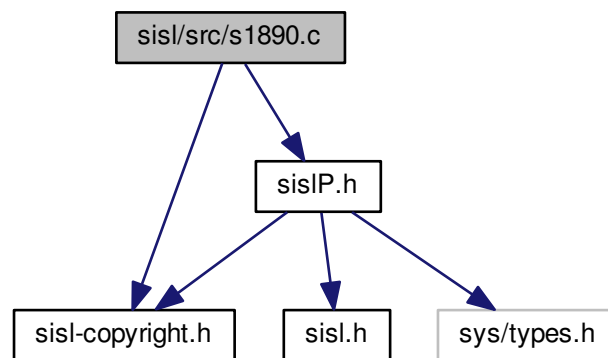
Definition at line 58 of file s1880.c.

## 30.2575 sisl/src/s1890.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1890.c:



### Macros

- #define [S1890](#)

### Functions

- void [s1890](#) (oknots, int oik, int oin, par, der, int \*jstat)



### 30.2575.1 Macro Definition Documentation

#### 30.2575.1.1 #define S1890

Definition at line 49 of file s1890.c.

### 30.2575.2 Function Documentation

#### 30.2575.2.1 void s1890 ( oknots , int oik, int oin, par , der , int \* jstat )

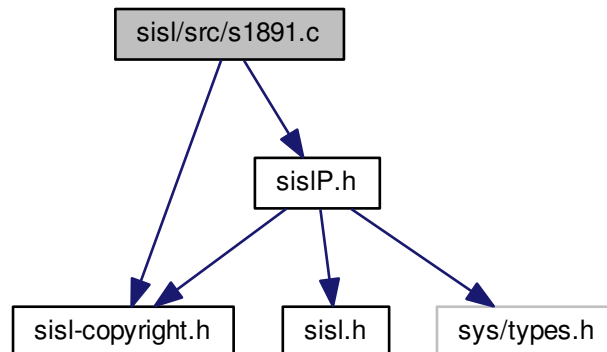
Definition at line 59 of file s1890.c.

## 30.2576 sisl/src/s1891.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1891.c:



### Macros

- #define [S1891](#)
- #define [MAX\\_SIZE](#) 50

### Functions

- void [s1891](#) (etau, epoint, int idim, int inbpt, int igrht, eder, int iopen, et, ebcoef, int \*in, int ik, int inlr, int incr, int \*jstat)

### 30.2576.1 Macro Definition Documentation

#### 30.2576.1.1 `#define MAX_SIZE 50`

Definition at line 52 of file s1891.c.

#### 30.2576.1.2 `#define S1891`

Definition at line 49 of file s1891.c.

### 30.2576.2 Function Documentation

#### 30.2576.2.1 `void s1891 ( etau , epoint , int idim , int inbpnt , int iright , eder , int iopen , et , ebcoef , int * in , int ik , int inlr , int inrc , int * jstat )`

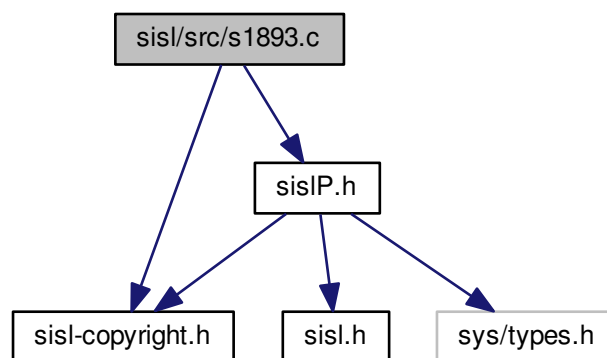
Definition at line 63 of file s1891.c.

### 30.2577 sisl/src/s1893.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1893.c:



### Macros

- `#define S1893`

## Functions

- void [s1893](#) ([SISLCurve](#) \*orig, earray, int dimp1, int narr, int der1, int der2, [SISLCurve](#) \*\*ncurve, int \*jstat)

### 30.2577.1 Macro Definition Documentation

#### 30.2577.1.1 #define S1893

Definition at line 49 of file s1893.c.

### 30.2577.2 Function Documentation

#### 30.2577.2.1 void s1893 ( [SISLCurve](#) \* orig, earray , int dimp1, int narr, int der1, int der2, [SISLCurve](#) \*\* ncurve, int \* jstat )

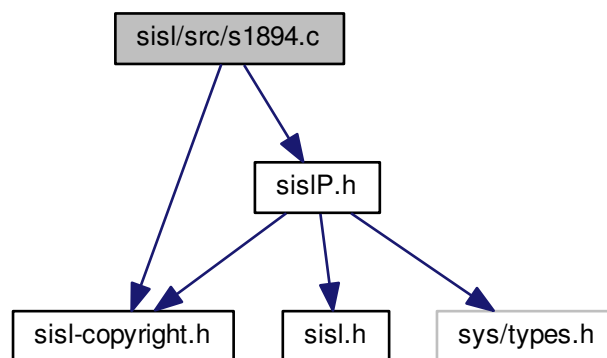
Definition at line 60 of file s1893.c.

## 30.2578 sisl/src/s1894.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1894.c:



## Macros

- #define [S1894](#)

## Functions

- void [s1894](#) (oknots, int oik, int oin, int der1, int der2, earray, int dimp1, int narr, nknots, int \*nik, int \*nin, int \*jstat)

### 30.2578.1 Macro Definition Documentation

#### 30.2578.1.1 #define S1894

Definition at line 49 of file s1894.c.

### 30.2578.2 Function Documentation

#### 30.2578.2.1 void s1894 ( oknots , int oik, int oin, int der1, int der2, earray , int dimp1, int narr, nknots , int \* nik, int \* nin, int \* jstat )

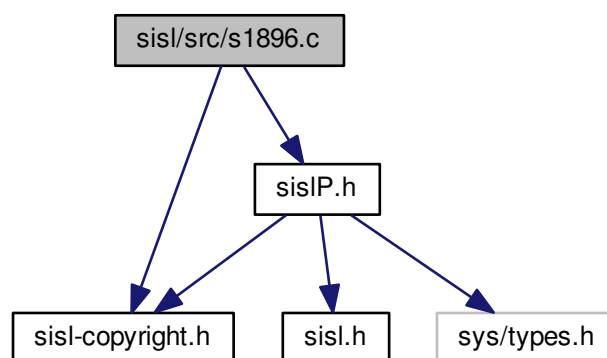
Definition at line 59 of file s1894.c.

## 30.2579 sisl/src/s1896.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1896.c:



## Macros

- #define [S1896](#)

## Functions

- void [s1896](#) ([SISLSurf](#) \*osurf, earray, int dimp1, int narr, ders1, dert1, ders2, dert2, [SISLSurf](#) \*\*nsurf, int \*jstat)

### 30.2579.1 Macro Definition Documentation

#### 30.2579.1.1 #define S1896

Definition at line 49 of file s1896.c.

### 30.2579.2 Function Documentation

#### 30.2579.2.1 void s1896 ( [SISLSurf](#) \* *osurf*, earray , int *dimp1*, int *narr*, ders1 , dert1 , ders2 , dert2 , [SISLSurf](#) \*\* *nsurf*, int \* *jstat* )

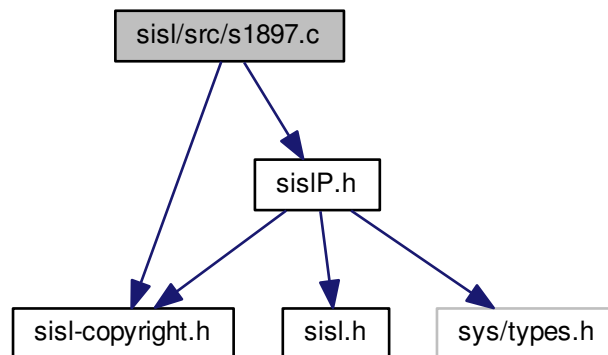
Definition at line 60 of file s1896.c.

## 30.2580 sisl/src/s1897.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1897.c:



## Macros

- #define [S1897](#)
- #define [MAX\\_IK](#) 50

## Functions

- void `s1897` (`et`, `int ik`, `double ax`, `int left`, `int deriv`, `ebiatx`, `int *jstat`)

### 30.2580.1 Macro Definition Documentation

#### 30.2580.1.1 `#define MAX_IK 50`

Definition at line 53 of file `s1897.c`.

#### 30.2580.1.2 `#define S1897`

Definition at line 49 of file `s1897.c`.

### 30.2580.2 Function Documentation

#### 30.2580.2.1 `void s1897 ( et , int ik , double ax , int left , int deriv , ebiatx , int * jstat )`

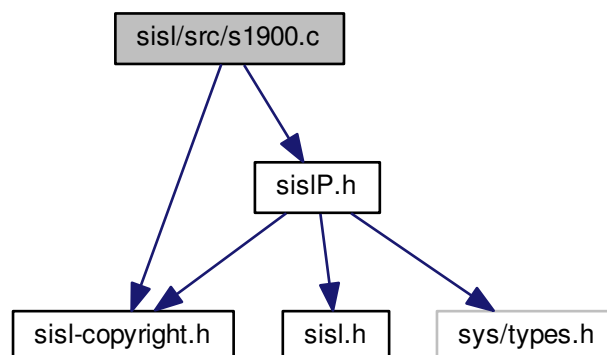
Definition at line 61 of file `s1897.c`.

## 30.2581 `sisl/src/s1900.c` File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for `s1900.c`:



## Macros

- `#define S1900`

## Functions

- void `s1900` (`param`, `knots`, `econd`, `ntype`, `int inpt`, `int ik`, `int idim`, `int iopen`, `double *cendpar`, `SISLCurve **rcurve`, `double **gpar`, `int *jnbpar`, `int *jstat`)

### 30.2581.1 Macro Definition Documentation

#### 30.2581.1.1 `#define S1900`

Definition at line 49 of file `s1900.c`.

### 30.2581.2 Function Documentation

#### 30.2581.2.1 `void s1900 ( param , knots , econd , ntype , int inpt , int ik , int idim , int iopen , double * cendpar , SISLCurve ** rcurve , double ** gpar , int * jnbpar , int * jstat )`

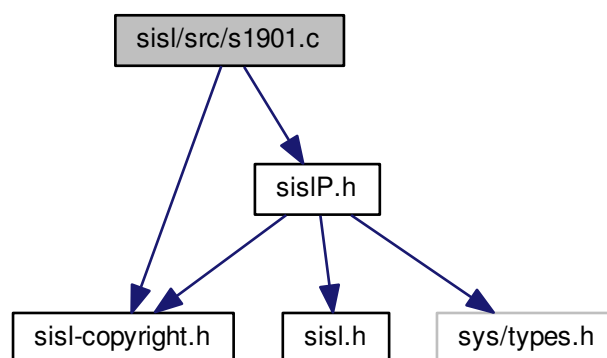
Definition at line 61 of file `s1900.c`.

## 30.2582 sisl/src/s1901.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for `s1901.c`:



## Macros

- `#define S1901`

## Typedefs

- typedef void(\* [fparamProc](#)) ()
- typedef void(\* [fknotsProc](#)) ()

## Functions

- void [s1901](#) ([fparamProc](#) *fparam*, [fknotsProc](#) *fknots*, *econd*, *ntype*, *int inpt*, [double](#) *astpar*, *int ik*, *int idim*, *int iopen*, [double](#) \**cendpar*, [SISLCurve](#) \*\**rcurve*, [double](#) \*\**gpar*, *int \*jnbpar*, *int \*jstat*)

### 30.2582.1 Macro Definition Documentation

#### 30.2582.1.1 #define S1901

Definition at line 49 of file s1901.c.

### 30.2582.2 Typedef Documentation

#### 30.2582.2.1 typedef void(\* [fknotsProc](#)) ()

Definition at line 59 of file s1901.c.

#### 30.2582.2.2 typedef void(\* [fparamProc](#)) ()

Definition at line 58 of file s1901.c.

### 30.2582.3 Function Documentation

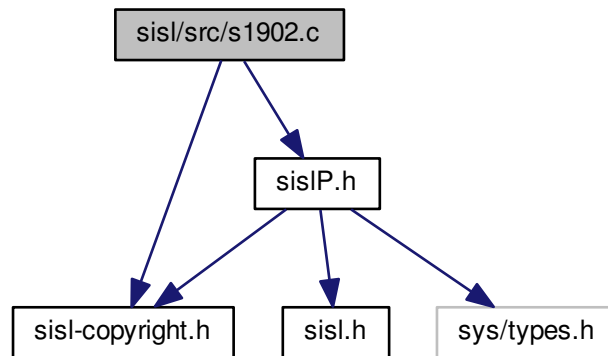
#### 30.2582.3.1 void [s1901](#) ( [fparamProc](#) *fparam*, [fknotsProc](#) *fknots*, *econd* , *ntype* , *int inpt*, [double](#) *astpar*, *int ik*, *int idim*, *int iopen*, [double](#) \* *cendpar*, [SISLCurve](#) \*\* *rcurve*, [double](#) \*\* *gpar*, *int \* jnbpar*, *int \* jstat* )

Definition at line 71 of file s1901.c.



## 30.2583 sisl/src/s1902.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for s1902.c:
```



### Macros

- `#define S1902`

### Functions

- `void s1902 (epar, int in, int ik, int cuopen, double **eknots, int *jstat)`

#### 30.2583.1 Macro Definition Documentation

##### 30.2583.1.1 `#define S1902`

Definition at line 49 of file `s1902.c`.

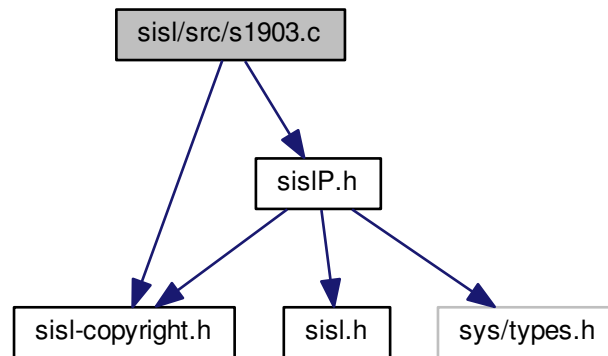
#### 30.2583.2 Function Documentation

##### 30.2583.2.1 `void s1902 (epar, int in, int ik, int cuopen, double **eknots, int *jstat)`

Definition at line 59 of file `s1902.c`.

## 30.2584 sisl/src/s1903.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for s1903.c:
```



### Macros

- `#define S1903`

### Functions

- `void s1903 (epar, int in, int ik, int cuopen, eknots, int *jstat)`

### 30.2584.1 Macro Definition Documentation

#### 30.2584.1.1 `#define S1903`

Definition at line 49 of file `s1903.c`.

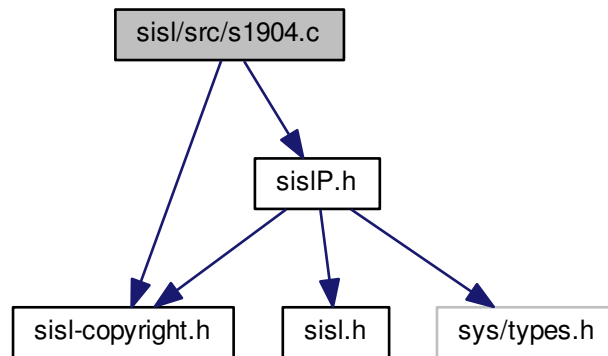
### 30.2584.2 Function Documentation

#### 30.2584.2.1 `void s1903 (epar, int in, int ik, int cuopen, eknots, int *jstat)`

Definition at line 59 of file `s1903.c`.

## 30.2585 sisl/src/s1904.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for s1904.c:
```



### Macros

- `#define S1904`

### Functions

- `void s1904 (epar, int in, int ik, int cuopen, eknots, int *jstat)`

#### 30.2585.1 Macro Definition Documentation

##### 30.2585.1.1 `#define S1904`

Definition at line 49 of file `s1904.c`.

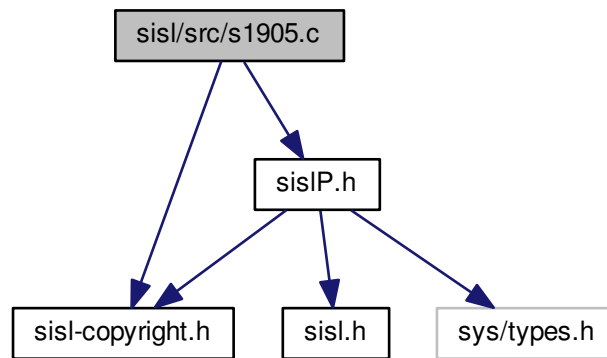
#### 30.2585.2 Function Documentation

##### 30.2585.2.1 `void s1904 (epar, int in, int ik, int cuopen, eknots, int *jstat)`

Definition at line 59 of file `s1904.c`.

## 30.2586 sisl/src/s1905.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for s1905.c:
```



### Macros

- `#define S1905`

### Functions

- void `s1905` ( `econd1`, `ntype1`, `int inpt1`, `int ik`, `int idim`, `int iopen`, `double **gcond2`, `int **mtype2`, `int *jnpt2`, `int *jstat` )

### 30.2586.1 Macro Definition Documentation

#### 30.2586.1.1 `#define S1905`

Definition at line 49 of file s1905.c.

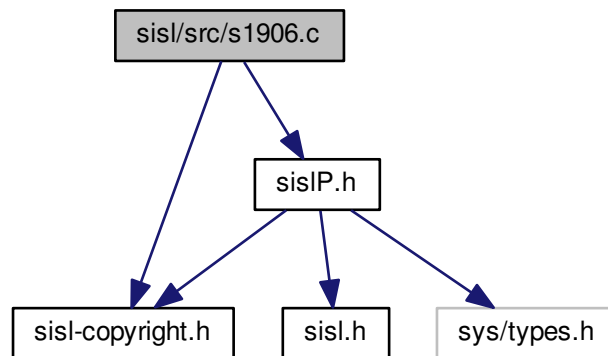
### 30.2586.2 Function Documentation

#### 30.2586.2.1 void `s1905` ( `econd1`, `ntype1`, `int inpt1`, `int ik`, `int idim`, `int iopen`, `double **gcond2`, `int **mtype2`, `int *jnpt2`, `int *jstat` )

Definition at line 59 of file s1905.c.

## 30.2587 sisl/src/s1906.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for s1906.c:
```



### Macros

- `#define S1906`

### Functions

- void `s1906` (`double *epoint`, `int *etype`, `int icnsta`, `int icnend`, `int inbpnt`, `int idim`, `double **opoint`, `int **otype`, `int *knbpnt`, `int *jstat`)

### 30.2587.1 Macro Definition Documentation

#### 30.2587.1.1 `#define S1906`

Definition at line 49 of file s1906.c.

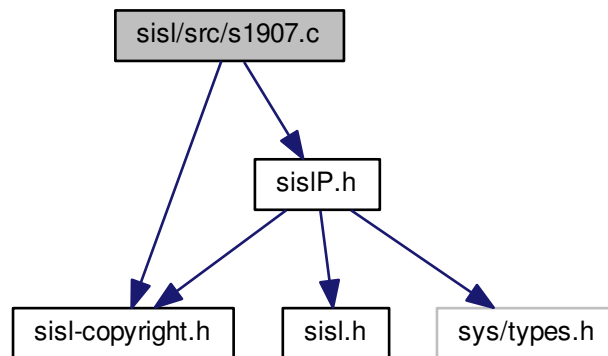
### 30.2587.2 Function Documentation

#### 30.2587.2.1 void `s1906` ( `double *epoint`, `int *etype`, `int icnsta`, `int icnend`, `int inbpnt`, `int idim`, `double **opoint`, `int **otype`, `int *knbpnt`, `int *jstat` )

Definition at line 59 of file s1906.c.

## 30.2588 sisl/src/s1907.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for s1907.c:
```



### Macros

- `#define S1907`

### Functions

- void `s1907` (`double *epoint`, `int *ntype`, `double *epar`, `int iopen`, `int icnsta`, `int icnend`, `int inbpnt`, `int idim`, `opoint`, `otype`, `opar`, `int *knbpnt`, `int *jstat`)

### 30.2588.1 Macro Definition Documentation

#### 30.2588.1.1 `#define S1907`

Definition at line 49 of file `s1907.c`.

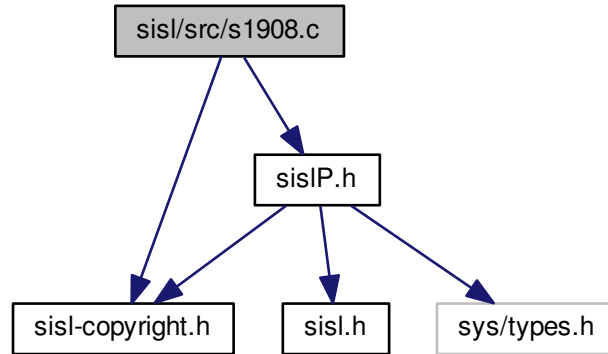
### 30.2588.2 Function Documentation

#### 30.2588.2.1 void `s1907` ( `double *epoint`, `int *ntype`, `double *epar`, `int iopen`, `int icnsta`, `int icnend`, `int inbpnt`, `int idim`, `opoint`, `otype`, `opar`, `int *knbpnt`, `int *jstat` )

Definition at line 58 of file `s1907.c`.

## 30.2589 sisl/src/s1908.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for s1908.c:
```



### Macros

- `#define S1908`
- `#define MAX_SIZE 30`

### Functions

- `void s1908 ( econd1, ntype1, epar, int inpt1, int ik, int idim, int iopen, gcond2, mtype2, mpar, int *jnpt2, int *jstat )`

#### 30.2589.1 Macro Definition Documentation

##### 30.2589.1.1 `#define MAX_SIZE 30`

Definition at line 52 of file `s1908.c`.

##### 30.2589.1.2 `#define S1908`

Definition at line 49 of file `s1908.c`.

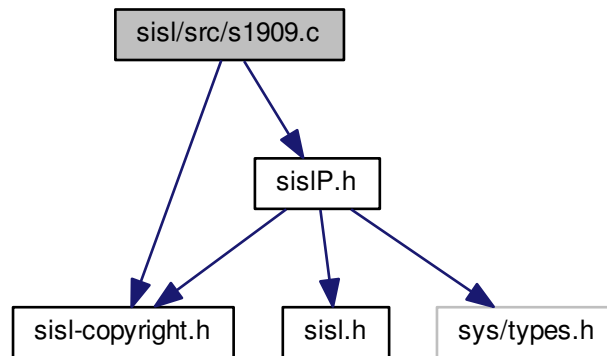
#### 30.2589.2 Function Documentation

##### 30.2589.2.1 `void s1908 ( econd1, ntype1, epar, int inpt1, int ik, int idim, int iopen, gcond2, mtype2, mpar, int *jnpt2, int *jstat )`

Definition at line 61 of file `s1908.c`.

## 30.2590 sisl/src/s1909.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for s1909.c:
```



### Macros

- `#define S1909`

### Functions

- void `s1909` (econd, ntype, int inpt, int idim, int iopen, double astpar, double \*cendpar, epar1, epar2, int \*jstat)

#### 30.2590.1 Macro Definition Documentation

##### 30.2590.1.1 `#define S1909`

Definition at line 49 of file `s1909.c`.

#### 30.2590.2 Function Documentation

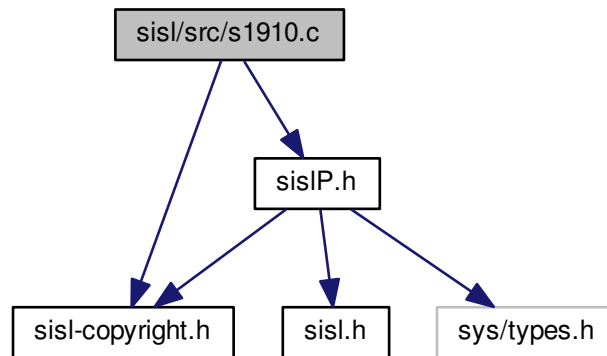
30.2590.2.1 void `s1909` ( econd , ntype , int inpt , int idim , int iopen , double astpar , double \* cendpar , epar1 , epar2 , int \* jstat )

Definition at line 60 of file `s1909.c`.



## 30.2591 sisl/src/s1910.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for s1910.c:
```



### Macros

- `#define S1910`

### Functions

- `void s1910 (econd, ntype, int inpt, int idim, int iopen, double astpar, double *cendpar, epar1, epar2, int *jstat)`

#### 30.2591.1 Macro Definition Documentation

##### 30.2591.1.1 `#define S1910`

Definition at line 49 of file `s1910.c`.

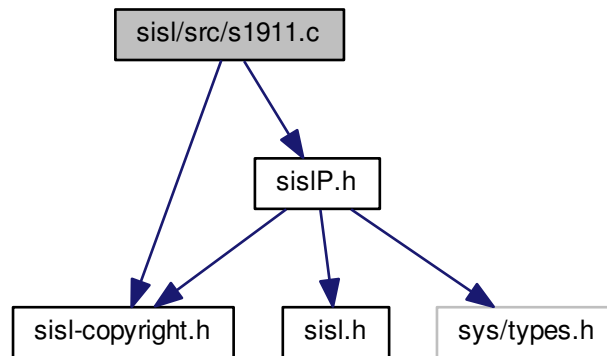
#### 30.2591.2 Function Documentation

##### 30.2591.2.1 `void s1910 ( econd , ntype , int inpt , int idim , int iopen , double astpar , double * cendpar , epar1 , epar2 , int * jstat )`

Definition at line 60 of file `s1910.c`.

## 30.2592 sisl/src/s1911.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for s1911.c:
```



### Macros

- `#define S1911`

### Functions

- void `s1911` (`econd`, `ntype`, `int inpt`, `int idim`, `int iopen`, `double astpar`, `double *cendpar`, `epar1`, `epar2`, `int *jstat`)

### 30.2592.1 Macro Definition Documentation

#### 30.2592.1.1 `#define S1911`

Definition at line 49 of file `s1911.c`.

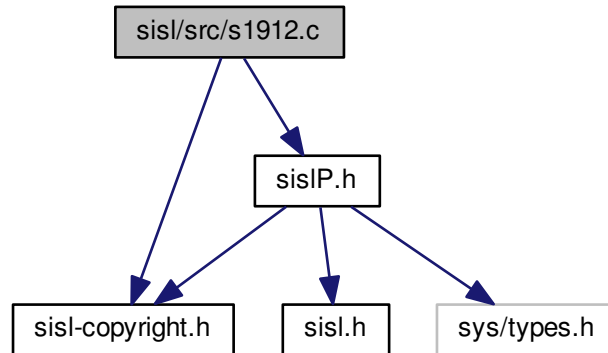
### 30.2592.2 Function Documentation

#### 30.2592.2.1 void `s1911` ( `econd` , `ntype` , `int inpt` , `int idim` , `int iopen` , `double astpar` , `double *cendpar` , `epar1` , `epar2` , `int *jstat` )

Definition at line 60 of file `s1911.c`.

## 30.2593 sisl/src/s1912.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for s1912.c:
```



### Macros

- `#define S1912`

### Typedefs

- `typedef void(* fparamProc) ()`
- `typedef void(* fknotsProc) ()`

### Functions

- `void s1912 (fparamProc fparam, fknotsProc fknots, econd, ntype, int inpt, double astpar, int ik, int idim, int iopen, double *cendpar, SISLCurve **rcurve, double **gpar, int *jnbpar, int *jstat)`

### 30.2593.1 Macro Definition Documentation

#### 30.2593.1.1 `#define S1912`

Definition at line 42 of file `s1912.c`.

### 30.2593.2 Typedef Documentation

#### 30.2593.2.1 `typedef void(* fknotsProc) ()`

Definition at line 52 of file `s1912.c`.

30.2593.2.2 `typedef void(* fparamProc) ()`

Definition at line 51 of file s1912.c.

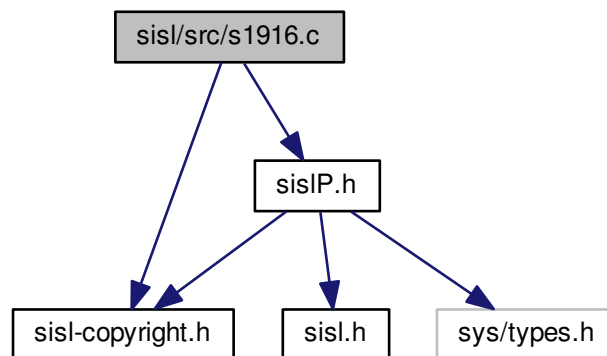
### 30.2593.3 Function Documentation

30.2593.3.1 `void s1912 ( fparamProc fparam, fknotsProc fknots, econd , ntype , int inpt, double astpar, int ik, int idim, int iopen, double * cendpar, SISLCurve ** rcurve, double ** gpar, int * jnbpar, int * jstat )`

Definition at line 64 of file s1912.c.

## 30.2594 sisl/src/s1916.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for s1916.c:
```



### Macros

- `#define S1916`

### Functions

- `void s1916 (int inbcrv, et2, ecoef, int in2, int iord, int idim, int iopen, par, der, int *jstat)`

### 30.2594.1 Macro Definition Documentation

30.2594.1.1 `#define S1916`

Definition at line 49 of file s1916.c.

## 30.2594.2 Function Documentation

30.2594.2.1 void s1916 ( int *inbcrv*, et2 , ecoef , int *in2*, int *iord*, int *idim*, int *iopen*, par , der , int \* *jstat* )

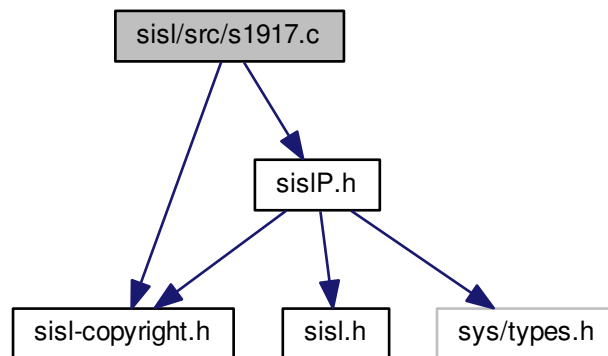
Definition at line 59 of file s1916.c.

## 30.2595 sisl/src/s1917.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1917.c:



### Macros

- #define [S1917](#)

### Functions

- void [s1917](#) (int *inbcrv*, ecoef, int *in2*, int *idim*, eptyp, double *astpar*, int *iopen*, par, der, int \**inumb*, int \**jstat*)

## 30.2595.1 Macro Definition Documentation

30.2595.1.1 #define S1917

Definition at line 49 of file s1917.c.

### 30.2595.2 Function Documentation

30.2595.2.1 void `s1917` ( int *inbcrv*, *ecoef*, int *in2*, int *idim*, *eptyp*, double *astpar*, int *iopen*, *par*, *der*, int \* *inumb*, int \* *jstat* )

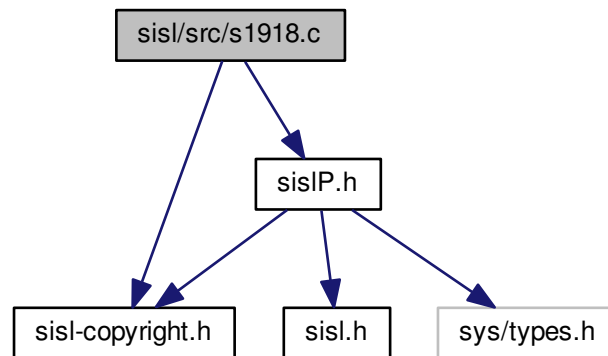
Definition at line 59 of file `s1917.c`.

### 30.2596 `sisl/src/s1918.c` File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for `s1918.c`:



#### Macros

- `#define S1918`

#### Functions

- void `s1918` (int *inbcrv*, *et2*, *ecoef*, int *in2*, int *iord*, int *idim*, *par*, *der*, int \**jstat*)

### 30.2596.1 Macro Definition Documentation

30.2596.1.1 `#define S1918`

Definition at line 49 of file `s1918.c`.

## 30.2596.2 Function Documentation

30.2596.2.1 void s1918 ( int *inbcrv*, et2 , ecoef , int *in2*, int *iord*, int *idim*, par , der , int \* *jstat* )

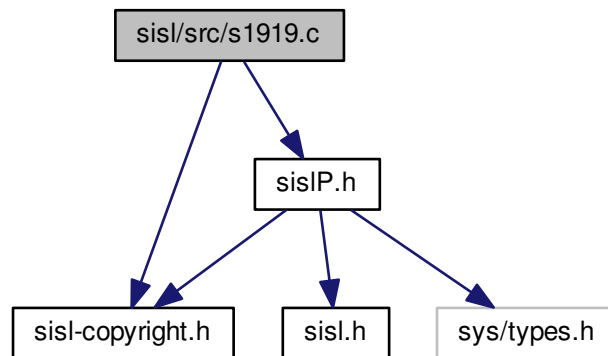
Definition at line 59 of file s1918.c.

## 30.2597 sisl/src/s1919.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1919.c:



### Macros

- #define [S1919](#)

### Functions

- void [s1919](#) (et, prev, curr, deriv, follow, int in, int ik, int idim, int iip, int iif, [double ap](#), [double ac](#), [double af](#), int \*jstat)

## 30.2597.1 Macro Definition Documentation

30.2597.1.1 #define S1919

Definition at line 49 of file s1919.c.

### 30.2597.2 Function Documentation

30.2597.2.1 `void s1919 ( et , prev , curr , deriv , follow , int in , int ik , int idim , int iip , int iif , double ap , double ac , double af , int * jstat )`

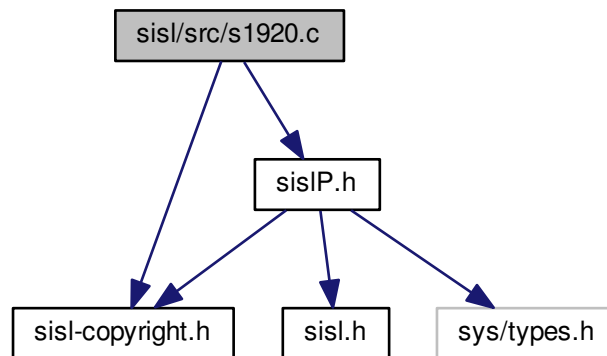
Definition at line 60 of file s1919.c.

### 30.2598 sisl/src/s1920.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1920.c:



#### Macros

- `#define S1920`

#### Functions

- `void s1920 (SISLCurve *pc1, edir, int idim, double aepsco, double aepsge, int *jpt, double **gpar, int *jcrv, SISLIntcurve ***wcurve, int *jstat)`

### 30.2598.1 Macro Definition Documentation

30.2598.1.1 `#define S1920`

Definition at line 49 of file s1920.c.



## 30.2598.2 Function Documentation

30.2598.2.1 `void s1920 ( SISLCurve * pc1, edir , int idim, double aepsco, double aepsge, int * jpt, double ** gpar, int * jcrv, SISLIntcurve *** wcurve, int * jstat )`

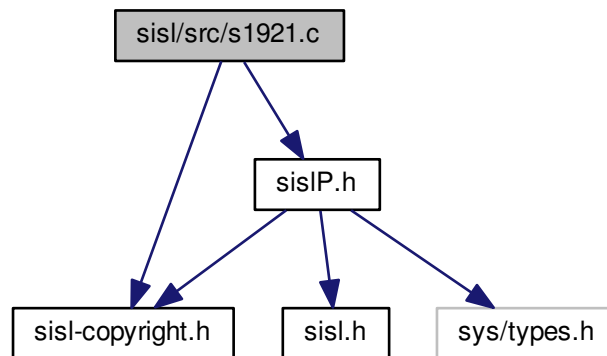
Definition at line 58 of file s1920.c.

## 30.2599 sisl/src/s1921.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1921.c:



### Macros

- `#define S1921`

### Functions

- `void s1921 ( SISLSurf *ps1, edir, int idim, double aepsco, double aepsge, int *jpt, double **gpar, int *jcrv, SISLIntcurve ***wcurve, int *jstat)`

## 30.2599.1 Macro Definition Documentation

30.2599.1.1 `#define S1921`

Definition at line 49 of file s1921.c.

### 30.2599.2 Function Documentation

30.2599.2.1 void s1921 ( SISLSurf \* *ps1*, edir , int *idim*, double *aepsco*, double *aepsge*, int \* *jpt*, double \*\* *gpar*, int \* *jcrv*, SISLIntcurve \*\*\* *wcurve*, int \* *jstat* )

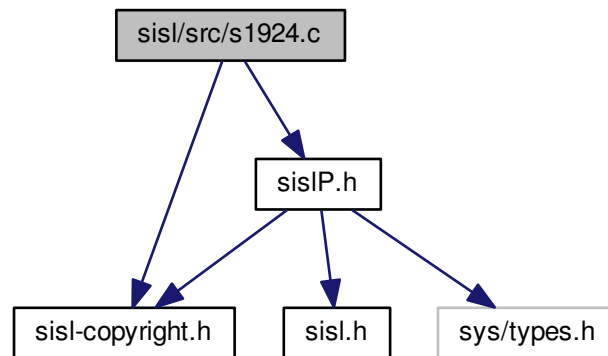
Definition at line 58 of file s1921.c.

### 30.2600 sisl/src/s1924.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1924.c:



#### Macros

- #define [S1924](#)

#### Functions

- void [s1924](#) (int id1, int id2, int id3, int id4, int in1, int in2, double \*\*ew, int \*jstat)

### 30.2600.1 Macro Definition Documentation

30.2600.1.1 #define S1924

Definition at line 49 of file s1924.c.

## 30.2600.2 Function Documentation

30.2600.2.1 void s1924 ( int *id1*, int *id2*, int *id3*, int *id4*, int *in1*, int *in2*, double \*\* *ew*, int \* *jstat* )

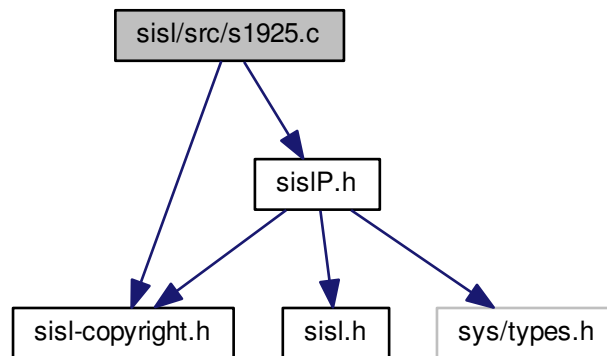
Definition at line 60 of file s1924.c.

## 30.2601 sisl/src/s1925.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1925.c:



### Macros

- #define [S1925](#)
- #define [s1925\\_MAX\\_ARRAY\\_SIZE](#) 50

### Functions

- void [s1925](#) (etau, epoint, int inbpnt, eder, et, ebcoef, int in, int ik, int irect, int [dim](#), ew1, int nur, ed, ew2, int inrc, ew3, int inlr, int \*jstat)

## 30.2601.1 Macro Definition Documentation

30.2601.1.1 #define [S1925](#)

Definition at line 49 of file s1925.c.

30.2601.1.2 `#define s1925_MAX_ARRAY_SIZE 50`

Definition at line 52 of file s1925.c.

### 30.2601.2 Function Documentation

30.2601.2.1 `void s1925 ( etau , epoint , int inbpnt , eder , et , ebcoef , int in , int ik , int irlight , int dim , ew1 , int nur , ed , ew2 , int inrc , ew3 , int inlr , int * jstat )`

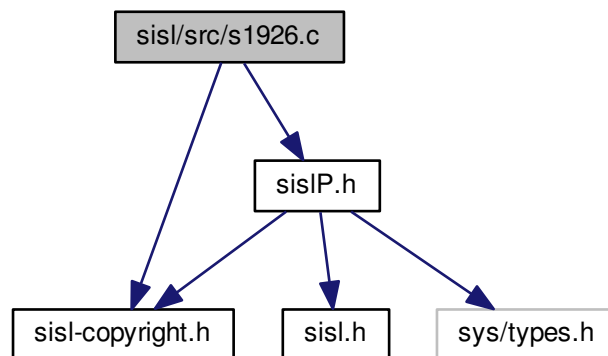
Definition at line 62 of file s1925.c.

## 30.2602 sisl/src/s1926.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1926.c:



### Macros

- `#define S1926`

### Functions

- `void s1926 (double *w1, int nur, int ik, int *ed, double *w2, int nrc, double *w3, int nlr, int *jstat)`

### 30.2602.1 Macro Definition Documentation

30.2602.1.1 `#define S1926`

Definition at line 49 of file s1926.c.

## 30.2602.2 Function Documentation

30.2602.2.1 void s1926 ( double \* w1, int nur, int ik, int \* ed, double \* w2, int nrc, double \* w3, int nlr, int \* jstat )

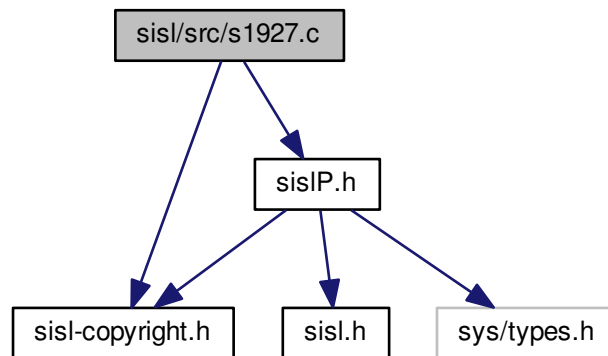
Definition at line 60 of file s1926.c.

## 30.2603 sisl/src/s1927.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1927.c:



### Macros

- #define [S1927](#)

### Functions

- void [s1927](#) (double \*w1, int nur, int ik, int \*ed, double \*w2, int nrc, double \*w3, int nlr, ex, double \*ey, int \*jstat)

## 30.2603.1 Macro Definition Documentation

30.2603.1.1 #define S1927

Definition at line 49 of file s1927.c.

### 30.2603.2 Function Documentation

30.2603.2.1 `void s1927 ( double * w1, int nur, int ik, int * ed, double * w2, int nrc, double * w3, int nlr, ex , double * ey, int * jstat )`

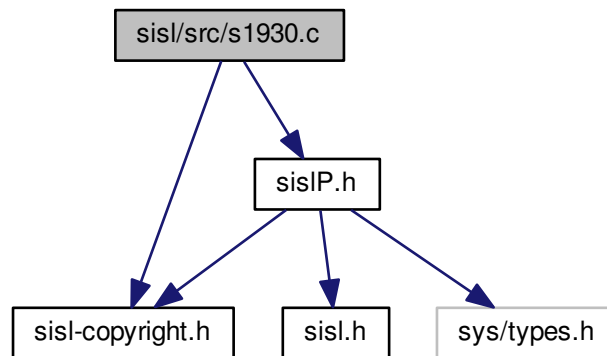
Definition at line 59 of file s1927.c.

### 30.2604 sisl/src/s1930.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1930.c:



#### Macros

- `#define S1930`

#### Functions

- `void s1930 (int inbcrv, SISLCurve **vpcrv, double **gknot2, double **gcoef2, int *jn2, int *jord2, int *jstat)`

### 30.2604.1 Macro Definition Documentation

30.2604.1.1 `#define S1930`

Definition at line 49 of file s1930.c.

## 30.2604.2 Function Documentation

30.2604.2.1 void `s1930` ( int *inbcrv*, SISLCurve \*\* *vpcrv*, double \*\* *gknot2*, double \*\* *gcoef2*, int \* *jn2*, int \* *jord2*, int \* *jstat* )

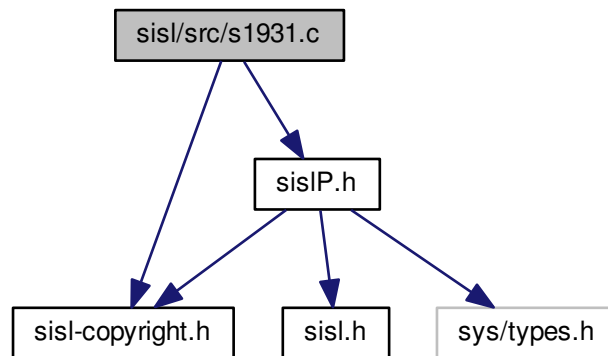
Definition at line 59 of file s1930.c.

## 30.2605 sisl/src/s1931.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1931.c:



### Macros

- #define `S1931`

### Functions

- void `s1931` (int *inbcrv*, SISLCurve \*\**vpcrv*, double \*\**gknot2*, double \*\**gcoef2*, int \**jn2*, int \**jord2*, int \**jstat*)

## 30.2605.1 Macro Definition Documentation

30.2605.1.1 #define `S1931`

Definition at line 49 of file s1931.c.

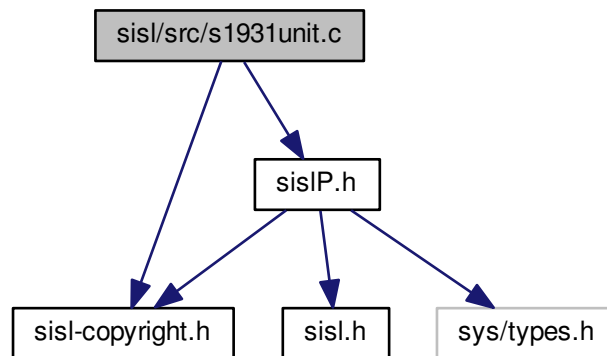
### 30.2605.2 Function Documentation

30.2605.2.1 `void s1931 ( int inbcrv, SISLCurve ** vpcrv, double ** gknot2, double ** gcoef2, int * jn2, int * jord2, int * jstat )`

Definition at line 59 of file s1931.c.

### 30.2606 sisl/src/s1931unit.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for s1931unit.c:
```



#### Macros

- `#define S1931UNIT`

#### Functions

- `void s1931unit (int inbcrv, SISLCurve **vpcrv, double **gknot2, double **gcoef2, int *jn2, int *jord2, int *jstat)`

### 30.2606.1 Macro Definition Documentation

30.2606.1.1 `#define S1931UNIT`

Definition at line 49 of file s1931unit.c.



## 30.2606.2 Function Documentation

30.2606.2.1 void `s1931unit` ( int *inbcrv*, SISLCurve \*\* *vpcrv*, double \*\* *gknot2*, double \*\* *gcoef2*, int \* *jn2*, int \* *jord2*, int \* *jstat* )

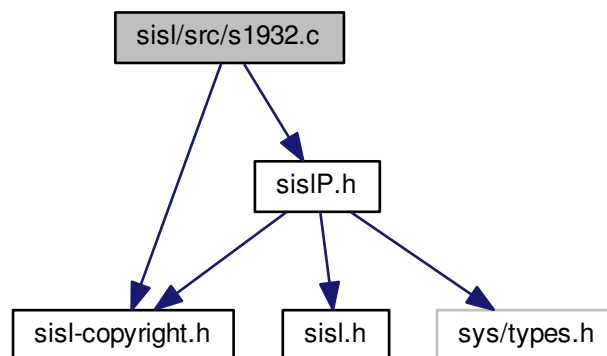
Definition at line 59 of file `s1931unit.c`.

## 30.2607 sisl/src/s1932.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for `s1932.c`:



### Macros

- `#define S1932`

### Functions

- void `s1932` (int *inbcrv*, SISLCurve \*\**crv*, double *start*, double *stop*, double \**et*, int *in*, int *iordr*, double \*\**iright*, int \**jstat*)

## 30.2607.1 Macro Definition Documentation

30.2607.1.1 `#define S1932`

Definition at line 49 of file `s1932.c`.

### 30.2607.2 Function Documentation

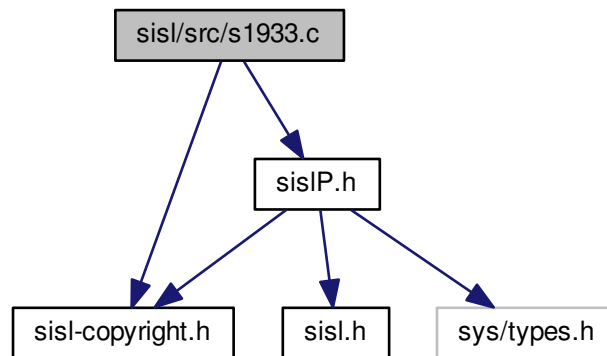
30.2607.2.1 void `s1932` ( int *inbcrv*, SISLCurve \*\* *crvarr*, double *start*, double *stop*, double \* *et*, int *in*, int *iordr*, double \*\* *iright*, int \* *jstat* )

Definition at line 60 of file `s1932.c`.

### 30.2608 sisl/src/s1933.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
```

Include dependency graph for `s1933.c`:



#### Macros

- `#define S1933`

#### Functions

- void `s1933` (int *inbcrv*, *crvarr*, double *start*, double *stop*, double \*\**it*, int \**in*, int \**iordr*, int \**jstat*)

### 30.2608.1 Macro Definition Documentation

30.2608.1.1 `#define S1933`

Definition at line 49 of file `s1933.c`.

## 30.2608.2 Function Documentation

30.2608.2.1 void s1933 ( int *inbcrv*, *crvarr* , double *start*, double *stop*, double \*\* *it*, int \* *in*, int \* *iordr*, int \* *jstat* )

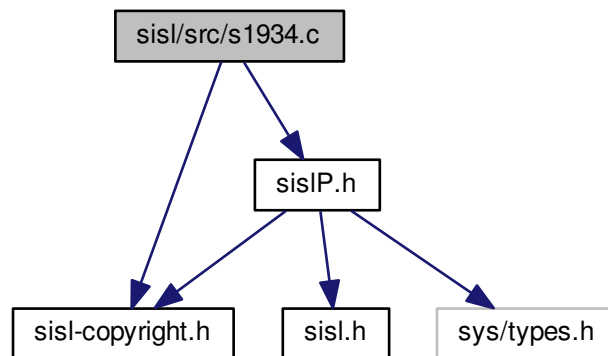
Definition at line 60 of file s1933.c.

## 30.2609 sisl/src/s1934.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1934.c:



### Macros

- #define [S1934](#)

### Functions

- void [s1934](#) (double \**et*, int *in*, int *ik*, double *start*, double *end*, int \**jstat*)

## 30.2609.1 Macro Definition Documentation

30.2609.1.1 #define S1934

Definition at line 49 of file s1934.c.

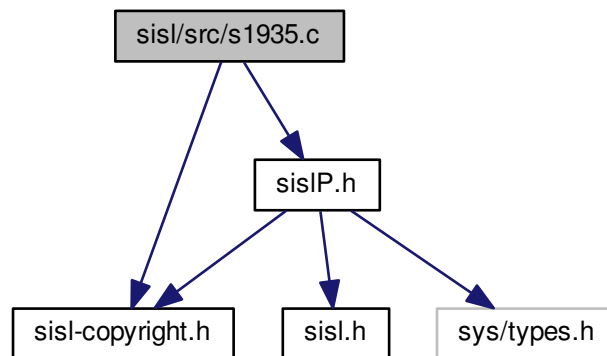
## 30.2609.2 Function Documentation

30.2609.2.1 `void s1934 ( double * et, int in, int ik, double start, double end, int * jstat )`

Definition at line 59 of file s1934.c.

## 30.2610 sisl/src/s1935.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for s1935.c:
```



### Macros

- `#define S1935`

### Functions

- `void s1935 (double *et1, int in1, double *et2, int in2, knt, int *in, int ik, int *jstat)`

## 30.2610.1 Macro Definition Documentation

30.2610.1.1 `#define S1935`

Definition at line 49 of file s1935.c.

## 30.2610.2 Function Documentation

30.2610.2.1 void s1935 ( double \* et1, int in1, double \* et2, int in2, knt , int \* in, int ik, int \* jstat )

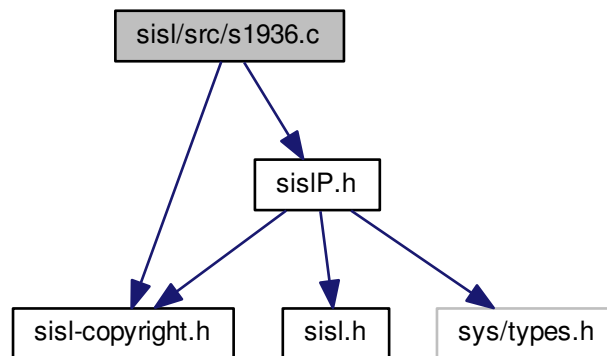
Definition at line 59 of file s1935.c.

## 30.2611 sisl/src/s1936.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1936.c:



### Macros

- #define [S1936](#)
- #define [MAX\\_SIZE](#) 50

### Functions

- void [s1936](#) ([SISLCurve](#) \*crv, etd, int ind, [double](#) \*curvd, int \*jstat)

## 30.2611.1 Macro Definition Documentation

30.2611.1.1 #define [MAX\\_SIZE](#) 50

Definition at line 52 of file s1936.c.

30.2611.1.2 `#define S1936`

Definition at line 49 of file s1936.c.

## 30.2611.2 Function Documentation

30.2611.2.1 `void s1936 ( SISLCurve * crv, etd , int ind, double * curvd, int * jstat )`

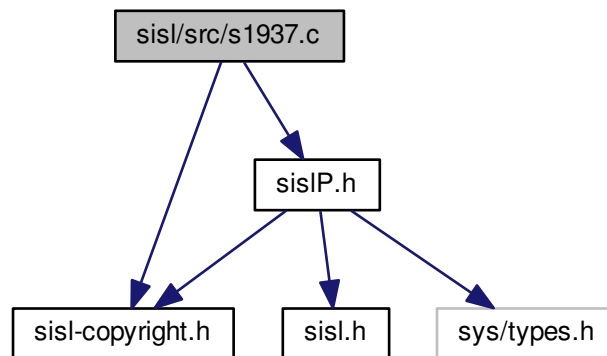
Definition at line 61 of file s1936.c.

## 30.2612 sisl/src/s1937.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1937.c:



### Macros

- `#define S1937`

### Functions

- `void s1937 (et, int iodr, int ref, int left, alfa, etref)`

## 30.2612.1 Macro Definition Documentation

30.2612.1.1 `#define S1937`

Definition at line 49 of file s1937.c.

## 30.2612.2 Function Documentation

30.2612.2.1 void s1937 ( et , int iorder, int ref, int left, alfa , etref )

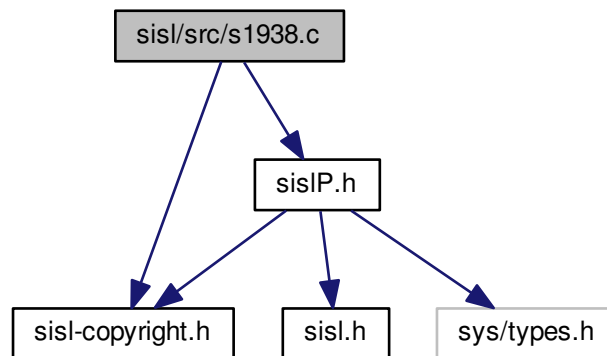
Definition at line 59 of file s1937.c.

## 30.2613 sisl/src/s1938.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1938.c:



### Macros

- #define [S1938](#)
- #define [MAX\\_SIZE](#) 50

### Functions

- void [s1938](#) ([SISLSurf](#) \*srf, etr1, int inr1, etr2, int inr2, [double](#) \*\*surfr, int \*jstat)

## 30.2613.1 Macro Definition Documentation

30.2613.1.1 #define [MAX\\_SIZE](#) 50

Definition at line 52 of file s1938.c.

30.2613.1.2 `#define S1938`

Definition at line 49 of file s1938.c.

## 30.2613.2 Function Documentation

30.2613.2.1 `void s1938 ( SISLSurf * srf, etr1 , int inr1, etr2 , int inr2, double ** surf, int * jstat )`

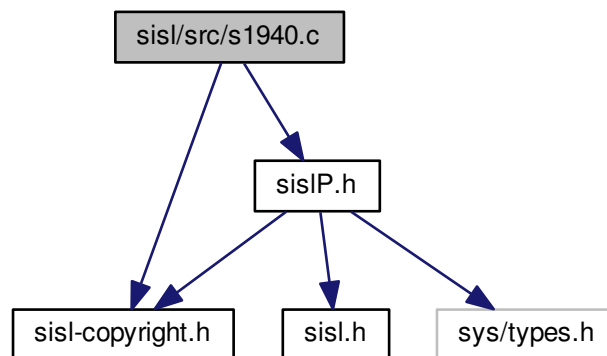
Definition at line 61 of file s1938.c.

## 30.2614 sisl/src/s1940.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1940.c:



### Macros

- `#define S1940`

### Functions

- `void s1940 (SISLCurve *oldcurve, eps, int startfix, int endfix, int iopen, int itmax, SISLCurve **newcurve, maxerr, int *stat)`

## 30.2614.1 Macro Definition Documentation

30.2614.1.1 `#define S1940`

Definition at line 42 of file s1940.c.



## 30.2614.2 Function Documentation

30.2614.2.1 void s1940 ( SISLCurve \* oldcurve, eps , int startfix, int endfix, int iopen, int itmax, SISLCurve \*\* newcurve, maxerr , int \* stat )

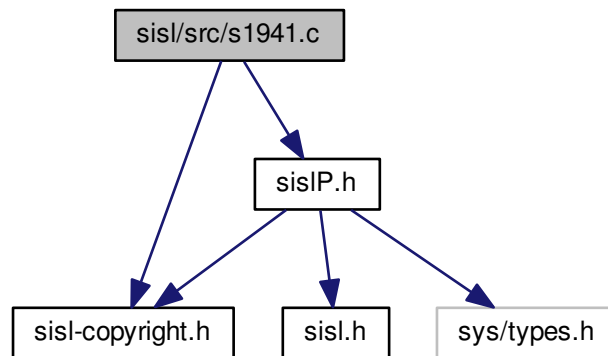
Definition at line 53 of file s1940.c.

## 30.2615 sisl/src/s1941.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1941.c:



### Macros

- #define [S1941](#)

### Functions

- void [s1941](#) (SISLCurve \*pcurve, int icont, int \*jstat)

## 30.2615.1 Macro Definition Documentation

30.2615.1.1 #define S1941

Definition at line 44 of file s1941.c.

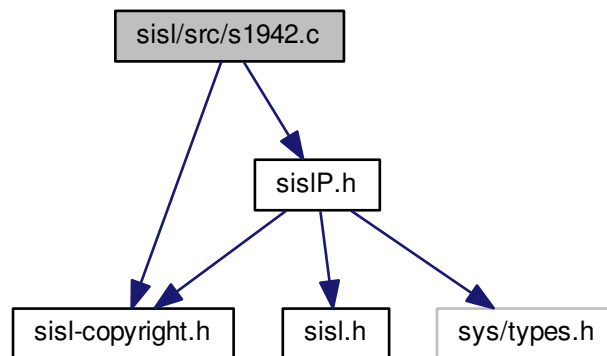
## 30.2615.2 Function Documentation

30.2615.2.1 void s1941 ( SISLCurve \* pcurve, int icon, int \* jstat )

Definition at line 52 of file s1941.c.

## 30.2616 sisl/src/s1942.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for s1942.c:
```



### Macros

- #define [S1942](#)

### Functions

- void [s1942](#) (SISLCurve \*pc1, SISLCurve \*pc2, int idim, ea, nstart, nstop, emxerr, el2err, int \*jstat)

## 30.2616.1 Macro Definition Documentation

30.2616.1.1 #define S1942

Definition at line 42 of file s1942.c.

## 30.2616.2 Function Documentation

30.2616.2.1 void s1942 ( SISLCurve \* pc1, SISLCurve \* pc2, int idim, ea, nstart, nstop, emxerr, el2err, int \* jstat )

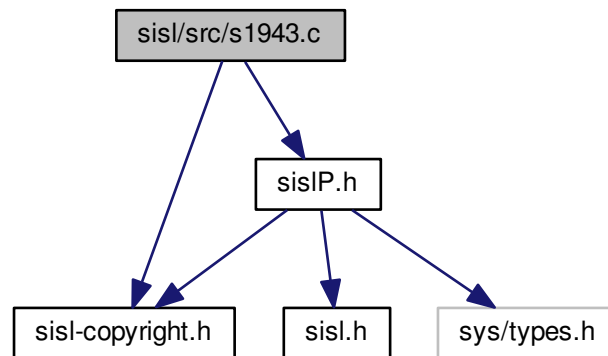
Definition at line 51 of file s1942.c.

## 30.2617 sisl/src/s1943.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1943.c:



### Macros

- #define [S1943](#)

### Functions

- void [s1943](#) (SISLCurve \*pcurve, etau, int ik, int in, int ileftfix, int irlightfix, int incont, SISLCurve \*\*rnewcurve, gmaxerr, gl2err, int \*jstat)

## 30.2617.1 Macro Definition Documentation

30.2617.1.1 #define S1943

Definition at line 43 of file s1943.c.

### 30.2617.2 Function Documentation

30.2617.2.1 `void s1943 ( SISLCurve * pcurve, etau , int ik, int in, int ileftfix, int irightfix, int incont, SISLCurve ** newcurve, gmaxerr , gl2err , int * jstat )`

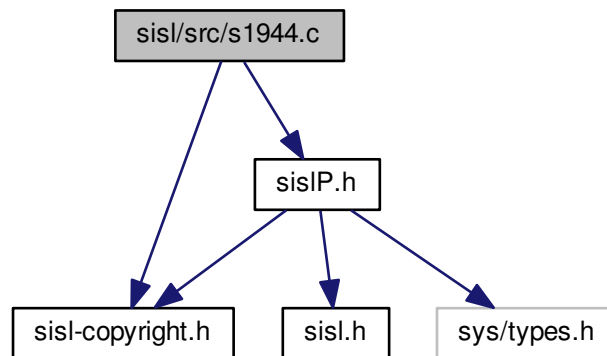
Definition at line 53 of file s1943.c.

### 30.2618 sisl/src/s1944.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1944.c:



#### Macros

- `#define S1944`

#### Functions

- `void s1944 (etau, int ik, int in, int idim, et, ed, int im, int inlc, int inlr, int inorm, ea, ew1, nfirst, nlast, eb, ew2, n2sta, ec, int *jstat)`

### 30.2618.1 Macro Definition Documentation

30.2618.1.1 `#define S1944`

Definition at line 43 of file s1944.c.

## 30.2618.2 Function Documentation

30.2618.2.1 void s1944 ( etau , int ik, int in, int idim, et , ed , int im, int inlc, int inlr, int inorm, ea , ew1 , nfirst , nlast , eb , ew2 , n2sta , ec , int \* jstat )

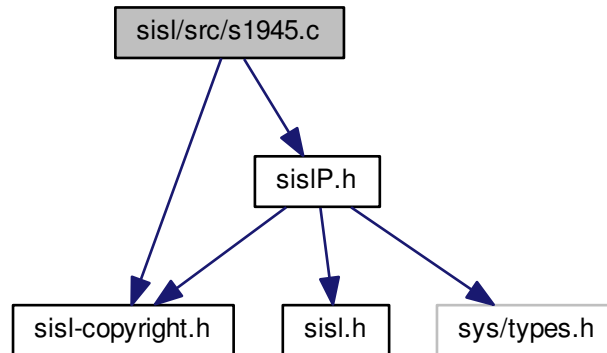
Definition at line 54 of file s1944.c.

## 30.2619 sisl/src/s1945.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1945.c:



### Macros

- #define [S1945](#)

### Functions

- void [s1945](#) (etau, int ik, int in, int idim, et, ed, int im, int ilend, int irend, int inlc, int inlr, int inorm, ea, ew1, int inh, nfirst, nlast, eb, ew2, ec, n2sta, int \*jstat)

## 30.2619.1 Macro Definition Documentation

30.2619.1.1 #define S1945

Definition at line 42 of file s1945.c.

### 30.2619.2 Function Documentation

30.2619.2.1 void s1945 ( etau , int ik, int in, int idim, et , ed , int im, int ilend, int irend, int inlc, int inlr, int inorm, ea , ew1 , int inh, nfirst, nlast , eb , ew2 , ec , n2sta, int \* jstat )

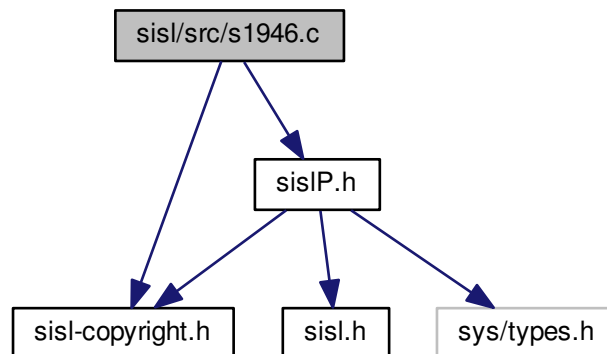
Definition at line 54 of file s1945.c.

### 30.2620 sisl/src/s1946.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1946.c:



#### Macros

- #define [S1946](#)

#### Functions

- void [s1946](#) (ea, ew1, nfirst, nlast, ed, ec, int ik, int in, int im, int idim, int ilend, int irend, int inlr, int inlc, int \*jstat)

### 30.2620.1 Macro Definition Documentation

30.2620.1.1 #define S1946

Definition at line 43 of file s1946.c.

## 30.2620.2 Function Documentation

30.2620.2.1 void s1946 ( ea , ew1 , nfirst , nlast , ed , ec , int ik, int in, int im, int idim, int ilend, int irend, int inlr, int inlc, int \* jstat )

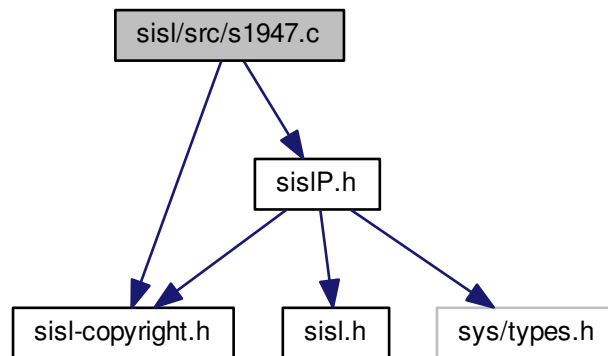
Definition at line 53 of file s1946.c.

## 30.2621 sisl/src/s1947.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1947.c:



### Macros

- #define [S1947](#)

### Functions

- void [s1947](#) (ea, nfirst, nlast, int ik, int im, etau, int in, int incont, ew, int inlr, int \*jnred, efac, int \*jstat)

## 30.2621.1 Macro Definition Documentation

30.2621.1.1 #define S1947

Definition at line 43 of file s1947.c.

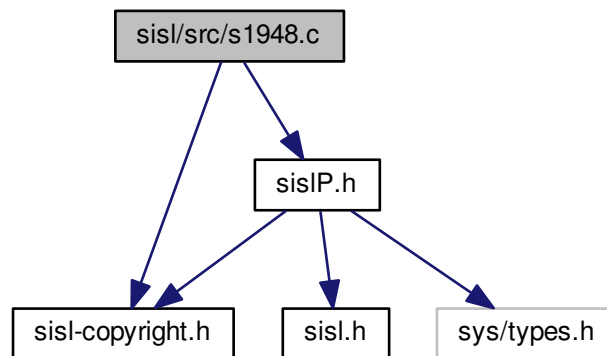
## 30.2621.2 Function Documentation

30.2621.2.1 `void s1947 ( ea , nfirst , nlast , int ik , int im , etau , int in , int incont , ew , int inlr , int * jnred , efac , int * jstat )`

Definition at line 53 of file s1947.c.

## 30.2622 sisl/src/s1948.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for s1948.c:
```



### Macros

- `#define S1948`

### Functions

- `void s1948 (double *ea, double *ew, int in, int ik, int inlr, int *nstart, int *jstat)`

## 30.2622.1 Macro Definition Documentation

30.2622.1.1 `#define S1948`

Definition at line 43 of file s1948.c.



## 30.2622.2 Function Documentation

30.2622.2.1 void s1948 ( double \* ea, double \* ew, int in, int ik, int inlr, int \* nstart, int \* jstat )

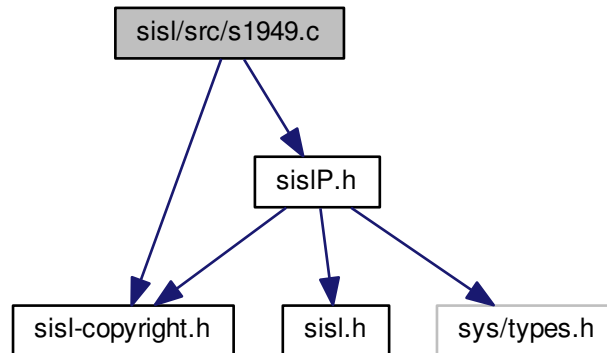
Definition at line 52 of file s1948.c.

## 30.2623 sisl/src/s1949.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1949.c:



### Macros

- #define [S1949](#)

### Functions

- void [s1949](#) (double \*ea, double \*ew, double \*eb, int in, int ik, int inlr, int idim, int \*nstart, int \*jstat)

## 30.2623.1 Macro Definition Documentation

30.2623.1.1 #define S1949

Definition at line 48 of file s1949.c.

### 30.2623.2 Function Documentation

30.2623.2.1 void s1949 ( double \* ea, double \* ew, double \* eb, int in, int ik, int inlr, int idim, int \* nstart, int \* jstat )

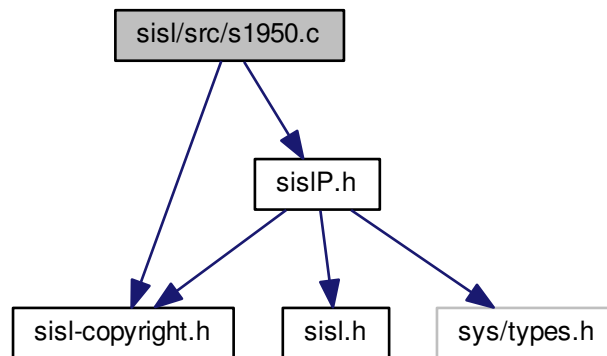
Definition at line 57 of file s1949.c.

### 30.2624 sisl/src/s1950.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1950.c:



#### Macros

- #define [S1950](#)

#### Functions

- void [s1950](#) (SISLCurve \*oldcurve, SISLCurve \*rankcurve, rank\_info \*ranking, eps, epsco, int startfix, int endfix, int \*jncont, int mini, int maxi, SISLCurve \*\*newcurve, maxerr, int \*stat)

### 30.2624.1 Macro Definition Documentation

30.2624.1.1 #define S1950

Definition at line 43 of file s1950.c.

## 30.2624.2 Function Documentation

30.2624.2.1 void s1950 ( SISLCurve \* *oldcurve*, SISLCurve \* *rankcurve*, rank\_info \* *ranking*, eps , epsco , int *startfix*, int *endfix*, int \* *jncont*, int *mini*, int *maxi*, SISLCurve \*\* *newcurve*, maxerr , int \* *stat* )

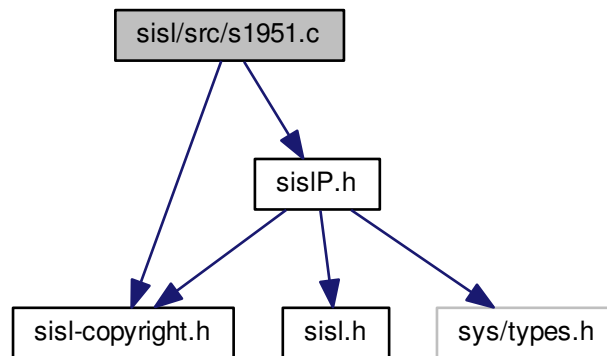
Definition at line 52 of file s1950.c.

## 30.2625 sisl/src/s1951.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1951.c:



### Macros

- #define [S1951](#)

### Functions

- void [s1951](#) (etau, ecoef, int in, int ik, int idim, int ilend, int irend, int incont, efac)

## 30.2625.1 Macro Definition Documentation

30.2625.1.1 #define S1951

Definition at line 43 of file s1951.c.

## 30.2625.2 Function Documentation

30.2625.2.1 void s1951 ( etau , ecoef , int in, int ik, int idim, int ilend, int irend, int incont, efac )

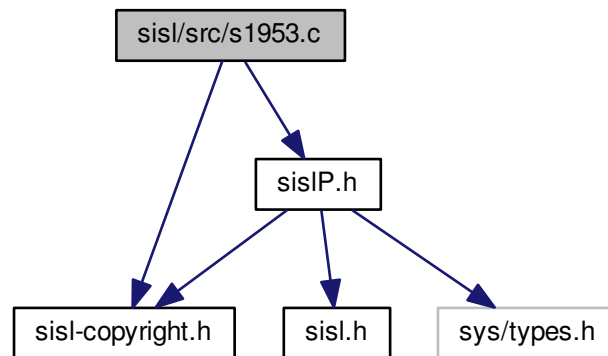
Definition at line 52 of file s1951.c.

## 30.2626 sisl/src/s1953.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1953.c:



### Macros

- #define [S1953](#)

### Functions

- void [s1953](#) (SISLCurve \*pcurve, epoint, int idim, double aepsco, double aepsge, int \*jpt, double \*\*gpar, int \*jcrv, SISLIntcurve \*\*\*wcurve, int \*jstat)

## 30.2626.1 Macro Definition Documentation

30.2626.1.1 #define S1953

Definition at line 49 of file s1953.c.

## 30.2626.2 Function Documentation

30.2626.2.1 void s1953 ( SISLCurve \* pcurve, epoint , int idim, double aepsco, double aepsge, int \* jpt, double \*\* gpar, int \* jcrv, SISLIntcurve \*\*\* wcurve, int \* jstat )

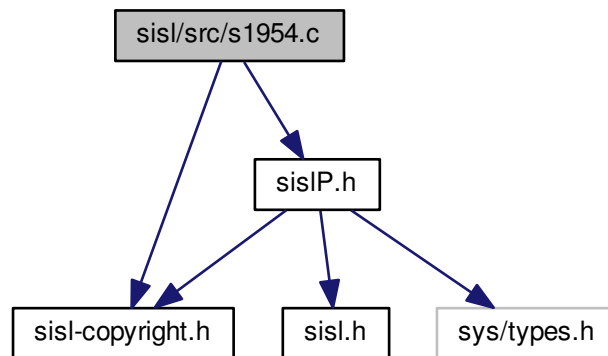
Definition at line 58 of file s1953.c.

## 30.2627 sisl/src/s1954.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1954.c:



## Macros

- #define [S1954](#)

## Functions

- void [s1954](#) (SISLSurf \*psurf, epoint, int idim, double aepsco, double aepsge, int \*jpt, double \*\*gpar, int \*jcrv, SISLIntcurve \*\*\*wcurve, int \*jstat)

## 30.2627.1 Macro Definition Documentation

30.2627.1.1 #define S1954

Definition at line 49 of file s1954.c.

### 30.2627.2 Function Documentation

30.2627.2.1 `void s1954 ( SISLSurf * psurf, epoint, int idim, double aepsco, double aepsge, int * jpt, double ** gpar, int * jcrv, SISLIntcurve *** wcurve, int * jstat )`

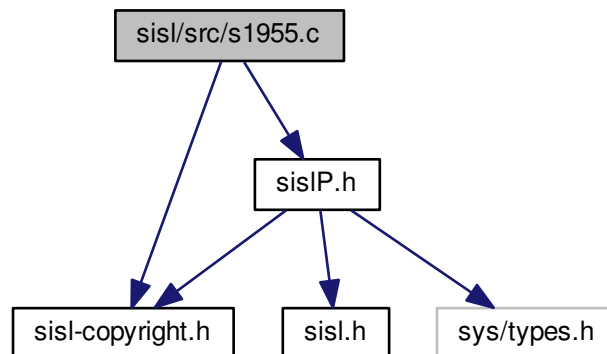
Definition at line 58 of file s1954.c.

### 30.2628 sisl/src/s1955.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1955.c:



#### Macros

- `#define S1955`

#### Functions

- `void s1955 ( SISLCurve *pc1, SISLCurve *pc2, double aepsco, double aepsge, int *jpt, double **gpar1, double **gpar2, int *jcrv, SISLIntcurve ***wcurve, int *jstat)`

### 30.2628.1 Macro Definition Documentation

30.2628.1.1 `#define S1955`

Definition at line 49 of file s1955.c.

## 30.2628.2 Function Documentation

30.2628.2.1 void s1955 ( SISLCurve \* pc1, SISLCurve \* pc2, double aepsco, double aepsge, int \* jpt, double \*\* gpar1, double \*\* gpar2, int \* jcrv, SISLIntcurve \*\*\* wcurve, int \* jstat )

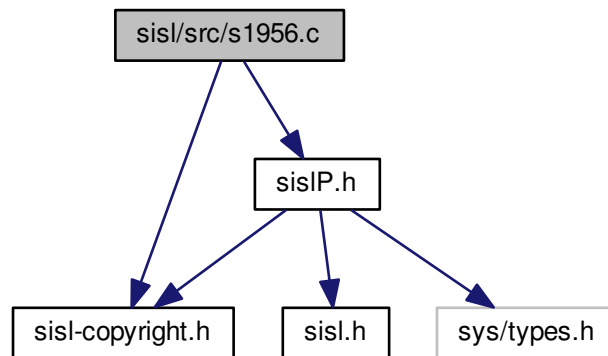
Definition at line 58 of file s1955.c.

## 30.2629 sisl/src/s1956.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1956.c:



### Macros

- #define [S1956](#)

### Functions

- void [s1956](#) (SISLCurve \*pc1, SISLCurve \*pc2, SISLSurf \*\*rsurf, int \*jstat)

## 30.2629.1 Macro Definition Documentation

30.2629.1.1 #define S1956

Definition at line 49 of file s1956.c.

## 30.2629.2 Function Documentation

30.2629.2.1 void s1956 ( SISLCurve \* pc1, SISLCurve \* pc2, SISLSurf \*\* rsurf, int \* jstat )

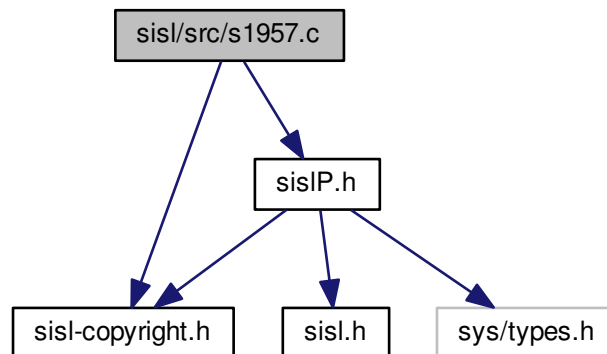
Definition at line 57 of file s1956.c.

## 30.2630 sisl/src/s1957.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1957.c:



### Macros

- #define [S1957](#)

### Functions

- void [s1957](#) (SISLCurve \*pcurve, epoint, int idim, double aepsco, double aepsge, double \*gpar, double \*dist, int \*jstat)

## 30.2630.1 Macro Definition Documentation

30.2630.1.1 #define S1957

Definition at line 49 of file s1957.c.



## 30.2630.2 Function Documentation

30.2630.2.1 void s1957 ( SISLCurve \* *pcurve*, epoint , int *idim*, double *aepsco*, double *aepsge*, double \* *gpar*, double \* *dist*, int \* *jstat* )

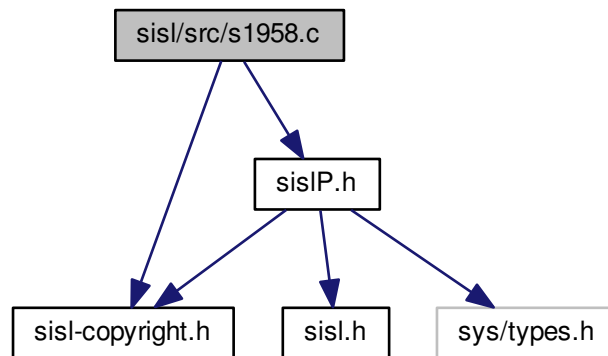
Definition at line 58 of file s1957.c.

## 30.2631 sisl/src/s1958.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1958.c:



### Macros

- #define [S1958](#)

### Functions

- void [s1958](#) (SISLSurf \**psurf*, epoint, int *idim*, double *aepsco*, double *aepsge*, gpar, double \**dist*, int \**jstat*)

## 30.2631.1 Macro Definition Documentation

30.2631.1.1 #define S1958

Definition at line 49 of file s1958.c.

## 30.2631.2 Function Documentation

30.2631.2.1 void `s1958` ( `SISLSurf * psurf`, `epoint`, `int idim`, `double aepsco`, `double aepsge`, `gpar`, `double * dist`, `int * jstat` )

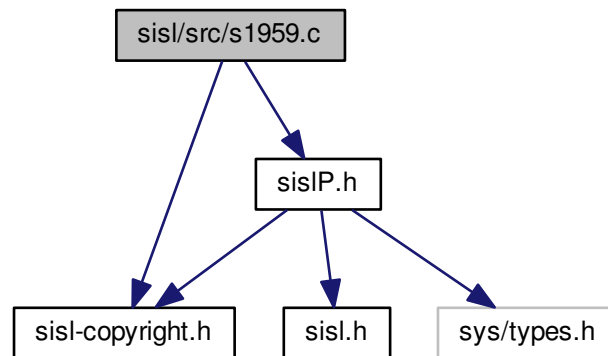
Definition at line 58 of file `s1958.c`.

## 30.2632 `sisl/src/s1959.c` File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for `s1959.c`:



### Macros

- `#define S1959`

### Functions

- void `s1959` (`SISLPoint *ppoint`, `SISLCurve *pcurve`, `double *gpos`, `int *jstat`)

## 30.2632.1 Macro Definition Documentation

30.2632.1.1 `#define S1959`

Definition at line 47 of file `s1959.c`.

## 30.2632.2 Function Documentation

30.2632.2.1 void s1959 ( SISLPoint \* ppoint, SISLCurve \* pcurve, double \* gpos, int \* jstat )

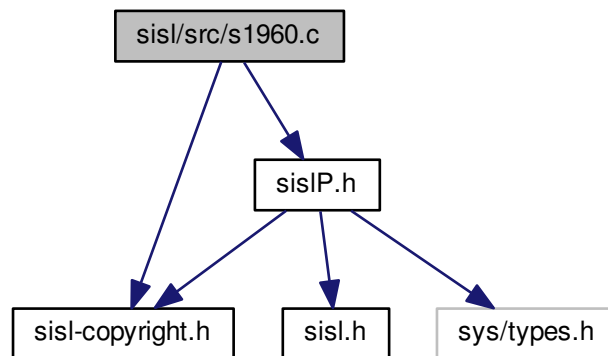
Definition at line 55 of file s1959.c.

## 30.2633 sisl/src/s1960.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1960.c:



### Macros

- #define [S1960](#)

### Functions

- void [s1960](#) (SISLPoint \*ppoint, SISLSurf \*psurf, gpos, int \*jstat)

## 30.2633.1 Macro Definition Documentation

30.2633.1.1 #define S1960

Definition at line 47 of file s1960.c.

### 30.2633.2 Function Documentation

30.2633.2.1 void s1960 ( SISLPoint \* ppoint, SISLSurf \* psurf, gpos , int \* jstat )

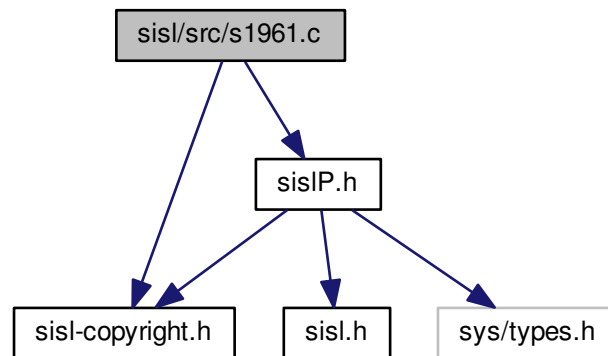
Definition at line 55 of file s1960.c.

### 30.2634 sisl/src/s1961.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1961.c:



#### Macros

- #define [S1961](#)

#### Functions

- void [s1961](#) (ep, int im, int idim, int ipar, epar, eeps, int ilend, int irend, int iopen, double afctol, int itmax, int ik, [SISLCurve](#) \*\*rc, emxerr, int \*jstat)

### 30.2634.1 Macro Definition Documentation

30.2634.1.1 #define S1961

Definition at line 42 of file s1961.c.

## 30.2634.2 Function Documentation

30.2634.2.1 void s1961 ( ep , int im, int idim, int ipar, epar , eeps , int ilend, int irend, int iopen, double afctol, int itmax, int ik, SISLCurve \*\* rc, emxerr , int \* jstat )

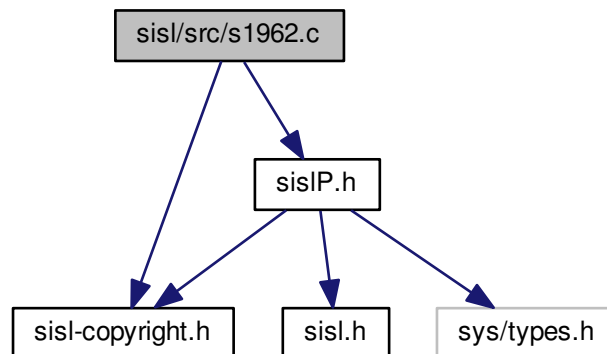
Definition at line 52 of file s1961.c.

## 30.2635 sisl/src/s1962.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1962.c:



### Macros

- #define [S1962](#)

### Functions

- void [s1962](#) (ep, ev, int im, int idim, int ipar, epar, eeps, int ilend, int irend, int iopen, int itmax, [SISLCurve](#) \*\*rc, emxerr, int \*jstat)

## 30.2635.1 Macro Definition Documentation

30.2635.1.1 #define S1962

Definition at line 43 of file s1962.c.

## 30.2635.2 Function Documentation

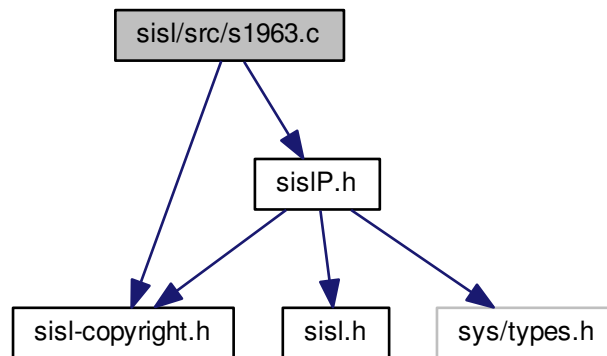
30.2635.2.1 void `s1962` ( `ep` , `ev` , int `im` , int `idim` , int `ipar` , `epar` , `eeps` , int `ilend` , int `irend` , int `iopen` , int `itmax` , `SISLCurve` `**rc` , `emxerr` , int `*jstat` )

Definition at line 52 of file `s1962.c`.

## 30.2636 `sisl/src/s1963.c` File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
```

Include dependency graph for `s1963.c`:



### Macros

- `#define` [S1963](#)

### Functions

- void [s1963](#) (`SISLCurve` `*pc` , `eeps` , int `ilend` , int `irend` , int `iopen` , int `itmax` , `SISLCurve` `**rc` , int `*jstat`)

## 30.2636.1 Macro Definition Documentation

30.2636.1.1 `#define` `S1963`

Definition at line 42 of file `s1963.c`.

## 30.2636.2 Function Documentation

30.2636.2.1 void s1963 ( SISLCurve \* pc, eeps , int ilend, int irend, int iopen, int itmax, SISLCurve \*\* rc, int \* jstat )

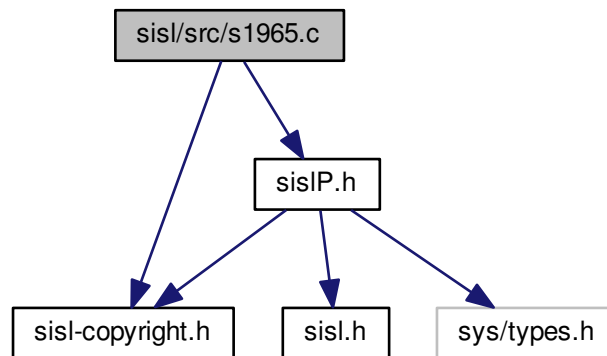
Definition at line 50 of file s1963.c.

## 30.2637 sisl/src/s1965.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1965.c:



### Macros

- #define [S1965](#)

### Functions

- void [s1965](#) (SISLSurf \*oldsurf, eps, edgefix, int iopen1, int iopen2, edgeps, int opt, int itmax, SISLSurf \*\*newsurf, maxerr, int \*stat)

## 30.2637.1 Macro Definition Documentation

30.2637.1.1 #define S1965

Definition at line 43 of file s1965.c.

### 30.2637.2 Function Documentation

30.2637.2.1 `void s1965 ( SISLSurf * oldsurf, eps , edgefix , int iopen1, int iopen2, edgeps , int opt, int itmax, SISLSurf ** newsurf, maxerr , int * stat )`

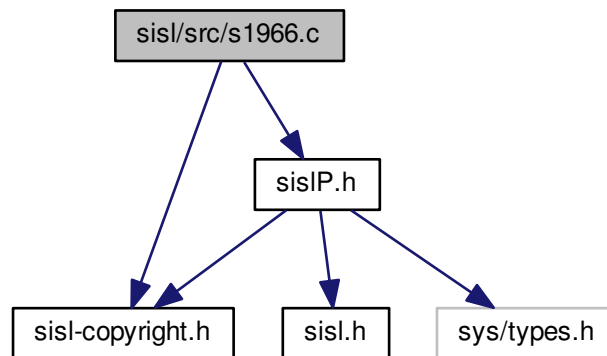
Definition at line 53 of file s1965.c.

### 30.2638 sisl/src/s1966.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1966.c:



#### Macros

- `#define S1966`

#### Functions

- `void s1966 (ep, int im1, int im2, int idim, int ipar, epar1, epar2, eeps, nend, int iopen1, int iopen2, edgeps, double afctol, int iopt, int itmax, int ik1, int ik2, SISLSurf **rs, emxerr, int *jstat)`

### 30.2638.1 Macro Definition Documentation

30.2638.1.1 `#define S1966`

Definition at line 43 of file s1966.c.



## 30.2638.2 Function Documentation

30.2638.2.1 void s1966 ( ep , int im1, int im2, int idim, int ipar, epar1 , epar2 , eeps , nend , int iopen1, int iopen2, edgeps , double afctol, int iopt, int itmax, int ik1, int ik2, SISLSurf \*\* rs, emxerr , int \* jstat )

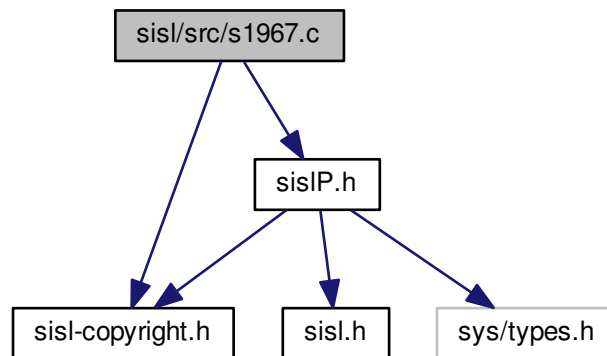
Definition at line 53 of file s1966.c.

## 30.2639 sisl/src/s1967.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1967.c:



### Macros

- #define [S1967](#)

### Functions

- void [s1967](#) (ep, etang1, etang2, eder11, int im1, int im2, int idim, int ipar, epar1, epar2, eeps, nend, int iopen1, int iopen2, edgeps, int iopt, int itmax, [SISLSurf](#) \*\*rs, emxerr, int \*jstat)

## 30.2639.1 Macro Definition Documentation

30.2639.1.1 #define S1967

Definition at line 42 of file s1967.c.

### 30.2639.2 Function Documentation

30.2639.2.1 void `s1967` ( `ep` , `etang1` , `etang2` , `eder11` , `int im1` , `int im2` , `int idim` , `int ipar` , `epar1` , `epar2` , `eeps` , `nend` , `int iopen1` , `int iopen2` , `edgeps` , `int iopt` , `int itmax` , `SISLSurf **rs` , `emxerr` , `int *jstat` )

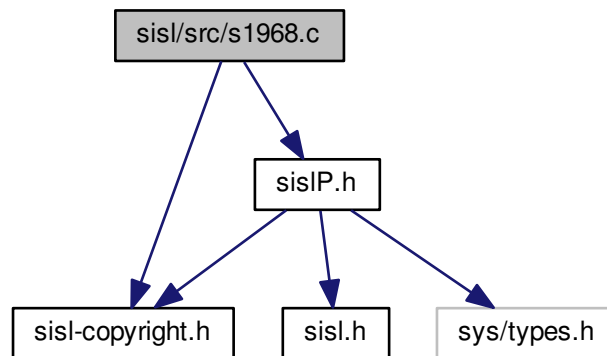
Definition at line 53 of file `s1967.c`.

### 30.2640 sisl/src/s1968.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for `s1968.c`:



#### Macros

- `#define S1968`

#### Functions

- void `s1968` (`SISLSurf *ps` , `eeps` , `nend` , `int iopen1` , `int iopen2` , `edgeps` , `int iopt` , `int itmax` , `SISLSurf **rs` , `int *jstat`)

### 30.2640.1 Macro Definition Documentation

30.2640.1.1 `#define S1968`

Definition at line 43 of file `s1968.c`.

## 30.2640.2 Function Documentation

30.2640.2.1 void s1968 ( SISLSurf \* ps, eeps , nend , int iopen1, int iopen2, edgeps , int iopt, int itmax, SISLSurf \*\* rs, int \* jstat )

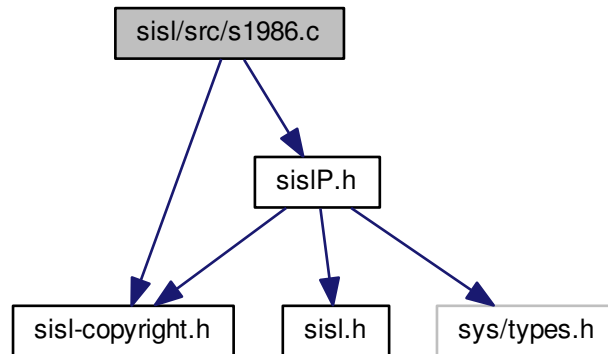
Definition at line 52 of file s1968.c.

## 30.2641 sisl/src/s1986.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1986.c:



### Macros

- #define [S1986](#)

### Functions

- void [s1986](#) (SISLCurve \*pc, double aepsge, int \*jgtpi, double \*\*gaxis, double \*cang, int \*jstat)

## 30.2641.1 Macro Definition Documentation

30.2641.1.1 #define S1986

Definition at line 42 of file s1986.c.

## 30.2641.2 Function Documentation

30.2641.2.1 `void s1986 ( SISLCurve * pc, double aepsge, int * jgtpi, double ** gaxis, double * cang, int * jstat )`

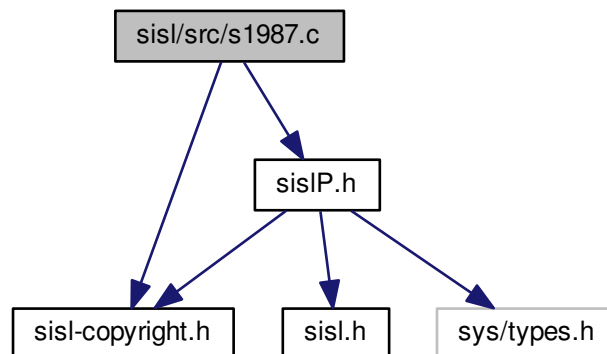
Definition at line 51 of file s1986.c.

## 30.2642 sisl/src/s1987.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1987.c:



### Macros

- `#define S1987`

### Functions

- `void s1987 (SISLSurf *ps, double aepsge, int *jgtpi, double **gaxis, double *cang, int *jstat)`

## 30.2642.1 Macro Definition Documentation

30.2642.1.1 `#define S1987`

Definition at line 42 of file s1987.c.

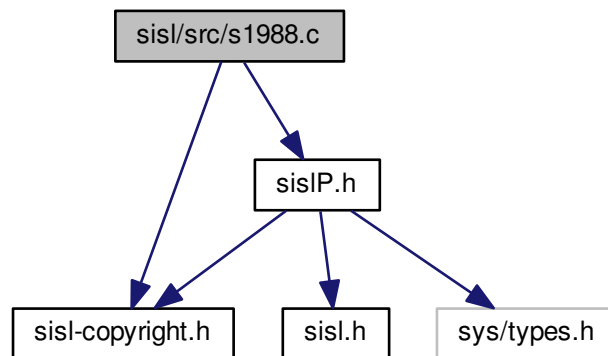
## 30.2642.2 Function Documentation

30.2642.2.1 `void s1987 ( SISLSurf * ps, double aepsge, int * jgtpi, double ** gaxis, double * cang, int * jstat )`

Definition at line 51 of file s1987.c.

## 30.2643 sisl/src/s1988.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for s1988.c:
```



### Macros

- `#define S1988`

### Functions

- `void s1988 (SISLCurve *pc, double **emax, double **emin, int *jstat)`

## 30.2643.1 Macro Definition Documentation

30.2643.1.1 `#define S1988`

Definition at line 49 of file s1988.c.

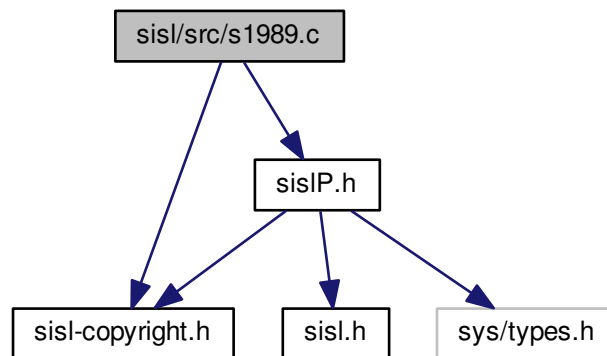
## 30.2643.2 Function Documentation

30.2643.2.1 `void s1988 ( SISLCurve * pc, double ** emax, double ** emin, int * jstat )`

Definition at line 57 of file s1988.c.

## 30.2644 sisl/src/s1989.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for s1989.c:
```



### Macros

- `#define S1989`

### Functions

- `void s1989 (SISLSurf *ps, double **emax, double **emin, int *jstat)`

## 30.2644.1 Macro Definition Documentation

30.2644.1.1 `#define S1989`

Definition at line 49 of file s1989.c.

## 30.2644.2 Function Documentation

30.2644.2.1 void s1989 ( SISLSurf \* ps, double \*\* emax, double \*\* emin, int \* jstat )

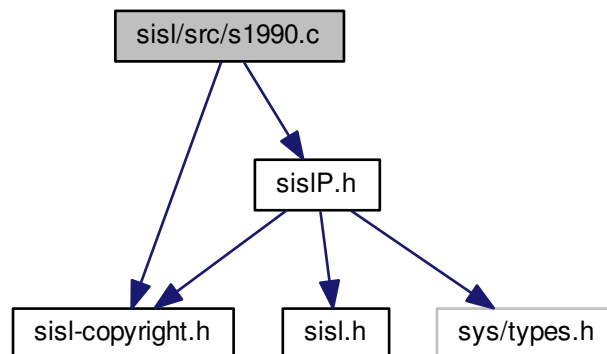
Definition at line 57 of file s1989.c.

## 30.2645 sisl/src/s1990.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1990.c:



### Macros

- #define [S1990](#)

### Functions

- void [s1990](#) (SISLSurf \*ps, double aepsge, int \*jstat)

## 30.2645.1 Macro Definition Documentation

30.2645.1.1 #define S1990

Definition at line 49 of file s1990.c.

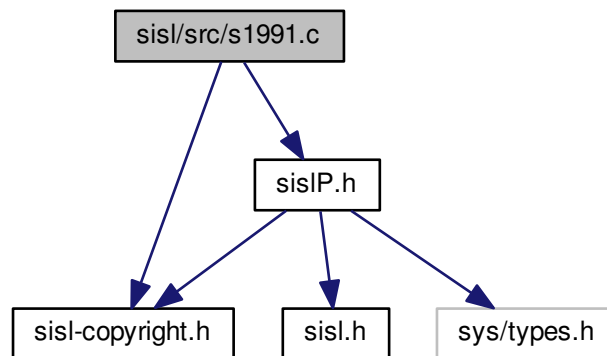
## 30.2645.2 Function Documentation

30.2645.2.1 `void s1990 ( SISLSurf * ps, double aepsge, int * jstat )`

Definition at line 72 of file s1990.c.

## 30.2646 sisl/src/s1991.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for s1991.c:
```



### Macros

- `#define S1991`

### Functions

- `void s1991 (SISLCurve *pc, double aepsge, int *jstat)`

## 30.2646.1 Macro Definition Documentation

30.2646.1.1 `#define S1991`

Definition at line 49 of file s1991.c.



## 30.2646.2 Function Documentation

30.2646.2.1 void s1991 ( SISLCurve \* pc, double aepsge, int \* jstat )

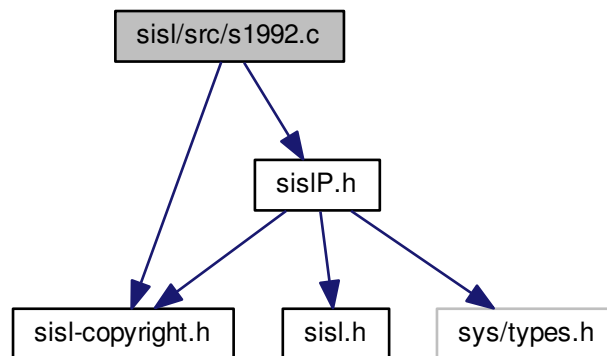
Definition at line 64 of file s1991.c.

## 30.2647 sisl/src/s1992.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1992.c:



### Macros

- #define [S1992](#)

### Functions

- void [s1992](#) (SISLObject \*po, int \*jstat)
- void [s1992cu](#) (SISLCurve \*pc, int \*jstat)
- void [s1992su](#) (SISLSurf \*ps, int \*jstat)

## 30.2647.1 Macro Definition Documentation

30.2647.1.1 #define S1992

Definition at line 49 of file s1992.c.

## 30.2647.2 Function Documentation

30.2647.2.1 void `s1992` ( `SISLObject * po`, int \* `jstat` )

Definition at line 75 of file `s1992.c`.

30.2647.2.2 void `s1992cu` ( `SISLCurve * pc`, int \* `jstat` )

Definition at line 193 of file `s1992.c`.

30.2647.2.3 void `s1992su` ( `SISLSurf * ps`, int \* `jstat` )

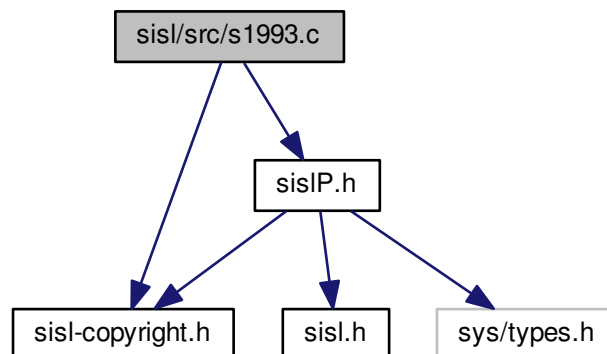
Definition at line 265 of file `s1992.c`.

## 30.2648 `sisl/src/s1993.c` File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for `s1993.c`:



### Macros

- `#define S1993`

### Functions

- void `s1993` (`SISLCurve *c1`, int \*`jstat`)

## 30.2648.1 Macro Definition Documentation

### 30.2648.1.1 #define S1993

Definition at line 49 of file s1993.c.

## 30.2648.2 Function Documentation

### 30.2648.2.1 void s1993 ( SISLCurve \* c1, int \* jstat )

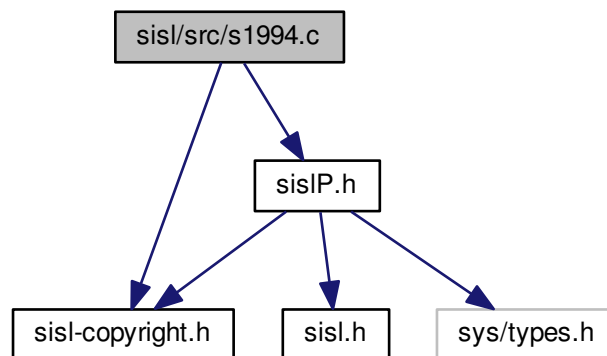
Definition at line 57 of file s1993.c.

## 30.2649 sisl/src/s1994.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s1994.c:



## Macros

- #define [S1994](#)

## Functions

- void [s1994](#) ([SISLSurf](#) \*s1, int \*jstat)

### 30.2649.1 Macro Definition Documentation

#### 30.2649.1.1 #define S1994

Definition at line 49 of file s1994.c.

### 30.2649.2 Function Documentation

#### 30.2649.2.1 void s1994 ( SISLSurf \* s1, int \* jstat )

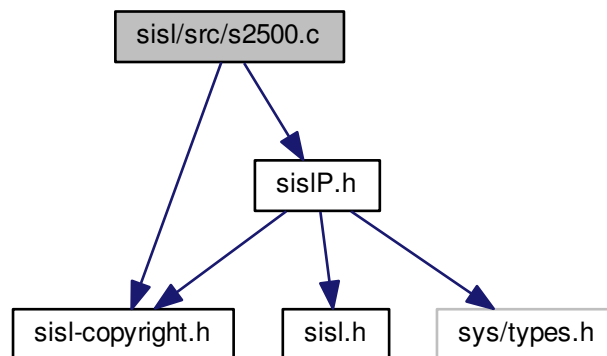
Definition at line 57 of file s1994.c.

## 30.2650 sisl/src/s2500.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s2500.c:



### Macros

- #define [S2500](#)

### Functions

- void [s2500](#) (SISLSurf \*surf, int ider, int iside1, int iside2, parvalue, int \*leftknot1, int \*leftknot2, double \*gaussian, int \*jstat)

## 30.2650.1 Macro Definition Documentation

### 30.2650.1.1 #define S2500

Definition at line 49 of file s2500.c.

## 30.2650.2 Function Documentation

### 30.2650.2.1 void s2500 ( SISLSurf \* surf, int ider, int iside1, int iside2, parvalue , int \* leftknot1, int \* leftknot2, double \* gaussian, int \* jstat )

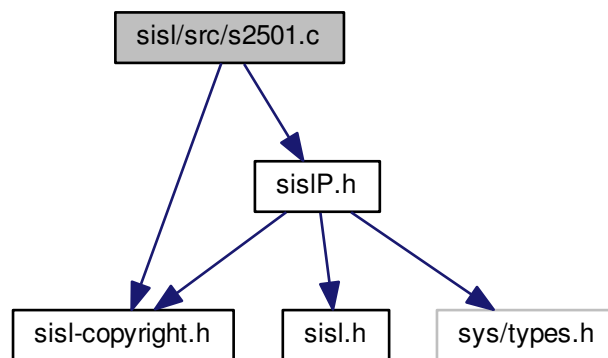
Definition at line 58 of file s2500.c.

## 30.2651 sisl/src/s2501.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s2501.c:



## Macros

- #define [S2501](#)

## Functions

- void [s2501](#) (SISLSurf \*surf, int ider, derive, normal, double \*gaussian, int \*jstat)

### 30.2651.1 Macro Definition Documentation

#### 30.2651.1.1 #define S2501

Definition at line 49 of file s2501.c.

### 30.2651.2 Function Documentation

#### 30.2651.2.1 void s2501 ( SISLSurf \* surf, int nder, derive , normal , double \* gaussian, int \* jstat )

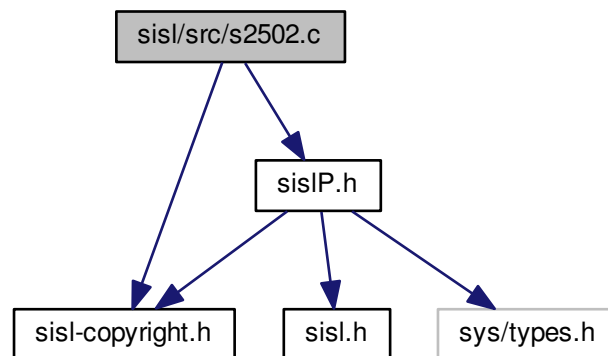
Definition at line 58 of file s2501.c.

## 30.2652 sisl/src/s2502.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s2502.c:



### Macros

- #define [S2502](#)

### Functions

- void [s2502](#) (SISLSurf \*surf, int nder, int iside1, int iside2, parvalue, int \*leftknot1, int \*leftknot2, double \*meancurvature, int \*jstat)

## 30.2652.1 Macro Definition Documentation

### 30.2652.1.1 #define S2502

Definition at line 49 of file s2502.c.

## 30.2652.2 Function Documentation

### 30.2652.2.1 void s2502 ( SISLSurf \* surf, int ider, int iside1, int iside2, parvalue , int \* leftknot1, int \* leftknot2, double \* meancurvature, int \* jstat )

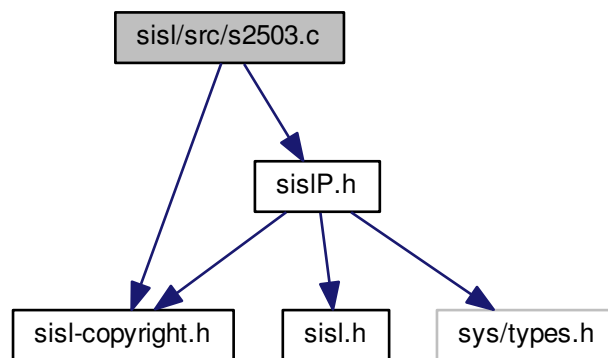
Definition at line 58 of file s2502.c.

## 30.2653 sisl/src/s2503.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s2503.c:



## Macros

- #define [S2503](#)

## Functions

- void [s2503](#) (SISLSurf \*surf, int ider, derive, normal, double \*meancurvature, int \*jstat)

### 30.2653.1 Macro Definition Documentation

#### 30.2653.1.1 #define S2503

Definition at line 49 of file s2503.c.

### 30.2653.2 Function Documentation

#### 30.2653.2.1 void s2503 ( SISLSurf \* surf, int ider, derive , normal , double \* meancurvature, int \* jstat )

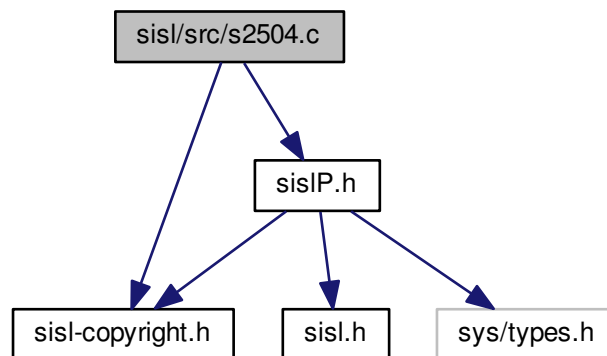
Definition at line 57 of file s2503.c.

## 30.2654 sisl/src/s2504.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s2504.c:



### Macros

- #define [S2504](#)

### Functions

- void [s2504](#) (SISLSurf \*surf, int ider, int iside1, int iside2, parvalue, int \*leftknot1, int \*leftknot2, double \*abs← Curvature, int \*jstat)



### 30.2654.1 Macro Definition Documentation

#### 30.2654.1.1 #define S2504

Definition at line 49 of file s2504.c.

### 30.2654.2 Function Documentation

#### 30.2654.2.1 void s2504 ( SISLSurf \* surf, int nder, int iside1, int iside2, parvalue , int \* leftknot1, int \* leftknot2, double \* absCurvature, int \* jstat )

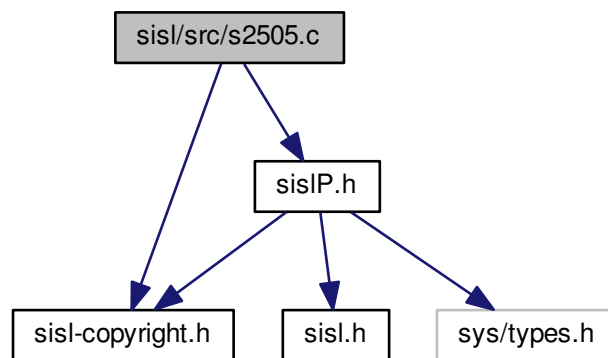
Definition at line 58 of file s2504.c.

## 30.2655 sisl/src/s2505.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s2505.c:



### Macros

- #define [S2505](#)

### Functions

- void [s2505](#) (SISLSurf \*surf, int der, derive, normal, double \*absCurvature, int \*jstat)

### 30.2655.1 Macro Definition Documentation

#### 30.2655.1.1 #define S2505

Definition at line 49 of file s2505.c.

### 30.2655.2 Function Documentation

#### 30.2655.2.1 void s2505 ( SISLSurf \* surf, int der, derive , normal , double \* absCurvature, int \* jstat )

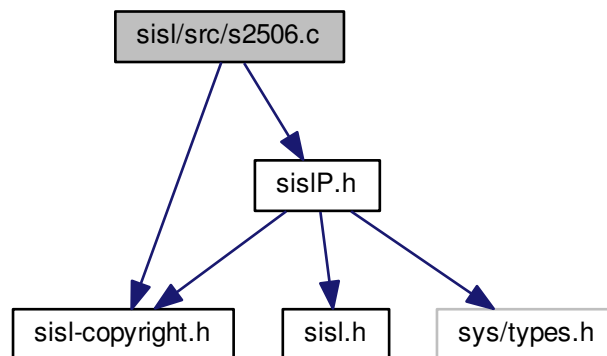
Definition at line 58 of file s2505.c.

## 30.2656 sisl/src/s2506.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s2506.c:



### Macros

- #define [S2506](#)

### Functions

- void [s2506](#) (SISLSurf \*surf, int ider, int iside1, int iside2, parvalue, int \*leftknot1, int \*leftknot2, double \*total← Curvature, int \*jstat)

## 30.2656.1 Macro Definition Documentation

### 30.2656.1.1 #define S2506

Definition at line 49 of file s2506.c.

## 30.2656.2 Function Documentation

### 30.2656.2.1 void s2506 ( SISLSurf \* surf, int ider, int iside1, int iside2, parvalue , int \* leftknot1, int \* leftknot2, double \* totalCurvature, int \* jstat )

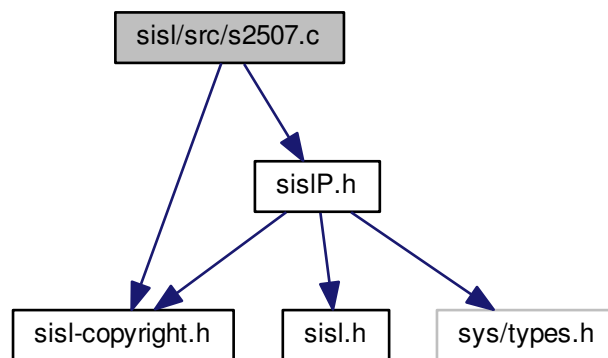
Definition at line 58 of file s2506.c.

## 30.2657 sisl/src/s2507.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s2507.c:



## Macros

- #define [S2507](#)

## Functions

- void [s2507](#) (SISLSurf \*surf, int ider, derive, normal, double \*totalCurvature, int \*jstat)

### 30.2657.1 Macro Definition Documentation

#### 30.2657.1.1 #define S2507

Definition at line 49 of file s2507.c.

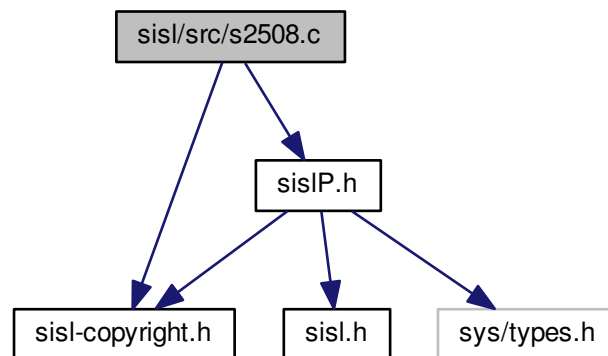
### 30.2657.2 Function Documentation

#### 30.2657.2.1 void s2507 ( SISLSurf \* surf, int ider, derive , normal , double \* totalCurvature, int \* jstat )

Definition at line 58 of file s2507.c.

## 30.2658 sisl/src/s2508.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for s2508.c:
```



### Macros

- #define [S2508](#)

### Functions

- void [s2508](#) (SISLSurf \*surf, int ider, int iside1, int iside2, parvalue, int \*leftknot1, int \*leftknot2, double \*mehlum, int \*jstat)

## 30.2658.1 Macro Definition Documentation

### 30.2658.1.1 #define S2508

Definition at line 49 of file s2508.c.

## 30.2658.2 Function Documentation

### 30.2658.2.1 void s2508 ( SISLSurf \* surf, int ider, int iside1, int iside2, parvalue , int \* leftknot1, int \* leftknot2, double \* mehlum, int \* jstat )

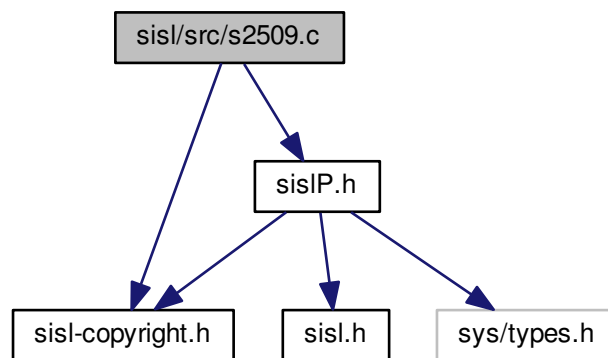
Definition at line 58 of file s2508.c.

## 30.2659 sisl/src/s2509.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s2509.c:



## Macros

- #define [S2509](#)

## Functions

- void [s2509](#) (SISLSurf \*surf, int ider, derive, normal, double \*mehlum, int \*jstat)

### 30.2659.1 Macro Definition Documentation

#### 30.2659.1.1 #define S2509

Definition at line 49 of file s2509.c.

### 30.2659.2 Function Documentation

#### 30.2659.2.1 void s2509 ( SISLSurf \* surf, int iber, derive , normal , double \* mehlum, int \* jstat )

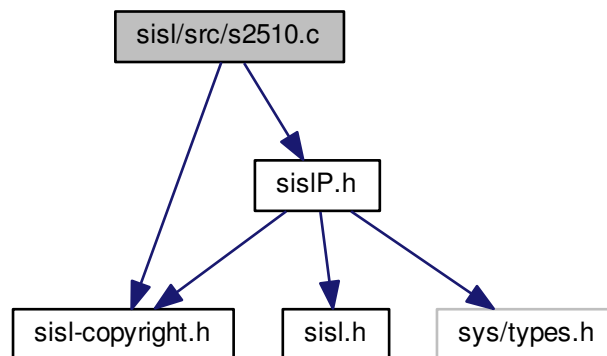
Definition at line 58 of file s2509.c.

## 30.2660 sisl/src/s2510.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s2510.c:



### Macros

- #define [S2510](#)

### Functions

- void [s2510](#) (SISLSurf \*surf, int iber, int iside1, int iside2, parvalue, int \*leftknot1, int \*leftknot2, double \*mehlum, int \*jstat)

## 30.2660.1 Macro Definition Documentation

### 30.2660.1.1 #define S2510

Definition at line 49 of file s2510.c.

## 30.2660.2 Function Documentation

### 30.2660.2.1 void s2510 ( SISLSurf \* surf, int ider, int iside1, int iside2, parvalue , int \* leftknot1, int \* leftknot2, double \* mehlum, int \* jstat )

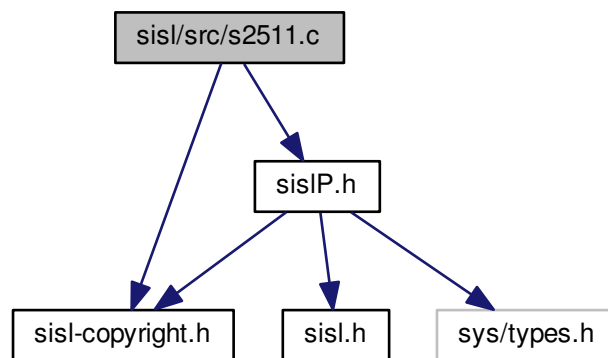
Definition at line 58 of file s2510.c.

## 30.2661 sisl/src/s2511.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s2511.c:



## Macros

- #define [S2511](#)

## Functions

- void [s2511](#) (SISLSurf \*surf, int ider, derive, [normal](#), [double](#) \*mehlum, int \*jstat)

### 30.2661.1 Macro Definition Documentation

#### 30.2661.1.1 #define S2511

Definition at line 49 of file s2511.c.

### 30.2661.2 Function Documentation

#### 30.2661.2.1 void s2511 ( SISLSurf \* surf, int iber, derive , normal , double \* mehlum, int \* jstat )

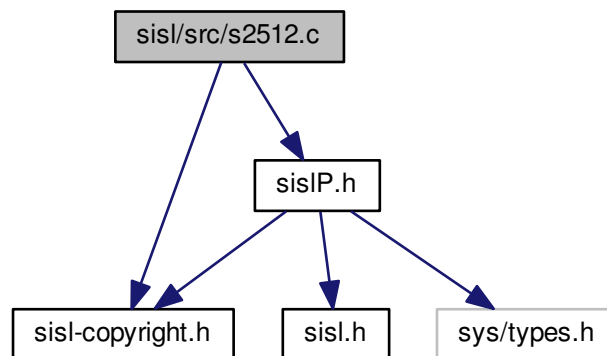
Definition at line 58 of file s2511.c.

## 30.2662 sisl/src/s2512.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s2512.c:



### Macros

- #define [S2512](#)

### Functions

- void [s2512](#) (SISLSurf \*surf, int iber, int iside1, int iside2, parvalue, int \*leftknot1, int \*leftknot2, gaussian, int \*stat)



## 30.2662.1 Macro Definition Documentation

### 30.2662.1.1 #define S2512

Definition at line 49 of file s2512.c.

## 30.2662.2 Function Documentation

### 30.2662.2.1 void s2512 ( SISLSurf \* surf, int ider, int iside1, int iside2, parvalue , int \* leftknot1, int \* leftknot2, gaussian , int \* stat )

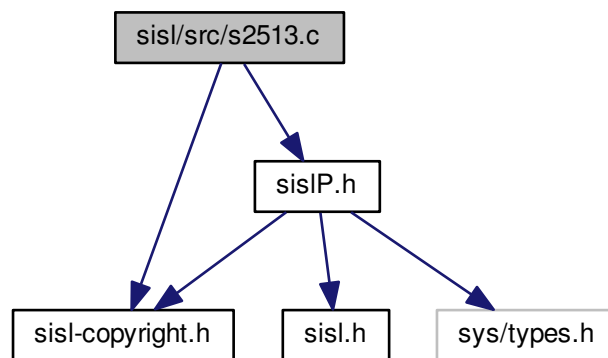
Definition at line 58 of file s2512.c.

## 30.2663 sisl/src/s2513.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s2513.c:



## Macros

- #define [S2513](#)

## Functions

- void [s2513](#) (SISLSurf \*surf, int ider, int type, int normalized, derive, [normal](#), fundform, int \*stat)

### 30.2663.1 Macro Definition Documentation

#### 30.2663.1.1 #define S2513

Definition at line 49 of file s2513.c.

### 30.2663.2 Function Documentation

#### 30.2663.2.1 void s2513 ( SISLSurf \* surf, int ider, int type, int normalized, derive , normal , fundform , int \* stat )

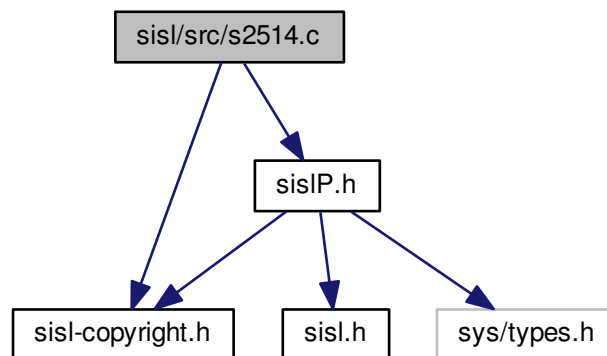
Definition at line 58 of file s2513.c.

## 30.2664 sisl/src/s2514.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s2514.c:



### Macros

- #define [S2514](#)

### Functions

- void [s2514](#) (SISLSurf \*surf, int ider, derive, [normal](#), gaussian, int \*stat)

### 30.2664.1 Macro Definition Documentation

#### 30.2664.1.1 #define S2514

Definition at line 49 of file s2514.c.

### 30.2664.2 Function Documentation

#### 30.2664.2.1 void s2514 ( SISLSurf \* surf, int ider, derive , normal , gaussian , int \* stat )

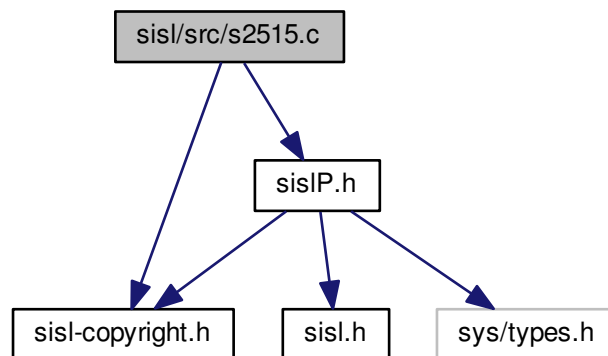
Definition at line 58 of file s2514.c.

## 30.2665 sisl/src/s2515.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s2515.c:



### Macros

- #define [S2515](#)

### Functions

- void [s2515](#) (SISLSurf \*surf, int ider, int iside1, int iside2, parvalue, int \*leftknot1, int \*leftknot2, mehlum, int \*stat)

### 30.2665.1 Macro Definition Documentation

#### 30.2665.1.1 #define S2515

Definition at line 49 of file s2515.c.

### 30.2665.2 Function Documentation

#### 30.2665.2.1 void s2515 ( SISLSurf \* surf, int ider, int iside1, int iside2, parvalue , int \* leftknot1, int \* leftknot2, mehlum , int \* stat )

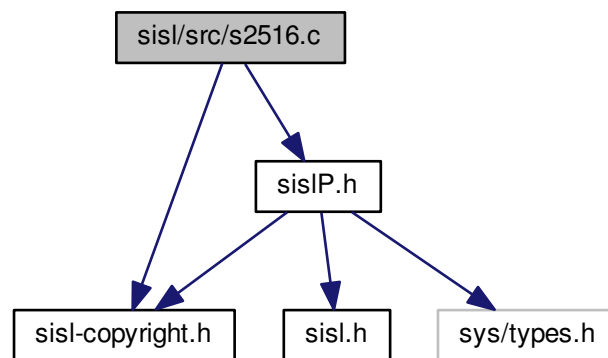
Definition at line 58 of file s2515.c.

### 30.2666 sisl/src/s2516.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s2516.c:



#### Macros

- #define [S2516](#)

#### Functions

- void [s2516](#) (SISLSurf \*surf, int ider, derive, [normal](#), mehlum, int \*stat)

### 30.2666.1 Macro Definition Documentation

#### 30.2666.1.1 #define S2516

Definition at line 49 of file s2516.c.

### 30.2666.2 Function Documentation

#### 30.2666.2.1 void s2516 ( SISLSurf \* surf, int nder, derive , normal , mehlum , int \* stat )

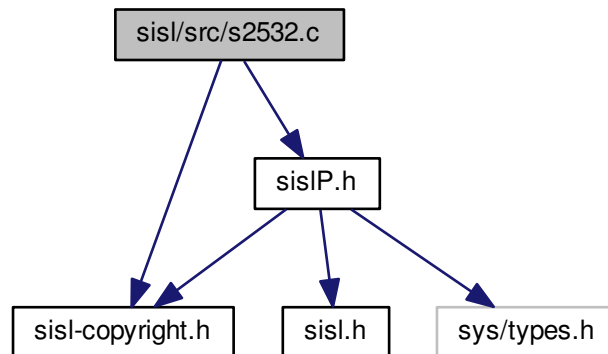
Definition at line 58 of file s2516.c.

## 30.2667 sisl/src/s2532.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s2532.c:



### Macros

- #define [S2532](#)

### Functions

- void [s2532](#) (SISLSurf \*surf, int u\_continuity, int v\_continuity, int \*u\_surfnumb, int \*v\_surfnumb, SISLSurf \*\*\*gauss\_surf, int \*stat)

### 30.2667.1 Macro Definition Documentation

#### 30.2667.1.1 #define S2532

Definition at line 48 of file s2532.c.

### 30.2667.2 Function Documentation

#### 30.2667.2.1 void s2532 ( SISLSurf \* surf, int u\_continuity, int v\_continuity, int \* u\_surfnumb, int \* v\_surfnumb, SISLSurf \*\*\* gauss\_surf, int \* stat )

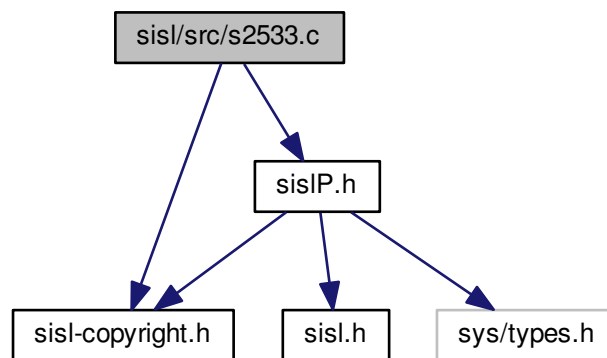
Definition at line 59 of file s2532.c.

## 30.2668 sisl/src/s2533.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s2533.c:



### Macros

- #define [S2533](#)

### Functions

- void [s2533](#) (double \*et, int ik, int in, int multinc, int newik, int \*newin, double \*\*newet, int \*stat)

## 30.2668.1 Macro Definition Documentation

### 30.2668.1.1 #define S2533

Definition at line 49 of file s2533.c.

## 30.2668.2 Function Documentation

### 30.2668.2.1 void s2533 ( double \* et, int ik, int in, int multinc, int newik, int \* newin, double \*\* newet, int \* stat )

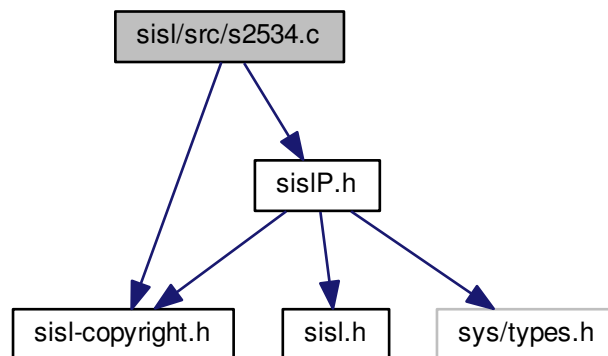
Definition at line 57 of file s2533.c.

## 30.2669 sisl/src/s2534.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s2534.c:



## Macros

- #define [S2534](#)

## Functions

- void [s2534](#) ([SISLSurf](#) \*surf, int u\_multinc, int v\_multinc, int newik1, int newik2, evalp, int eval\_dim, [SISLSurf](#) \*\*rsurf, int \*stat)

### 30.2669.1 Macro Definition Documentation

#### 30.2669.1.1 #define S2534

Definition at line 49 of file s2534.c.

### 30.2669.2 Function Documentation

#### 30.2669.2.1 void s2534 ( SISLSurf \* surf, int u\_multinc, int v\_multinc, int newik1, int newik2, evalp , int eval\_dim, SISLSurf \*\* rsurf, int \* stat )

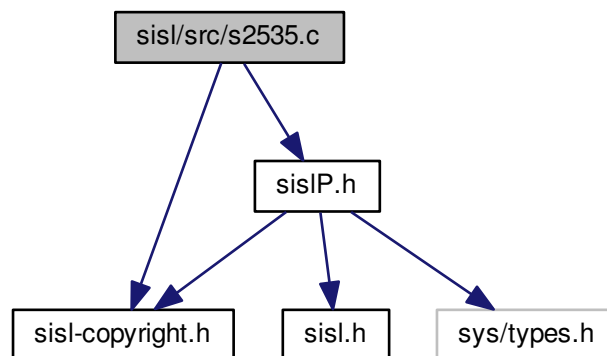
Definition at line 68 of file s2534.c.

## 30.2670 sisl/src/s2535.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s2535.c:



### Macros

- #define [S2535](#)

### Functions

- void [s2535](#) (SISLSurf \*surf, int u\_continuity, int v\_continuity, int \*u\_surfnumb, int \*v\_surfnumb, SISLSurf \*\*\*patches, int \*stat)



## 30.2670.1 Macro Definition Documentation

### 30.2670.1.1 #define S2535

Definition at line 48 of file s2535.c.

## 30.2670.2 Function Documentation

### 30.2670.2.1 void s2535 ( SISLSurf \* surf, int u\_continuity, int v\_continuity, int \* u\_surfnumb, int \* v\_surfnumb, SISLSurf \*\*\* patches, int \* stat )

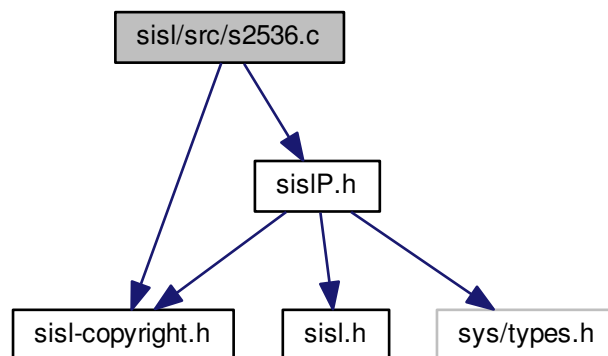
Definition at line 59 of file s2535.c.

## 30.2671 sisl/src/s2536.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s2536.c:



## Macros

- #define [S2536](#)

## Functions

- void [s2536](#) (SISLSurf \*surf, int u\_continuity, int v\_continuity, int \*u\_surfnumb, int \*v\_surfnumb, SISLSurf \*\*\*mehlum\_surf, int \*stat)

### 30.2671.1 Macro Definition Documentation

#### 30.2671.1.1 #define S2536

Definition at line 49 of file s2536.c.

### 30.2671.2 Function Documentation

#### 30.2671.2.1 void s2536 ( SISLSurf \* surf, int u\_continuity, int v\_continuity, int \* u\_surfnumb, int \* v\_surfnumb, SISLSurf \*\*\* mehlum\_surf, int \* stat )

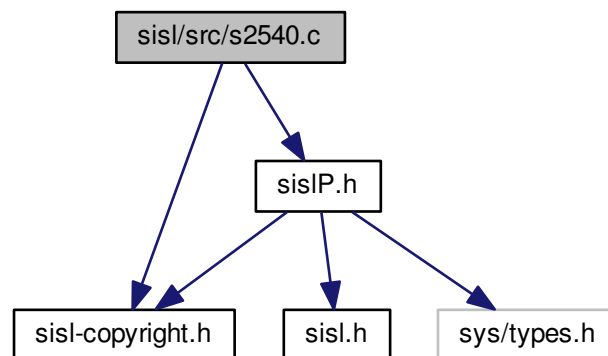
Definition at line 60 of file s2536.c.

## 30.2672 sisl/src/s2540.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s2540.c:



### Macros

- #define [S2540](#)

### Functions

- void [s2540](#) (SISLSurf \*surf, int curvature\_type, int export\_par\_val, int pick\_subpart, boundary, int n\_u, int n\_v, double \*\*garr, int \*stat)

## 30.2672.1 Macro Definition Documentation

### 30.2672.1.1 #define S2540

Definition at line 48 of file s2540.c.

## 30.2672.2 Function Documentation

### 30.2672.2.1 void s2540 ( SISLSurf \* surf, int curvature\_type, int export\_par\_val, int pick\_subpart, boundary, int n\_u, int n\_v, double \*\* garr, int \* stat )

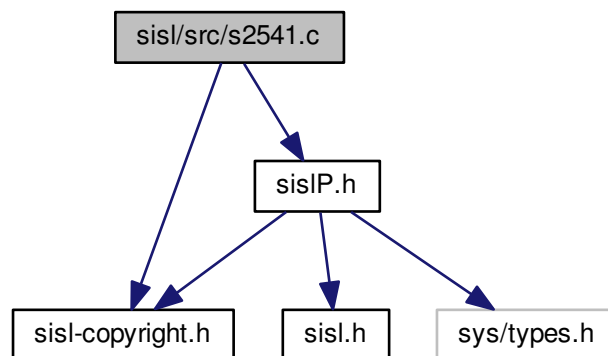
Definition at line 61 of file s2540.c.

## 30.2673 sisl/src/s2541.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s2541.c:



## Macros

- #define [S2541](#)

## Functions

- void [s2541](#) (SISLSurf \*surf, evalp, int dim, int export\_par\_val, int pick\_subpart, boundary, int n\_u, int n\_v, double \*\*garr, int \*stat)

### 30.2673.1 Macro Definition Documentation

#### 30.2673.1.1 #define S2541

Definition at line 48 of file s2541.c.

### 30.2673.2 Function Documentation

#### 30.2673.2.1 void s2541 ( SISLSurf \* surf, evalp , int dim, int export\_par\_val, int pick\_subpart, boundary , int n\_u, int n\_v, double \*\* garr, int \* stat )

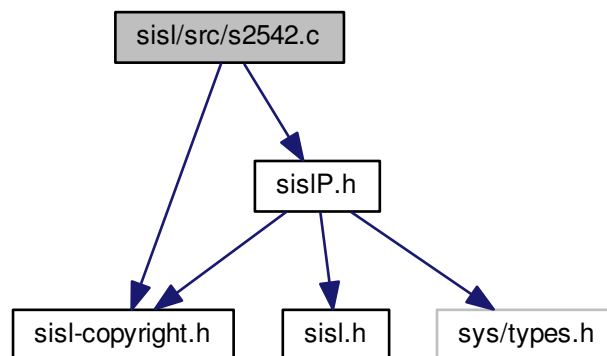
Definition at line 71 of file s2541.c.

## 30.2674 sisl/src/s2542.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s2542.c:



### Macros

- #define [S2542](#)

### Functions

- void [s2542](#) (SISLSurf \*surf, int ider, int iside1, int iside2, parvalue, int \*leftknot1, int \*leftknot2, double \*k1, double \*k2, d1, d2, int \*jstat)

### 30.2674.1 Macro Definition Documentation

#### 30.2674.1.1 #define S2542

Definition at line 49 of file s2542.c.

### 30.2674.2 Function Documentation

#### 30.2674.2.1 void s2542 ( SISLSurf \* surf, int ider, int iside1, int iside2, parvalue , int \* leftknot1, int \* leftknot2, double \* k1, double \* k2, d1 , d2 , int \* jstat )

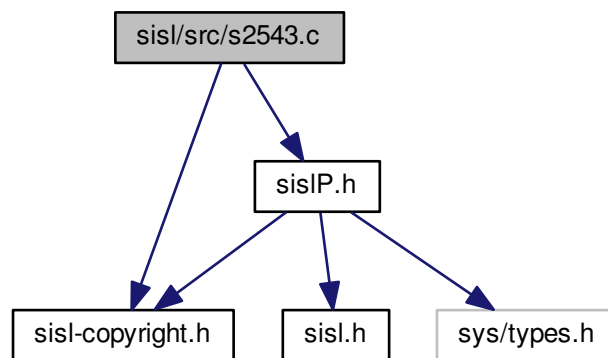
Definition at line 59 of file s2542.c.

## 30.2675 sisl/src/s2543.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s2543.c:



### Macros

- #define [S2543](#)

### Functions

- void [s2543](#) (SISLSurf \*surf, int ider, derive, normal, double \*k1, double \*k2, d1, d2, int \*jstat)

### 30.2675.1 Macro Definition Documentation

#### 30.2675.1.1 #define S2543

Definition at line 49 of file s2543.c.

### 30.2675.2 Function Documentation

#### 30.2675.2.1 void s2543 ( SISLSurf \* surf, int iber, derive , normal , double \* k1, double \* k2, d1 , d2 , int \* jstat )

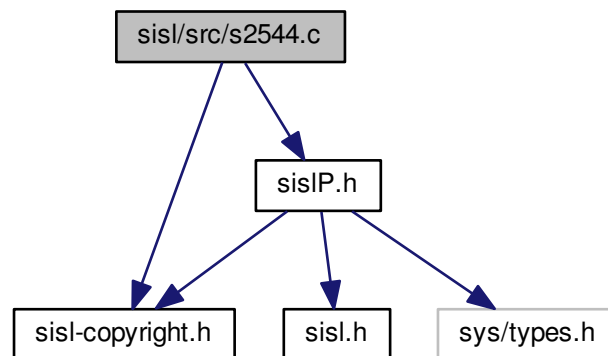
Definition at line 58 of file s2543.c.

## 30.2676 sisl/src/s2544.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s2544.c:



### Macros

- #define [S2544](#)

### Functions

- void [s2544](#) (SISLSurf \*surf, int iber, int iside1, int iside2, parvalue, int \*leftknot1, int \*leftknot2, norcurv, int \*jstat)

## 30.2676.1 Macro Definition Documentation

### 30.2676.1.1 #define S2544

Definition at line 49 of file s2544.c.

## 30.2676.2 Function Documentation

### 30.2676.2.1 void s2544 ( SISLSurf \* surf, int iber, int iside1, int iside2, parvalue , int \* leftknot1, int \* leftknot2, norcurv , int \* jstat )

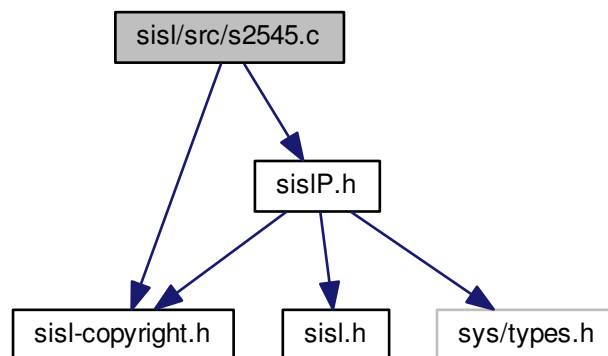
Definition at line 58 of file s2544.c.

## 30.2677 sisl/src/s2545.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s2545.c:



## Macros

- #define [S2545](#)

## Functions

- void [s2545](#) (SISLSurf \*surf, int curvature\_type, int export\_par\_val, int pick\_subpart, boundary, int n\_u, int n\_v, double scale, double \*\*garr, int \*stat)

## 30.2677.1 Macro Definition Documentation

### 30.2677.1.1 #define S2545

Definition at line 48 of file s2545.c.

## 30.2677.2 Function Documentation

### 30.2677.2.1 void s2545 ( SISLSurf \* surf, int curvature\_type, int export\_par\_val, int pick\_subpart, boundary , int n\_u, int n\_v, double scale, double \*\* garr, int \* stat )

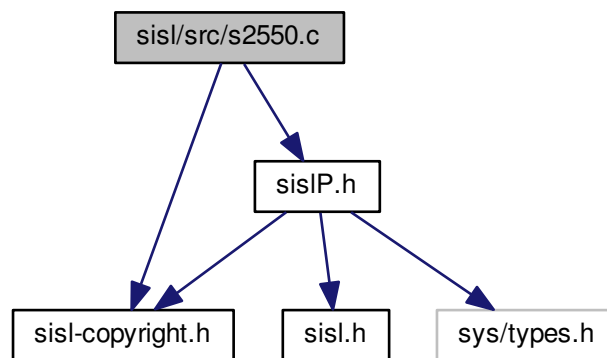
Definition at line 61 of file s2545.c.

## 30.2678 sisl/src/s2550.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s2550.c:



### Macros

- #define [S2550](#)

### Functions

- void [s2550](#) (SISLCurve \*curve, ax, int num\_ax, curvature, int \*jstat)



## 30.2678.1 Macro Definition Documentation

### 30.2678.1.1 #define S2550

Definition at line 48 of file s2550.c.

## 30.2678.2 Function Documentation

### 30.2678.2.1 void s2550 ( SISLCurve \* curve, ax , int num\_ax, curvature , int \* jstat )

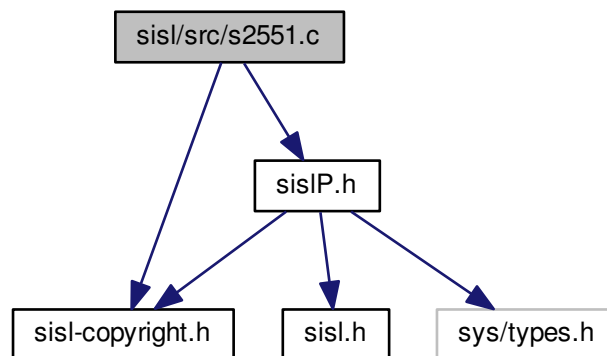
Definition at line 60 of file s2550.c.

## 30.2679 sisl/src/s2551.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s2551.c:



## Macros

- #define [S2551](#)

## Functions

- void [s2551](#) (SISLCurve \*curve, double parvalue, int \*leftknot, derive, double \*curvature, int \*jstat)

## 30.2679.1 Macro Definition Documentation

### 30.2679.1.1 #define S2551

Definition at line 48 of file s2551.c.

## 30.2679.2 Function Documentation

### 30.2679.2.1 void s2551 ( SISLCurve \* curve, double parvalue, int \* leftknot, derive , double \* curvature, int \* jstat )

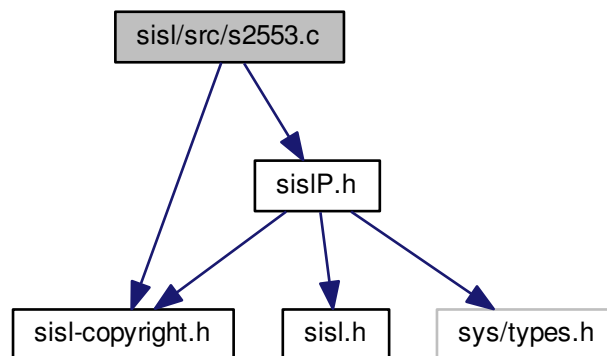
Definition at line 61 of file s2551.c.

## 30.2680 sisl/src/s2553.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s2553.c:



### Macros

- #define [S2553](#)

### Functions

- void [s2553](#) ([SISLCurve](#) \*curve, ax, int num\_ax, torsion, int \*jstat)

## 30.2680.1 Macro Definition Documentation

### 30.2680.1.1 #define S2553

Definition at line 48 of file s2553.c.

## 30.2680.2 Function Documentation

### 30.2680.2.1 void s2553 ( SISLCurve \* curve, ax , int num\_ax, torsion , int \* jstat )

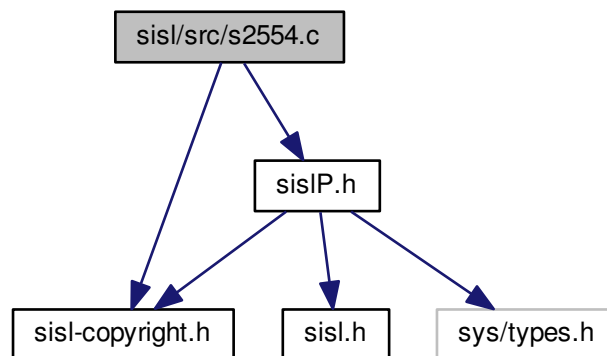
Definition at line 60 of file s2553.c.

## 30.2681 sisl/src/s2554.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s2554.c:



## Macros

- #define [S2554](#)

## Functions

- void [s2554](#) (SISLCurve \*curve, double parvalue, int \*leftknot, derive, double \*torsion, int \*jstat)

## 30.2681.1 Macro Definition Documentation

### 30.2681.1.1 #define S2554

Definition at line 48 of file s2554.c.

## 30.2681.2 Function Documentation

### 30.2681.2.1 void s2554 ( SISLCurve \* curve, double parvalue, int \* leftknot, derive , double \* torsion, int \* jstat )

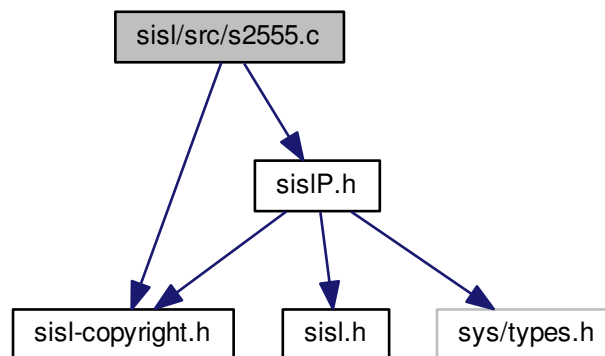
Definition at line 61 of file s2554.c.

## 30.2682 sisl/src/s2555.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s2555.c:



## Macros

- #define [S2555](#)

## Functions

- void [s2555](#) (derive, [double](#) \*torsion, int \*jstat)

## 30.2682.1 Macro Definition Documentation

### 30.2682.1.1 #define S2555

Definition at line 49 of file s2555.c.

## 30.2682.2 Function Documentation

### 30.2682.2.1 void s2555 ( derive , double \* torsion, int \* jstat )

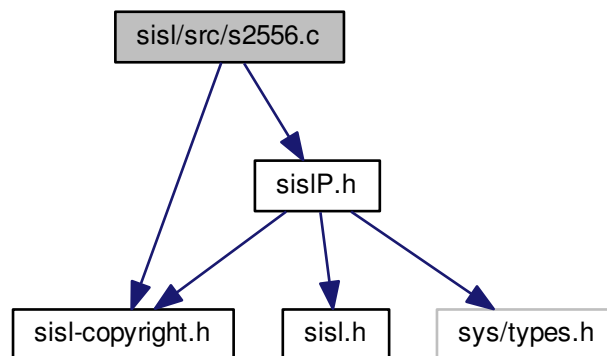
Definition at line 59 of file s2555.c.

## 30.2683 sisl/src/s2556.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s2556.c:



## Macros

- #define [S2556](#)

## Functions

- void [s2556](#) ([SISLCurve](#) \*curve, ax, int num\_ax, VoC, int \*jstat)

### 30.2683.1 Macro Definition Documentation

#### 30.2683.1.1 #define S2556

Definition at line 48 of file s2556.c.

### 30.2683.2 Function Documentation

#### 30.2683.2.1 void s2556 ( SISLCurve \* curve, ax , int num\_ax, VoC , int \* jstat )

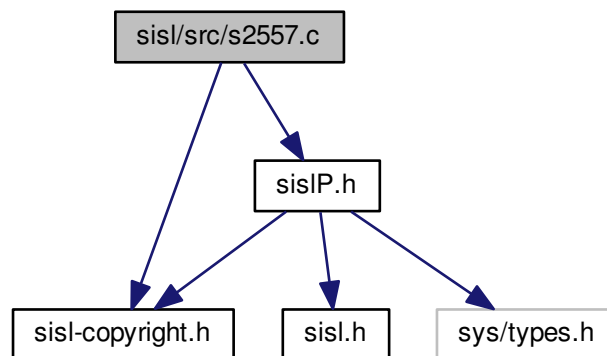
Definition at line 60 of file s2556.c.

## 30.2684 sisl/src/s2557.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s2557.c:



### Macros

- #define [S2557](#)

### Functions

- void [s2557](#) (SISLCurve \*curve, double parvalue, int \*leftknot, derive, double \*VoC, int \*jstat)

## 30.2684.1 Macro Definition Documentation

### 30.2684.1.1 #define S2557

Definition at line 48 of file s2557.c.

## 30.2684.2 Function Documentation

### 30.2684.2.1 void s2557 ( SISLCurve \* curve, double parvalue, int \* leftknot, derive , double \* VoC, int \* jstat )

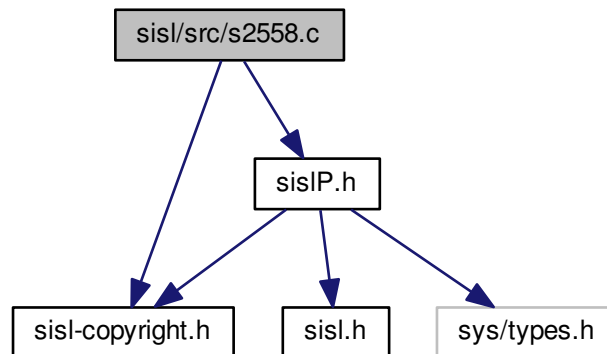
Definition at line 61 of file s2557.c.

## 30.2685 sisl/src/s2558.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s2558.c:



## Macros

- #define [S2558](#)

## Functions

- void [s2558](#) (derive, int idim, [double](#) \*VoC, int \*jstat)

## 30.2685.1 Macro Definition Documentation

### 30.2685.1.1 #define S2558

Definition at line 49 of file s2558.c.

## 30.2685.2 Function Documentation

### 30.2685.2.1 void s2558 ( derive , int *idim*, double \* *VoC*, int \* *jstat* )

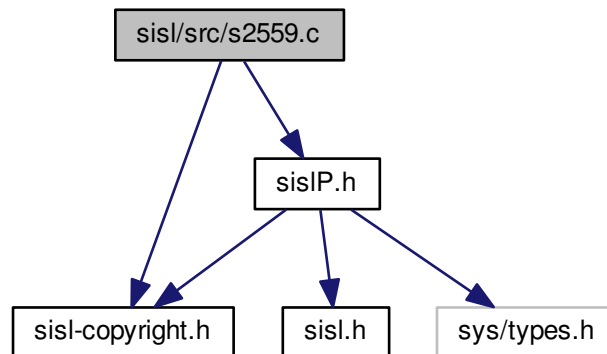
Definition at line 60 of file s2558.c.

## 30.2686 sisl/src/s2559.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s2559.c:



## Macros

- #define [S2559](#)

## Functions

- void [s2559](#) ([SISLCurve](#) \**curve*, *ax*, int *num\_ax*, *p*, *t*, *n*, *b*, int \**jstat*)



## 30.2686.1 Macro Definition Documentation

### 30.2686.1.1 #define S2559

Definition at line 48 of file s2559.c.

## 30.2686.2 Function Documentation

### 30.2686.2.1 void s2559 ( SISLCurve \* curve, ax , int num\_ax, p , t , n , b , int \* jstat )

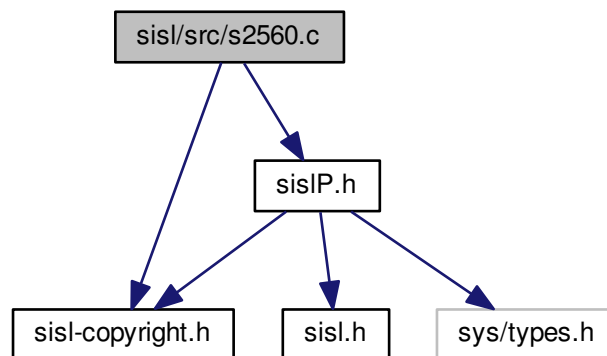
Definition at line 63 of file s2559.c.

## 30.2687 sisl/src/s2560.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s2560.c:



## Macros

- #define [S2560](#)

## Functions

- void [s2560](#) (SISLCurve \*curve, double parvalue, int \*leftknot, derive, p, t, n, b, int \*jstat)

## 30.2687.1 Macro Definition Documentation

### 30.2687.1.1 #define S2560

Definition at line 48 of file s2560.c.

## 30.2687.2 Function Documentation

### 30.2687.2.1 void s2560 ( SISLCurve \* curve, double parvalue, int \* leftknot, derive , p , t , n , b , int \* jstat )

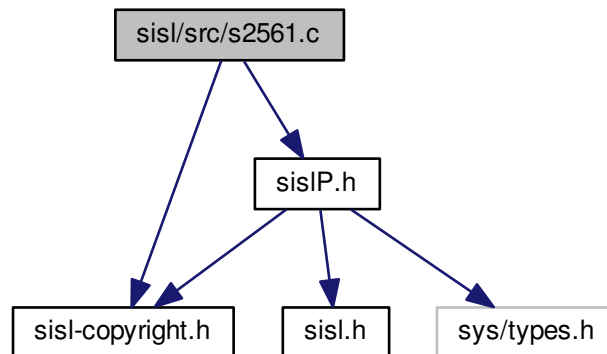
Definition at line 64 of file s2560.c.

## 30.2688 sisl/src/s2561.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s2561.c:



## Macros

- #define [S2561](#)

## Functions

- void [s2561](#) (derive, int idim, p, t, n, b, int \*jstat)

## 30.2688.1 Macro Definition Documentation

### 30.2688.1.1 #define S2561

Definition at line 49 of file s2561.c.

## 30.2688.2 Function Documentation

### 30.2688.2.1 void s2561 ( derive , int *idim*, p , t , n , b , int \* *jstat* )

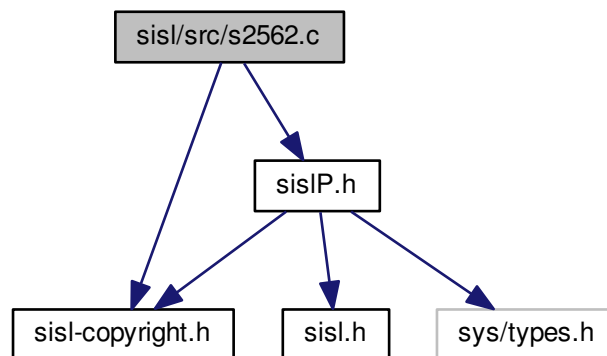
Definition at line 63 of file s2561.c.

## 30.2689 sisl/src/s2562.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s2562.c:



## Macros

- #define [S2562](#)

## Functions

- void [s2562](#) ([SISLCurve](#) \**curve*, ax, int num\_ax, int val\_flag, p, t, n, b, val, int \*jstat)

## 30.2689.1 Macro Definition Documentation

### 30.2689.1.1 #define S2562

Definition at line 48 of file s2562.c.

## 30.2689.2 Function Documentation

### 30.2689.2.1 void s2562 ( SISLCurve \* curve, ax , int num\_ax, int val\_flag, p , t , n , b , val , int \* jstat )

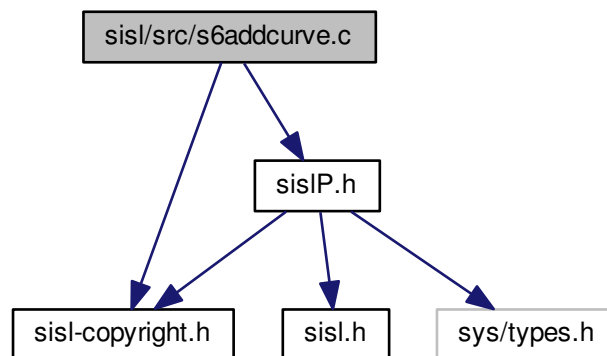
Definition at line 65 of file s2562.c.

## 30.2690 sisl/src/s6addcurve.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s6addcurve.c:



## Macros

- #define [S6ADDCURVE](#)

## Functions

- void [s6addcurve](#) (SISLCurve \*pc1, SISLCurve \*pc2, int isign, SISLCurve \*\*rcurve, int \*jstat)

## 30.2690.1 Macro Definition Documentation

### 30.2690.1.1 #define S6ADDCURVE

Definition at line 49 of file s6addcurve.c.

## 30.2690.2 Function Documentation

### 30.2690.2.1 void s6addcurve ( SISLCurve \* pc1, SISLCurve \* pc2, int isign, SISLCurve \*\* rcurve, int \* jstat )

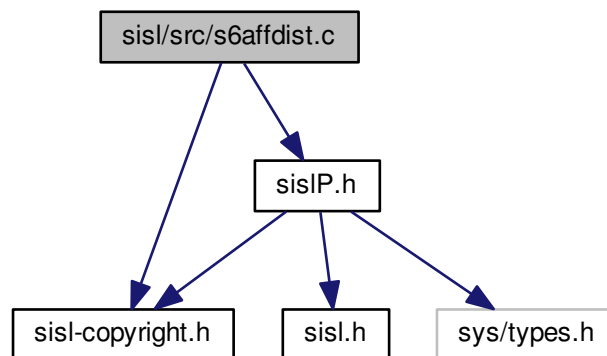
Definition at line 64 of file s6addcurve.c.

## 30.2691 sisl/src/s6affdist.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s6affdist.c:



## Macros

- #define [S6AFFDIST](#)

## Functions

- [double s6affdist](#) (e1, e2, emat, int idim)

## 30.2691.1 Macro Definition Documentation

### 30.2691.1.1 #define S6AFFDIST

Definition at line 49 of file s6affdist.c.

## 30.2691.2 Function Documentation

### 30.2691.2.1 double s6affdist ( e1 , e2 , emat , int idim )

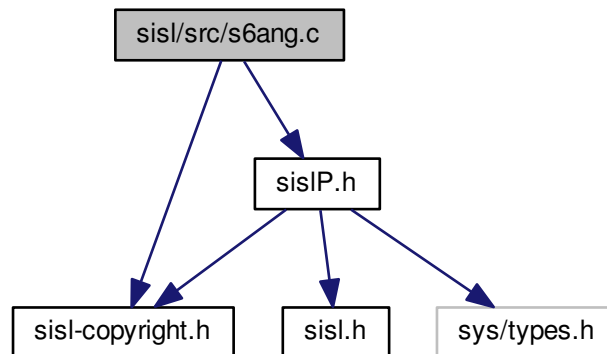
Definition at line 57 of file s6affdist.c.

## 30.2692 sisl/src/s6ang.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s6ang.c:



## Macros

- #define `S6ANG`

## Functions

- `double s6ang` (evec1, evec2, int idim)

## 30.2692.1 Macro Definition Documentation

### 30.2692.1.1 #define S6ANG

Definition at line 49 of file s6ang.c.

## 30.2692.2 Function Documentation

### 30.2692.2.1 double s6ang ( evec1 , evec2 , int idim )

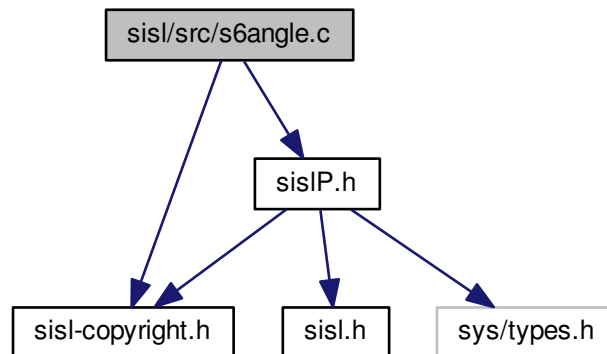
Definition at line 57 of file s6ang.c.

## 30.2693 sisl/src/s6angle.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s6angle.c:



## Macros

- #define [S6ANGLE](#)

## Functions

- [double s6angle](#) (evec1, evec2, enorm, int idim, int \*jstat)

## 30.2693.1 Macro Definition Documentation

### 30.2693.1.1 #define S6ANGLE

Definition at line 49 of file s6angle.c.

## 30.2693.2 Function Documentation

### 30.2693.2.1 double s6angle ( evec1 , evec2 , enorm , int *idim*, int \* *jstat* )

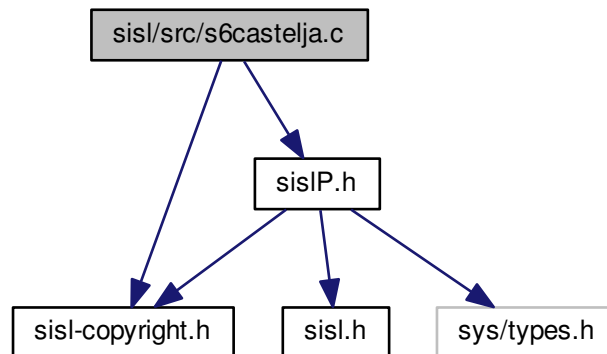
Definition at line 57 of file s6angle.c.

## 30.2694 sisl/src/s6castelja.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s6castelja.c:



## Macros

- #define [S6DECASTELJAU](#)

## Functions

- void [s6deCasteljau](#) (C, double a, double b, double t, int k, D, int \*jstat)



## 30.2694.1 Macro Definition Documentation

### 30.2694.1.1 #define S6DECASTELJAU

Definition at line 49 of file s6castelja.c.

## 30.2694.2 Function Documentation

### 30.2694.2.1 void s6deCasteljau ( C , double a , double b , double t , int k , D , int\* jstat )

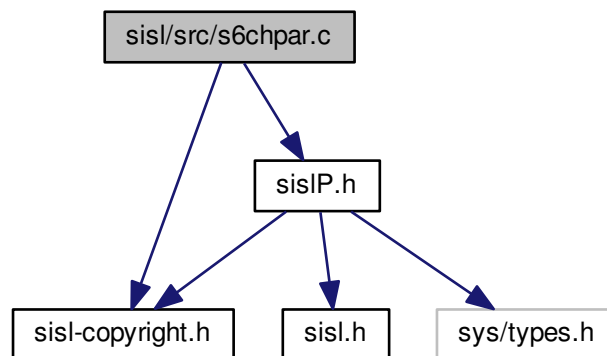
Definition at line 58 of file s6castelja.c.

## 30.2695 sisl/src/s6chpar.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s6chpar.c:



## Macros

- #define [S6CHPAR](#)

## Functions

- void [s6chpar](#) (ecoef1, int in1, int in2, int idim, ecoef2)

## 30.2695.1 Macro Definition Documentation

### 30.2695.1.1 #define S6CHPAR

Definition at line 49 of file s6chpar.c.

## 30.2695.2 Function Documentation

### 30.2695.2.1 void s6chpar ( ecoef1 , int in1, int in2, int idim, ecoef2 )

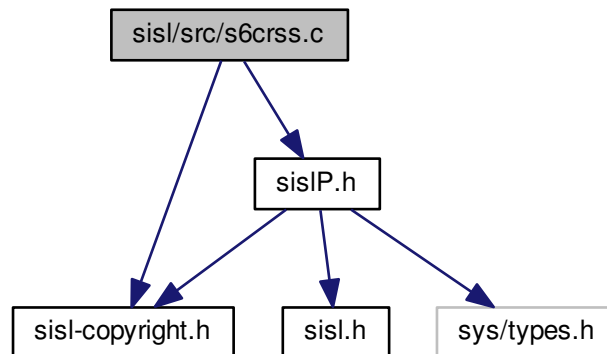
Definition at line 57 of file s6chpar.c.

## 30.2696 sisl/src/s6crss.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s6crss.c:



## Macros

- #define [S6CRSS](#)

## Functions

- void [s6crss](#) (e1, e2, e3)

## 30.2696.1 Macro Definition Documentation

### 30.2696.1.1 #define S6CRSS

Definition at line 49 of file s6crss.c.

## 30.2696.2 Function Documentation

### 30.2696.2.1 void s6crss ( e1 , e2 , e3 )

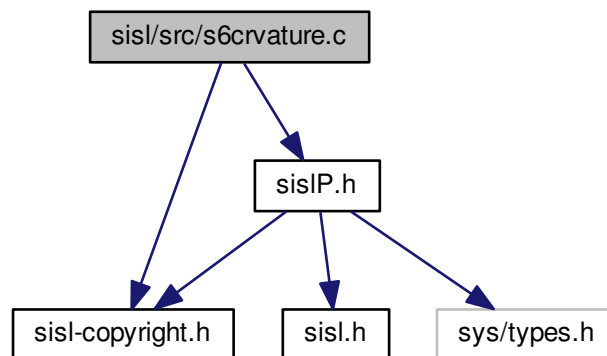
Definition at line 57 of file s6crss.c.

## 30.2697 sisl/src/s6crvature.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s6crvature.c:



## Macros

- #define [S6CURVATURE](#)

## Functions

- void [s6curvature](#) (eder, int idim, ecurv, int \*jstat)

## 30.2697.1 Macro Definition Documentation

### 30.2697.1.1 #define S6CURVATURE

Definition at line 49 of file s6crvature.c.

## 30.2697.2 Function Documentation

### 30.2697.2.1 void s6curvature ( eder , int *idim*, ecurv , int\* *jstat* )

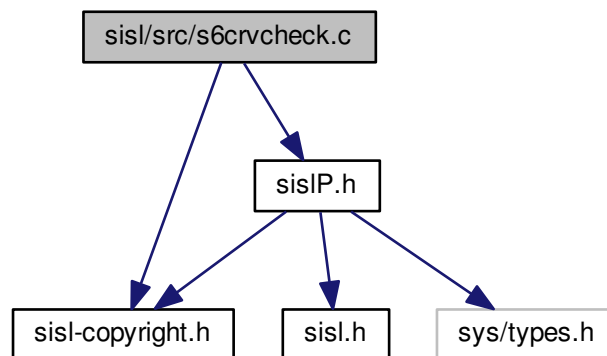
Definition at line 57 of file s6crvature.c.

## 30.2698 sisl/src/s6crvcheck.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s6crvcheck.c:



### Macros

- #define [S6CRVCHECK](#)

### Functions

- void [s6crvcheck](#) (SISLCurve \*pc, int \*jstat)

## 30.2698.1 Macro Definition Documentation

### 30.2698.1.1 #define S6CRVCHECK

Definition at line 49 of file s6crvcheck.c.

## 30.2698.2 Function Documentation

### 30.2698.2.1 void s6crvcheck ( SISLCurve \* *pc*, int \* *jstat* )

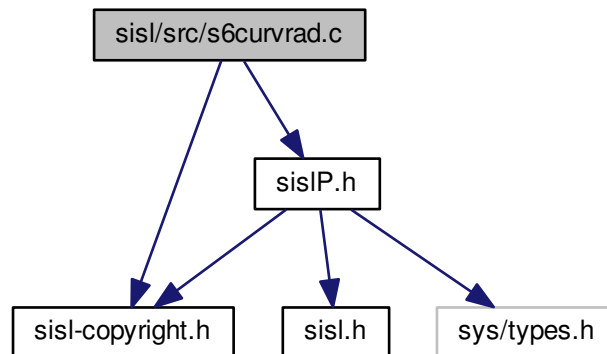
Definition at line 57 of file s6crvcheck.c.

## 30.2699 sisl/src/s6curvad.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s6curvad.c:



## Macros

- #define [S6CURVRAD](#)

## Functions

- void [s6curvad](#) (epnt1, epnt2, etang, int idim,\*crad, int \*jstat)

## 30.2699.1 Macro Definition Documentation

### 30.2699.1.1 #define S6CURVRAD

Definition at line 49 of file s6curvrad.c.

## 30.2699.2 Function Documentation

### 30.2699.2.1 void s6curvrad ( epnt1 , epnt2 , etang , int *idim* , \* *crad* , int\* *jstat* )

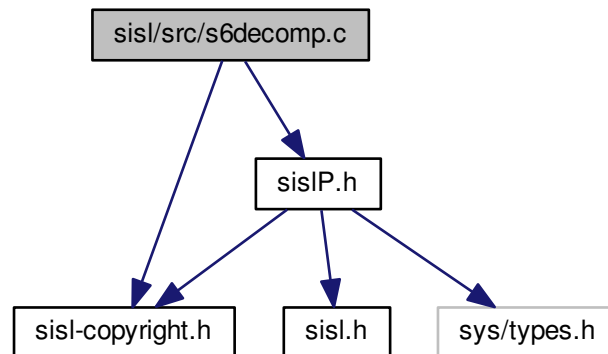
Definition at line 58 of file s6curvrad.c.

## 30.2700 sisl/src/s6decomp.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s6decomp.c:



## Macros

- #define [S6DECOMP](#)

## Functions

- void [s6decomp](#) (ea, gx, eb1, eb2, eb3, int \*jstat)

## 30.2700.1 Macro Definition Documentation

### 30.2700.1.1 #define S6DECOMP

Definition at line 49 of file s6decomp.c.

## 30.2700.2 Function Documentation

### 30.2700.2.1 void s6decomp ( ea , gx , eb1 , eb2 , eb3 , int \* jstat )

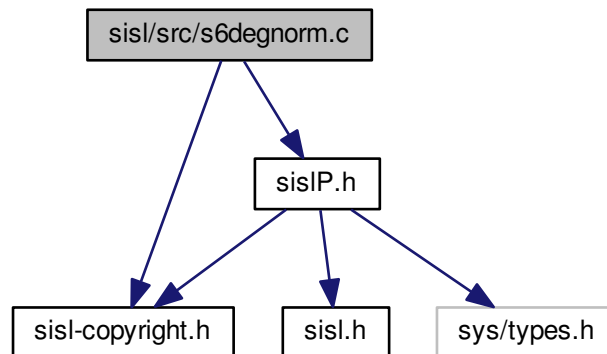
Definition at line 57 of file s6decomp.c.

## 30.2701 sisl/src/s6degnorm.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s6degnorm.c:



## Macros

- #define [S6DEGNORM](#)

## Functions

- void [s6degnorm](#) (SISLSurf \*ps1, int ider, epar, eder, utang, vtang, enorm, int \*jstat)

## 30.2701.1 Macro Definition Documentation

### 30.2701.1.1 #define S6DEGNORM

Definition at line 49 of file s6degnorm.c.

## 30.2701.2 Function Documentation

### 30.2701.2.1 void s6degnorm ( SISLSurf \* *ps1*, int *ider*, epar , eder , utang , vtang , enorm , int \* *jstat* )

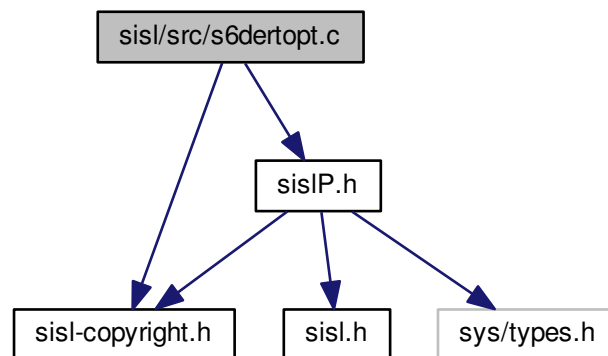
Definition at line 58 of file s6degnorm.c.

## 30.2702 sisl/src/s6dertopt.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s6dertopt.c:



## Macros

- #define [S6DERTOPT](#)

## Functions

- void [s6dertopt](#) (eder, ntype, int inpt, int idim, epoint, int \*jstat)



## 30.2702.1 Macro Definition Documentation

### 30.2702.1.1 #define S6DERTOPT

Definition at line 49 of file s6dertopt.c.

## 30.2702.2 Function Documentation

### 30.2702.2.1 void s6dertopt ( eder , ntype , int *inpt*, int *idim*, epoint , int \* *jstat* )

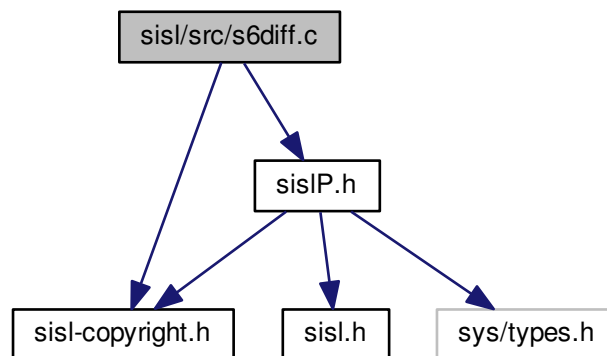
Definition at line 58 of file s6dertopt.c.

## 30.2703 sisl/src/s6diff.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s6diff.c:



## Macros

- #define [S6DIFF](#)

## Functions

- void [s6diff](#) (e1, e2, int idim, e3)

### 30.2703.1 Macro Definition Documentation

#### 30.2703.1.1 #define S6DIFF

Definition at line 49 of file s6diff.c.

### 30.2703.2 Function Documentation

#### 30.2703.2.1 void s6diff ( e1 , e2 , int idim, e3 )

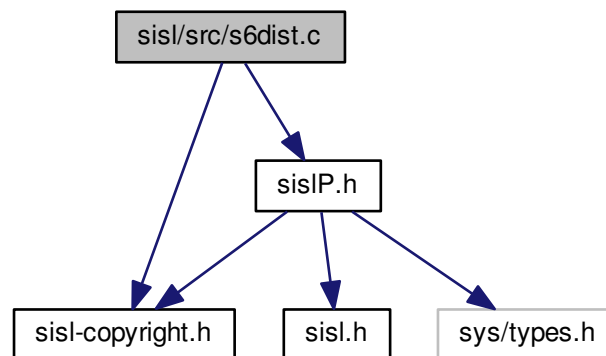
Definition at line 57 of file s6diff.c.

## 30.2704 sisl/src/s6dist.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s6dist.c:



### Macros

- #define [S6DIST](#)

### Functions

- [double s6dist](#) (epoint1, epoint2, int idim)

## 30.2704.1 Macro Definition Documentation

### 30.2704.1.1 #define S6DIST

Definition at line 49 of file s6dist.c.

## 30.2704.2 Function Documentation

### 30.2704.2.1 double s6dist ( epoint1 , epoint2 , int idim )

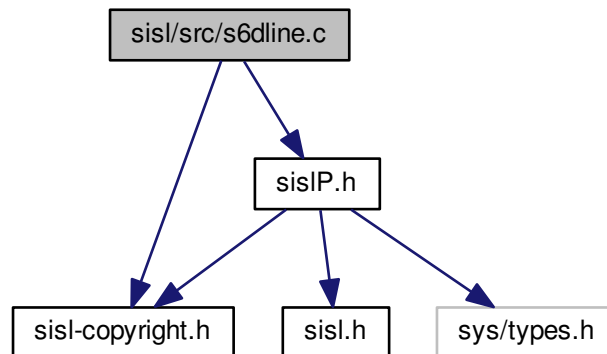
Definition at line 57 of file s6dist.c.

## 30.2705 sisl/src/s6dline.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s6dline.c:



## Macros

- #define [S6DLINE](#)

## Functions

- [double s6dline](#) (estart, eend, epoint, int idim, int \*jstat)

## 30.2705.1 Macro Definition Documentation

### 30.2705.1.1 #define S6DLINE

Definition at line 49 of file s6dline.c.

## 30.2705.2 Function Documentation

### 30.2705.2.1 double s6dline ( estart , eend , epoint , int idim, int \* jstat )

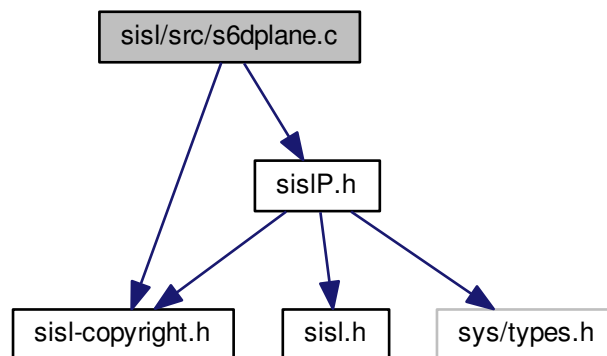
Definition at line 58 of file s6dline.c.

## 30.2706 sisl/src/s6dplane.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s6dplane.c:



## Macros

- #define [S6DPLANE](#)

## Functions

- [double s6dplane](#) (eq1, eq2, eq3, epoint, int idim, int \*jstat)

### 30.2706.1 Macro Definition Documentation

#### 30.2706.1.1 #define S6DPLANE

Definition at line 49 of file s6dplane.c.

### 30.2706.2 Function Documentation

#### 30.2706.2.1 double s6dplane ( eq1 , eq2 , eq3 , epoint , int idim , int \* jstat )

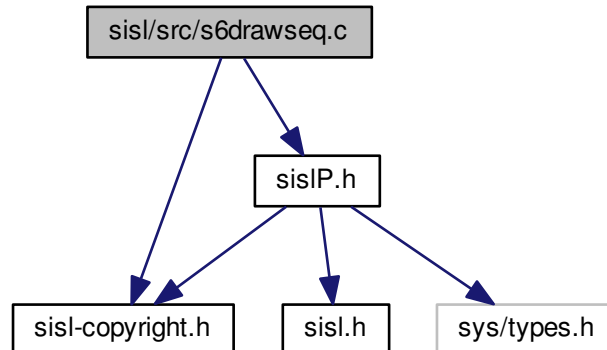
Definition at line 58 of file s6dplane.c.

## 30.2707 sisl/src/s6drawseq.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s6drawseq.c:



### Macros

- #define [S6DRAWSEQ](#)

### Functions

- void [s6move](#) ()
- void [s6line](#) ()
- void [s6drawseq](#) (epoint, int ipoint)

## 30.2707.1 Macro Definition Documentation

### 30.2707.1.1 #define S6DRAWSEQ

Definition at line 49 of file s6drawseq.c.

## 30.2707.2 Function Documentation

### 30.2707.2.1 void s6drawseq ( epoint , int ipoint )

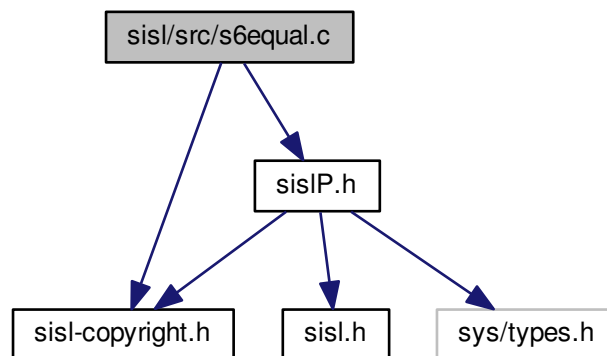
Definition at line 65 of file s6drawseq.c.

### 30.2707.2.2 void s6line ( )

### 30.2707.2.3 void s6move ( )

## 30.2708 sisl/src/s6equal.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for s6equal.c:
```



### Macros

- #define [S6EQUAL](#)

### Functions

- int [s6equal](#) (double a1, double a2, double aref)

## 30.2708.1 Macro Definition Documentation

### 30.2708.1.1 #define S6EQUAL

Definition at line 49 of file s6equal.c.

## 30.2708.2 Function Documentation

### 30.2708.2.1 int s6equal ( double a1, double a2, double aref )

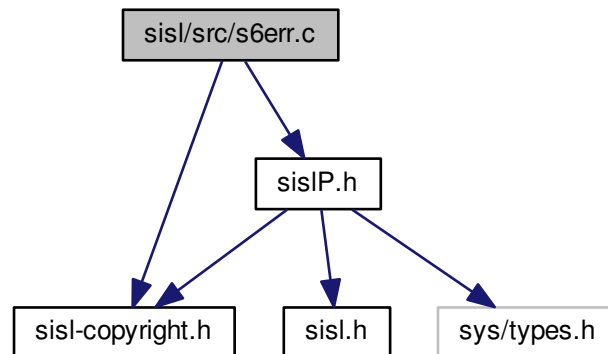
Definition at line 57 of file s6equal.c.

## 30.2709 sisl/src/s6err.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s6err.c:



## Macros

- #define [S6ERR](#)

## Functions

- void [s6err](#) (char \*rut, int jstat, int ipos)

## 30.2709.1 Macro Definition Documentation

### 30.2709.1.1 #define S6ERR

Definition at line 49 of file s6err.c.

## 30.2709.2 Function Documentation

### 30.2709.2.1 void s6err ( char \* rut, int jstat, int ipos )

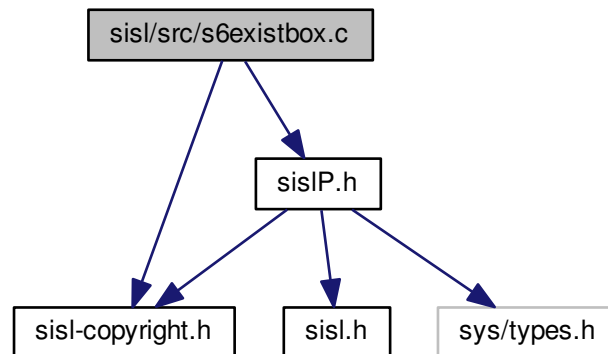
Definition at line 56 of file s6err.c.

## 30.2710 sisl/src/s6existbox.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s6existbox.c:



## Macros

- #define `S6EXISTBOX`

## Functions

- int `s6existbox` (`SISLbox` \*pbox, int itype, `double` aeapsge)



### 30.2710.1 Macro Definition Documentation

#### 30.2710.1.1 #define S6EXISTBOX

Definition at line 49 of file s6existbox.c.

### 30.2710.2 Function Documentation

#### 30.2710.2.1 int s6existbox ( SISLbox \* pbox, int itype, double aepsge )

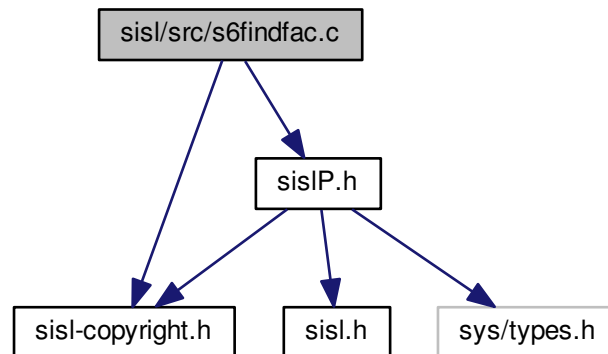
Definition at line 56 of file s6existbox.c.

## 30.2711 sisl/src/s6findfac.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s6findfac.c:



### Macros

- #define [S6FINDFAC](#)

### Functions

- void [s6findfac](#) (evecu, evectv, evectw, etang, int idim, int isign, double \*coef1, double \*coef2, double \*coef3, int \*jstat)

### 30.2711.1 Macro Definition Documentation

#### 30.2711.1.1 #define S6FINDFAC

Definition at line 49 of file s6findfac.c.

### 30.2711.2 Function Documentation

#### 30.2711.2.1 void s6findfac ( evecu , evecv , evecw , etang , int idim, int isign, double \* coef1, double \* coef2, double \* coef3, int \* jstat )

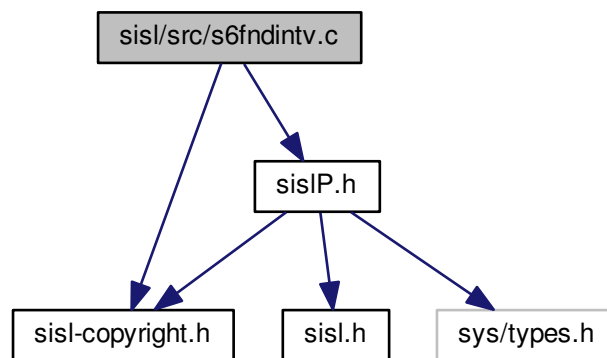
Definition at line 59 of file s6findfac.c.

## 30.2712 sisl/src/s6fndintv.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s6fndintv.c:



### Macros

- #define [SFNDINTVL](#)

### Functions

- void [s6fndintvl](#) (double \*et, int ik, int in, int \*ileft, double ax1, double ax2, int mu\_max, int \*jstat)

### 30.2712.1 Macro Definition Documentation

#### 30.2712.1.1 #define SFNDINTVL

Definition at line 42 of file s6findintv.c.

### 30.2712.2 Function Documentation

#### 30.2712.2.1 void s6findintvl ( double \* *et*, int *ik*, int *in*, int \* *ileft*, double *ax1*, double *ax2*, int *mu\_max*, int \* *jstat* )

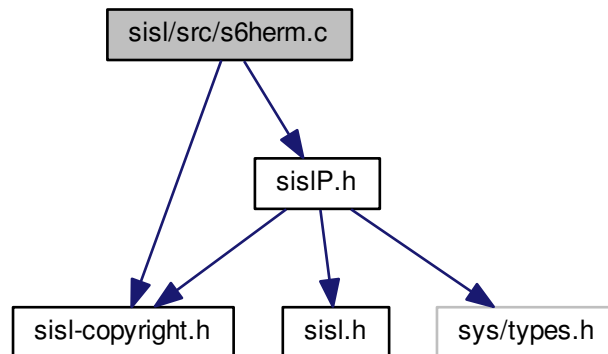
Definition at line 51 of file s6findintv.c.

## 30.2713 sisl/src/s6herm.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s6herm.c:



### Macros

- #define `S6HERM`

### Functions

- void `s6herm` (double \**pt*, double \**uknots*, double \**vknots*, int *unum*, int *vnum*, int *dim*, int *uindex*, int *vindex*, *herminfo*, int \**jstat*)

### 30.2713.1 Macro Definition Documentation

#### 30.2713.1.1 #define S6HERM

Definition at line 49 of file s6herm.c.

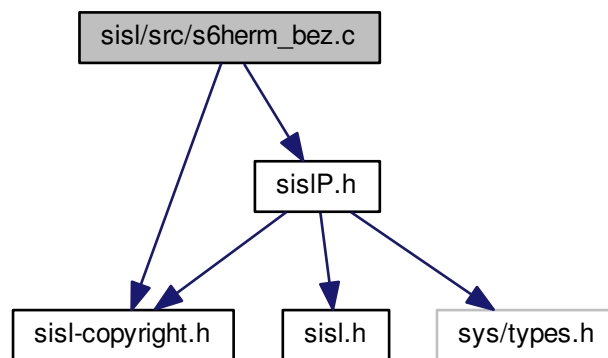
### 30.2713.2 Function Documentation

#### 30.2713.2.1 void s6herm ( double \* pt, double \* uknots, double \* vknots, int unum, int vnum, int dim, int uindex, int vindex, herminfo , int \* jstat )

Definition at line 59 of file s6herm.c.

## 30.2714 sisl/src/s6herm\_bez.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for s6herm_bez.c:
```



### Macros

- #define [S6HERMITE\\_BEZIER](#)

### Functions

- void [s6hermite\\_bezier](#) (SISLSurf \*s, a, b, int idim, c, int \*jstat)

## 30.2714.1 Macro Definition Documentation

### 30.2714.1.1 #define S6HERMITE\_BEZIER

Definition at line 49 of file s6herm\_bez.c.

## 30.2714.2 Function Documentation

### 30.2714.2.1 void s6hermite\_bezier ( SISLSurf \* s , a , b , int *idim* , c , int \* *jstat* )

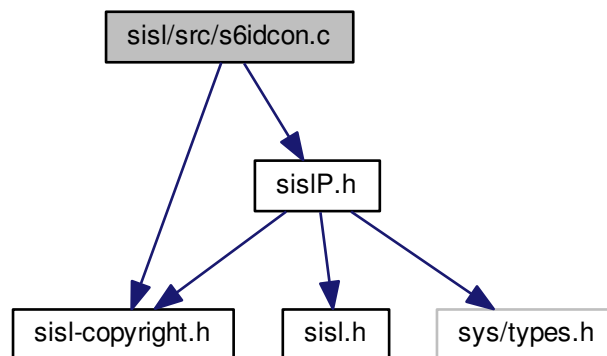
Definition at line 58 of file s6herm\_bez.c.

## 30.2715 sisl/src/s6idcon.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s6idcon.c:



## Macros

- #define [S6IDCON](#)

## Functions

- void [s6idcon](#) (SISLIntdat \*\*pintdat, SISLIntpt \*\*pintpt1, SISLIntpt \*\*pintpt2, int \*jstat)

## 30.2715.1 Macro Definition Documentation

### 30.2715.1.1 #define S6IDCON

Definition at line 49 of file s6idcon.c.

## 30.2715.2 Function Documentation

### 30.2715.2.1 void s6idcon ( SISLIntdat \*\* pintdat, SISLIntpt \*\* pintpt1, SISLIntpt \*\* pintpt2, int \* jstat )

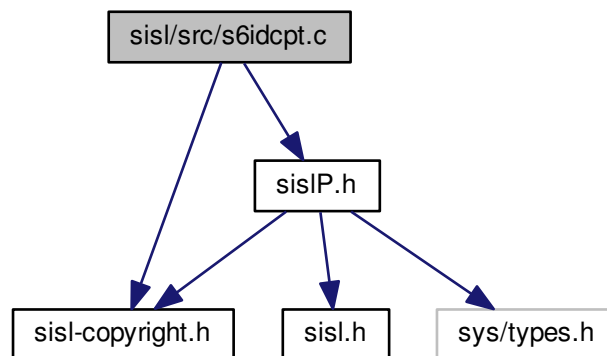
Definition at line 72 of file s6idcon.c.

## 30.2716 sisl/src/s6idcpt.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s6idcpt.c:



### Macros

- #define [S6IDCPT](#)

### Functions

- void [s6idcpt](#) ( [SISLIntdat](#) \*pintdat, [SISLIntpt](#) \*pintpt, [SISLIntpt](#) \*\*rintpt )

### 30.2716.1 Macro Definition Documentation

#### 30.2716.1.1 #define S6IDCPT

Definition at line 49 of file s6idcpt.c.

### 30.2716.2 Function Documentation

#### 30.2716.2.1 void s6idcpt ( SISLIntdat \* pintdat, SISLIntpt \* pintpt, SISLIntpt \*\* rintpt )

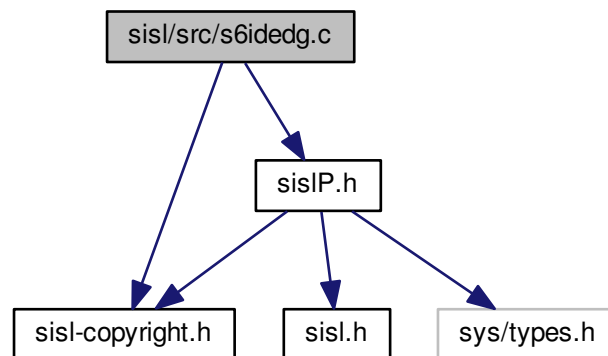
Definition at line 57 of file s6idcpt.c.

## 30.2717 sisl/src/s6idedg.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s6idedg.c:



### Macros

- #define [S6IDEDG](#)

### Functions

- void [s6idedg](#) (SISLObject \*po1, SISLObject \*po2, int iobj, int ipar, double apar, SISLIntdat \*pintdat, SISLIntpt \*\*rintpt, int \*jnum, int \*jstat)

### 30.2717.1 Macro Definition Documentation

#### 30.2717.1.1 #define S6IDEDG

Definition at line 49 of file s6idedg.c.

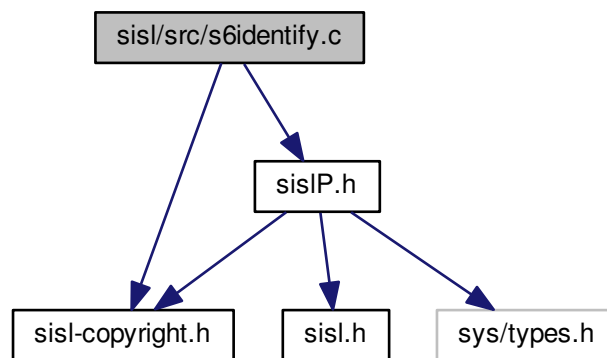
### 30.2717.2 Function Documentation

#### 30.2717.2.1 void s6idedg ( SISLObject \* po1, SISLObject \* po2, int iobj, int ipar, double apar, SISLIntdat \* pintdat, SISLPtedge \*\* rptedge, int \* jnum, int \* jstat )

Definition at line 58 of file s6idedg.c.

## 30.2718 sisl/src/s6identify.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for s6identify.c:
```



### Macros

- #define [S6IDENTIFY](#)

### Functions

- void [s6identify](#) (SISLSurf \*s, a, b, double level\_val, double eps1, double eps2, int \*jstat)



## 30.2718.1 Macro Definition Documentation

### 30.2718.1.1 #define S6IDENTIFY

Definition at line 49 of file s6identify.c.

## 30.2718.2 Function Documentation

### 30.2718.2.1 void s6identify ( SISLSurf\* s, a , b , double level\_val, double eps1, double eps2, int\* jstat )

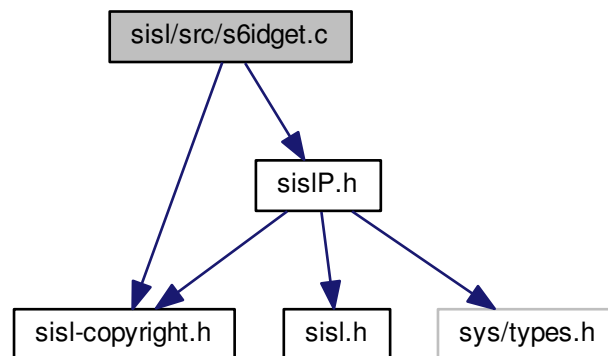
Definition at line 58 of file s6identify.c.

## 30.2719 sisl/src/s6idget.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s6idget.c:



## Macros

- #define [S6IDGET](#)

## Functions

- void [s6idget](#) (SISLObject \*po1, SISLObject \*po2, int ipar, double apar, SISLIntdat \*pintdat, SISLIntdat \*\*rintdat, int \*jstat)

### 30.2719.1 Macro Definition Documentation

#### 30.2719.1.1 #define S6IDGET

Definition at line 49 of file s6idget.c.

### 30.2719.2 Function Documentation

#### 30.2719.2.1 void s6idget ( SISLObject \* po1, SISLObject \* po2, int ipar, double apar, SISLIntdat \* pintdat, SISLIntdat \*\* rintdat, int \* jstat )

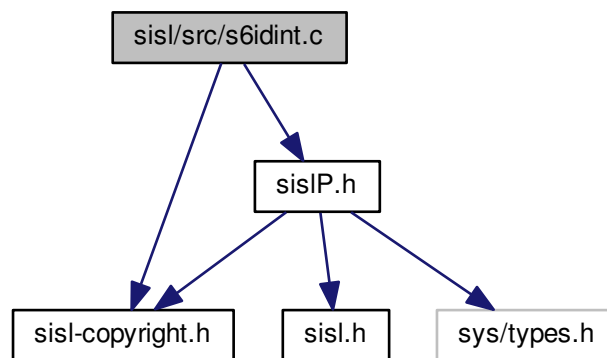
Definition at line 58 of file s6idget.c.

### 30.2720 sisl/src/s6idint.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s6idint.c:



#### Macros

- #define [S6IDINT](#)

#### Functions

- void [s6idint](#) (SISLObject \*po1, SISLObject \*po2, SISLIntdat \*pintdat, SISLIntpt \*\*rpt, int iob)

## 30.2720.1 Macro Definition Documentation

### 30.2720.1.1 #define S6IDINT

Definition at line 49 of file s6idint.c.

## 30.2720.2 Function Documentation

### 30.2720.2.1 void s6idint ( SISLObject \* po1, SISLObject \* po2, SISLIntdat \* pintdat, SISLIntpt \*\* rpt, int iob )

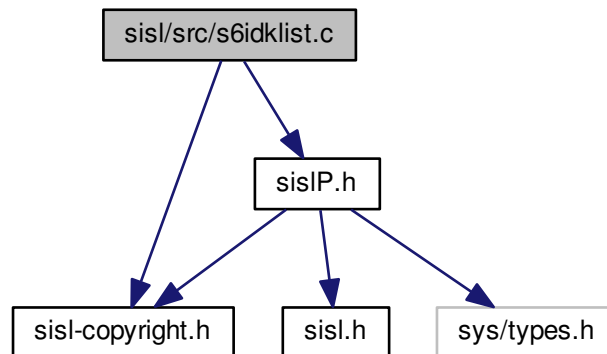
Definition at line 57 of file s6idint.c.

## 30.2721 sisl/src/s6idklist.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s6idklist.c:



## Macros

- #define `S6IDKLIST`

## Functions

- void `s6idklist` (SISLIntdat \*\*pintdat, SISLIntlist \*pintlist, int \*jstat)

### 30.2721.1 Macro Definition Documentation

#### 30.2721.1.1 #define S6IDKLIST

Definition at line 49 of file s6idklist.c.

### 30.2721.2 Function Documentation

#### 30.2721.2.1 void s6idklist ( SISLIntdat \*\* pintdat, SISLIntlist \* pintlist, int \* jstat )

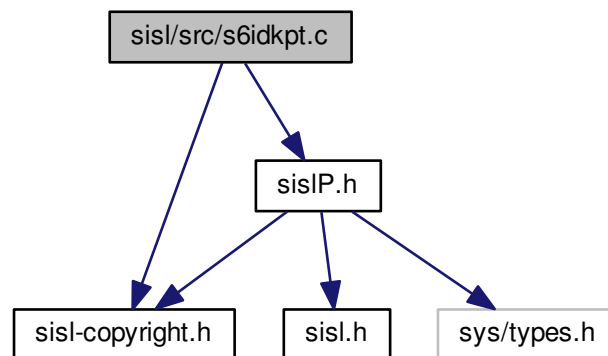
Definition at line 57 of file s6idklist.c.

## 30.2722 sisl/src/s6idkpt.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s6idkpt.c:



### Macros

- #define [S6IDKPT](#)

### Functions

- void [s6idkpt](#) ( [SISLIntdat](#) \*\*pintdat, [SISLIntpt](#) \*\*pintpt, [SISLIntpt](#) \*\*rtpt, [SISLIntpt](#) \*\*rfpt, int \*jstat)

## 30.2722.1 Macro Definition Documentation

### 30.2722.1.1 #define S6IDKPT

Definition at line 49 of file s6idkpt.c.

## 30.2722.2 Function Documentation

### 30.2722.2.1 void s6idkpt ( SISLIntdat \*\* pintdat, SISLIntpt \*\* pintpt, SISLIntpt \*\* rpt, SISLIntpt \*\* rfpt, int \* jstat )

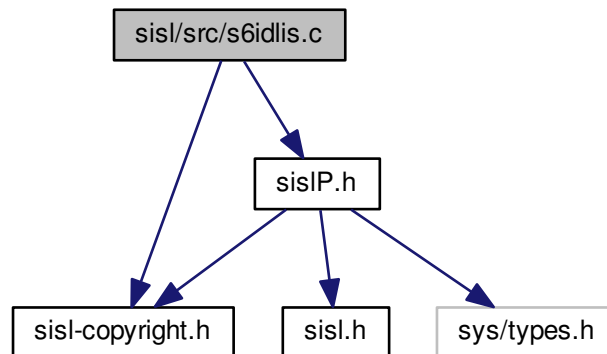
Definition at line 57 of file s6idkpt.c.

## 30.2723 sisl/src/s6idlis.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s6idlis.c:



## Macros

- #define [S6IDLIS](#)

## Functions

- void [s6idlis](#) (SISLObject \*po1, SISLObject \*po2, SISLIntdat \*\*pintdat, int \*jstat)

### 30.2723.1 Macro Definition Documentation

#### 30.2723.1.1 #define S6IDLIS

Definition at line 47 of file s6idlis.c.

### 30.2723.2 Function Documentation

#### 30.2723.2.1 void s6idlis ( SISLObject \* po1, SISLObject \* po2, SISLIntdat \*\* pintdat, int \* jstat )

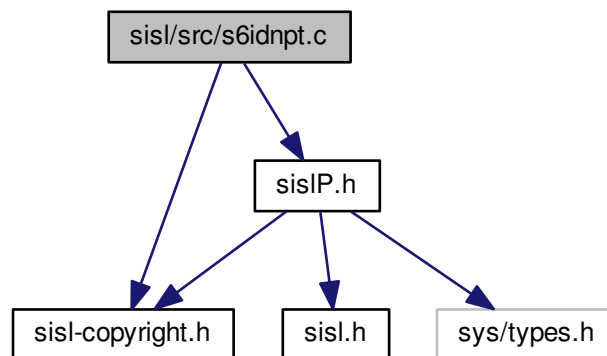
Definition at line 68 of file s6idlis.c.

## 30.2724 sisl/src/s6idnpt.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s6idnpt.c:



### Macros

- #define [S6IDNPT](#)

### Functions

- void [s6idnpt](#) (SISLIntdat \*\*pintdat, SISLIntpt \*\*pintpt, int itest, int \*jstat)

## 30.2724.1 Macro Definition Documentation

### 30.2724.1.1 #define S6IDNPT

Definition at line 49 of file s6idnpt.c.

## 30.2724.2 Function Documentation

### 30.2724.2.1 void s6idnpt ( SISLIntdat \*\* pintdat, SISLIntpt \*\* pintpt, int itest, int \* jstat )

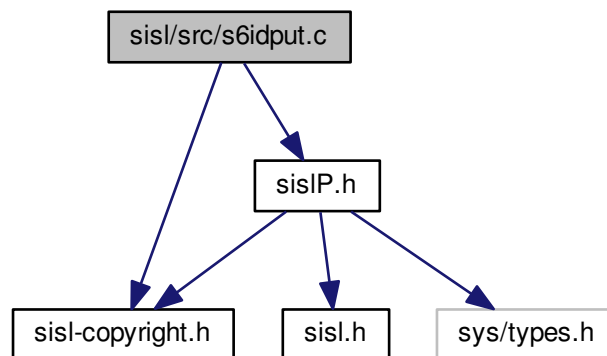
Definition at line 57 of file s6idnpt.c.

## 30.2725 sisl/src/s6idput.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s6idput.c:



## Macros

- #define [S6IDPUT](#)

## Functions

- void [s6idput](#) (SISLIntdat \*\*rintdat, SISLIntdat \*pintdat, int inr, double apar, int \*jstat)

## 30.2725.1 Macro Definition Documentation

### 30.2725.1.1 #define S6IDPUT

Definition at line 49 of file s6idput.c.

## 30.2725.2 Function Documentation

### 30.2725.2.1 void s6idput ( SISLIntdat \*\* *rintdat*, SISLIntdat \* *pintdat*, int *inr*, double *apar*, int \* *jstat* )

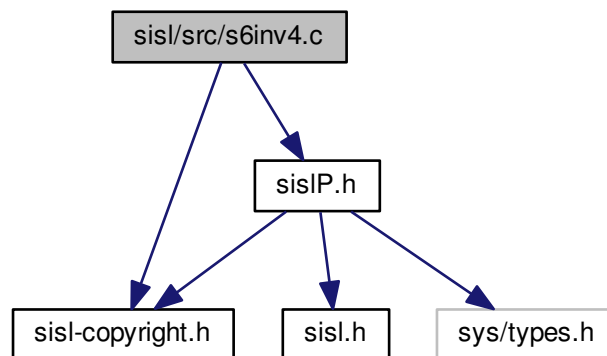
Definition at line 57 of file s6idput.c.

## 30.2726 sisl/src/s6inv4.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s6inv4.c:



### Macros

- #define [S6INV4](#)

### Functions

- void [s6inv4](#) (em, einv, int \*jstat)



## 30.2726.1 Macro Definition Documentation

### 30.2726.1.1 #define S6INV4

Definition at line 50 of file s6inv4.c.

## 30.2726.2 Function Documentation

### 30.2726.2.1 void s6inv4 ( em , einv , int \* jstat )

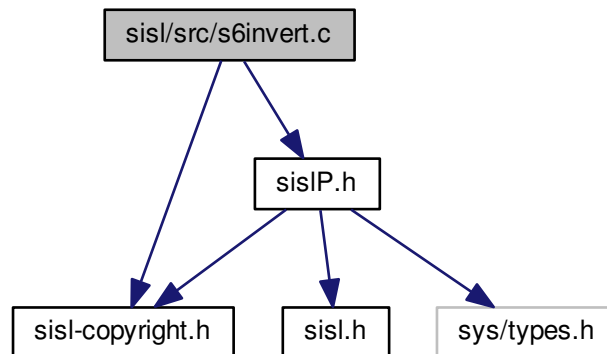
Definition at line 59 of file s6inv4.c.

## 30.2727 sisl/src/s6invert.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s6invert.c:



## Macros

- #define [S6INVERT](#)

## Functions

- void [s6invert](#) (emat, int im, einvertmat, int \*jstat)

## 30.2727.1 Macro Definition Documentation

### 30.2727.1.1 #define S6INVERT

Definition at line 49 of file s6invert.c.

## 30.2727.2 Function Documentation

### 30.2727.2.1 void s6invert ( emat , int im, einvertmat , int \* jstat )

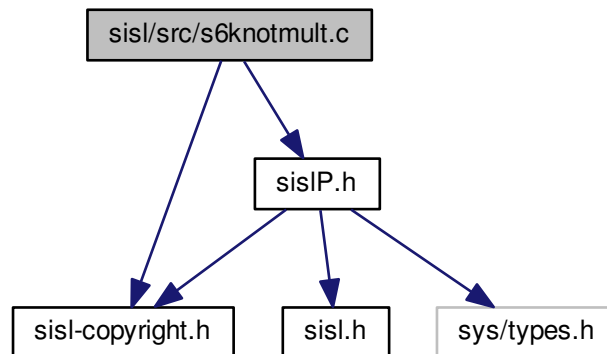
Definition at line 57 of file s6invert.c.

## 30.2728 sisl/src/s6knotmult.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s6knotmult.c:



## Macros

- #define `S6KNOTMULT`

## Functions

- int `s6knotmult` (et, int ik, int in, int \*ileft, double ax, int \*jstat)

## 30.2728.1 Macro Definition Documentation

### 30.2728.1.1 #define S6KNOTMULT

Definition at line 49 of file s6knotmult.c.

## 30.2728.2 Function Documentation

### 30.2728.2.1 int s6knotmult ( et , int ik, int in, int \* ileft, double ax, int \* jstat )

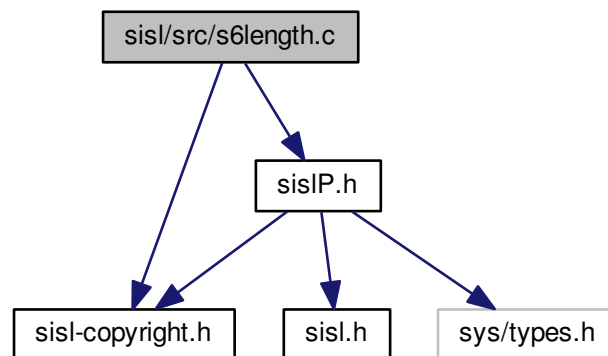
Definition at line 57 of file s6knotmult.c.

## 30.2729 sisl/src/s6length.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s6length.c:



## Macros

- #define `S6LENGTH`

## Functions

- `double s6length` (e1, int idim, int \*jstat)

## 30.2729.1 Macro Definition Documentation

### 30.2729.1.1 #define S6LENGTH

Definition at line 47 of file s6length.c.

## 30.2729.2 Function Documentation

### 30.2729.2.1 double s6length ( e1 , int *idim*, int \* *jstat* )

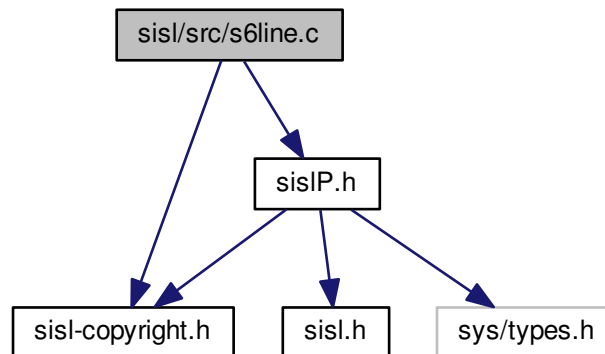
Definition at line 55 of file s6length.c.

## 30.2730 sisl/src/s6line.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s6line.c:



### Macros

- #define [S6LINE](#)

### Functions

- void [s6line](#) (epoint)

## 30.2730.1 Macro Definition Documentation

### 30.2730.1.1 #define S6LINE

Definition at line 49 of file s6line.c.

## 30.2730.2 Function Documentation

### 30.2730.2.1 void s6line ( epoint )

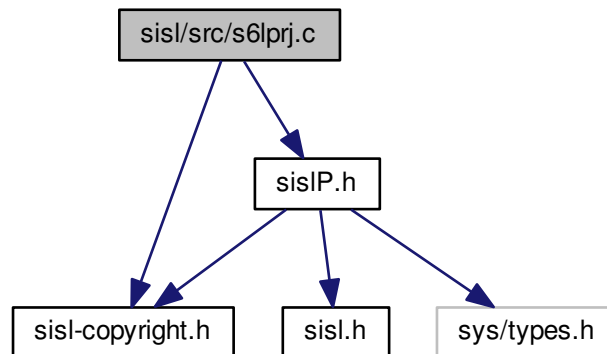
Definition at line 57 of file s6line.c.

## 30.2731 sisl/src/s6lprj.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s6lprj.c:



## Macros

- #define [S6LPRJ](#)

## Functions

- [double s6lprj](#) (e1, e2, int idim)

### 30.2731.1 Macro Definition Documentation

#### 30.2731.1.1 #define S6LPRJ

Definition at line 47 of file s6lprj.c.

### 30.2731.2 Function Documentation

#### 30.2731.2.1 double s6lprj ( e1 , e2 , int idim )

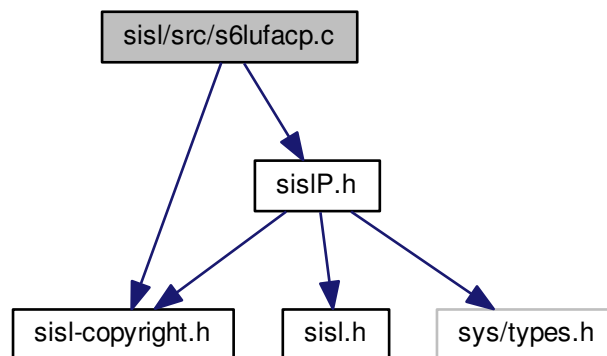
Definition at line 55 of file s6lprj.c.

## 30.2732 sisl/src/s6lufacp.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s6lufacp.c:



### Macros

- #define `S6LUFACP`

### Functions

- void `s6lufacp` (ea, nl, int im, int \*jstat)

### 30.2732.1 Macro Definition Documentation

#### 30.2732.1.1 #define S6LUFACP

Definition at line 49 of file s6lufacp.c.

### 30.2732.2 Function Documentation

#### 30.2732.2.1 void s6lufacp ( ea , nl , int im, int \* jstat )

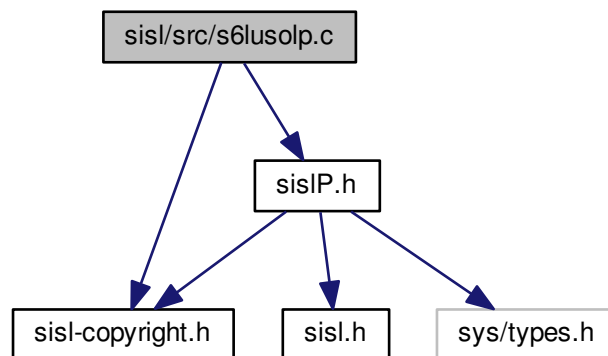
Definition at line 57 of file s6lufacp.c.

## 30.2733 sisl/src/s6lusolp.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s6lusolp.c:



### Macros

- #define [S6LUSOLP](#)

### Functions

- void [s6lusolp](#) (ea, eb, nl, int im, int \*jstat)

### 30.2733.1 Macro Definition Documentation

#### 30.2733.1.1 #define S6LUSOLP

Definition at line 49 of file s6lusolp.c.

### 30.2733.2 Function Documentation

#### 30.2733.2.1 void s6lusolp ( ea , eb , nl , int im, int \* jstat )

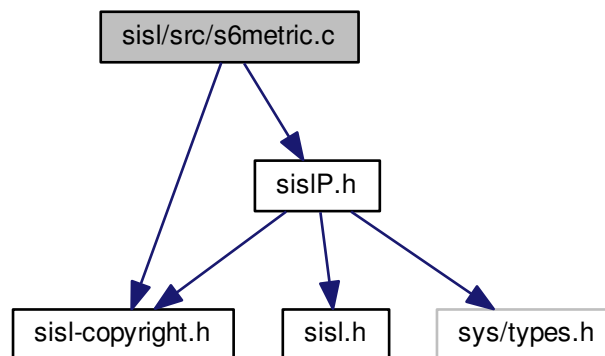
Definition at line 57 of file s6lusolp.c.

## 30.2734 sisl/src/s6metric.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s6metric.c:



### Macros

- #define [S6METRIC](#)

### Functions

- void [s6metric](#) (epoint, int in, int idim, emat, int \*jstat)



## 30.2734.1 Macro Definition Documentation

### 30.2734.1.1 #define S6METRIC

Definition at line 49 of file s6metric.c.

## 30.2734.2 Function Documentation

### 30.2734.2.1 void s6metric ( epoint , int in, int idim, emat , int \* jstat )

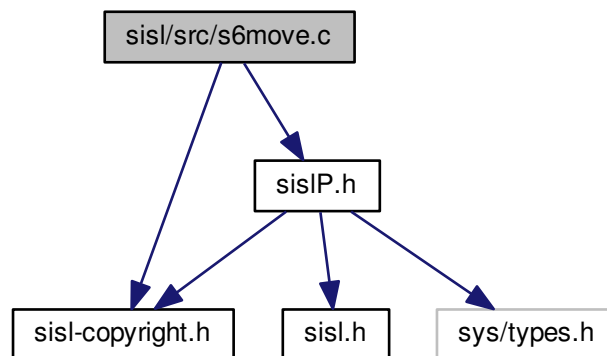
Definition at line 57 of file s6metric.c.

## 30.2735 sisl/src/s6move.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s6move.c:



## Macros

- #define [S6MOVE](#)

## Functions

- void [s6move](#) (epoint)

## 30.2735.1 Macro Definition Documentation

### 30.2735.1.1 #define S6MOVE

Definition at line 49 of file s6move.c.

## 30.2735.2 Function Documentation

### 30.2735.2.1 void s6move ( epoint )

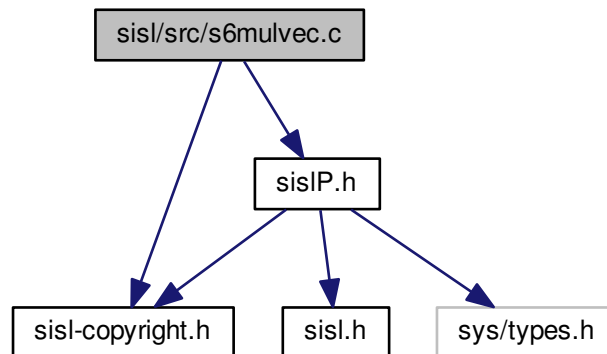
Definition at line 57 of file s6move.c.

## 30.2736 sisl/src/s6mulvec.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s6mulvec.c:



## Macros

- #define `S6MULVEC`

## Functions

- void `s6mulvec` (ematrix, evect, eright)

## 30.2736.1 Macro Definition Documentation

### 30.2736.1.1 #define S6MULVEC

Definition at line 49 of file s6mulvec.c.

## 30.2736.2 Function Documentation

### 30.2736.2.1 void s6mulvec ( ematrix , evec1 , evec2 )

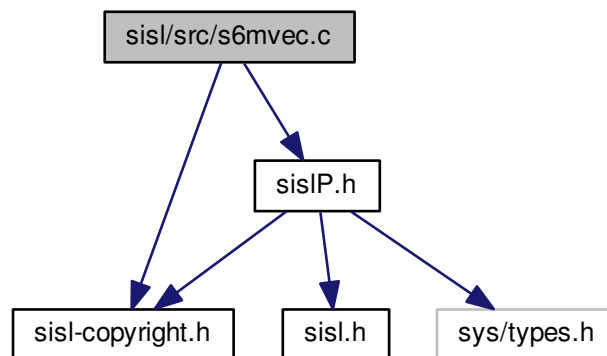
Definition at line 58 of file s6mulvec.c.

## 30.2737 sisl/src/s6mvec.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s6mvec.c:



## Macros

- #define [S6MVEC](#)

## Functions

- void [s6mulvec](#) (emat, evec1, int invec, evec2)

### 30.2737.1 Macro Definition Documentation

#### 30.2737.1.1 #define S6MVEC

Definition at line 49 of file s6mvec.c.

### 30.2737.2 Function Documentation

#### 30.2737.2.1 void s6mvec ( emat , evec1 , int inbvec, evec2 )

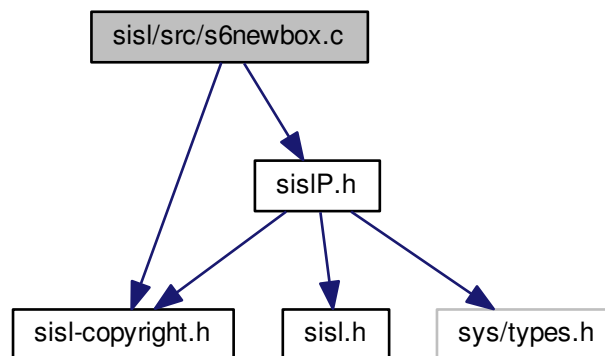
Definition at line 57 of file s6mvec.c.

## 30.2738 sisl/src/s6newbox.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s6newbox.c:



### Macros

- #define [S6NEWBOX](#)

### Functions

- void [s6newbox](#) (SISLbox \*pbox, int inum, int itype, [double aepsge](#), int \*jstat)

## 30.2738.1 Macro Definition Documentation

### 30.2738.1.1 #define S6NEWBOX

Definition at line 49 of file s6newbox.c.

## 30.2738.2 Function Documentation

### 30.2738.2.1 void s6newbox ( SISLbox \* pbox, int inum, int itype, double aepsge, int \* jstat )

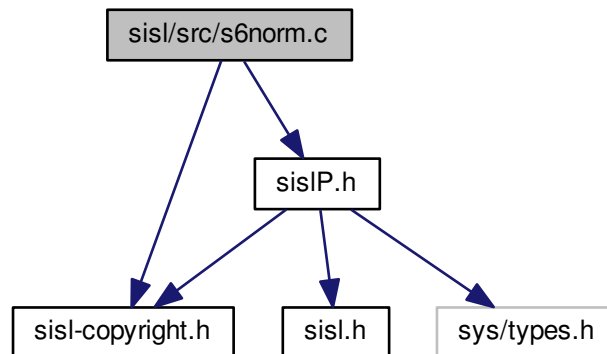
Definition at line 57 of file s6newbox.c.

## 30.2739 sisl/src/s6norm.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s6norm.c:



## Macros

- #define [S6NORM](#)

## Functions

- [double s6norm](#) (e1, int idim, e2, int \*jstat)

## 30.2739.1 Macro Definition Documentation

### 30.2739.1.1 #define S6NORM

Definition at line 47 of file s6norm.c.

## 30.2739.2 Function Documentation

### 30.2739.2.1 double s6norm ( e1 , int idim, e2 , int \* jstat )

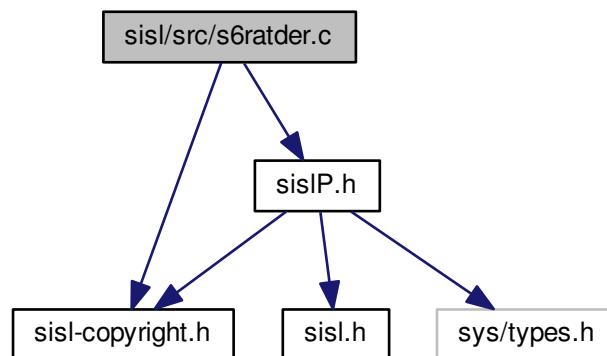
Definition at line 55 of file s6norm.c.

## 30.2740 sisl/src/s6ratder.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s6ratder.c:



## Macros

- #define [S6RATDER](#)

## Functions

- void [s6ratder](#) (eder, int idim, int ider, gder, int \*jstat)

## 30.2740.1 Macro Definition Documentation

### 30.2740.1.1 #define S6RATDER

Definition at line 49 of file s6ratder.c.

## 30.2740.2 Function Documentation

### 30.2740.2.1 void s6ratder ( eder , int idim, int nder, gder , int \* jstat )

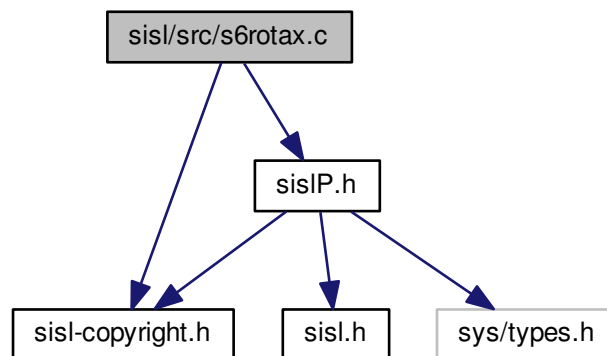
Definition at line 57 of file s6ratder.c.

## 30.2741 sisl/src/s6rotax.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s6rotax.c:



## Macros

- #define [S6ROTAX](#)

## Functions

- void [s6rotax](#) (ep, eaxis, expnt, emat, int \*jstat)

## 30.2741.1 Macro Definition Documentation

### 30.2741.1.1 #define S6ROTAX

Definition at line 49 of file s6rotax.c.

## 30.2741.2 Function Documentation

### 30.2741.2.1 void s6rotax ( ep , eaxis , expnt , emat , int \* jstat )

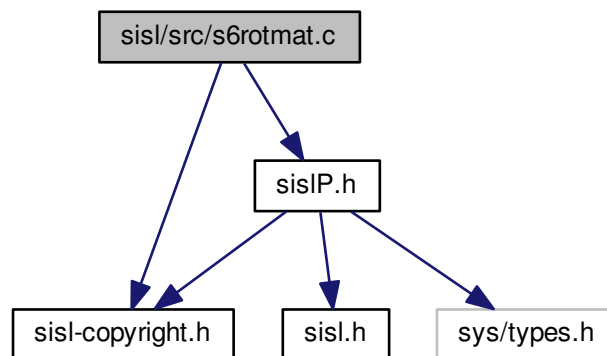
Definition at line 57 of file s6rotax.c.

## 30.2742 sisl/src/s6rotmat.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s6rotmat.c:



## Macros

- #define [S6ROTMAT](#)

## Functions

- void [s6rotmat](#) (eorigo, eaxis, enorm, ematrix, int \*jstat)



## 30.2742.1 Macro Definition Documentation

### 30.2742.1.1 #define S6ROTMAT

Definition at line 49 of file s6rotmat.c.

## 30.2742.2 Function Documentation

### 30.2742.2.1 void s6rotmat ( eorigo , exaxis , enorm , ematrix , int \* jstat )

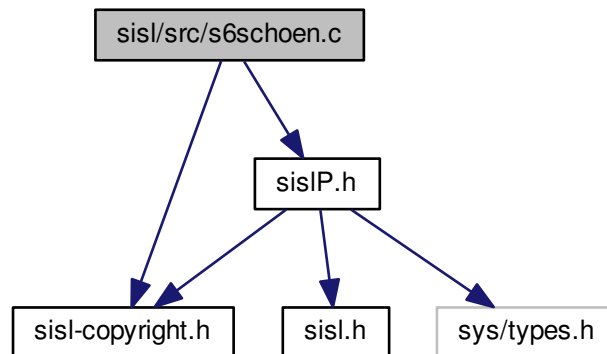
Definition at line 59 of file s6rotmat.c.

## 30.2743 sisl/src/s6schoen.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s6schoen.c:



## Macros

- #define [S6SCHOEN](#)

## Functions

- [double s6schoen](#) (et, int ik, int index)

### 30.2743.1 Macro Definition Documentation

#### 30.2743.1.1 #define S6SCHOEN

Definition at line 50 of file s6schoen.c.

### 30.2743.2 Function Documentation

#### 30.2743.2.1 double s6schoen ( et , int ik, int index )

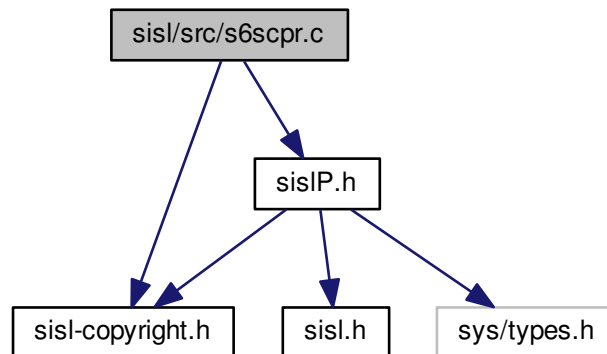
Definition at line 57 of file s6schoen.c.

## 30.2744 sisl/src/s6scpr.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s6scpr.c:



### Macros

- #define [S6SCPR](#)

### Functions

- [double s6scpr](#) (e1, e2, int idim)

## 30.2744.1 Macro Definition Documentation

### 30.2744.1.1 #define S6SCPR

Definition at line 49 of file s6scpr.c.

## 30.2744.2 Function Documentation

### 30.2744.2.1 double s6scpr ( e1 , e2 , int *idim* )

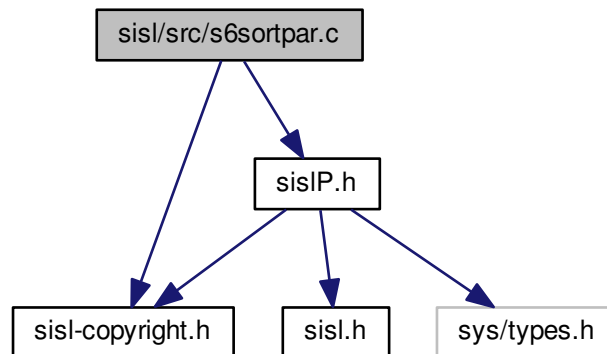
Definition at line 57 of file s6scpr.c.

## 30.2745 sisl/src/s6sortpar.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s6sortpar.c:



## Macros

- #define [S6SORTPAR](#)

## Functions

- void [s6sortpar](#) (evec1, epar1, int ipar, int idim, evec2, epar2, int \*jstat)

## 30.2745.1 Macro Definition Documentation

### 30.2745.1.1 #define S6SORTPAR

Definition at line 49 of file s6sortpar.c.

## 30.2745.2 Function Documentation

### 30.2745.2.1 void s6sortpar ( evec1 , epar1 , int ipar, int idim, evec2 , epar2 , int \* jstat )

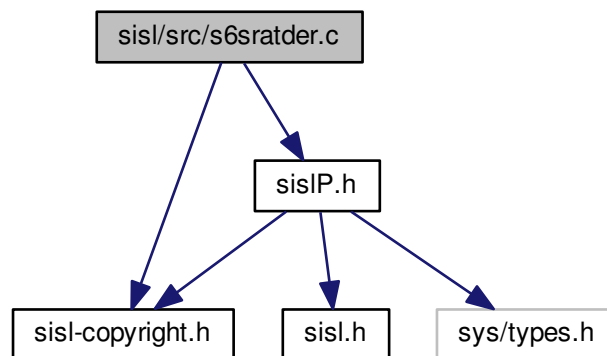
Definition at line 58 of file s6sortpar.c.

## 30.2746 sisl/src/s6sratder.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s6sratder.c:



## Macros

- #define [S6SRATDER](#)

## Functions

- void [s6sratder](#) (eder, int idim, int ider1, int ider2, gder, int \*jstat)

## 30.2746.1 Macro Definition Documentation

### 30.2746.1.1 #define S6SRATDER

Definition at line 49 of file s6sratder.c.

## 30.2746.2 Function Documentation

### 30.2746.2.1 void s6sratder ( eder , int idim, int ider1, int ider2, gder , int \* jstat )

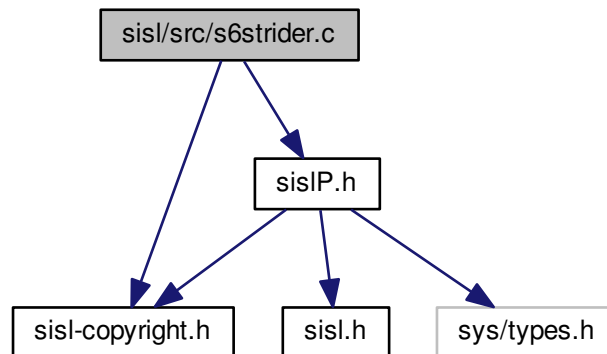
Definition at line 57 of file s6sratder.c.

## 30.2747 sisl/src/s6strider.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s6strider.c:



## Macros

- #define [S6STRIDER](#)

## Functions

- void [s6strider](#) (eder, int idim, int ider, gder, int \*jstat)

## 30.2747.1 Macro Definition Documentation

### 30.2747.1.1 #define S6STRIDER

Definition at line 49 of file s6strider.c.

## 30.2747.2 Function Documentation

### 30.2747.2.1 void s6strider ( eder , int idim, int nder, gder , int \* jstat )

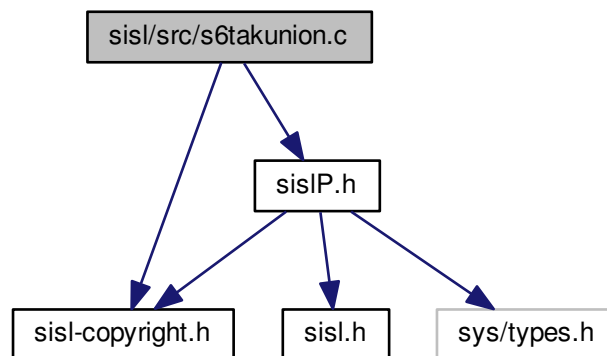
Definition at line 57 of file s6strider.c.

## 30.2748 sisl/src/s6takunion.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s6takunion.c:



## Macros

- #define [S6TAKEUNION](#)

## Functions

- void [s6takeunion](#) (evec1, int ielem1, evec2, int ielem2,\*\*gunion, int \*jnmbelem, int \*jstat)

## 30.2748.1 Macro Definition Documentation

### 30.2748.1.1 #define S6TAKEUNION

Definition at line 49 of file s6takunion.c.

## 30.2748.2 Function Documentation

### 30.2748.2.1 void s6takeunion ( evec1 , int ielem1, evec2 , int ielem2, \*\* gunion, int\* jnmelem, int\* jstat )

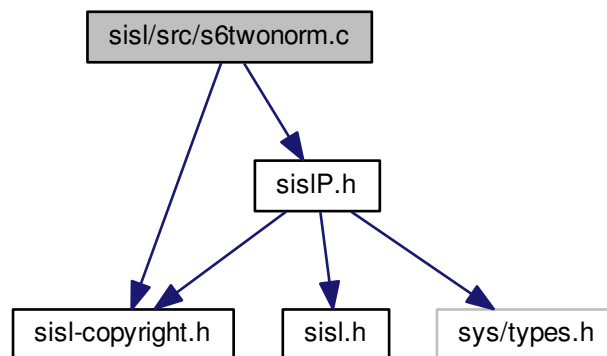
Definition at line 57 of file s6takunion.c.

## 30.2749 sisl/src/s6twonorm.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s6twonorm.c:



## Macros

- #define [S6TWONORM](#)

## Functions

- void [s6twonorm](#) (evec, enorm1, enorm2, int \*jstat)

### 30.2749.1 Macro Definition Documentation

#### 30.2749.1.1 #define S6TWONORM

Definition at line 49 of file s6twonorm.c.

### 30.2749.2 Function Documentation

#### 30.2749.2.1 void s6twonorm ( evec , enorm1 , enorm2 , int \* jstat )

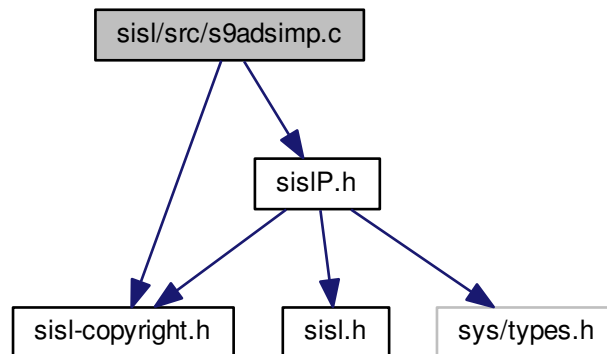
Definition at line 57 of file s6twonorm.c.

## 30.2750 sisl/src/s9adsimp.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s9adsimp.c:



### Macros

- #define [S9ADSIMP](#)

### Functions

- [double s9adsimp](#) (epnt1, epar1, eimpli, int ideg, egd1, epgd1, etang, eptan, [double](#) astep, int \*jstat)



### 30.2750.1 Macro Definition Documentation

#### 30.2750.1.1 #define S9ADSIMP

Definition at line 49 of file s9adsimp.c.

### 30.2750.2 Function Documentation

#### 30.2750.2.1 double s9adsimp ( epnt1 , epar1 , eimpli , int ideg , egd1 , epgd1 , etang , eptan , double astep, int \* jstat )

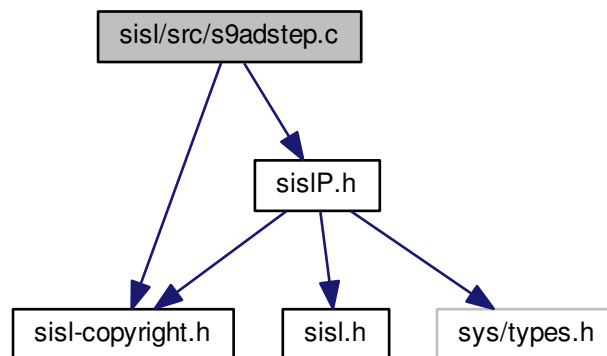
Definition at line 58 of file s9adsimp.c.

## 30.2751 sisl/src/s9adstep.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s9adstep.c:



### Macros

- #define [S9ADSTEP](#)

### Functions

- [double s9adstep](#) (epnt1, epar1, epnt2, epar2, egd1, epgd1, egd2, epgd2, etang, eptan1, eptan2, [double astep](#), int \*jstat)

### 30.2751.1 Macro Definition Documentation

#### 30.2751.1.1 #define S9ADSTEP

Definition at line 49 of file s9adstep.c.

### 30.2751.2 Function Documentation

#### 30.2751.2.1 double s9adstep ( epnt1 , epar1 , epnt2 , epar2 , egd1 , epgd1 , egd2 , epgd2 , etang , eptan1 , eptan2 , double astep, int \* jstat )

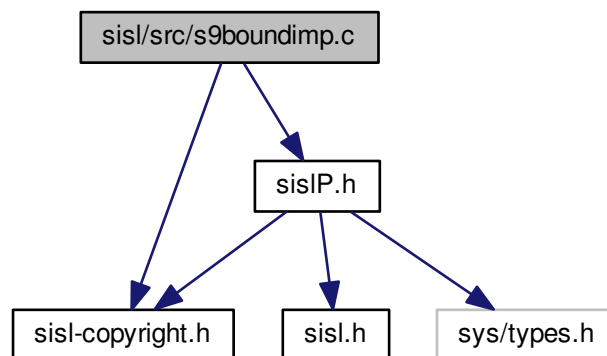
Definition at line 59 of file s9adstep.c.

## 30.2752 sisl/src/s9boundimp.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s9boundimp.c:



### Macros

- #define [S9BOUNDIMP](#)

### Functions

- void [s9boundimp](#) (epnt1, epar1, [SISLSurf](#) \*psurf1, eimpli, int ideg, [double](#) apar, int idir, [double](#) aepsge, gpnt1, gpar1, int \*jstat)

## 30.2752.1 Macro Definition Documentation

### 30.2752.1.1 #define S9BOUNDIMP

Definition at line 49 of file s9boundimp.c.

## 30.2752.2 Function Documentation

### 30.2752.2.1 void s9boundimp ( epnt1 , epar1 , SISLSurf \* psurf1, eimpli , int ideg, double apar, int idir, double aepsge, gpnt1 , gpar1 , int \* jstat )

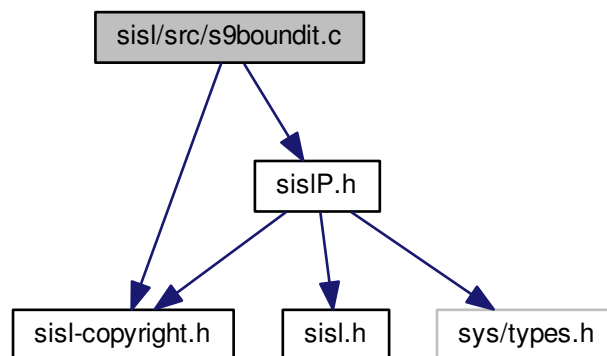
Definition at line 59 of file s9boundimp.c.

## 30.2753 sisl/src/s9boundit.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s9boundit.c:



## Macros

- #define [S9BOUNDIT](#)

## Functions

- void [s9boundit](#) (epnt1, epnt2, epar1, epar2, [SISLSurf](#) \*psurf1, [SISLSurf](#) \*psurf2, [double](#) apar, int idir, [double](#) aepsge, gpnt1, gpnt2, gpar1, gpar2, int \*jstat)

### 30.2753.1 Macro Definition Documentation

#### 30.2753.1.1 #define S9BOUNDIT

Definition at line 49 of file s9boundit.c.

### 30.2753.2 Function Documentation

#### 30.2753.2.1 void s9boundit ( epnt1 , epnt2 , epar1 , epar2 , SISLSurf \* psurf1, SISLSurf \* psurf2, double apar, int idir, double aepsge, gpnt1 , gpnt2 , gpar1 , gpar2 , int \* jstat )

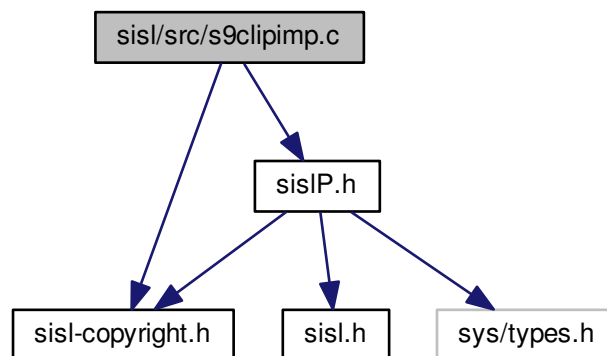
Definition at line 59 of file s9boundit.c.

## 30.2754 sisl/src/s9clipimp.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s9clipimp.c:



### Macros

- #define [S9CLIPIMP](#)

### Functions

- void [s9clipimp](#) (epar1, epar2, [SISLSurf](#) \*psurf1, eimpli, int ideg, euval, evval, [double](#) aepsge, gpnt1, gpar1, int \*jstat)

### 30.2754.1 Macro Definition Documentation

#### 30.2754.1.1 #define S9CLIPIMP

Definition at line 49 of file s9clipimp.c.

### 30.2754.2 Function Documentation

#### 30.2754.2.1 void s9clipimp ( epar1 , epar2 , SISLSurf \* psurf1, eimpli , int ideg, euval , evval , double aepsge, gpnt1 , gpar1 , int \* jstat )

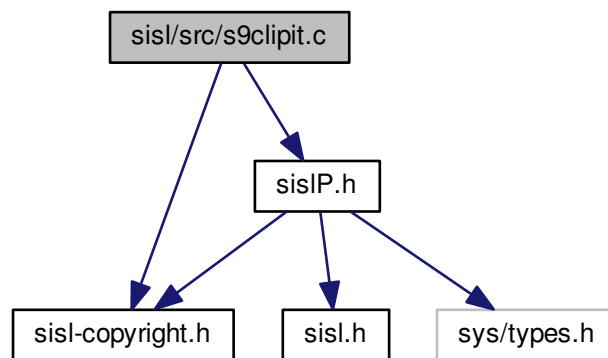
Definition at line 59 of file s9clipimp.c.

## 30.2755 sisl/src/s9clipit.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s9clipit.c:



### Macros

- #define [S9CLIPIT](#)

### Functions

- void [s9clipit](#) (epar11, epar12, epar21, epar22, [SISLSurf](#) \*psurf1, [SISLSurf](#) \*psurf2, euval, evval, esval, etval, [double](#) aepsge, gpnt1, gpnt2, gpar1, gpar2, int \*jstat)

### 30.2755.1 Macro Definition Documentation

#### 30.2755.1.1 #define S9CLIPIT

Definition at line 49 of file s9clipit.c.

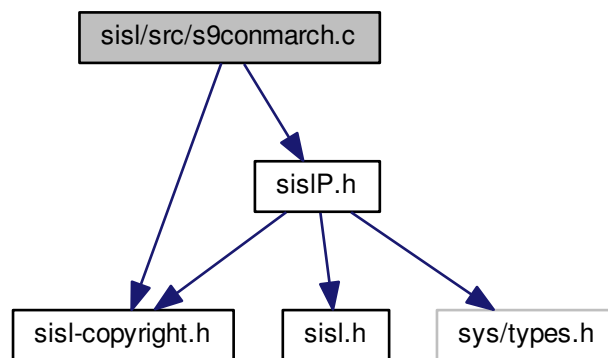
### 30.2755.2 Function Documentation

30.2755.2.1 void s9clipit ( epar11 , epar12 , epar21 , epar22 , SISLSurf \* psurf1, SISLSurf \* psurf2, euval , evval , esval , etval , double aepsge, gpnt1 , gpnt2 , gpar1 , gpar2 , int \* jstat )

Definition at line 60 of file s9clipit.c.

## 30.2756 sisl/src/s9conmarch.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for s9conmarch.c:
```



### Macros

- #define [S9CONMARCH](#)

### Functions

- void [s9conmarch](#) (SISLSurf \*ps, double alevel, epar, ndir, int ipoint, gpar, mpar, int \*jpoint, int \*jstat)

## 30.2756.1 Macro Definition Documentation

### 30.2756.1.1 #define S9CONMARCH

Definition at line 49 of file s9conmarch.c.

## 30.2756.2 Function Documentation

### 30.2756.2.1 void s9conmarch ( SISLSurf \* ps, double alevel, epar, ndir, int ipoint, gpar, mpar, int \* jpoint, int \* jstat )

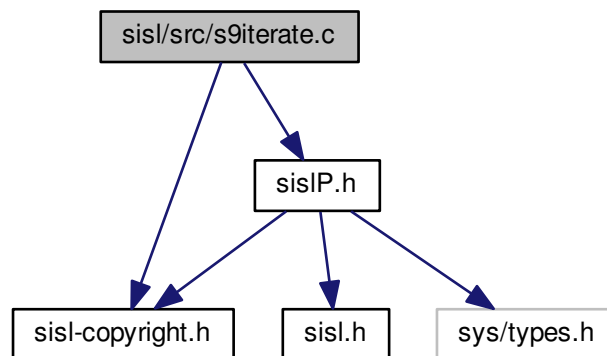
Definition at line 58 of file s9conmarch.c.

## 30.2757 sisl/src/s9iterate.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s9iterate.c:



### Macros

- #define [S9ITERATE](#)

### Functions

- void [s9iterate](#) (epoint, epnt1, epnt2, epar1, epar2, [SISLSurf](#) \*psurf1, [SISLSurf](#) \*psurf2, double astep, double [aepsge](#), gpnt1, gpnt2, gpar1, gpar2, int \*jstat)

### 30.2757.1 Macro Definition Documentation

#### 30.2757.1.1 #define S9ITERATE

Definition at line 47 of file s9iterate.c.

### 30.2757.2 Function Documentation

30.2757.2.1 void s9iterate ( epoint , epnt1 , epnt2 , epar1 , epar2 , SISLSurf \* psurf1, SISLSurf \* psurf2, double astep, double aepsge, gpnt1 , gpnt2 , gpar1 , gpar2 , int \* jstat )

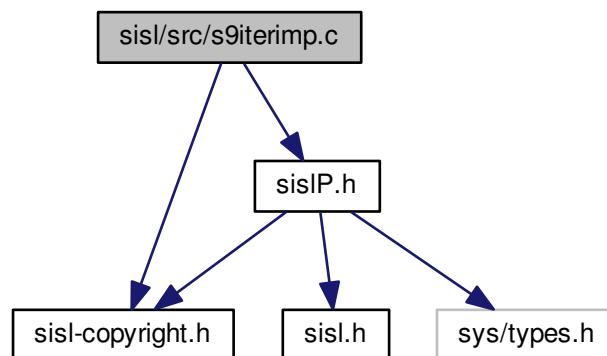
Definition at line 58 of file s9iterate.c.

## 30.2758 sisl/src/s9iterimp.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s9iterimp.c:



### Macros

- #define [S9ITERIMP](#)

### Functions

- void [s9iterimp](#) (epoint, epnt1, epar1, [SISLSurf](#) \*psurf1, eimpli, int ideg, [double](#) astep, [double](#) aepsge, gpnt1, gpar1, int \*jstat)



## 30.2758.1 Macro Definition Documentation

### 30.2758.1.1 #define S9ITERIMP

Definition at line 47 of file s9iterimp.c.

## 30.2758.2 Function Documentation

### 30.2758.2.1 void s9iterimp ( epoint , epnt1 , epar1 , SISLSurf \* *psurf1*, eimpli , int *ideg*, double *astep*, double *aepsge*, gpnt1 , gpar1 , int \* *jstat* )

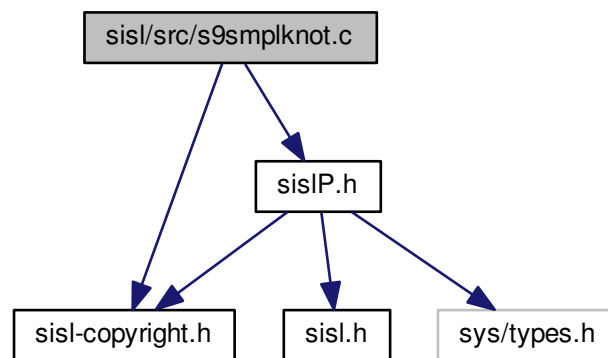
Definition at line 57 of file s9iterimp.c.

## 30.2759 sisl/src/s9smplknot.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s9smplknot.c:



## Macros

- #define [S9SIMPLE\\_KNOT](#)

## Functions

- void [s9simple\\_knot](#) (SISLSurf \*surf, int idiv, epar, int \*fixflag, int \*jstat)

### 30.2759.1 Macro Definition Documentation

#### 30.2759.1.1 #define S9SIMPLE\_KNOT

Definition at line 49 of file s9smplknot.c.

### 30.2759.2 Function Documentation

#### 30.2759.2.1 void s9simple\_knot ( SISLSurf\* surf, int idiv, epar , int \* fixflag, int \* jstat )

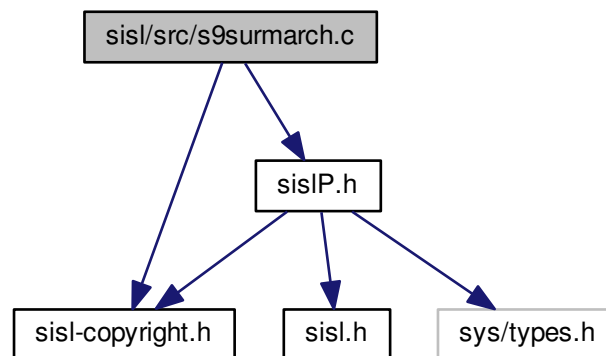
Definition at line 58 of file s9smplknot.c.

## 30.2760 sisl/src/s9surmarch.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for s9surmarch.c:



### Macros

- #define [S9SURMARCH](#)

### Functions

- void [s9surmarch](#) (SISLSurf \*ps1, SISLSurf \*ps2, epar, ndir, int ipoint, gpar, mpar, int \*jpoint, int \*jstat)

## 30.2760.1 Macro Definition Documentation

### 30.2760.1.1 #define S9SURMARCH

Definition at line 49 of file s9surmarch.c.

## 30.2760.2 Function Documentation

### 30.2760.2.1 void s9surmarch ( SISLSurf \* ps1, SISLSurf \* ps2, epar , ndir , int ipoint, gpar , mpar , int \* jpoint, int \* jstat )

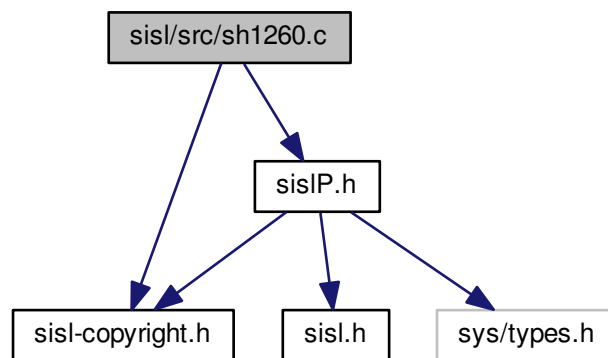
Definition at line 58 of file s9surmarch.c.

## 30.2761 sisl/src/sh1260.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for sh1260.c:



## Macros

- #define [SH1260](#)

## Functions

- void [sh1260](#) (double aconst, vcurve, int icurve, int \*jstat)

### 30.2761.1 Macro Definition Documentation

#### 30.2761.1.1 #define SH1260

Definition at line 42 of file sh1260.c.

### 30.2761.2 Function Documentation

#### 30.2761.2.1 void sh1260 ( double *aconst*, *vcurve* , int *icurve*, int \* *jstat* )

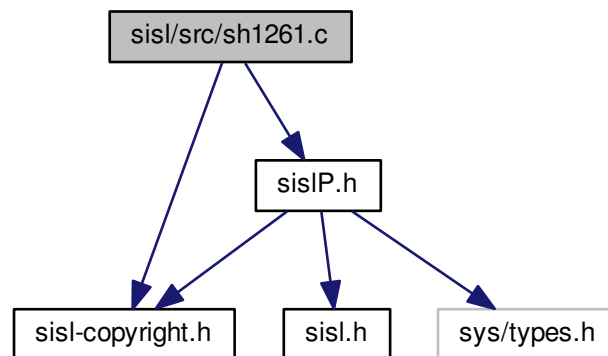
Definition at line 50 of file sh1260.c.

## 30.2762 sisl/src/sh1261.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for sh1261.c:



### Macros

- #define [SH1261](#)

### Functions

- void [sh1261](#) ([SISLCurve](#) \**pcurve1*, [SISLCurve](#) \**pcurve2*, *ecoef1*, int *ik1*, *ecoef2*, int *ik2*, [SISLCurve](#) \*\**rcrtanc*, int \**jstat*)

## 30.2762.1 Macro Definition Documentation

### 30.2762.1.1 #define SH1261

Definition at line 42 of file sh1261.c.

## 30.2762.2 Function Documentation

### 30.2762.2.1 void sh1261 ( SISLCurve \* pcurve1, SISLCurve\* pcurve2, ecoef1 , int ik1, ecoef2 , int ik2, SISLCurve\*\* rcartanc, int\* jstat )

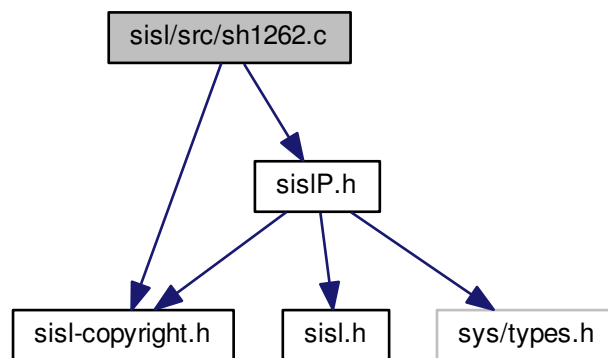
Definition at line 58 of file sh1261.c.

## 30.2763 sisl/src/sh1262.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for sh1262.c:



### Macros

- #define [SH1262](#)

### Functions

- void [sh1262](#) (vcurve, int iedge, int inmbx, ecoef, int \*jstat)

### 30.2763.1 Macro Definition Documentation

#### 30.2763.1.1 #define SH1262

Definition at line 42 of file sh1262.c.

### 30.2763.2 Function Documentation

#### 30.2763.2.1 void sh1262 ( vcurve , int iedge, int inmbx, ecoef , int\* jstat )

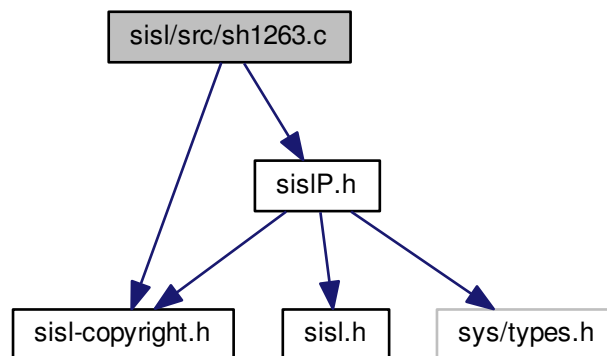
Definition at line 63 of file sh1262.c.

## 30.2764 sisl/src/sh1263.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for sh1263.c:



### Macros

- #define [SH1263](#)

### Functions

- void [sh1263](#) (vcurve, int iedge, vboundc, int \*jstat)

### 30.2764.1 Macro Definition Documentation

#### 30.2764.1.1 #define SH1263

Definition at line 42 of file sh1263.c.

### 30.2764.2 Function Documentation

#### 30.2764.2.1 void sh1263 ( vcurve , int iedge , vboundc , int \* jstat )

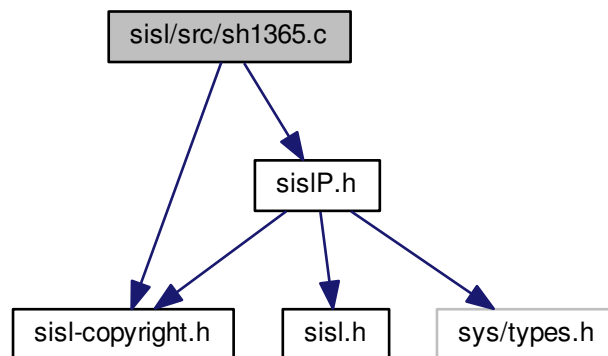
Definition at line 58 of file sh1263.c.

## 30.2765 sisl/src/sh1365.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for sh1365.c:



### Macros

- #define [SH1365](#)

### Functions

- void [sh1365](#) ([SISLCurve](#) \*pcurve, etau, int ik, int in, int ileftfix, int irlightfix, [SISLCurve](#) \*\*rnewcurve, double \*\*gmaxerr, double \*\*gl2err, int \*jstat)

### 30.2765.1 Macro Definition Documentation

#### 30.2765.1.1 #define SH1365

Definition at line 48 of file sh1365.c.

### 30.2765.2 Function Documentation

#### 30.2765.2.1 void sh1365 ( SISLCurve \* pcurve, etau , int ik, int in, int ileftfix, int irlightfix, SISLCurve \*\* rnewcurve, double \*\* gmaxerr, double \*\* gl2err, int \* jstat )

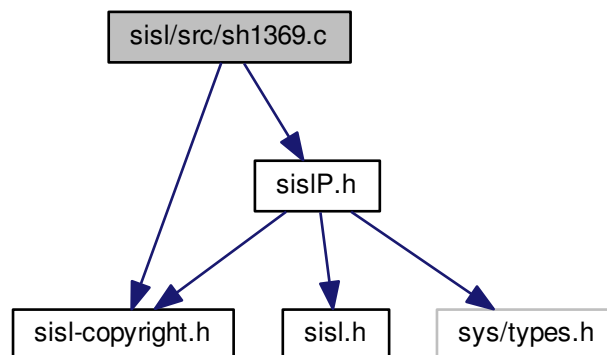
Definition at line 58 of file sh1365.c.

## 30.2766 sisl/src/sh1369.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for sh1369.c:



### Macros

- #define [SH1369](#)

### Functions

- void [sh1369](#) (SISLSurf \*ps, ecentr, enorm, double abigr, double asmalr, int idim, double aepsco, double aepsge, int trackflag, int \*jtrack, SISLTrack \*\*\*wtrack, int \*jpt, double \*\*gpar, int \*\*pretop, int \*jcrv, SISLIntcurve \*\*\*wcurve, int \*jsurf, SISLIntsurf \*\*\*wsurf, int \*jstat)



### 30.2766.1 Macro Definition Documentation

#### 30.2766.1.1 #define SH1369

Definition at line 49 of file sh1369.c.

### 30.2766.2 Function Documentation

30.2766.2.1 void sh1369 ( SISLSurf \* ps, ecentr , enorm , double abigr, double asmalr, int idim, double aepsco, double aepsge, int trackflag, int \* jtrack, SISLTrack \*\*\* wtrack, int \* jpt, double \*\* gpar, int \*\* pretop, int \* jcrv, SISLIntcurve \*\*\* wcurve, int \* jsurf, SISLIntsurf \*\*\* wsurf, int \* jstat )

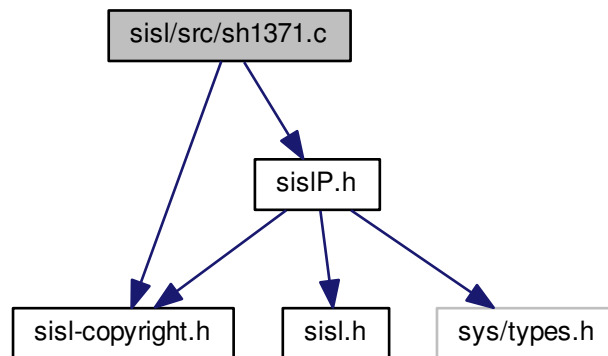
Definition at line 60 of file sh1369.c.

## 30.2767 sisl/src/sh1371.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for sh1371.c:



### Macros

- #define [SH1371](#)

### Functions

- void [sh1371](#) (SISLCurve \*pc1, ecentr, double aradiu, int idim, double aepsco, double aepsge, int trackflag, int \*jtrack, SISLTrack \*\*\*wtrack, int \*jpt, double \*\*gpar, int \*\*pretop, int \*jcrv, SISLIntcurve \*\*\*wcurve, int \*jstat)

### 30.2767.1 Macro Definition Documentation

#### 30.2767.1.1 #define SH1371

Definition at line 49 of file sh1371.c.

### 30.2767.2 Function Documentation

30.2767.2.1 void sh1371 ( SISLCurve \* pc1, ecentr , double aradiu, int idim, double aepsco, double aepsge, int trackflag, int \* jtrack, SISLTrack \*\*\* wtrack, int \* jpt, double \*\* gpar, int \*\* pretop, int \* jcrv, SISLIntcurve \*\*\* wcurve, int \* jstat )

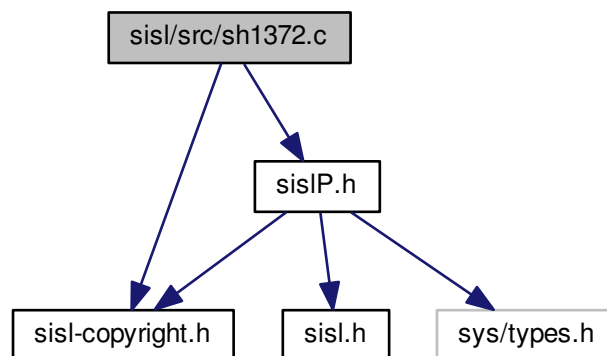
Definition at line 61 of file sh1371.c.

## 30.2768 sisl/src/sh1372.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for sh1372.c:



### Macros

- #define [SH1372](#)

### Functions

- void [sh1372](#) (SISLCurve \*pc1, epoint, edirec, double aradiu, int idim, double aepsco, double aepsge, int trackflag, int \*jtrack, SISLTrack \*\*\*wtrack, int \*jpt, double \*\*gpar, int \*\*pretop, int \*jcrv, SISLIntcurve \*\*\*wcurve, int \*jstat)

### 30.2768.1 Macro Definition Documentation

#### 30.2768.1.1 #define SH1372

Definition at line 49 of file sh1372.c.

### 30.2768.2 Function Documentation

30.2768.2.1 void sh1372 ( SISLCurve \* *pc1*, epoint, edirec, double *aradiu*, int *idim*, double *aepsco*, double *aepsge*, int *trackflag*, int \* *jtrack*, SISLTrack \*\*\* *wtrack*, int \* *jpt*, double \*\* *gpar*, int \*\* *pretop*, int \* *jcrv*, SISLIntcurve \*\*\* *wcurve*, int \* *jstat* )

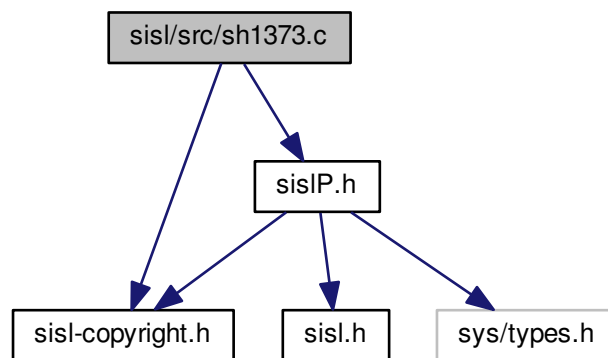
Definition at line 60 of file sh1372.c.

## 30.2769 sisl/src/sh1373.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for sh1373.c:



### Macros

- #define [SH1373](#)

### Functions

- void [sh1373](#) (SISLCurve \**pc1*, etop, eaxis, econc, int *idim*, double *aepsco*, double *aepsge*, int *trackflag*, int \**jtrack*, SISLTrack \*\*\**wtrack*, int \**jpt*, double \*\**gpar*, int \*\**pretop*, int \**jcrv*, SISLIntcurve \*\*\**wcurve*, int \**jstat*)

### 30.2769.1 Macro Definition Documentation

#### 30.2769.1.1 #define SH1373

Definition at line 49 of file sh1373.c.

### 30.2769.2 Function Documentation

30.2769.2.1 void sh1373 ( SISLCurve \* pc1, etop , eaxis , econe , int idim, double aepsco, double aepsge, int trackflag, int \* jtrack, SISLTrack \*\*\* wtrack, int \* jpt, double \*\* gpar, int \*\* pretop, int \* jcrv, SISLIntcurve \*\*\* wcurve, int \* jstat )

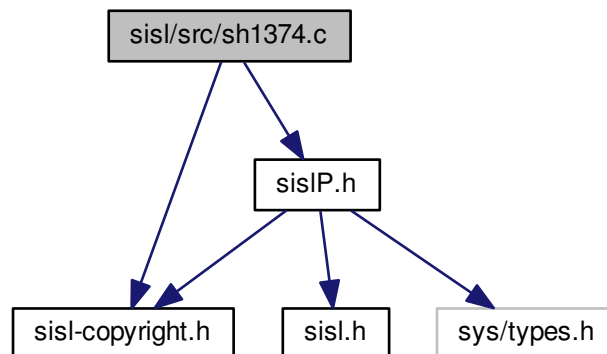
Definition at line 60 of file sh1373.c.

## 30.2770 sisl/src/sh1374.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for sh1374.c:



### Macros

- #define [SH1374](#)

### Functions

- void [sh1374](#) (SISLCurve \*pc1, earray, int idim, [double](#) aepsco, [double](#) aepsge, int trackflag, int \*jtrack, [SISLTrack](#) \*\*\*wtrack, int \*jpt, [double](#) \*\*gpar, int \*\*pretop, int \*jcrv, [SISLIntcurve](#) \*\*\*wcurve, int \*jstat)

### 30.2770.1 Macro Definition Documentation

#### 30.2770.1.1 #define SH1374

Definition at line 49 of file sh1374.c.

### 30.2770.2 Function Documentation

30.2770.2.1 void sh1374 ( SISLCurve \*pc1, earray , int idim, double aepsco, double aepsge, int trackflag, int \*jtrack, SISLTrack \*\*\*wtrack, int \*jpt, double \*\*gpar, int \*\*pretop, int \*jcrv, SISLIntcurve \*\*\*wcurve, int \*jstat )

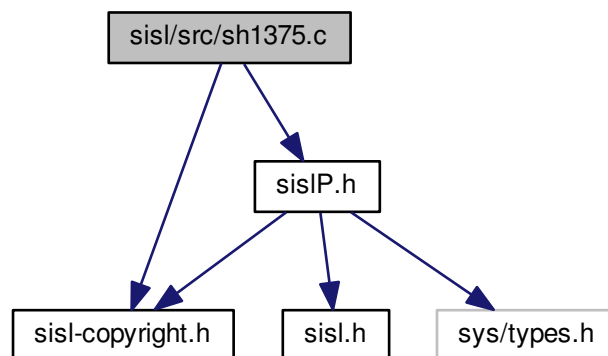
Definition at line 60 of file sh1374.c.

## 30.2771 sisl/src/sh1375.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for sh1375.c:



### Macros

- #define [SH1375](#)

### Functions

- void [sh1375](#) (SISLCurve \*pc1, ecentr, enorm, double abigr, double asmalr, int idim, double aepsco, double aepsge, int trackflag, int \*jtrack, SISLTrack \*\*\*wtrack, int \*jpt, double \*\*gpar, int \*\*pretop, int \*jcrv, SISLIntcurve \*\*\*wcurve, int \*jstat)

### 30.2771.1 Macro Definition Documentation

#### 30.2771.1.1 #define SH1375

Definition at line 49 of file sh1375.c.

### 30.2771.2 Function Documentation

30.2771.2.1 void sh1375 ( SISLCurve \* *pc1*, *ecentr*, *enorm*, double *abigr*, double *asmalr*, int *idim*, double *aepsco*, double *aepsge*, int *trackflag*, int \* *jtrack*, SISLTrack \*\*\* *wtrack*, int \* *jpt*, double \*\* *gpar*, int \*\* *pretop*, int \* *jcrv*, SISLIntcurve \*\*\* *wcurve*, int \* *jstat* )

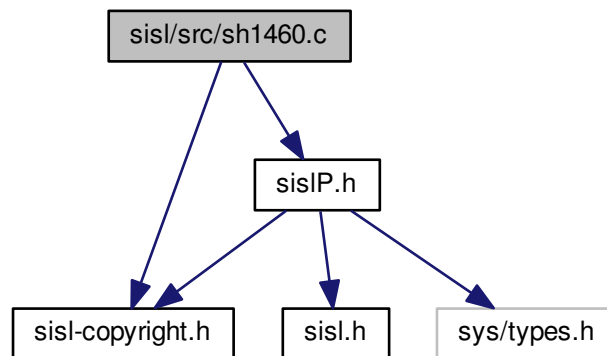
Definition at line 60 of file sh1375.c.

### 30.2772 sisl/src/sh1460.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for sh1460.c:



#### Macros

- #define [SH1460](#)

#### Typedefs

- typedef void(\* [fshapeProc](#)) ()
- typedef void(\* [fevalmidProc](#)) ()

## Functions

- void `sh1460` (`fshapeProc` `fshape`, `vboundc`, `int` `icurv`, `SISLSurf` `***wsurf`, `int` `*jstat`)

### 30.2772.1 Macro Definition Documentation

#### 30.2772.1.1 `#define SH1460`

Definition at line 49 of file `sh1460.c`.

### 30.2772.2 Typedef Documentation

#### 30.2772.2.1 `typedef void(* fevalmidProc)()`

Definition at line 64 of file `sh1460.c`.

#### 30.2772.2.2 `typedef void(* fshapeProc)()`

Definition at line 54 of file `sh1460.c`.

### 30.2772.3 Function Documentation

#### 30.2772.3.1 `void sh1460 ( fshapeProc fshape, vboundc , int icurv, SISLSurf *** wsurf, int * jstat )`

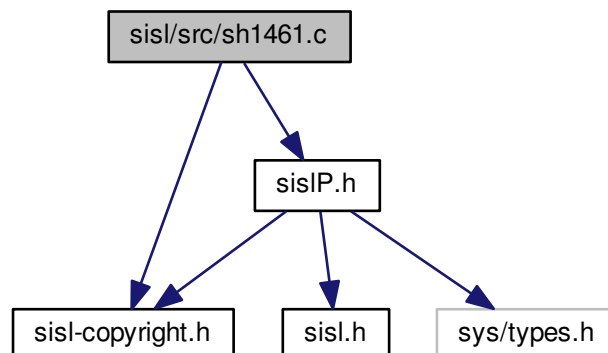
Definition at line 80 of file `sh1460.c`.

## 30.2773 sisl/src/sh1461.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for `sh1461.c`:



## Macros

- `#define SH1461`

## Typedefs

- `typedef void(* fshapeProc) ()`
- `typedef void(* initProc) ()`

## Functions

- `void sh1461 (fshapeProc fshape, initProc f_initmid, vboundc, int icurv, vsurf, int *jstat)`

### 30.2773.1 Macro Definition Documentation

#### 30.2773.1.1 `#define SH1461`

Definition at line 49 of file sh1461.c.

### 30.2773.2 Typedef Documentation

#### 30.2773.2.1 `typedef void(* fshapeProc) ()`

Definition at line 58 of file sh1461.c.

#### 30.2773.2.2 `typedef void(* initProc) ()`

Definition at line 59 of file sh1461.c.

### 30.2773.3 Function Documentation

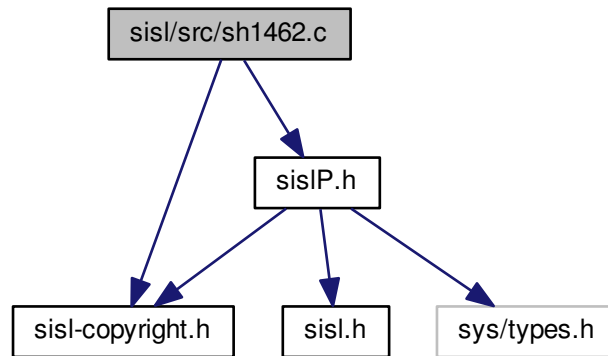
#### 30.2773.3.1 `void sh1461 ( fshapeProc fshape, initProc f_initmid, vboundc , int icurv, vsurf , int * jstat )`

Definition at line 87 of file sh1461.c.



## 30.2774 sisl/src/sh1462.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh1462.c:
```



### Macros

- `#define SH1462`

### Typedefs

- `typedef void(* fshapeProc) ()`

### Functions

- `void sh1462 (fshapeProc fshape, vboundc, int icurv, etwist, etang, eder, int *jstat)`

#### 30.2774.1 Macro Definition Documentation

##### 30.2774.1.1 `#define SH1462`

Definition at line 49 of file `sh1462.c`.

#### 30.2774.2 Typedef Documentation

##### 30.2774.2.1 `typedef void(* fshapeProc) ()`

Definition at line 56 of file `sh1462.c`.

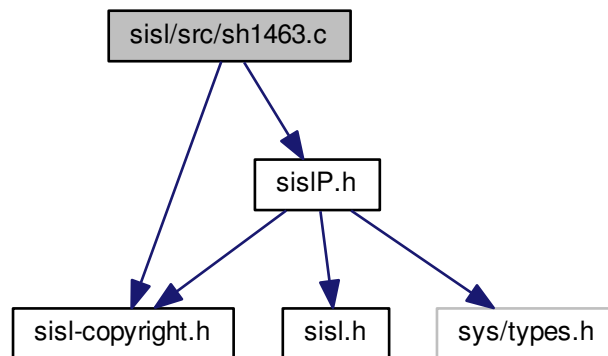
### 30.2774.3 Function Documentation

30.2774.3.1 void sh1462 ( fshapeProc fshape, vboundc , int icurv, etwist , etang , eder , int\* jstat )

Definition at line 64 of file sh1462.c.

### 30.2775 sisl/src/sh1463.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh1463.c:
```



#### Macros

- #define [SH1463](#)

#### Typedefs

- typedef void(\* [fshapeProc](#)) ()

#### Functions

- void [sh1463](#) ([fshapeProc](#) fshape, vboundc, int icurv, etwist, etang, eder, int \*jstat)

### 30.2775.1 Macro Definition Documentation

30.2775.1.1 #define SH1463

Definition at line 49 of file sh1463.c.

## 30.2775.2 Typedef Documentation

### 30.2775.2.1 typedef void(\* fshapeProc) ()

Definition at line 56 of file sh1463.c.

## 30.2775.3 Function Documentation

### 30.2775.3.1 void sh1463 ( fshapeProc fshape, vboundc , int icurv, etwist , etang , eder , int\* jstat )

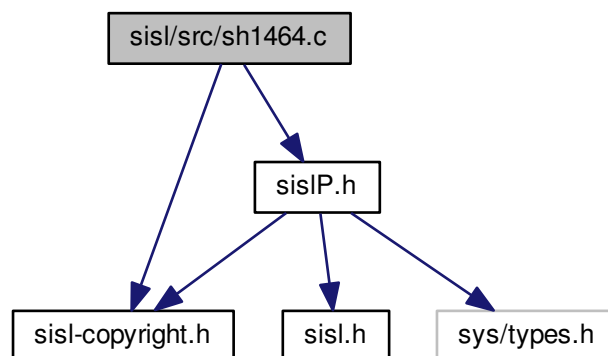
Definition at line 65 of file sh1463.c.

## 30.2776 sisl/src/sh1464.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for sh1464.c:



## Macros

- `#define SH1464`

## Typedefs

- `typedef void(* fshapeProc) ()`

## Functions

- `void sh1464 (fshapeProc fshape, vboundc, int icurv, etwist, etang, eder, int *jstat)`

### 30.2776.1 Macro Definition Documentation

#### 30.2776.1.1 #define SH1464

Definition at line 49 of file sh1464.c.

### 30.2776.2 Typedef Documentation

#### 30.2776.2.1 typedef void(\* fshapeProc) ()

Definition at line 56 of file sh1464.c.

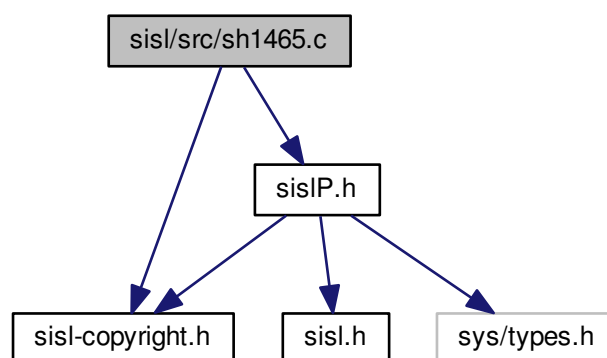
### 30.2776.3 Function Documentation

#### 30.2776.3.1 void sh1464 ( fshapeProc fshape, vboundc , int icurv, etwist , etang , eder , int\* jstat )

Definition at line 64 of file sh1464.c.

## 30.2777 sisl/src/sh1465.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh1465.c:
```



### Macros

- #define [SH1465](#)

## Typedefs

- typedef void(\* [fshapeProc](#)) ()

## Functions

- void [sh1465](#) ([fshapeProc](#) fshape, vboundc, int icurv, etwist, etang, eder, int \*jstat)

### 30.2777.1 Macro Definition Documentation

#### 30.2777.1.1 #define SH1465

Definition at line 49 of file sh1465.c.

### 30.2777.2 Typedef Documentation

#### 30.2777.2.1 typedef void(\* [fshapeProc](#)) ()

Definition at line 59 of file sh1465.c.

### 30.2777.3 Function Documentation

#### 30.2777.3.1 void [sh1465](#) ( [fshapeProc](#) fshape, vboundc , int icurv, etwist , etang , eder , int\* jstat )

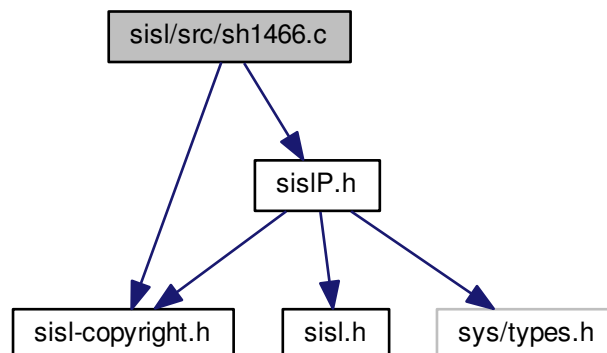
Definition at line 67 of file sh1465.c.

## 30.2778 sisl/src/sh1466.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for sh1466.c:



## Macros

- #define [SH1466](#)

## Functions

- void [sh1466](#) (ecurve, etwist, int ider, ebar, eval, int \*jstat)

### 30.2778.1 Macro Definition Documentation

#### 30.2778.1.1 #define SH1466

Definition at line 49 of file sh1466.c.

### 30.2778.2 Function Documentation

#### 30.2778.2.1 void sh1466 ( ecurve , etwist , int ider , ebar , eval , int\* jstat )

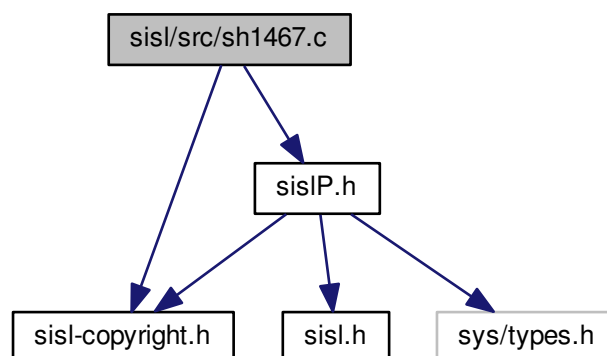
Definition at line 58 of file sh1466.c.

### 30.2779 sisl/src/sh1467.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for sh1467.c:



## Macros

- #define [SH1467](#)

## Functions

- void [sh1467](#) (ecurve, etwist, int ider, ebar, eval, int \*jstat)

## 30.2779.1 Macro Definition Documentation

### 30.2779.1.1 #define SH1467

Definition at line 49 of file sh1467.c.

## 30.2779.2 Function Documentation

### 30.2779.2.1 void sh1467 ( ecurve , etwist , int ider , ebar , eval , int\* jstat )

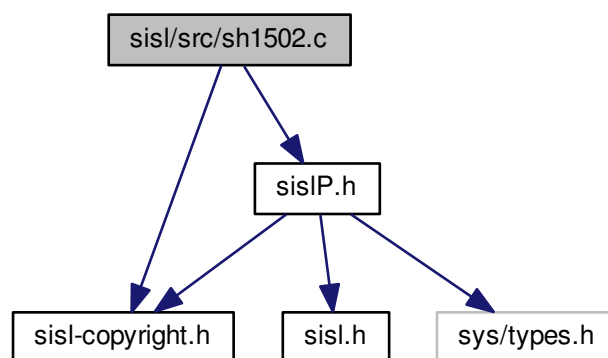
Definition at line 68 of file sh1467.c.

## 30.2780 sisl/src/sh1502.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for sh1502.c:



## Macros

- #define [SH1502](#)

## Functions

- void [sh1502](#) ([SISLCurve](#) \*pc1, base, norm, axisA, alpha, ratio, int idim, [double](#) aepsco, [double](#) aepsge, int trackflag, int \*jtrack, [SISLTrack](#) \*\*\*wtrack, int \*jpt, [double](#) \*\*gpar, int \*\*pretop, int \*jcrv, [SISLIntcurve](#) \*\*\*wcurve, int \*jstat)

### 30.2780.1 Macro Definition Documentation

#### 30.2780.1.1 #define SH1502

Definition at line 49 of file sh1502.c.

### 30.2780.2 Function Documentation

#### 30.2780.2.1 void sh1502 ( [SISLCurve](#) \* pc1, base , norm , axisA , alpha , ratio , int idim, [double](#) aepsco, [double](#) aepsge, int trackflag, int \* jtrack, [SISLTrack](#) \*\*\* wtrack, int \* jpt, [double](#) \*\* gpar, int \*\* pretop, int \* jcrv, [SISLIntcurve](#) \*\*\* wcurve, int \* jstat )

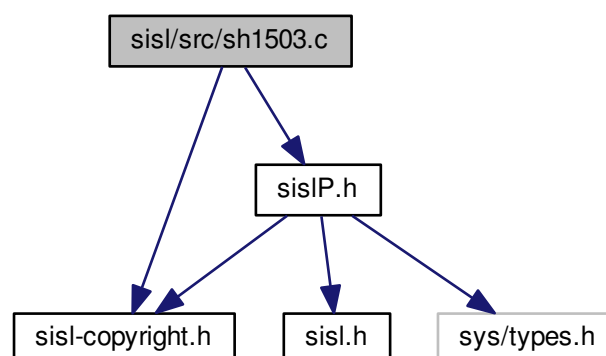
Definition at line 60 of file sh1502.c.

### 30.2781 sisl/src/sh1503.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for sh1503.c:



## Macros

- #define [SH1503](#)



## Functions

- void `sh1503` (`SISLSurf *ps1`, `base`, `norm`, `axisA`, `double alpha`, `double ratio`, `int idim`, `double aepsco`, `double aepsge`, `int trackflag`, `int *jtrack`, `SISLTrack ***wtrack`, `int *jpt`, `double **gpar`, `int **pretop`, `int *jcrv`, `SISLIntcurve ***wcurve`, `int *jsurf`, `SISLIntsurf ***wsurf`, `int *jstat`)

### 30.2781.1 Macro Definition Documentation

#### 30.2781.1.1 #define SH1503

Definition at line 49 of file sh1503.c.

### 30.2781.2 Function Documentation

- 30.2781.2.1 void `sh1503` ( `SISLSurf * ps1`, `base` , `norm` , `axisA` , `double alpha`, `double ratio`, `int idim`, `double aepsco`, `double aepsge`, `int trackflag`, `int * jtrack`, `SISLTrack *** wtrack`, `int * jpt`, `double ** gpar`, `int ** pretop`, `int * jcrv`, `SISLIntcurve *** wcurve`, `int * jsurf`, `SISLIntsurf *** wsurf`, `int * jstat` )

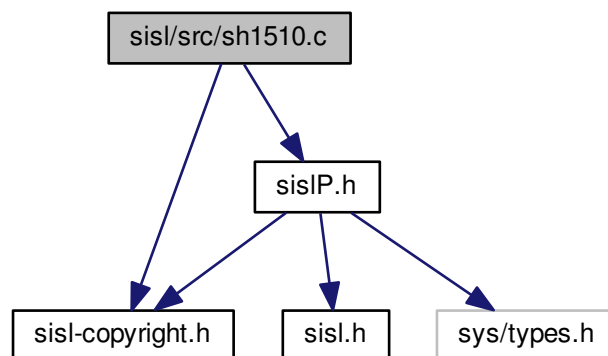
Definition at line 61 of file sh1503.c.

## 30.2782 sisl/src/sh1510.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for sh1510.c:



## Macros

- #define `SH1510`

## Functions

- void `sh1510` (`SISLSurf` \*ps, eyepoint, int idim, double aepsco, double aepsge, int trackflag, int \*jtrack, `SISLTrack` \*\*\*wtrack, int \*jpt, double \*\*gpar, int \*\*pretop, int \*jcrv, `SISLIntcurve` \*\*\*wcurve, int \*jsurf, `SISLIntsurf` \*\*\*wsurf, int \*jstat)

### 30.2782.1 Macro Definition Documentation

#### 30.2782.1.1 #define SH1510

Definition at line 49 of file sh1510.c.

### 30.2782.2 Function Documentation

- 30.2782.2.1 void `sh1510` ( `SISLSurf` \* ps, eyepoint , int idim, double aepsco, double aepsge, int trackflag, int \* jtrack, `SISLTrack` \*\*\* wtrack, int \* jpt, double \*\* gpar, int \*\* pretop, int \* jcrv, `SISLIntcurve` \*\*\* wcurve, int \* jsurf, `SISLIntsurf` \*\*\* wsurf, int \* jstat )

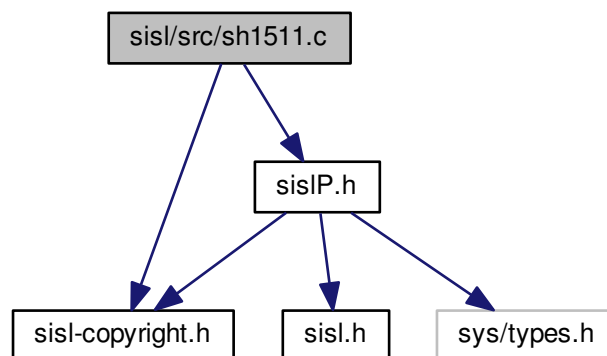
Definition at line 59 of file sh1510.c.

### 30.2783 sisl/src/sh1511.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for sh1511.c:



## Macros

- #define `SH1511`

## Functions

- void [sh1511](#) ([SISLSurf](#) \*ps, qpoint, bvec, int idim, [double](#) aepsco, [double](#) aepsge, int trackflag, int \*jtrack, [SISLTrack](#) \*\*\*wtrack, int \*jpt, [double](#) \*\*gpar, int \*\*pretop, int \*jcrv, [SISLIntcurve](#) \*\*\*wcurve, int \*jsurf, [SISLIntsurf](#) \*\*\*wsurf, int \*jstat)

### 30.2783.1 Macro Definition Documentation

#### 30.2783.1.1 #define SH1511

Definition at line 49 of file sh1511.c.

### 30.2783.2 Function Documentation

- 30.2783.2.1 void [sh1511](#) ( [SISLSurf](#) \* ps, qpoint , bvec , int idim, [double](#) aepsco, [double](#) aepsge, int trackflag, int \* jtrack, [SISLTrack](#) \*\*\* wtrack, int \* jpt, [double](#) \*\* gpar, int \*\* pretop, int \* jcrv, [SISLIntcurve](#) \*\*\* wcurve, int \* jsurf, [SISLIntsurf](#) \*\*\* wsurf, int \* jstat )

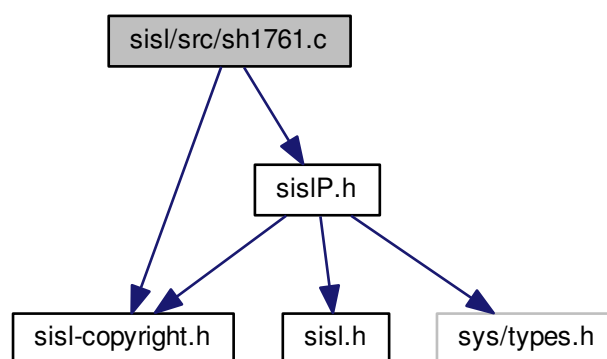
Definition at line 59 of file sh1511.c.

## 30.2784 sisl/src/sh1761.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for sh1761.c:



## Macros

- #define [SH1761](#)

## Functions

- void [sh1761](#) ([SISLObject](#) \*po1, [SISLObject](#) \*po2, [double](#) *aepsge*, [SISLIntdat](#) \*\*pintdat, int \*jstat)

### 30.2784.1 Macro Definition Documentation

#### 30.2784.1.1 #define SH1761

Definition at line 49 of file sh1761.c.

### 30.2784.2 Function Documentation

#### 30.2784.2.1 void sh1761 ( [SISLObject](#) \* *po1*, [SISLObject](#) \* *po2*, [double](#) *aepsge*, [SISLIntdat](#) \*\* *pintdat*, int \* *jstat* )

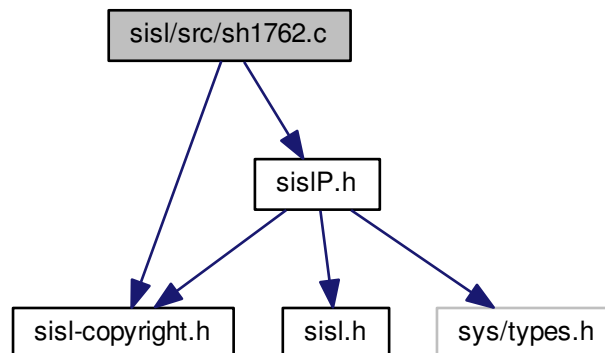
Definition at line 59 of file sh1761.c.

## 30.2785 sisl/src/sh1762.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for sh1762.c:



## Macros

- #define [SH1762](#)

## Functions

- void [sh1762](#) ([SISLObject](#) \*po1, [SISLObject](#) \*po2, [double](#) *aepsge*, [SISLIntdat](#) \*\*pintdat, *vedge*, int \*jstat)

### 30.2785.1 Macro Definition Documentation

#### 30.2785.1.1 #define SH1762

Definition at line 49 of file sh1762.c.

### 30.2785.2 Function Documentation

#### 30.2785.2.1 void sh1762 ( SISLObject \* po1, SISLObject \* po2, double aepsge, SISLIntdat \*\* pintdat, vedge , int \* jstat )

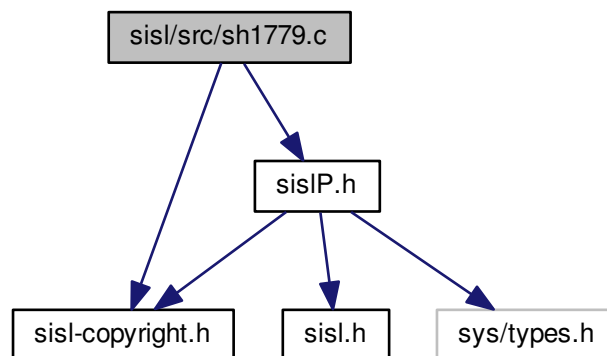
Definition at line 118 of file sh1762.c.

## 30.2786 sisl/src/sh1779.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for sh1779.c:



### Macros

- #define [SH1779](#)

### Functions

- void [sh1779](#) (SISLObject \*po1, SISLObject \*po2, double aepsge, SISLIntdat \*\*rintdat, SISLIntpt \*pintpt, int \*jnewpt, int \*jstat)

### 30.2786.1 Macro Definition Documentation

#### 30.2786.1.1 #define SH1779

Definition at line 49 of file sh1779.c.

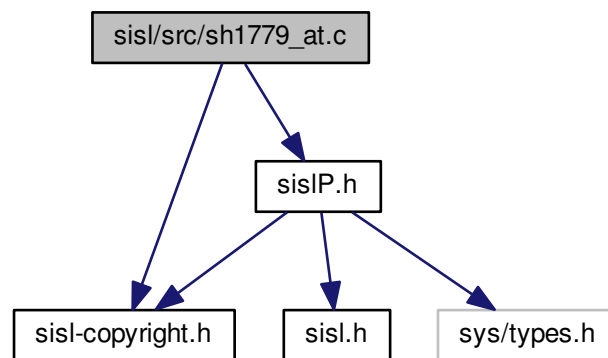
### 30.2786.2 Function Documentation

#### 30.2786.2.1 void sh1779 ( SISLObject \* po1, SISLObject \* po2, double aepsge, SISLIntdat \*\* rintdat, SISLIntpt \* pintpt, int \* jnewpt, int \* jstat )

Definition at line 61 of file sh1779.c.

### 30.2787 sisl/src/sh1779\_at.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh1779_at.c:
```



#### Macros

- #define [SH1779\\_AT](#)

#### Functions

- void [sh1779\\_at](#) (SISLObject \*po1, SISLObject \*po2, SISLIntpt \*pintpt, int \*jstat)

### 30.2787.1 Macro Definition Documentation

#### 30.2787.1.1 #define SH1779\_AT

Definition at line 49 of file sh1779\_at.c.

### 30.2787.2 Function Documentation

#### 30.2787.2.1 void sh1779\_at ( SISLObject \* po1, SISLObject \* po2, SISLIntpt \* pintpt, int \* jstat )

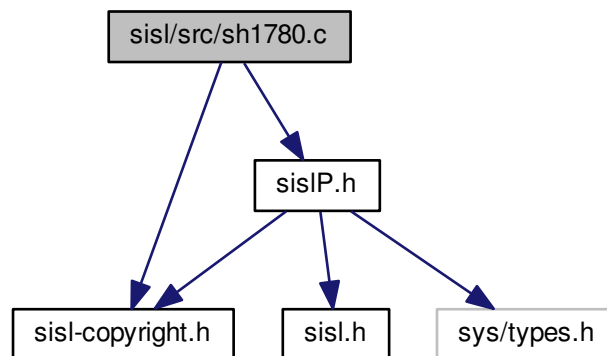
Definition at line 60 of file sh1779\_at.c.

## 30.2788 sisl/src/sh1780.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for sh1780.c:



### Macros

- #define [SH1780](#)

### Functions

- void [sh1780](#) (SISLObject \*po1, SISLObject \*po2, double aepsge, SISLIntdat \*\*rintdat, SISLIntpt \*pintpt, int \*jnewpt, int \*jstat)

### 30.2788.1 Macro Definition Documentation

#### 30.2788.1.1 #define SH1780

Definition at line 49 of file sh1780.c.

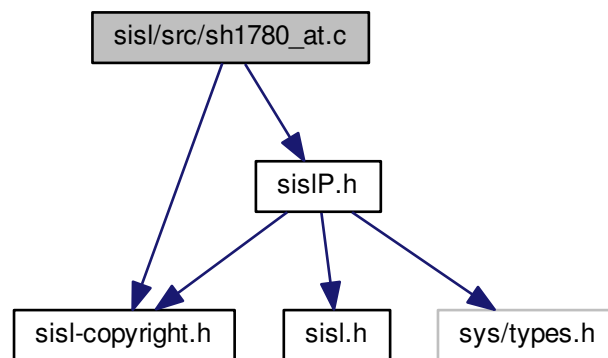
### 30.2788.2 Function Documentation

#### 30.2788.2.1 void sh1780 ( SISLObject \* po1, SISLObject \* po2, double aepsge, SISLIntdat \*\* rintdat, SISLIntpt \* pintpt, int \* jnewpt, int \* jstat )

Definition at line 59 of file sh1780.c.

## 30.2789 sisl/src/sh1780\_at.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh1780_at.c:
```



### Macros

- #define [SH1780\\_AT](#)

### Functions

- void [sh1780\\_at](#) (SISLObject \*po1, SISLObject \*po2, SISLIntpt \*pintpt, int \*jstat)



### 30.2789.1 Macro Definition Documentation

#### 30.2789.1.1 #define SH1780\_AT

Definition at line 49 of file sh1780\_at.c.

### 30.2789.2 Function Documentation

#### 30.2789.2.1 void sh1780\_at ( SISLObject \* po1, SISLObject \* po2, SISLIntpt \* pintpt, int \* jstat )

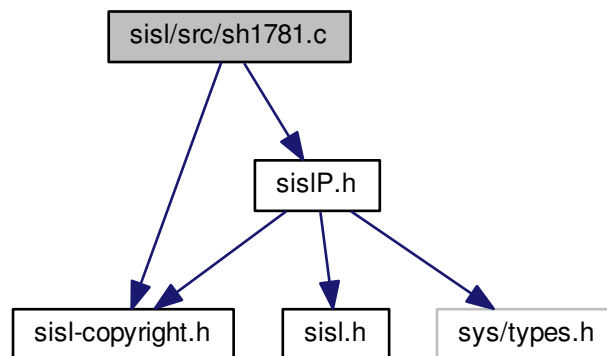
Definition at line 60 of file sh1780\_at.c.

## 30.2790 sisl/src/sh1781.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for sh1781.c:



### Macros

- #define [SH1781](#)

### Functions

- void [sh1781](#) (SISLObject \*po1, SISLObject \*po2, double aepsge, SISLIntdat \*\*rintdat, SISLIntpt \*pintpt, int \*jnewpt, int \*jstat)

### 30.2790.1 Macro Definition Documentation

#### 30.2790.1.1 #define SH1781

Definition at line 49 of file sh1781.c.

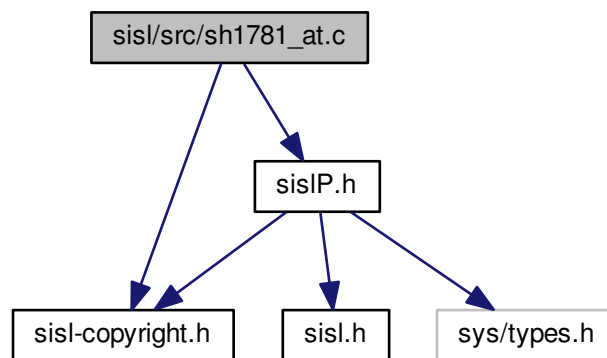
### 30.2790.2 Function Documentation

#### 30.2790.2.1 void sh1781 ( SISLObject \* po1, SISLObject \* po2, double aepsge, SISLIntdat \*\* rintdat, SISLIntpt \* pintpt, int \* jnewpt, int \* jstat )

Definition at line 61 of file sh1781.c.

## 30.2791 sisl/src/sh1781\_at.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh1781_at.c:
```



### Macros

- #define [SH1781\\_AT](#)

### Functions

- void [sh1781\\_at](#) (SISLObject \*po1, SISLObject \*po2, SISLIntpt \*pintpt, int \*jstat)

### 30.2791.1 Macro Definition Documentation

#### 30.2791.1.1 #define SH1781\_AT

Definition at line 49 of file sh1781\_at.c.

### 30.2791.2 Function Documentation

#### 30.2791.2.1 void sh1781\_at ( SISLObject \* po1, SISLObject \* po2, SISLIntpt \* pintpt, int \* jstat )

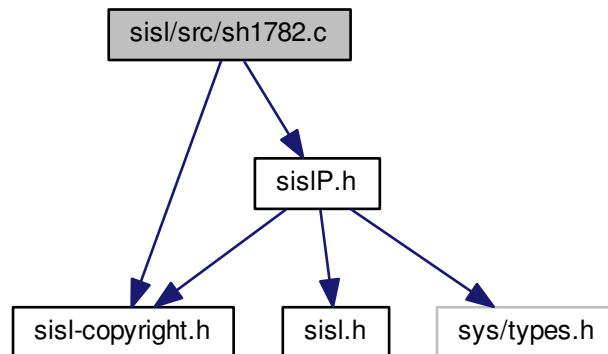
Definition at line 62 of file sh1781\_at.c.

## 30.2792 sisl/src/sh1782.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for sh1782.c:



### Macros

- #define [SH1782](#)

### Functions

- void [sh1782](#) (SISLObject \*po1, SISLObject \*po2, double aepsge, SISLIntdat \*pintdat, int ipar, double apar, SISLIntdat \*\*rintdat, int \*jnewpt, int \*jstat)

### 30.2792.1 Macro Definition Documentation

#### 30.2792.1.1 #define SH1782

Definition at line 49 of file sh1782.c.

### 30.2792.2 Function Documentation

#### 30.2792.2.1 void sh1782 ( SISLObject \* po1, SISLObject \* po2, double aepsge, SISLIntdat \* pintdat, int ipar, double apar, SISLIntdat \*\* rintdat, int \* jnewpt, int \* jstat )

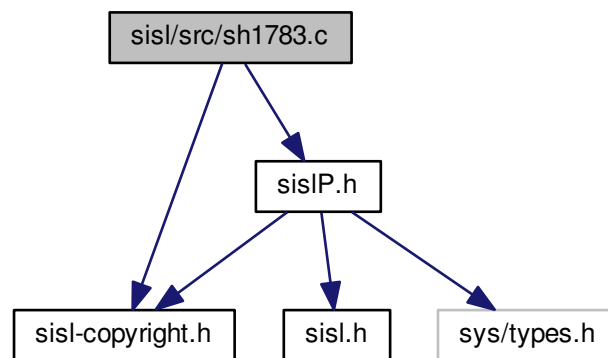
Definition at line 78 of file sh1782.c.

## 30.2793 sisl/src/sh1783.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for sh1783.c:



### Macros

- #define [SH1783](#)

### Typedefs

- typedef void(\* [fevalProc](#)) ()

## Functions

- void [sh1783](#) ([SISLCurve](#) \*pc1, [SISLCurve](#) \*pc2, [double](#) [aepsge](#), epar, int idir1, int idir2, elast, enext, int \*jstat)

### 30.2793.1 Macro Definition Documentation

#### 30.2793.1.1 #define SH1783

Definition at line 49 of file sh1783.c.

### 30.2793.2 Typedef Documentation

#### 30.2793.2.1 typedef void(\* fevalProc)()

Definition at line 56 of file sh1783.c.

### 30.2793.3 Function Documentation

#### 30.2793.3.1 void sh1783 ( [SISLCurve](#) \* pc1, [SISLCurve](#) \* pc2, [double](#) [aepsge](#), epar , int [idir1](#), int [idir2](#), elast , enext , int \* [jstat](#) )

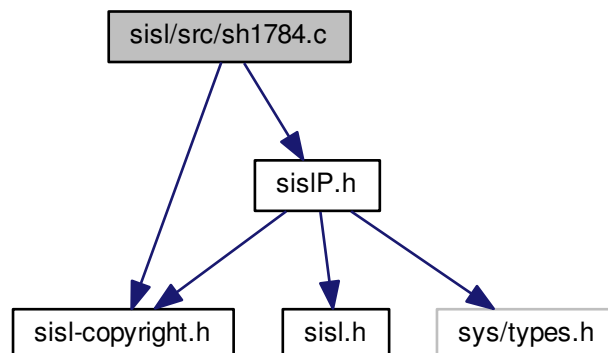
Definition at line 73 of file sh1783.c.

## 30.2794 sisl/src/sh1784.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for sh1784.c:



## Macros

- #define [SH1784](#)

## Typedefs

- typedef void(\* [fevalcProc](#)) ()

## Functions

- void [sh1784](#) ([SISLCurve](#) \**pcurve*, [SISLSurf](#) \**psurf*, [double](#) *aepsge*, *epar*, *icur*, *idirc*, *elast*, *enext*, *int* \**jstat*)

### 30.2794.1 Macro Definition Documentation

#### 30.2794.1.1 #define SH1784

Definition at line 49 of file sh1784.c.

### 30.2794.2 Typedef Documentation

#### 30.2794.2.1 typedef void(\* fevalcProc) ()

Definition at line 54 of file sh1784.c.

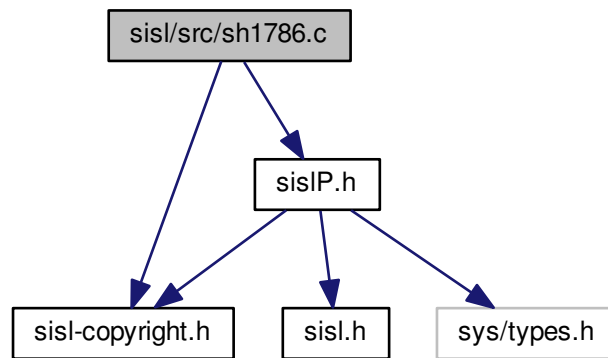
### 30.2794.3 Function Documentation

#### 30.2794.3.1 void sh1784 ( [SISLCurve](#) \* *pcurve*, [SISLSurf](#) \* *psurf*, [double](#) *aepsge*, *epar* , *icur*, *idirc*, *elast* , *enext* , *int* \* *jstat* )

Definition at line 73 of file sh1784.c.

## 30.2795 sisl/src/sh1786.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh1786.c:
```



### Macros

- `#define SH1786`

### Functions

- `void sh1786 (SISLObject *po1, SISLObject *po2, double aepsge, SISLIntdat **rintdat, SISLIntpt *pintpt, int *jnewpt, int *jstat)`

#### 30.2795.1 Macro Definition Documentation

##### 30.2795.1.1 `#define SH1786`

Definition at line 49 of file `sh1786.c`.

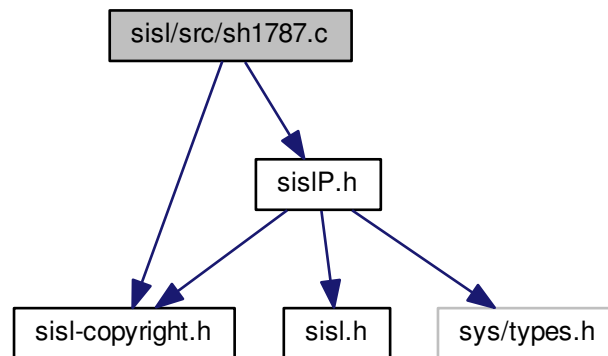
#### 30.2795.2 Function Documentation

##### 30.2795.2.1 `void sh1786 ( SISLObject * po1, SISLObject * po2, double aepsge, SISLIntdat ** rintdat, SISLIntpt * pintpt, int * jnewpt, int * jstat )`

Definition at line 61 of file `sh1786.c`.

## 30.2796 sisl/src/sh1787.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh1787.c:
```



### Macros

- `#define SH1787`

### Functions

- void `sh1787` (SISLObject \*po1, SISLObject \*po2, double aepsge, SISLIntdat \*\*rintdat, SISLIntpt \*pintpt, int \*jnewpt, int \*jstat)

### 30.2796.1 Macro Definition Documentation

#### 30.2796.1.1 #define SH1787

Definition at line 49 of file sh1787.c.

### 30.2796.2 Function Documentation

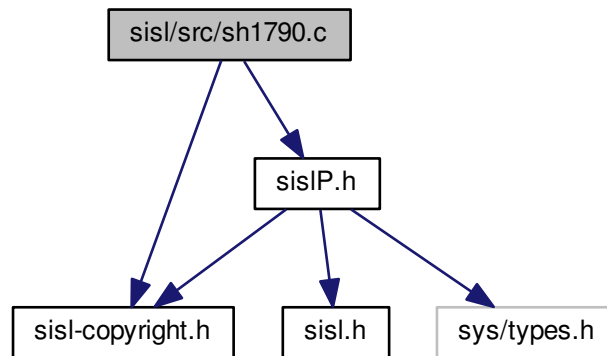
#### 30.2796.2.1 void sh1787 ( SISLObject \* po1, SISLObject \* po2, double aepsge, SISLIntdat \*\* rintdat, SISLIntpt \* pintpt, int \* jnewpt, int \* jstat )

Definition at line 62 of file sh1787.c.



## 30.2797 sisl/src/sh1790.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh1790.c:
```



### Macros

- `#define SH1790`

### Functions

- `void sh1790 (SISLObject *po1, SISLObject *po2, int itype, double aepsge, int *jstat)`

#### 30.2797.1 Macro Definition Documentation

##### 30.2797.1.1 `#define SH1790`

Definition at line 49 of file `sh1790.c`.

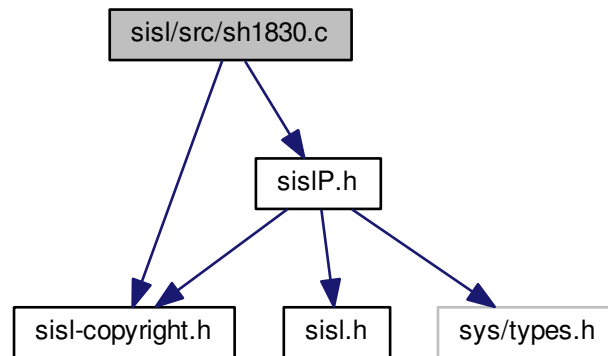
#### 30.2797.2 Function Documentation

##### 30.2797.2.1 `void sh1790 ( SISLObject * po1, SISLObject * po2, int itype, double aepsge, int * jstat )`

Definition at line 57 of file `sh1790.c`.

## 30.2798 sisl/src/sh1830.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh1830.c:
```



### Macros

- `#define SH1830`

### Functions

- `void sh1830 (SISLObject *po1, SISLObject *po2, double aepsge, int *jstat)`

### 30.2798.1 Macro Definition Documentation

#### 30.2798.1.1 `#define SH1830`

Definition at line 49 of file `sh1830.c`.

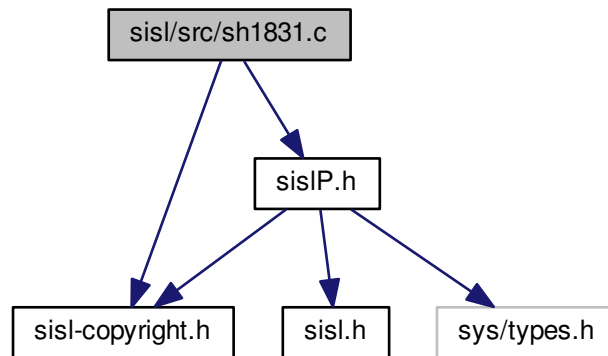
### 30.2798.2 Function Documentation

#### 30.2798.2.1 `void sh1830 ( SISLObject * po1, SISLObject * po2, double aepsge, int * jstat )`

Definition at line 57 of file `sh1830.c`.

## 30.2799 sisl/src/sh1831.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh1831.c:
```



### Macros

- `#define SH1831`

### Functions

- `void sh1831 (SISLCurve *pc1, SISLCurve *pc2, int isign, epoint, enorm, double aepsge, int *jstat)`

#### 30.2799.1 Macro Definition Documentation

##### 30.2799.1.1 `#define SH1831`

Definition at line 42 of file `sh1831.c`.

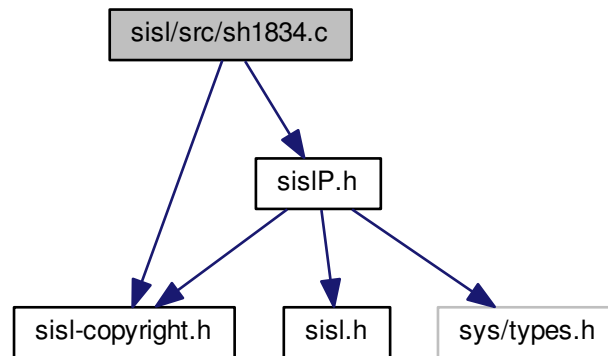
#### 30.2799.2 Function Documentation

##### 30.2799.2.1 `void sh1831 ( SISLCurve * pc1, SISLCurve * pc2, int isign, epoint , enorm , double aepsge, int * jstat )`

Definition at line 51 of file `sh1831.c`.

## 30.2800 sisl/src/sh1834.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh1834.c:
```



### Macros

- `#define SH1834`

### Functions

- `void sh1834 (SISLObject *po1, SISLObject *po2, double aepsge, int idim, edir1, edir2, int *jstat)`

#### 30.2800.1 Macro Definition Documentation

##### 30.2800.1.1 `#define SH1834`

Definition at line 49 of file `sh1834.c`.

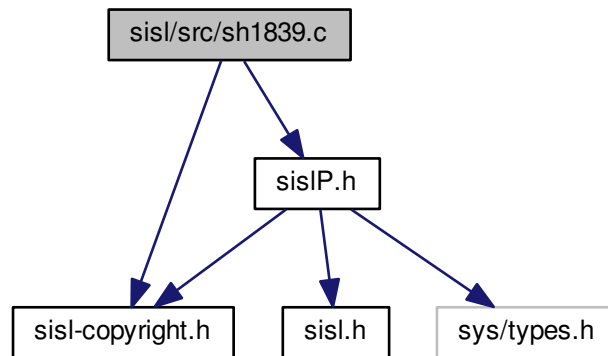
#### 30.2800.2 Function Documentation

##### 30.2800.2.1 `void sh1834 ( SISLObject * po1, SISLObject * po2, double aepsge, int idim, edir1 , edir2 , int * jstat )`

Definition at line 70 of file `sh1834.c`.

## 30.2801 sisl/src/sh1839.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh1839.c:
```



### Macros

- `#define SH1839`

### Functions

- `void sh1839 (SISLObject *po1, SISLObject *po2, double aepsge, int *jstat)`

#### 30.2801.1 Macro Definition Documentation

##### 30.2801.1.1 `#define SH1839`

Definition at line 49 of file `sh1839.c`.

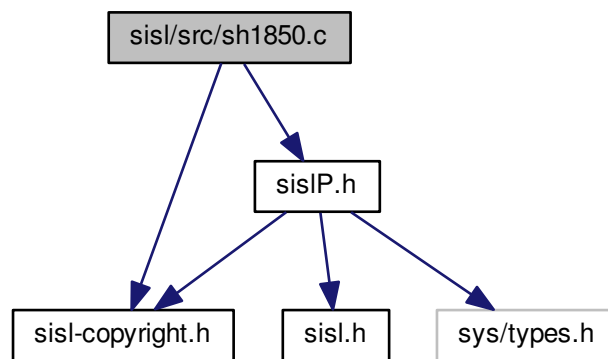
#### 30.2801.2 Function Documentation

##### 30.2801.2.1 `void sh1839 ( SISLObject * po1, SISLObject * po2, double aepsge, int * jstat )`

Definition at line 56 of file `sh1839.c`.

## 30.2802 sisl/src/sh1850.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh1850.c:
```



### Macros

- `#define SH1850`

### Functions

- void `sh1850` (`SISLCurve *pc1`, `epoint`, `enorm`, `int idim`, `double aepsco`, `double aepsge`, `int trackflag`, `int *jtrack`, `SISLTrack ***wtrack`, `int *jpt`, `double **gpar`, `int **pretop`, `int *jcrv`, `SISLIntcurve ***wcurve`, `int *jstat`)

### 30.2802.1 Macro Definition Documentation

#### 30.2802.1.1 #define SH1850

Definition at line 49 of file `sh1850.c`.

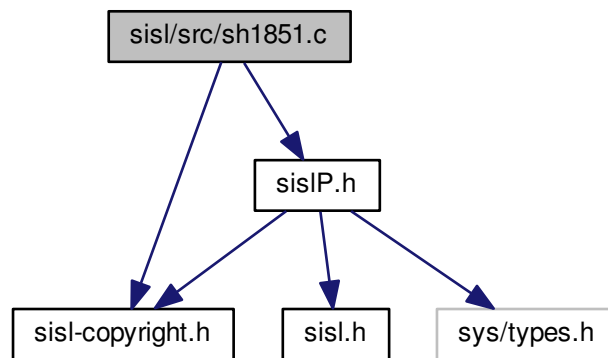
### 30.2802.2 Function Documentation

30.2802.2.1 void `sh1850` ( `SISLCurve * pc1`, `epoint` , `enorm` , `int idim`, `double aepsco`, `double aepsge`, `int trackflag`, `int * jtrack`, `SISLTrack *** wtrack`, `int * jpt`, `double ** gpar`, `int ** pretop`, `int * jcrv`, `SISLIntcurve *** wcurve`, `int * jstat` )

Definition at line 60 of file `sh1850.c`.

## 30.2803 sisl/src/sh1851.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh1851.c:
```



### Macros

- `#define SH1851`

### Functions

- void `sh1851` (`SISLSurf *ps1`, `epoint`, `enorm`, `int idim`, `double aepsco`, `double aepsge`, `int trackflag`, `int *jtrack`, `SISLTrack ***wtrack`, `int *jpt`, `double **gpar`, `int **pretop`, `int *jcrv`, `SISLIntcurve ***wcurve`, `int *jsurf`, `SISLIntsurf ***wsurf`, `int *jstat`)

### 30.2803.1 Macro Definition Documentation

#### 30.2803.1.1 #define SH1851

Definition at line 49 of file sh1851.c.

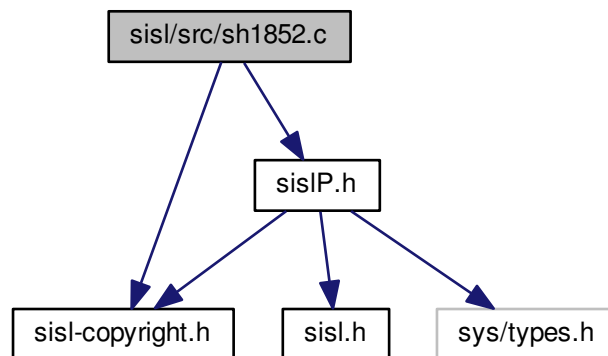
### 30.2803.2 Function Documentation

#### 30.2803.2.1 void sh1851 ( `SISLSurf * ps1`, `epoint` , `enorm` , `int idim`, `double aepsco`, `double aepsge`, `int trackflag`, `int * jtrack`, `SISLTrack *** wtrack`, `int * jpt`, `double ** gpar`, `int ** pretop`, `int * jcrv`, `SISLIntcurve *** wcurve`, `int * jsurf`, `SISLIntsurf *** wsurf`, `int * jstat` )

Definition at line 60 of file sh1851.c.

## 30.2804 sisl/src/sh1852.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh1852.c:
```



### Macros

- #define [SH1852](#)

### Functions

- void [sh1852](#) (SISLSurf \*ps1, ecenter, double radius, int idim, double aepsco, double aepsge, int trackflag, int \*jtrack, SISLTrack \*\*\*wtrack, int \*jpt, double \*\*gpar, int \*\*pretop, int \*jcrv, SISLIntcurve \*\*\*wcurve, int \*jsurf, SISLIntsurf \*\*\*wsurf, int \*jstat)

### 30.2804.1 Macro Definition Documentation

#### 30.2804.1.1 #define SH1852

Definition at line 49 of file sh1852.c.

### 30.2804.2 Function Documentation

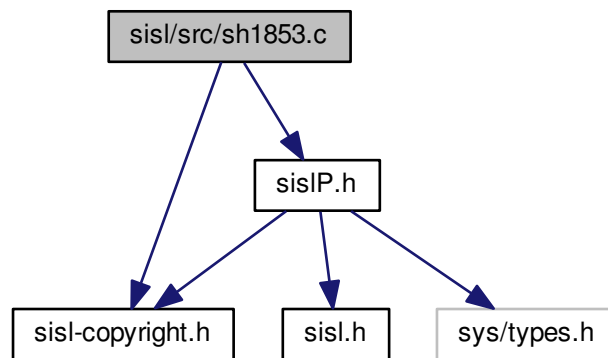
30.2804.2.1 void [sh1852](#) ( SISLSurf \* *ps1*, ecenter , double *radius*, int *idim*, double *aepsco*, double *aepsge*, int *trackflag*, int \* *jtrack*, SISLTrack \*\*\* *wtrack*, int \* *jpt*, double \*\* *gpar*, int \*\* *pretop*, int \* *jcrv*, SISLIntcurve \*\*\* *wcurve*, int \* *jsurf*, SISLIntsurf \*\*\* *wsurf*, int \* *jstat* )

Definition at line 61 of file sh1852.c.



## 30.2805 sisl/src/sh1853.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh1853.c:
```



### Macros

- `#define SH1853`

### Functions

- void `sh1853` (SISLSurf \*ps1, epoint, edirec, double aradius, int idim, double aepsco, double aepsge, int trackflag, int \*jtrack, SISLTrack \*\*\*wtrack, int \*jpt, double \*\*gpar, int \*\*pretop, int \*jcrv, SISLIntcurve \*\*\*wcurve, int \*jsurf, SISLIntsurf \*\*\*wsurf, int \*jstat)

### 30.2805.1 Macro Definition Documentation

#### 30.2805.1.1 #define SH1853

Definition at line 49 of file sh1853.c.

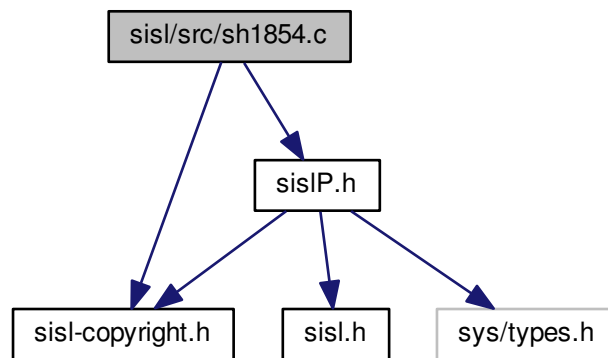
### 30.2805.2 Function Documentation

#### 30.2805.2.1 void sh1853 ( SISLSurf \* ps1, epoint , edirec , double aradius, int idim, double aepsco, double aepsge, int trackflag, int \* jtrack, SISLTrack \*\*\* wtrack, int \* jpt, double \*\* gpar, int \*\* pretop, int \* jcrv, SISLIntcurve \*\*\* wcurve, int \* jsurf, SISLIntsurf \*\*\* wsurf, int \* jstat )

Definition at line 61 of file sh1853.c.

## 30.2806 sisl/src/sh1854.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh1854.c:
```



### Macros

- `#define SH1854`

### Functions

- void `sh1854` (`SISLSurf *ps1`, `etop`, `eaxis`, `econe`, `int idim`, `double aepsco`, `double aepsge`, `int trackflag`, `int *jtrack`, `SISLTrack ***wtrack`, `int *jpt`, `double **gpar`, `int **pretop`, `int *jcrv`, `SISLIntcurve ***wcurve`, `int *jsurf`, `SISLIntsurf ***wsurf`, `int *jstat`)

### 30.2806.1 Macro Definition Documentation

#### 30.2806.1.1 #define SH1854

Definition at line 49 of file sh1854.c.

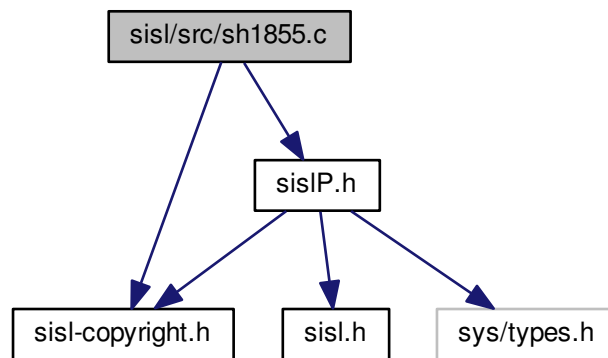
### 30.2806.2 Function Documentation

30.2806.2.1 void `sh1854` ( `SISLSurf * ps1`, `etop` , `eaxis` , `econe` , `int idim`, `double aepsco`, `double aepsge`, `int trackflag`, `int * jtrack`, `SISLTrack *** wtrack`, `int * jpt`, `double ** gpar`, `int ** pretop`, `int * jcrv`, `SISLIntcurve *** wcurve`, `int * jsurf`, `SISLIntsurf *** wsurf`, `int * jstat` )

Definition at line 61 of file sh1854.c.

## 30.2807 sisl/src/sh1855.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh1855.c:
```



### Macros

- `#define SH1855`

### Functions

- void `sh1855` (`SISLSurf *ps1`, `ecentr`, `double aradius`, `enorm`, `int idim`, `double aepsco`, `double aepsge`, `int trackflag`, `int *jtrack`, `SISLTrack ***wtrack`, `int *jpt`, `double **gpar`, `int **pretop`, `int *jcrv`, `SISLIntcurve ***wcurve`, `int *jstat`)

### 30.2807.1 Macro Definition Documentation

#### 30.2807.1.1 #define SH1855

Definition at line 49 of file sh1855.c.

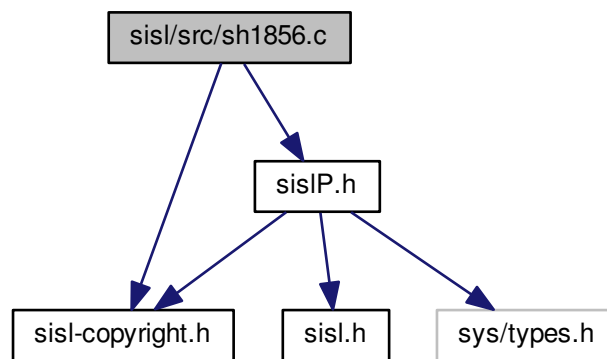
### 30.2807.2 Function Documentation

#### 30.2807.2.1 void sh1855 ( `SISLSurf * ps1`, `ecentr` , `double aradius`, `enorm` , `int idim`, `double aepsco`, `double aepsge`, `int trackflag`, `int * jtrack`, `SISLTrack *** wtrack`, `int * jpt`, `double ** gpar`, `int ** pretop`, `int * jcrv`, `SISLIntcurve *** wcurve`, `int * jstat` )

Definition at line 60 of file sh1855.c.

## 30.2808 sisl/src/sh1856.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh1856.c:
```



### Macros

- #define [SH1856](#)

### Functions

- void [sh1856](#) ([SISLSurf](#) \*ps1, epoint, edir, int idim, [double](#) aepsco, [double](#) aepsge, int trackflag, int \*jtrack, [SISLTrack](#) \*\*\*wtrack, int \*jpt, [double](#) \*\*gpar, int \*\*pretop, int \*jcrv, [SISLIntcurve](#) \*\*\*wcurve, int \*jstat)

### 30.2808.1 Macro Definition Documentation

#### 30.2808.1.1 #define SH1856

Definition at line 49 of file sh1856.c.

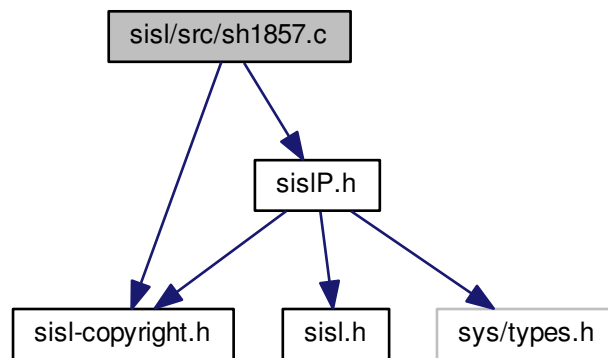
### 30.2808.2 Function Documentation

30.2808.2.1 void [sh1856](#) ( [SISLSurf](#) \* ps1, epoint , edir , int idim, [double](#) aepsco, [double](#) aepsge, int trackflag, int \* jtrack, [SISLTrack](#) \*\*\* wtrack, int \* jpt, [double](#) \*\* gpar, int \*\* pretop, int \* jcrv, [SISLIntcurve](#) \*\*\* wcurve, int \* jstat )

Definition at line 60 of file sh1856.c.

## 30.2809 sisl/src/sh1857.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh1857.c:
```



### Macros

- `#define SH1857`

### Functions

- void `sh1857` (SISLCurve \*pc1, SISLCurve \*pc2, double aepsco, double aepsge, int trackflag, int \*jtrack, SISLTrack \*\*\*wtrack, int \*jpt, double \*\*gpar1, double \*\*gpar2, int \*\*pretop, int \*jcrv, SISLIntcurve \*\*\*wcurve, int \*jstat)

### 30.2809.1 Macro Definition Documentation

#### 30.2809.1.1 #define SH1857

Definition at line 49 of file sh1857.c.

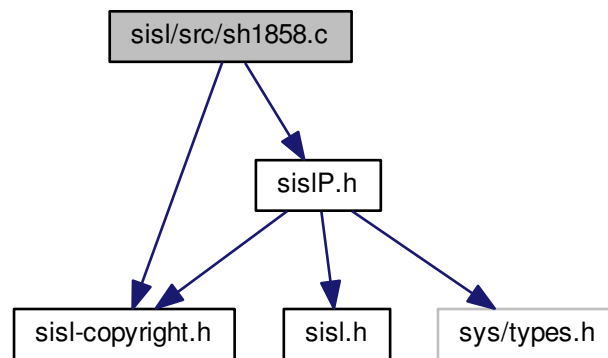
### 30.2809.2 Function Documentation

#### 30.2809.2.1 void sh1857 ( SISLCurve \* pc1, SISLCurve \* pc2, double aepsco, double aepsge, int trackflag, int \* jtrack, SISLTrack \*\*\* wtrack, int \* jpt, double \*\* gpar1, double \*\* gpar2, int \*\* pretop, int \* jcrv, SISLIntcurve \*\*\* wcurve, int \* jstat )

Definition at line 59 of file sh1857.c.

## 30.2810 sisl/src/sh1858.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh1858.c:
```



### Macros

- `#define SH1858`

### Functions

- void `sh1858` (SISLSurf \*ps1, SISLCurve \*pc1, double aepsco, double aepsge, int trackflag, int \*jtrack, SISLTrack \*\*\*wtrack, int \*jpt, double \*\*gpar1, double \*\*gpar2, int \*\*pretop, int \*jcrv, SISLIntcurve \*\*\*wcurve, int \*jstat)

#### 30.2810.1 Macro Definition Documentation

##### 30.2810.1.1 #define SH1858

Definition at line 49 of file sh1858.c.

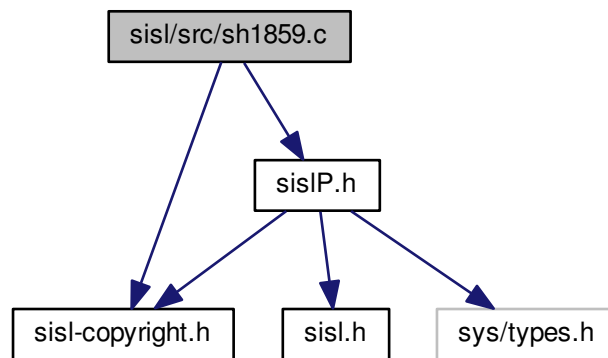
#### 30.2810.2 Function Documentation

##### 30.2810.2.1 void sh1858 ( SISLSurf \* ps1, SISLCurve \* pc1, double aepsco, double aepsge, int trackflag, int \* jtrack, SISLTrack \*\*\* wtrack, int \* jpt, double \*\* gpar1, double \*\* gpar2, int \*\* pretop, int \* jcrv, SISLIntcurve \*\*\* wcurve, int \* jstat )

Definition at line 59 of file sh1858.c.

## 30.2811 sisl/src/sh1859.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh1859.c:
```



### Macros

- `#define SH1859`

### Functions

- void `sh1859` (`SISLSurf *ps1`, `SISLSurf *ps2`, `double aepsco`, `double aepsge`, `int trackflag`, `int *jtrack`, `SISLTrack ***wtrack`, `int *jpt`, `double **gpar1`, `double **gpar2`, `int **pretop`, `int *jcrv`, `SISLIntcurve ***wcurve`, `int *jsurf`, `SISLIntsurf ***wsurf`, `int *jstat`)

### 30.2811.1 Macro Definition Documentation

#### 30.2811.1.1 #define SH1859

Definition at line 49 of file sh1859.c.

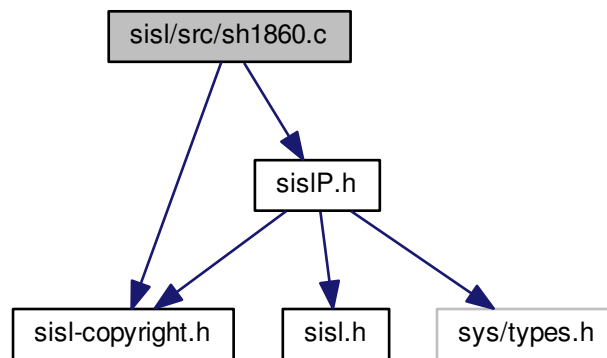
### 30.2811.2 Function Documentation

#### 30.2811.2.1 void sh1859 ( `SISLSurf * ps1`, `SISLSurf * ps2`, `double aepsco`, `double aepsge`, `int trackflag`, `int * jtrack`, `SISLTrack *** wtrack`, `int * jpt`, `double ** gpar1`, `double ** gpar2`, `int ** pretop`, `int * jcrv`, `SISLIntcurve *** wcurve`, `int * jsurf`, `SISLIntsurf *** wsurf`, `int * jstat` )

Definition at line 60 of file sh1859.c.

## 30.2812 sisl/src/sh1860.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh1860.c:
```



### Macros

- `#define SH1860`

### Functions

- void `sh1860` (SISLSurf \*ps, eview, int idim, double aepsco, double aepsge, int trackflag, int \*jtrack, SISLTrack \*\*\*wtrack, int \*jpt, double \*\*gpar, int \*\*pretop, int \*jcrv, SISLIntcurve \*\*\*wcurve, int \*jsurf, SISLIntsurf \*\*\*wsurf, int \*jstat)

### 30.2812.1 Macro Definition Documentation

#### 30.2812.1.1 #define SH1860

Definition at line 49 of file sh1860.c.

### 30.2812.2 Function Documentation

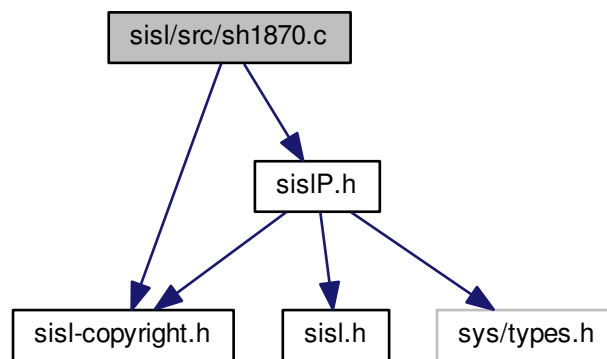
#### 30.2812.2.1 void sh1860 ( SISLSurf \* ps, eview , int idim, double aepsco, double aepsge, int trackflag, int \* jtrack, SISLTrack \*\*\* wtrack, int \* jpt, double \*\* gpar, int \*\* pretop, int \* jcrv, SISLIntcurve \*\*\* wcurve, int \* jsurf, SISLIntsurf \*\*\* wsurf, int \* jstat )

Definition at line 59 of file sh1860.c.



## 30.2813 sisl/src/sh1870.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh1870.c:
```



### Macros

- `#define SH1870`

### Functions

- `void sh1870 (SISLSurf *ps1, double *pt1, int idim, double aepsco, double aepsge, int trackflag, int *jtrack, SISLTrack ***wtrack, int *jpt, double **gpar1, int **pretop, int *jcrv, SISLIntcurve ***wcurve, int *jstat)`

#### 30.2813.1 Macro Definition Documentation

##### 30.2813.1.1 `#define SH1870`

Definition at line 49 of file `sh1870.c`.

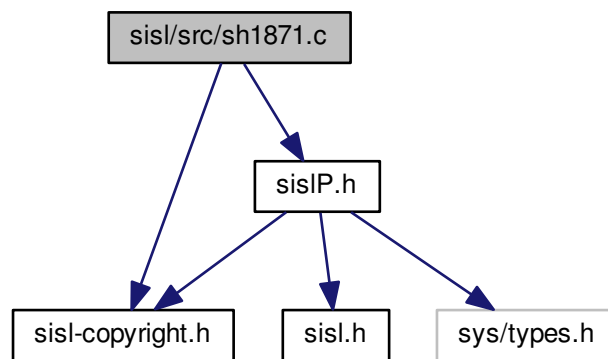
#### 30.2813.2 Function Documentation

30.2813.2.1 `void sh1870 ( SISLSurf * ps1, double * pt1, int idim, double aepsco, double aepsge, int trackflag, int * jtrack, SISLTrack *** wtrack, int * jpt, double ** gpar1, int ** pretop, int * jcrv, SISLIntcurve *** wcurve, int * jstat )`

Definition at line 59 of file `sh1870.c`.

## 30.2814 sisl/src/sh1871.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh1871.c:
```



### Macros

- #define [SH1871](#)

### Functions

- void [sh1871](#) (SISLCurve \*pc1, double \*pt1, int idim, double aepsco, double aepsge, int trackflag, int \*jtrack, SISLTrack \*\*\*wtrack, int \*jpt, double \*\*gpar1, int \*\*pretop, int \*jcrv, SISLIntcurve \*\*\*wcurve, int \*jstat)

### 30.2814.1 Macro Definition Documentation

#### 30.2814.1.1 #define SH1871

Definition at line 49 of file sh1871.c.

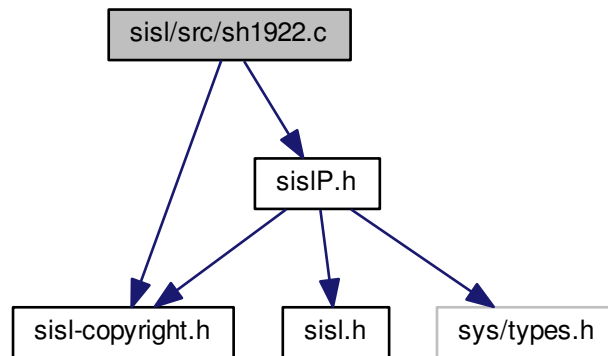
### 30.2814.2 Function Documentation

30.2814.2.1 void [sh1871](#) ( SISLCurve \* pc1, double \* pt1, int idim, double aepsco, double aepsge, int trackflag, int \* jtrack, SISLTrack \*\*\* wtrack, int \* jpt, double \*\* gpar1, int \*\* pretop, int \* jcrv, SISLIntcurve \*\*\* wcurve, int \* jstat )

Definition at line 59 of file sh1871.c.

## 30.2815 sisl/src/sh1922.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh1922.c:
```



### Macros

- `#define SH1922`

### Functions

- `void sh1922 (et, int im, int ik, etau, int in, ea, nfirst, nlast, int *jstat)`

#### 30.2815.1 Macro Definition Documentation

##### 30.2815.1.1 `#define SH1922`

Definition at line 48 of file `sh1922.c`.

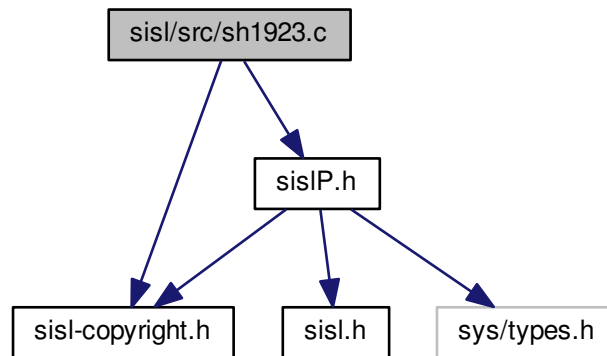
#### 30.2815.2 Function Documentation

##### 30.2815.2.1 `void sh1922 ( et , int im , int ik , etau , int in , ea , nfirst , nlast , int * jstat )`

Definition at line 57 of file `sh1922.c`.

## 30.2816 sisl/src/sh1923.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh1923.c:
```



### Macros

- `#define SH1923`

### Functions

- `void sh1923 (double *ea, int in, int ik, int *nstart, int *jstat)`

### 30.2816.1 Macro Definition Documentation

#### 30.2816.1.1 `#define SH1923`

Definition at line 48 of file `sh1923.c`.

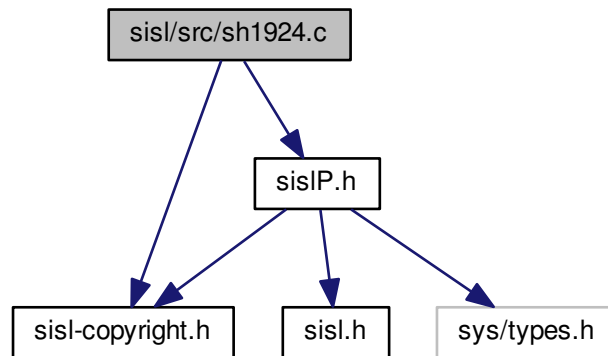
### 30.2816.2 Function Documentation

#### 30.2816.2.1 `void sh1923 ( double * ea, int in, int ik, int * nstart, int * jstat )`

Definition at line 56 of file `sh1923.c`.

## 30.2817 sisl/src/sh1924.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh1924.c:
```



### Macros

- `#define SH1924`

### Functions

- `void sh1924 (double *ea, double *eb, int in, int ik, int idim, int *nstart, int *jstat)`

#### 30.2817.1 Macro Definition Documentation

##### 30.2817.1.1 `#define SH1924`

Definition at line 48 of file `sh1924.c`.

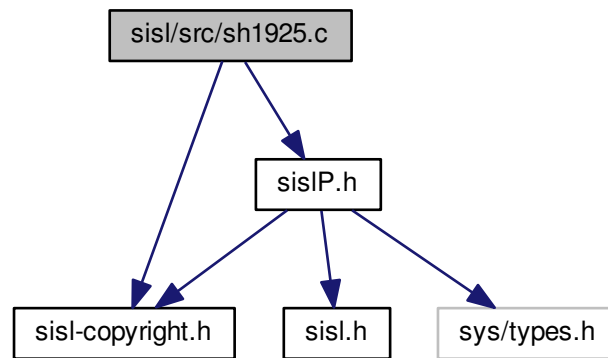
#### 30.2817.2 Function Documentation

##### 30.2817.2.1 `void sh1924 ( double * ea, double * eb, int in, int ik, int idim, int * nstart, int * jstat )`

Definition at line 57 of file `sh1924.c`.

## 30.2818 sisl/src/sh1925.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh1925.c:
```



### Macros

- `#define SH1925`

### Functions

- void `sh1925` (`SISLCurve *pc1`, `SISLCurve *pc2`, `int idim`, `ea`, `nstart`, `nstop`, `emxerr`, `el2err`, `int ilend`, `int irend`, `int *jstat`)

#### 30.2818.1 Macro Definition Documentation

##### 30.2818.1.1 `#define SH1925`

Definition at line 48 of file `sh1925.c`.

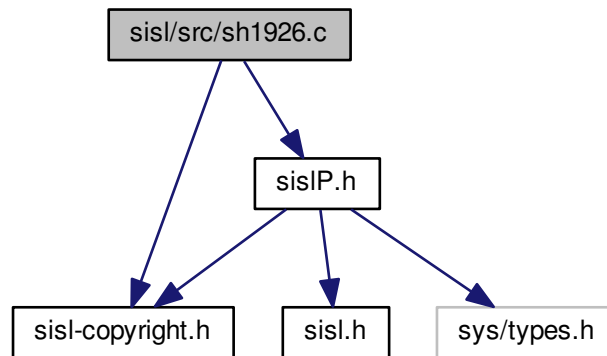
#### 30.2818.2 Function Documentation

##### 30.2818.2.1 void `sh1925` ( `SISLCurve * pc1`, `SISLCurve * pc2`, `int idim`, `ea`, `nstart`, `nstop`, `emxerr`, `el2err`, `int ilend`, `int irend`, `int * jstat` )

Definition at line 58 of file `sh1925.c`.

## 30.2819 sisl/src/sh1926.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh1926.c:
```



### Macros

- `#define SH1926`

### Functions

- void `sh1926` (etau, int ik, int in, int idim, et, ed, int im, ea, nfirst, nlast, eb, n2sta, ec, int \*jstat)

### 30.2819.1 Macro Definition Documentation

#### 30.2819.1.1 #define SH1926

Definition at line 48 of file `sh1926.c`.

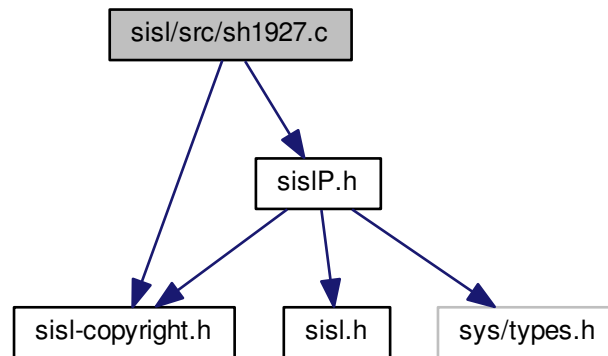
### 30.2819.2 Function Documentation

#### 30.2819.2.1 void `sh1926` ( etau , int ik , int in , int idim , et , ed , int im , ea , nfirst , nlast , eb , n2sta , ec , int \* jstat )

Definition at line 58 of file `sh1926.c`.

## 30.2820 sisl/src/sh1927.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh1927.c:
```



### Macros

- `#define SH1927`

### Functions

- `void sh1927 ( etau , int ik , int in , int idim , SISLCurve * pcurve , int ilend , int irend , ec , int * jstat )`

### 30.2820.1 Macro Definition Documentation

#### 30.2820.1.1 `#define SH1927`

Definition at line 48 of file `sh1927.c`.

### 30.2820.2 Function Documentation

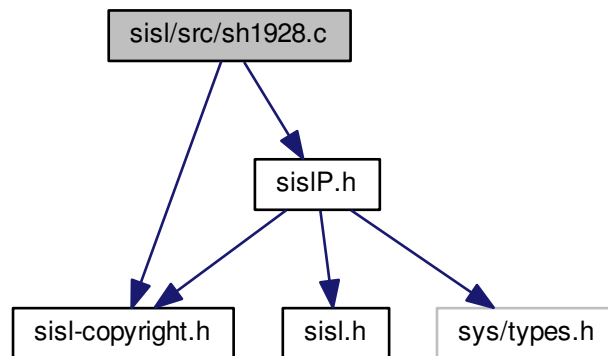
#### 30.2820.2.1 `void sh1927 ( etau , int ik , int in , int idim , SISLCurve * pcurve , int ilend , int irend , ec , int * jstat )`

Definition at line 57 of file `sh1927.c`.



## 30.2821 sisl/src/sh1928.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh1928.c:
```



### Macros

- `#define SH1928`

### Functions

- `void sh1928 ( etau, int ik, int in, int idim, et, ed, int im, int ilend, int irend, ea, int inh, nfirst, nlast, eb, ec, n2sta, int *jstat )`

#### 30.2821.1 Macro Definition Documentation

##### 30.2821.1.1 `#define SH1928`

Definition at line 48 of file `sh1928.c`.

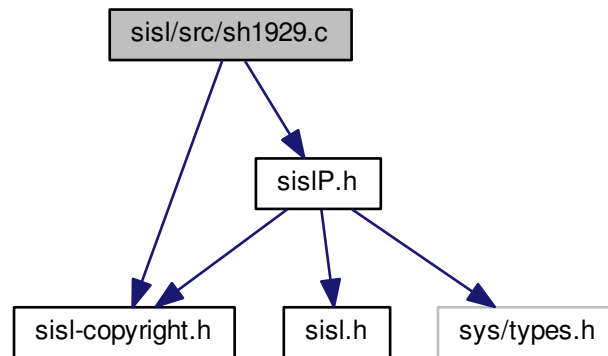
#### 30.2821.2 Function Documentation

##### 30.2821.2.1 `void sh1928 ( etau, int ik, int in, int idim, et, ed, int im, int ilend, int irend, ea, int inh, nfirst, nlast, eb, ec, n2sta, int *jstat )`

Definition at line 59 of file `sh1928.c`.

## 30.2822 sisl/src/sh1929.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh1929.c:
```



### Macros

- `#define SH1929`

### Functions

- `void sh1929 ( etau , int in , int ik , int imu , et , int im , int ij , eah , int *jmuprm , int *jnu , int *jstat )`

### 30.2822.1 Macro Definition Documentation

#### 30.2822.1.1 `#define SH1929`

Definition at line 48 of file `sh1929.c`.

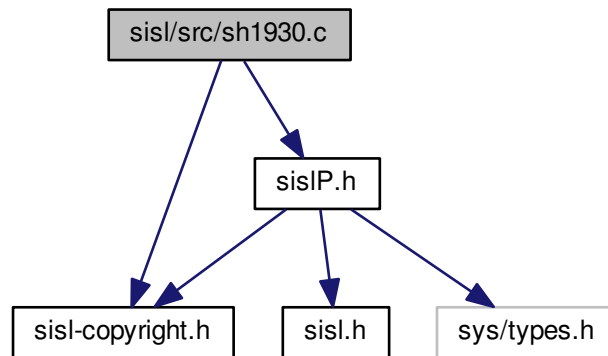
### 30.2822.2 Function Documentation

#### 30.2822.2.1 `void sh1929 ( etau , int in , int ik , int imu , et , int im , int ij , eah , int * jmuprm , int * jnu , int * jstat )`

Definition at line 57 of file `sh1929.c`.

## 30.2823 sisl/src/sh1930.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh1930.c:
```



### Macros

- `#define SH1930`

### Functions

- `void sh1930 (ea, nfirst, nlast, ed, ec, int ik, int in, int im, int idim, int ilend, int irend, int *jstat)`

#### 30.2823.1 Macro Definition Documentation

##### 30.2823.1.1 `#define SH1930`

Definition at line 48 of file `sh1930.c`.

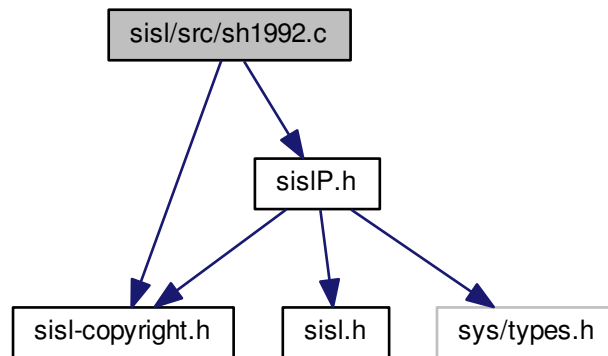
#### 30.2823.2 Function Documentation

##### 30.2823.2.1 `void sh1930 ( ea , nfirst , nlast , ed , ec , int ik , int in , int im , int idim , int ilend , int irend , int * jstat )`

Definition at line 57 of file `sh1930.c`.

## 30.2824 sisl/src/sh1992.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh1992.c:
```



### Macros

- #define [SH1992](#)

### Functions

- void [sh1992](#) (SISLObject \*po, int itype, double aepsge, int \*jstat)
- void [sh1992cu](#) (SISLCurve \*pc, int itype, double aepsge, int \*jstat)
- void [sh1992su](#) (SISLSurf \*ps, int itype, double aepsge, int \*jstat)

### 30.2824.1 Macro Definition Documentation

#### 30.2824.1.1 #define SH1992

Definition at line 49 of file sh1992.c.

### 30.2824.2 Function Documentation

#### 30.2824.2.1 void sh1992 ( SISLObject \* po, int itype, double aepsge, int \* jstat )

Definition at line 71 of file sh1992.c.

30.2824.2.2 void sh1992cu ( SISLCurve \* pc, int itype, double aepsge, int \* jstat )

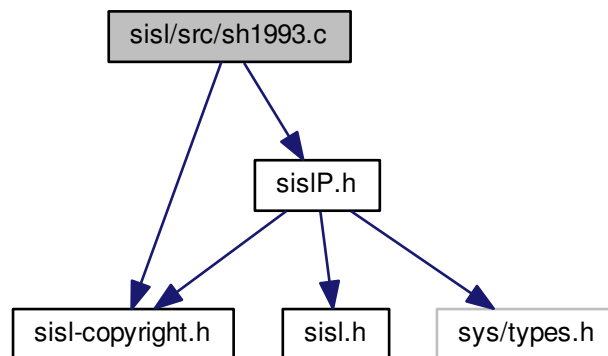
Definition at line 301 of file sh1992.c.

30.2824.2.3 void sh1992su ( SISLSurf \* ps, int itype, double aepsge, int \* jstat )

Definition at line 429 of file sh1992.c.

## 30.2825 sisl/src/sh1993.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh1993.c:
```



### Macros

- `#define SH1993`

### Functions

- void `sh1993` (SISLCurve \*c1, double aepsge, int \*jstat)

## 30.2825.1 Macro Definition Documentation

30.2825.1.1 `#define SH1993`

Definition at line 48 of file sh1993.c.

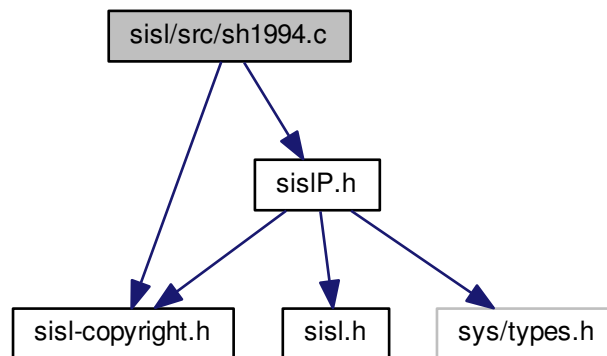
## 30.2825.2 Function Documentation

30.2825.2.1 void sh1993 ( SISLCurve \* c1, double aepsge, int \* jstat )

Definition at line 55 of file sh1993.c.

## 30.2826 sisl/src/sh1994.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh1994.c:
```



### Macros

- `#define SH1994`

### Functions

- void `sh1994` (SISLSurf \*s1, double aepsge, int \*jstat)

## 30.2826.1 Macro Definition Documentation

30.2826.1.1 `#define SH1994`

Definition at line 49 of file sh1994.c.

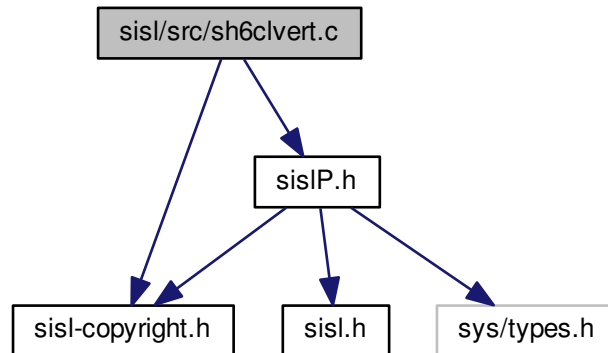
## 30.2826.2 Function Documentation

30.2826.2.1 void sh1994 ( SISLSurf \* s1, double aepsge, int \* jstat )

Definition at line 56 of file sh1994.c.

## 30.2827 sisl/src/sh6clvert.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh6clvert.c:
```



### Macros

- #define [SH6CLOSEVERT](#)

### Functions

- void [sh6closevert](#) (SISLCurve \*pcurve, SISLSurf \*psurf, double \*cpar1, epar2)

## 30.2827.1 Macro Definition Documentation

30.2827.1.1 #define SH6CLOSEVERT

Definition at line 47 of file sh6clvert.c.

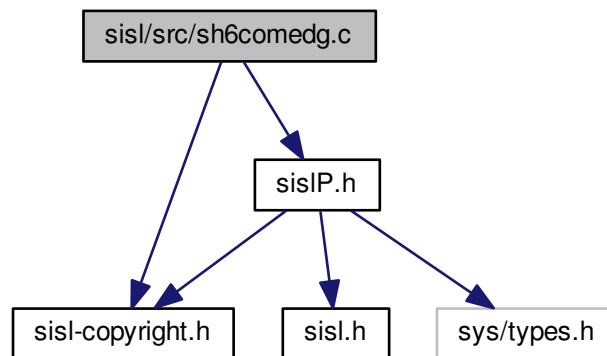
## 30.2827.2 Function Documentation

30.2827.2.1 void sh6closevert ( SISLCurve \* pcurve, SISLSurf \* psurf, double \* cpar1, epar2 )

Definition at line 57 of file sh6clvert.c.

## 30.2828 sisl/src/sh6comedg.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh6comedg.c:
```



### Macros

- #define `SH6COMEDG`

### Functions

- void `sh6comedg` (SISLObject \*po1, SISLObject \*po2, SISLIntpt \*pt1, SISLIntpt \*pt2, int \*jstat)

## 30.2828.1 Macro Definition Documentation

30.2828.1.1 #define SH6COMEDG

Definition at line 49 of file sh6comedg.c.



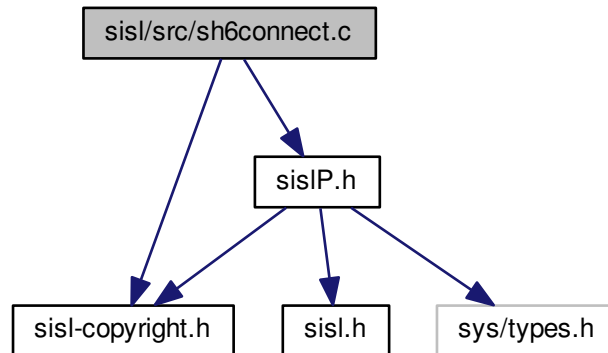
## 30.2828.2 Function Documentation

30.2828.2.1 void sh6comedg ( SISLObject \* po1, SISLObject \* po2, SISLIntpt \* pt1, SISLIntpt \* pt2, int \* jstat )

Definition at line 59 of file sh6comedg.c.

## 30.2829 sisl/src/sh6connect.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh6connect.c:
```



### Macros

- #define [SH6CONNECT](#)

### Functions

- void [sh6connect](#) (SISLIntpt \*pt1, SISLIntpt \*pt2, int \*jstat)

## 30.2829.1 Macro Definition Documentation

30.2829.1.1 #define SH6CONNECT

Definition at line 49 of file sh6connect.c.

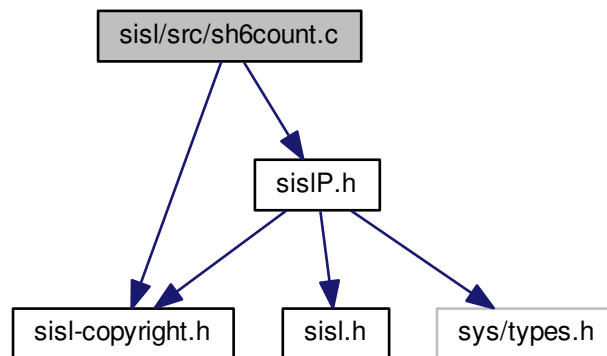
## 30.2829.2 Function Documentation

30.2829.2.1 void sh6connect ( SISLIntpt \* pt1, SISLIntpt \* pt2, int \* jstat )

Definition at line 59 of file sh6connect.c.

## 30.2830 sisl/src/sh6count.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh6count.c:
```



### Macros

- #define `SH6COUNT`

### Functions

- int `sh6count` (SISLIntpt \*pt, int \*jstat)

## 30.2830.1 Macro Definition Documentation

30.2830.1.1 #define SH6COUNT

Definition at line 49 of file sh6count.c.

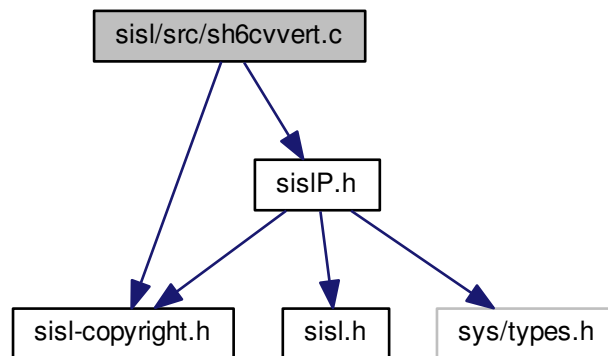
## 30.2830.2 Function Documentation

### 30.2830.2.1 int sh6count ( SISLIntpt \* pt, int \* jstat )

Definition at line 58 of file sh6count.c.

## 30.2831 sisl/src/sh6cvert.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh6cvert.c:
```



## Macros

- `#define SH6CVVERT`

## Functions

- void `sh6cvert` (SISLCurve \*pc1, SISLCurve \*pc2, double \*cpar1, double \*cpar2)

## 30.2831.1 Macro Definition Documentation

### 30.2831.1.1 #define SH6CVVERT

Definition at line 41 of file sh6cvert.c.

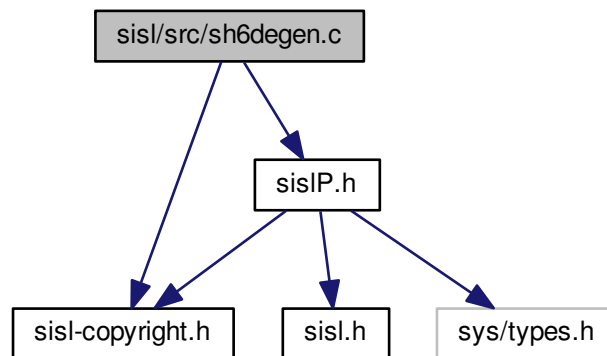
## 30.2831.2 Function Documentation

30.2831.2.1 void sh6cvvert ( SISLCurve \* pc1, SISLCurve \* pc2, double \* cpar1, double \* cpar2 )

Definition at line 50 of file sh6cvvert.c.

## 30.2832 sisl/src/sh6degen.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh6degen.c:
```



### Macros

- #define [SH6DEGEN](#)

### Functions

- void [sh6degen](#) (SISLObject \*po1, SISLObject \*po2, SISLIntdat \*\*pintdat, double aepsge, int \*jstat)

## 30.2832.1 Macro Definition Documentation

30.2832.1.1 #define SH6DEGEN

Definition at line 49 of file sh6degen.c.

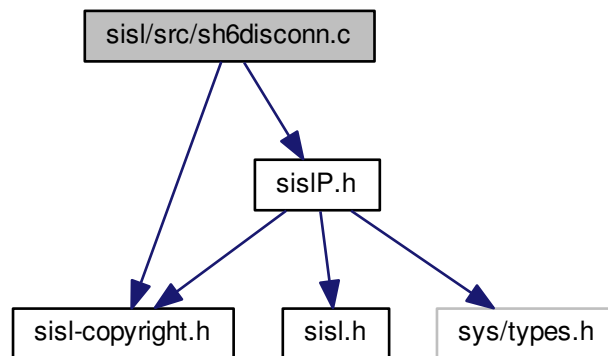
## 30.2832.2 Function Documentation

30.2832.2.1 void sh6degen ( SISLObject \* *po1*, SISLObject \* *po2*, SISLIntdat \*\* *pinmdat*, double *aepsge*, int \* *jstat* )

Definition at line 65 of file sh6degen.c.

## 30.2833 sisl/src/sh6disconn.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh6disconn.c:
```



### Macros

- #define `SH6DISCONNECT`

### Functions

- void `sh6disconnect` (SISLIntpt \**pt1*, SISLIntpt \**pt2*, int \**jstat*)

## 30.2833.1 Macro Definition Documentation

30.2833.1.1 #define SH6DISCONNECT

Definition at line 49 of file sh6disconn.c.

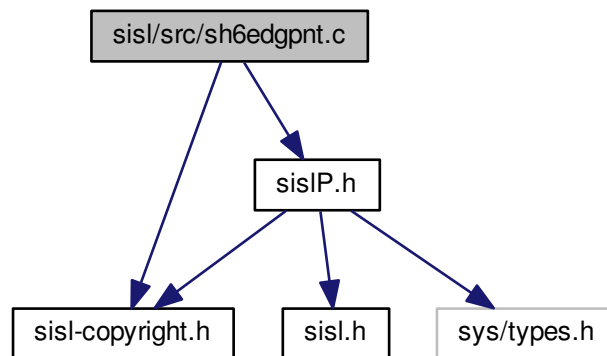
### 30.2833.2 Function Documentation

30.2833.2.1 void sh6disconnect ( SISLIntpt \* *pt1*, SISLIntpt \* *pt2*, int \* *jstat* )

Definition at line 57 of file sh6disconn.c.

### 30.2834 sisl/src/sh6edgpnt.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh6edgpnt.c:
```



#### Macros

- #define [SH6EDGPOINT](#)

#### Functions

- void [sh6edgpoint](#) (vedge, SISLIntpt \*\*\*wintpt, int \*jnum, int \*jstat)

### 30.2834.1 Macro Definition Documentation

30.2834.1.1 #define SH6EDGPOINT

Definition at line 49 of file sh6edgpnt.c.

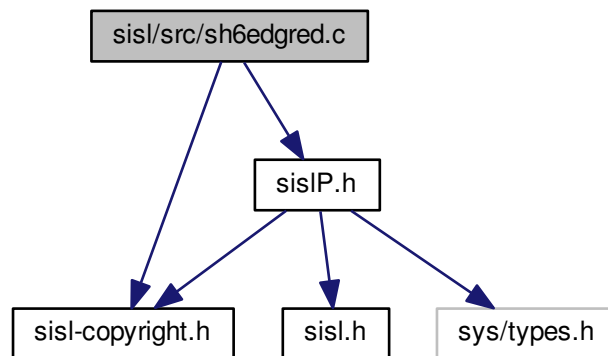
## 30.2834.2 Function Documentation

30.2834.2.1 void sh6edgpoint ( vedge , SISLIntpt \*\*\* wintpt, int \* jnum, int \* jstat )

Definition at line 57 of file sh6edgpnt.c.

## 30.2835 sisl/src/sh6edgred.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh6edgred.c:
```



### Macros

- #define [SH6EDGRED](#)

### Functions

- void [sh6edgred](#) (SISLObject \*po1, SISLObject \*po2, SISLIntdat \*pintdat, int \*jstat)

## 30.2835.1 Macro Definition Documentation

30.2835.1.1 #define SH6EDGRED

Definition at line 49 of file sh6edgred.c.

### 30.2835.2 Function Documentation

30.2835.2.1 void sh6edgred ( SISLObject \* *po1*, SISLObject \* *po2*, SISLIntdat \* *pintdat*, int \* *jstat* )

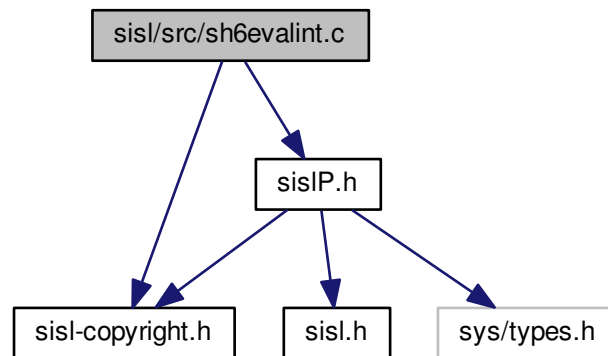
Definition at line 63 of file sh6edgred.c.

### 30.2836 sisl/src/sh6evalint.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for sh6evalint.c:



#### Macros

- #define [SH6EVALINT](#)

#### Functions

- void [sh6evalint](#) (SISLObject \**ob1*, SISLObject \**ob2*, eimpli, int *ideg*, SISLIntpt \**pt*, double *aepsge*, curve\_3d, curve\_2d\_1, curve\_2d\_2, int \**jstat*)

### 30.2836.1 Macro Definition Documentation

30.2836.1.1 #define SH6EVALINT

Definition at line 49 of file sh6evalint.c.



## 30.2836.2 Function Documentation

30.2836.2.1 void sh6evalint ( SISLObject \* ob1, SISLObject \* ob2, eimpli , int ideg, SISLIntpt \* pt, double aepsge, curve\_3d , curve\_2d\_1 , curve\_2d\_2 , int \* jstat )

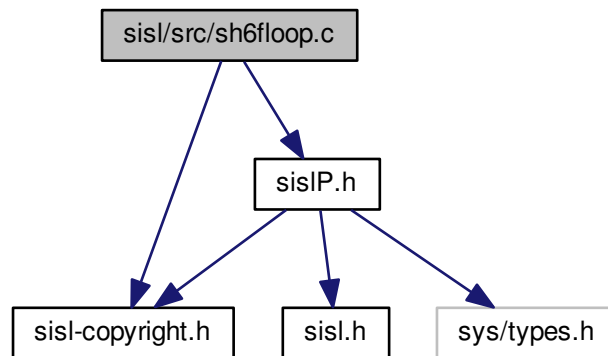
Definition at line 61 of file sh6evalint.c.

## 30.2837 sisl/src/sh6floop.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for sh6floop.c:



### Macros

- #define `SH6FLOOP`

### Functions

- void `sh6floop` (vedgept, int inum, int \*jpt, int \*jstat)

## 30.2837.1 Macro Definition Documentation

30.2837.1.1 #define SH6FLOOP

Definition at line 49 of file sh6floop.c.

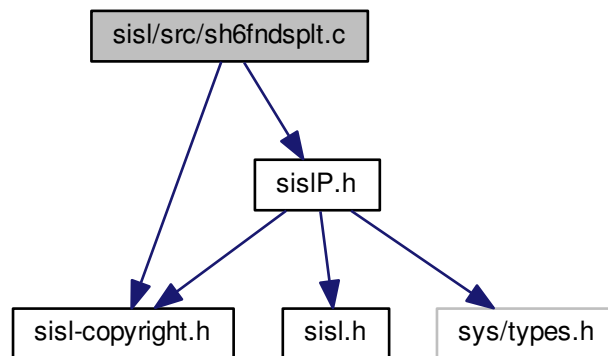
## 30.2837.2 Function Documentation

30.2837.2.1 void sh6floop ( vedgept , int inum, int \* jpt, int \* jstat )

Definition at line 58 of file sh6floop.c.

## 30.2838 sisl/src/sh6fndsplt.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh6fndsplt.c:
```



### Macros

- #define [SH6FINDSPLIT](#)

### Functions

- void [sh6findsplit](#) (SISLSurf \*ps1, SISLSurf \*ps2, double aepsge, int \*jstat)

## 30.2838.1 Macro Definition Documentation

30.2838.1.1 #define SH6FINDSPLIT

Definition at line 49 of file sh6fndsplt.c.

## 30.2838.2 Function Documentation

30.2838.2.1 void sh6findsplit ( SISLSurf \* *ps1*, SISLSurf \* *ps2*, double *aepsge*, int \* *jstat* )

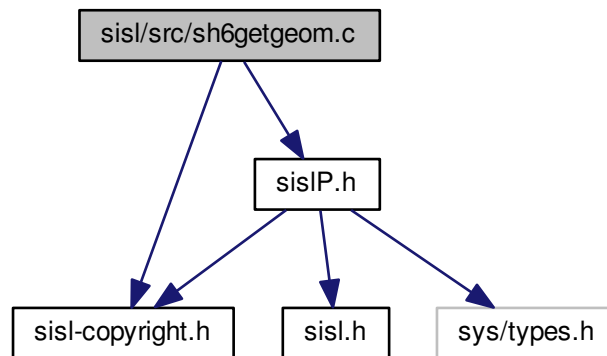
Definition at line 68 of file sh6fndsplt.c.

## 30.2839 sisl/src/sh6getgeom.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for sh6getgeom.c:



### Macros

- #define [SH6GETGEOM](#)

### Functions

- void [sh6getgeom](#) (SISLObject \**ob*, int *obnr*, SISLIntpt \**pt*, double \*\**geom*, double \*\**norm*, double *aepsge*, int \**jstat*)

## 30.2839.1 Macro Definition Documentation

30.2839.1.1 #define SH6GETGEOM

Definition at line 49 of file sh6getgeom.c.

## 30.2839.2 Function Documentation

30.2839.2.1 void sh6getgeom ( SISLObject \* ob, int obnr, SISLIntpt \* pt, double \*\* geom, double \*\* norm, double aepsge, int \* jstat )

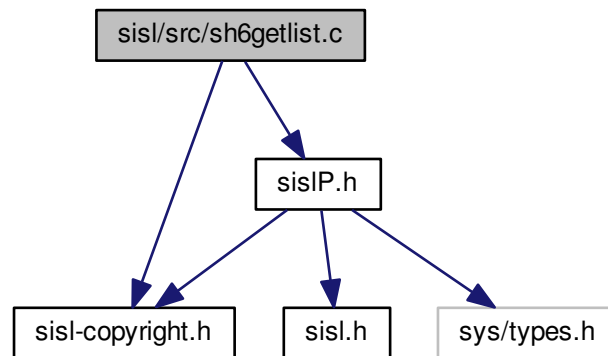
Definition at line 58 of file sh6getgeom.c.

## 30.2840 sisl/src/sh6getlist.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for sh6getlist.c:



### Macros

- #define `SH6GETLIST`

### Functions

- void `sh6getlist` (SISLIntpt \*pt1, SISLIntpt \*pt2, int \*index1, int \*index2, int \*jstat)

## 30.2840.1 Macro Definition Documentation

30.2840.1.1 #define `SH6GETLIST`

Definition at line 49 of file sh6getlist.c.

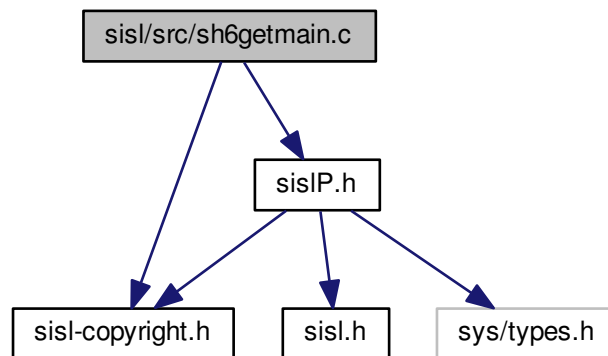
## 30.2840.2 Function Documentation

30.2840.2.1 void sh6getlist ( SISLIntpt \* *pt1*, SISLIntpt \* *pt2*, int \* *index1*, int \* *index2*, int \* *jstat* )

Definition at line 57 of file sh6getlist.c.

## 30.2841 sisl/src/sh6getmain.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh6getmain.c:
```



### Macros

- `#define SH6GETMAIN`

### Functions

- `SISLIntpt * sh6getmain (SISLIntpt *pt)`

## 30.2841.1 Macro Definition Documentation

30.2841.1.1 `#define SH6GETMAIN`

Definition at line 49 of file sh6getmain.c.

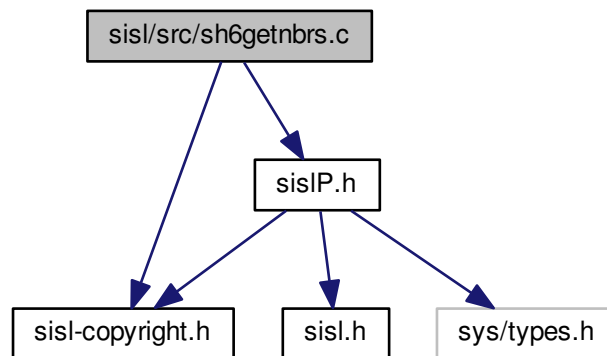
## 30.2841.2 Function Documentation

### 30.2841.2.1 SISLIntpt\* sh6getmain ( SISLIntpt \* pt )

Definition at line 58 of file sh6getmain.c.

## 30.2842 sisl/src/sh6getnbrs.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh6getnbrs.c:
```



### Macros

- `#define SH6GETNHBR`

### Functions

- `void sh6getnhbrs (SISLIntpt *pt, SISLIntpt **pt1, SISLIntpt **pt2, int *jstat)`

## 30.2842.1 Macro Definition Documentation

### 30.2842.1.1 #define SH6GETNHBR

Definition at line 49 of file sh6getnbrs.c.

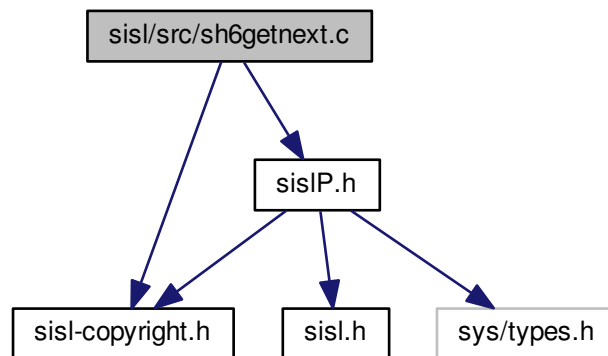
## 30.2842.2 Function Documentation

30.2842.2.1 void sh6getnhbrs ( SISLIntpt \* pt, SISLIntpt \*\* pt1, SISLIntpt \*\* pt2, int \* jstat )

Definition at line 57 of file sh6getnhrs.c.

## 30.2843 sisl/src/sh6getnext.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh6getnext.c:
```



### Macros

- `#define SH6GETNEXT`

### Functions

- `SISLIntpt * sh6getnext (SISLIntpt *pt, int index)`

## 30.2843.1 Macro Definition Documentation

30.2843.1.1 `#define SH6GETNEXT`

Definition at line 49 of file sh6getnext.c.

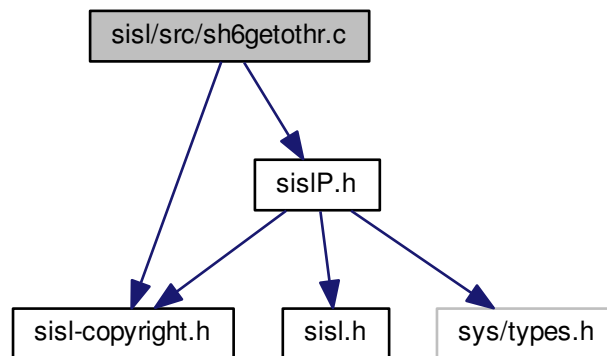
### 30.2843.2 Function Documentation

#### 30.2843.2.1 SISLIntpt\* sh6getnext ( SISLIntpt \* pt, int index )

Definition at line 57 of file sh6getnext.c.

### 30.2844 sisl/src/sh6getthr.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh6getthr.c:
```



#### Macros

- `#define SH6GETOTHER`

#### Functions

- void `sh6getother` (SISLIntpt \*pt, SISLIntpt \*pt1, SISLIntpt \*\*pt2, int \*jstat)

### 30.2844.1 Macro Definition Documentation

#### 30.2844.1.1 #define SH6GETOTHER

Definition at line 49 of file sh6getthr.c.



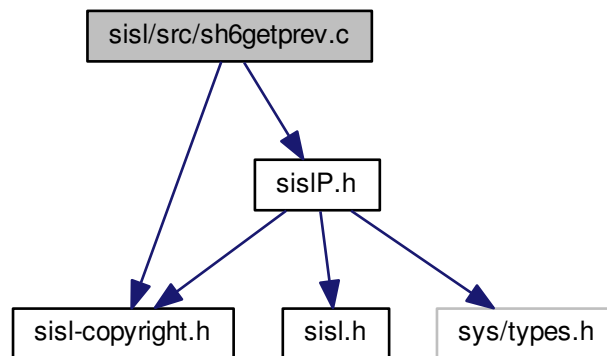
## 30.2844.2 Function Documentation

30.2844.2.1 void sh6getother ( SISLIntpt \* pt, SISLIntpt \* pt1, SISLIntpt \*\* pt2, int \* jstat )

Definition at line 57 of file sh6getthr.c.

## 30.2845 sisl/src/sh6getprev.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh6getprev.c:
```



### Macros

- #define `SH6GETPREV`

### Functions

- int `sh6getprev` (SISLIntpt \*pt1, SISLIntpt \*pt2)

## 30.2845.1 Macro Definition Documentation

30.2845.1.1 #define SH6GETPREV

Definition at line 49 of file sh6getprev.c.

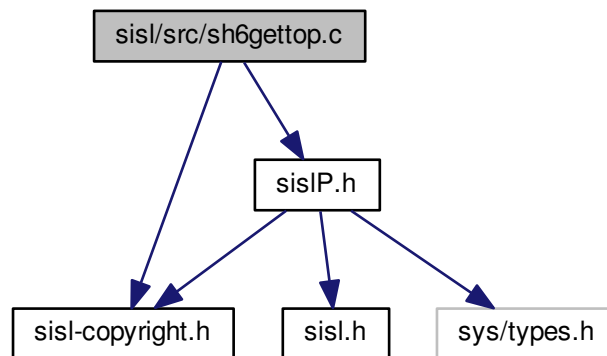
## 30.2845.2 Function Documentation

### 30.2845.2.1 `int sh6getprev ( SISLIntpt * pt1, SISLIntpt * pt2 )`

Definition at line 57 of file `sh6getprev.c`.

## 30.2846 `sisl/src/sh6gettop.c` File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh6gettop.c:
```



### Macros

- `#define SH6GETTOP`

### Functions

- `void sh6gettop ( SISLIntpt *pt, int ilist, int *left1, int *right1, int *left2, int *right2, int *jstat )`

## 30.2846.1 Macro Definition Documentation

### 30.2846.1.1 `#define SH6GETTOP`

Definition at line 49 of file `sh6gettop.c`.

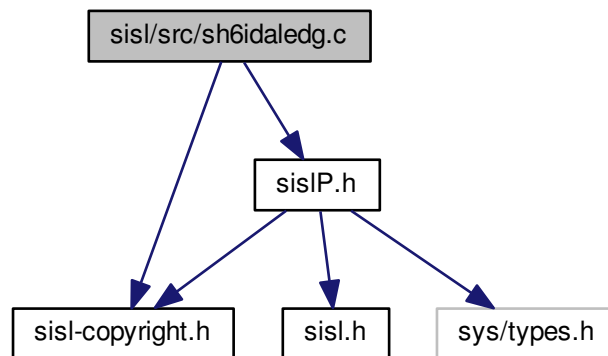
## 30.2846.2 Function Documentation

30.2846.2.1 void sh6gettop ( SISLIntpt \* *pt*, int \* *ilist*, int \* *left1*, int \* *right1*, int \* *left2*, int \* *right2*, int \* *jstat* )

Definition at line 57 of file sh6gettop.c.

## 30.2847 sisl/src/sh6idaledg.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh6idaledg.c:
```



### Macros

- #define `SH6ALLEDG`

### Functions

- void `sh6idaledg` (SISLObject \*pob1, SISLObject \*pob2, SISLIntdat \*pintdat, wedge, int \*jstat)

## 30.2847.1 Macro Definition Documentation

30.2847.1.1 #define SH6ALLEDG

Definition at line 49 of file sh6idaledg.c.

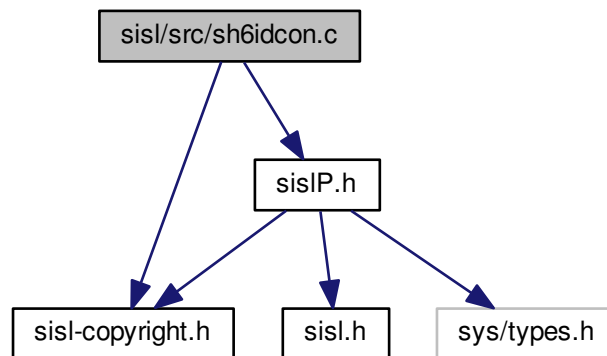
## 30.2847.2 Function Documentation

30.2847.2.1 void sh6idaledg ( SISLObject \* pob1, SISLObject \* pob2, SISLIntdat \* pintdat, wedge , int \* jstat )

Definition at line 57 of file sh6idaledg.c.

## 30.2848 sisl/src/sh6idcon.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh6idcon.c:
```



### Macros

- #define [SH6IDCON](#)

### Functions

- void [sh6idcon](#) (SISLIntdat \*\*pintdat, SISLIntpt \*\*pintpt1, SISLIntpt \*\*pintpt2, int \*jstat)

## 30.2848.1 Macro Definition Documentation

30.2848.1.1 #define SH6IDCON

Definition at line 49 of file sh6idcon.c.

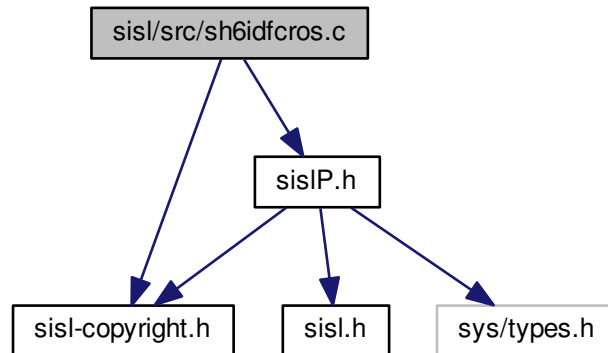
## 30.2848.2 Function Documentation

30.2848.2.1 void sh6idcon ( SISLIntdat \*\* pintdat, SISLIntpt \*\* pintpt1, SISLIntpt \*\* pintpt2, int \* jstat )

Definition at line 63 of file sh6idcon.c.

## 30.2849 sisl/src/sh6idfcros.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh6idfcros.c:
```



### Macros

- #define [SH6IDFCROSS](#)

### Functions

- void [sh6idfcross](#) (SISLIntdat \*pintdat, vcross, int \*jncross, int ipar1, int ipar2, int \*jstat)

## 30.2849.1 Macro Definition Documentation

30.2849.1.1 #define SH6IDFCROSS

Definition at line 49 of file sh6idfcros.c.

## 30.2849.2 Function Documentation

30.2849.2.1 void sh6idfcross ( SISLIntdat \* pintdat, vcross , int \* jncross, int ipar1, int ipar2, int \* jstat )

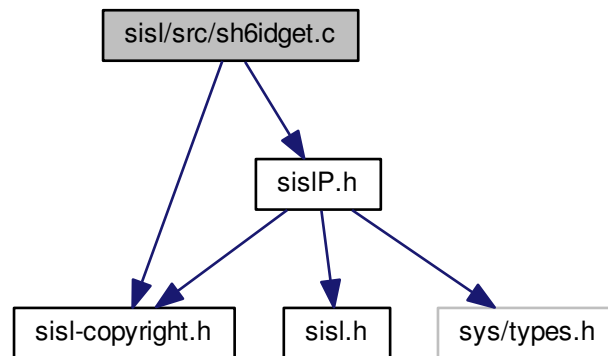
Definition at line 59 of file sh6idfcros.c.

## 30.2850 sisl/src/sh6idget.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for sh6idget.c:



### Macros

- #define [SH6IDGET](#)

### Functions

- void [sh6idget](#) (SISLObject \*po1, SISLObject \*po2, int ipar, double apar, SISLIntdat \*pintdat, SISLIntdat \*\*rintdat, double aepsge, int \*jstat)

## 30.2850.1 Macro Definition Documentation

30.2850.1.1 #define SH6IDGET

Definition at line 49 of file sh6idget.c.

## 30.2850.2 Function Documentation

30.2850.2.1 void sh6idget ( SISLObject \* *po1*, SISLObject \* *po2*, int *ipar*, double *apar*, SISLIntdat \* *pintdat*, SISLIntdat \*\* *rintdat*, double *aepsge*, int \* *jstat* )

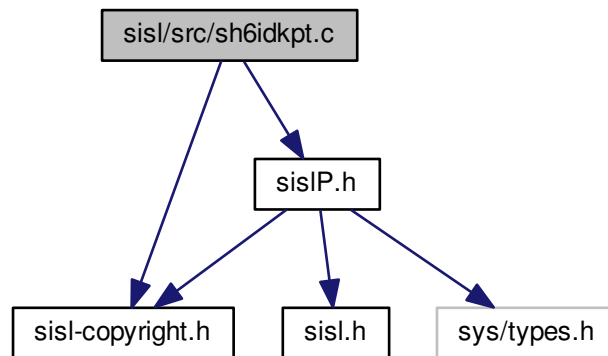
Definition at line 60 of file sh6idget.c.

## 30.2851 sisl/src/sh6idkpt.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for sh6idkpt.c:



### Macros

- #define `S6IDKPT`

### Functions

- void `sh6idget` (SISLIntdat \*\**pintdat*, SISLIntpt \*\**pintpt*, int *join*, int \**jstat*)

## 30.2851.1 Macro Definition Documentation

30.2851.1.1 #define `S6IDKPT`

Definition at line 49 of file sh6idkpt.c.

## 30.2851.2 Function Documentation

30.2851.2.1 void sh6idkpt ( SISLIntdat \*\* pintdat, SISLIntpt \*\* pintpt, int join, int \* jstat )

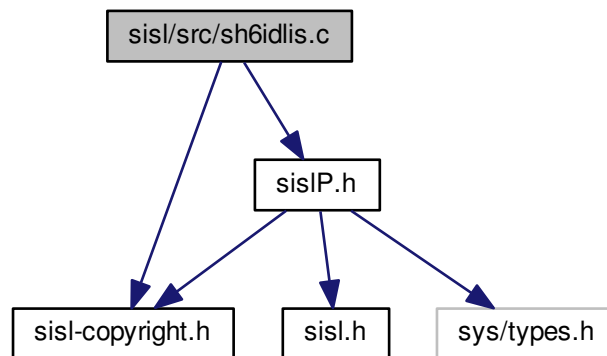
Definition at line 59 of file sh6idkpt.c.

## 30.2852 sisl/src/sh6idlis.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for sh6idlis.c:



### Macros

- #define `SH6IDLIS`

### Functions

- void `sh6idlis` (SISLObject \*po1, SISLObject \*po2, SISLIntdat \*\*pintdat, double aepsge, int \*jstat)

## 30.2852.1 Macro Definition Documentation

30.2852.1.1 #define SH6IDLIS

Definition at line 49 of file sh6idlis.c.



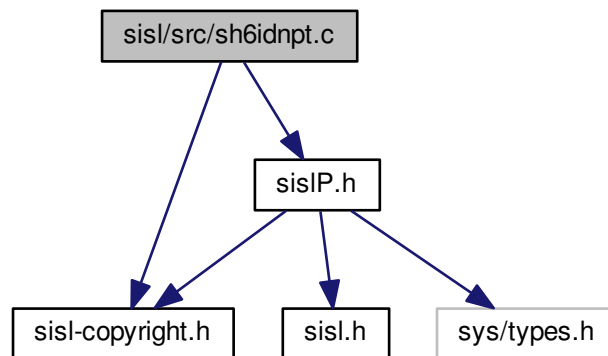
## 30.2852.2 Function Documentation

30.2852.2.1 void sh6idlis ( SISLObject \* *po1*, SISLObject \* *po2*, SISLIntdat \*\* *pintdat*, double *aepsge*, int \* *jstat* )

Definition at line 59 of file sh6idlis.c.

## 30.2853 sisl/src/sh6idnpt.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh6idnpt.c:
```



### Macros

- #define `SH6IDNPT`

### Functions

- void `sh6idnpt` (SISLIntdat \*\**pintdat*, SISLIntpt \*\**pintpt*, int *itest*, int \**jstat*)

## 30.2853.1 Macro Definition Documentation

30.2853.1.1 #define `SH6IDNPT`

Definition at line 49 of file sh6idnpt.c.

### 30.2853.2 Function Documentation

30.2853.2.1 void sh6idnpt ( SISLIntdat \*\* *pintdat*, SISLIntpt \*\* *pintpt*, int *itest*, int \* *jstat* )

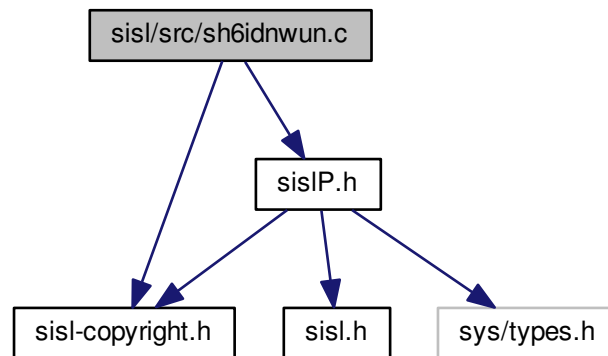
Definition at line 58 of file sh6idnpt.c.

### 30.2854 sisl/src/sh6idnwun.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for sh6idnwun.c:



#### Macros

- #define [SH6IDNEWUNITE](#)

#### Functions

- void [sh6idnewunite](#) (SISLObject \*po1, SISLObject \*po2, SISLIntdat \*\*intdat, SISLIntpt \*\*pt1, SISLIntpt \*\*pt2, double weight, double aepsge, int \*jstat)

### 30.2854.1 Macro Definition Documentation

30.2854.1.1 #define SH6IDNEWUNITE

Definition at line 49 of file sh6idnwun.c.

### 30.2854.2 Function Documentation

30.2854.2.1 void sh6idnewunite ( SISLObject \* *po1*, SISLObject \* *po2*, SISLIntdat \*\* *intdat*, SISLIntpt \*\* *pt1*, SISLIntpt \*\* *pt2*, double *weight*, double *aepsge*, int \* *jstat* )

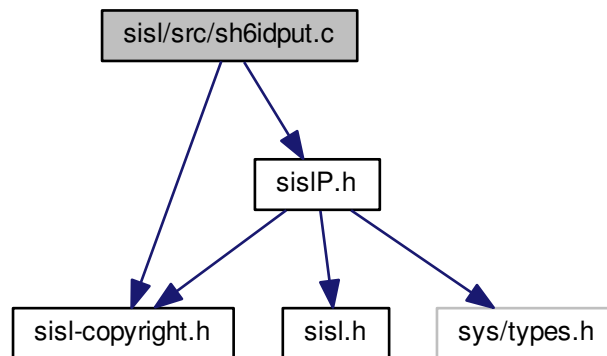
Definition at line 61 of file sh6idnwun.c.

### 30.2855 sisl/src/sh6idput.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for sh6idput.c:



#### Macros

- #define [SH6IDPUT](#)

#### Functions

- void [sh6idput](#) (SISLObject \**po1*, SISLObject \**po2*, SISLIntdat \*\**rintdat*, SISLIntdat \**pintdat*, int *inr*, double *apar*, SISLIntpt \*\*\**outintpt*, int \**npoint*, int \**jstat*)

### 30.2855.1 Macro Definition Documentation

30.2855.1.1 #define SH6IDPUT

Definition at line 49 of file sh6idput.c.

### 30.2855.2 Function Documentation

30.2855.2.1 void sh6idput ( SISLObject \* *po1*, SISLObject \* *po2*, SISLIntdat \*\* *rintdat*, SISLIntdat \* *pintdat*, int *inr*, double *apar*, SISLIntpt \*\*\* *outintpt*, int \* *npoint*, int \* *jstat* )

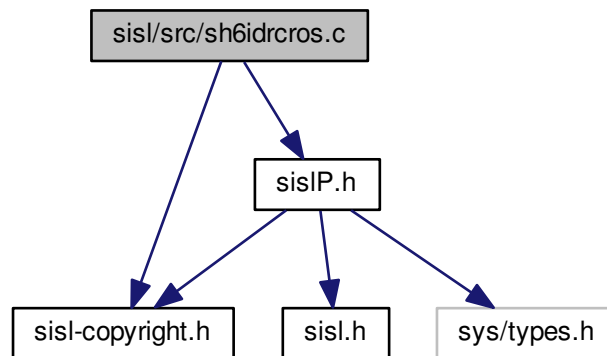
Definition at line 61 of file sh6idput.c.

### 30.2856 sisl/src/sh6idrcros.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for sh6idrcros.c:



#### Macros

- #define [SH6IDRMCROSS](#)

#### Functions

- void [sh6idrcross](#) (SISLObject \**po1*, SISLObject \**po2*, SISLIntdat \*\**pintdat*, *vcross*, int *incross*, *vpt*, int *inpt*, int \**jstat*)

### 30.2856.1 Macro Definition Documentation

30.2856.1.1 #define SH6IDRMCROSS

Definition at line 49 of file sh6idrcros.c.

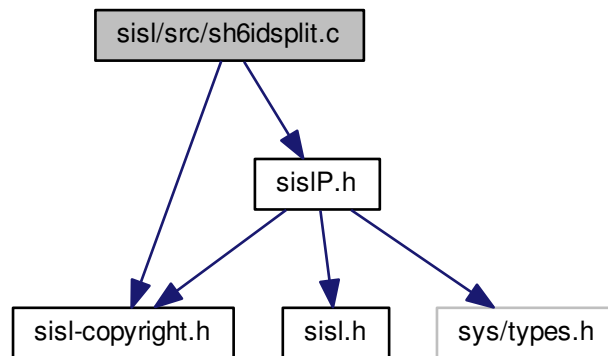
## 30.2856.2 Function Documentation

30.2856.2.1 void sh6idrcross ( SISLObject \* po1, SISLObject \* po2, SISLIntdat \*\* pintdat, vcross , int incross, vpt , int inpt, int \* jstat )

Definition at line 60 of file sh6idrcros.c.

## 30.2857 sisl/src/sh6idsplit.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh6idsplit.c:
```



### Macros

- #define [S6IDSPLIT](#)

### Functions

- void [sh6idsplit](#) (SISLIntdat \*\*pintdat, SISLIntpt \*psource, int \*jstat)

## 30.2857.1 Macro Definition Documentation

30.2857.1.1 #define S6IDSPLIT

Definition at line 49 of file sh6idsplit.c.

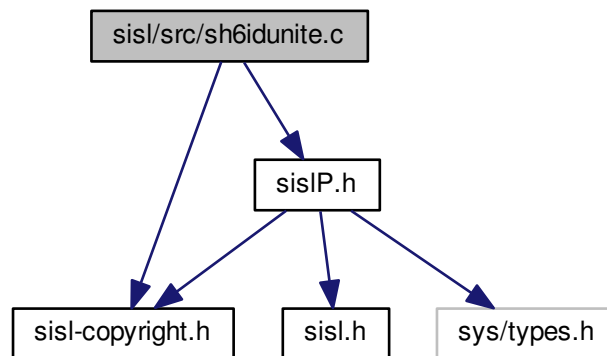
## 30.2857.2 Function Documentation

30.2857.2.1 void sh6idsplit ( SISLIntdat \*\* pintdat, SISLIntpt \* psource, int \* jstat )

Definition at line 59 of file sh6idsplit.c.

## 30.2858 sisl/src/sh6idunite.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh6idunite.c:
```



### Macros

- #define [SH6IDUNITE](#)

### Functions

- void [sh6idunite](#) (SISLIntdat \*\*intdat, SISLIntpt \*\*pt1, SISLIntpt \*\*pt2, double weight, int \*jstat)

## 30.2858.1 Macro Definition Documentation

30.2858.1.1 #define SH6IDUNITE

Definition at line 49 of file sh6idunite.c.

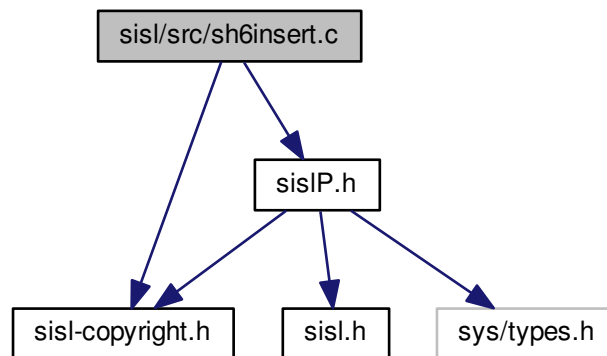
## 30.2858.2 Function Documentation

30.2858.2.1 void sh6idunite ( SISLIntdat \*\* *intdat*, SISLIntpt \*\* *pt1*, SISLIntpt \*\* *pt2*, double *weight*, int \* *jstat* )

Definition at line 60 of file sh6idunite.c.

## 30.2859 sisl/src/sh6insert.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh6insert.c:
```



### Macros

- #define [SH6INSERT](#)

### Functions

- void [sh6insert](#) (SISLIntdat \*\**pintdat*, SISLIntpt \**pt1*, SISLIntpt \**pt2*, SISLIntpt \*\**ptnew*, int \**jstat*)

## 30.2859.1 Macro Definition Documentation

30.2859.1.1 #define SH6INSERT

Definition at line 49 of file sh6insert.c.

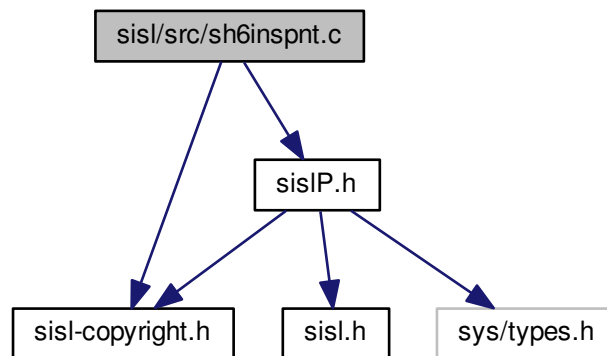
## 30.2859.2 Function Documentation

30.2859.2.1 void sh6insert ( SISLIntdat \*\* *pin*dat, SISLIntpt \* *pt1*, SISLIntpt \* *pt2*, SISLIntpt \*\* *ptnew*, int \* *jstat* )

Definition at line 58 of file sh6insert.c.

## 30.2860 sisl/src/sh6inspnt.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh6inspnt.c:
```



### Macros

- #define [SH6INSERTPT](#)

### Functions

- void [sh6insertpt](#) (SISLIntpt \**pt1*, SISLIntpt \**pt2*, SISLIntpt \**ptnew*, int \**jstat*)

## 30.2860.1 Macro Definition Documentation

30.2860.1.1 #define SH6INSERTPT

Definition at line 49 of file sh6inspnt.c.



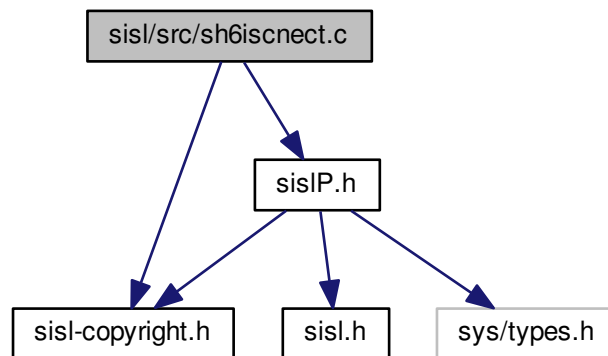
## 30.2860.2 Function Documentation

30.2860.2.1 void sh6insertpt ( SISLIntpt \* pt1, SISLIntpt \* pt2, SISLIntpt \* ptnew, int \* jstat )

Definition at line 59 of file sh6inspnt.c.

## 30.2861 sisl/src/sh6iscnect.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh6iscnect.c:
```



### Macros

- `#define SH6ISCONNECT`

### Functions

- int `sh6iscnect` (SISLIntpt \*pt0, SISLIntpt \*pt1, SISLIntpt \*pt2)

## 30.2861.1 Macro Definition Documentation

30.2861.1.1 `#define SH6ISCONNECT`

Definition at line 49 of file sh6iscnect.c.

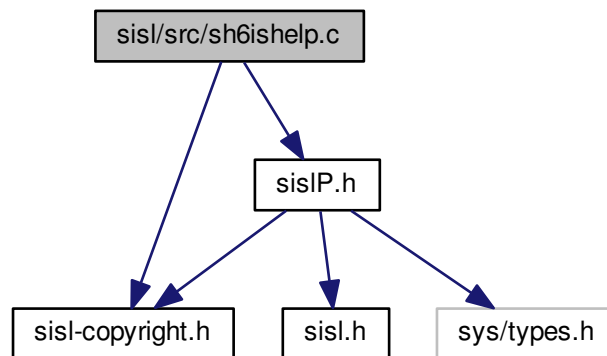
## 30.2861.2 Function Documentation

### 30.2861.2.1 `int sh6isconnect ( SISLIntpt * pt0, SISLIntpt * pt1, SISLIntpt * pt2 )`

Definition at line 57 of file sh6isconnect.c.

## 30.2862 sisl/src/sh6ishelp.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh6ishelp.c:
```



### Macros

- `#define SH6ISHHELP`

### Functions

- `int sh6ishelp (SISLIntpt *pt)`

## 30.2862.1 Macro Definition Documentation

### 30.2862.1.1 `#define SH6ISHHELP`

Definition at line 49 of file sh6ishelp.c.

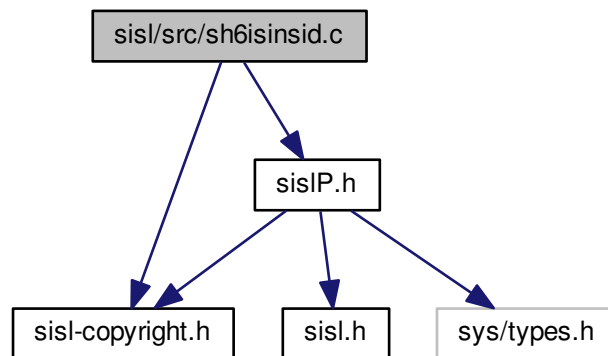
## 30.2862.2 Function Documentation

### 30.2862.2.1 int sh6ishelp ( SISLIntpt \* pt )

Definition at line 57 of file sh6ishelp.c.

## 30.2863 sisl/src/sh6isinsid.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh6isinsid.c:
```



## Macros

- `#define SH6ISINSIDE`

## Functions

- void `sh6isinside` (SISLObject \*po1, SISLObject \*po2, SISLIntpt \*intpt, int \*jstat)

## 30.2863.1 Macro Definition Documentation

### 30.2863.1.1 #define SH6ISINSIDE

Definition at line 49 of file sh6isinsid.c.

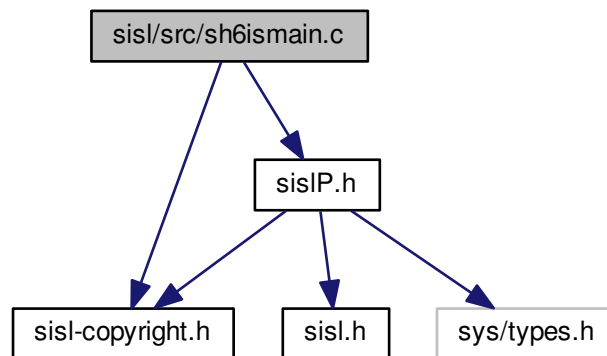
## 30.2863.2 Function Documentation

30.2863.2.1 void sh6isinside ( SISLObject \* *po1*, SISLObject \* *po2*, SISLIntpt \* *intpt*, int \* *jstat* )

Definition at line 59 of file sh6isinsid.c.

## 30.2864 sisl/src/sh6ismain.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh6ismain.c:
```



### Macros

- #define `SH6ISMAIN`

### Functions

- int `sh6ismain` (SISLIntpt \*pt)

## 30.2864.1 Macro Definition Documentation

30.2864.1.1 #define SH6ISMAIN

Definition at line 49 of file sh6ismain.c.

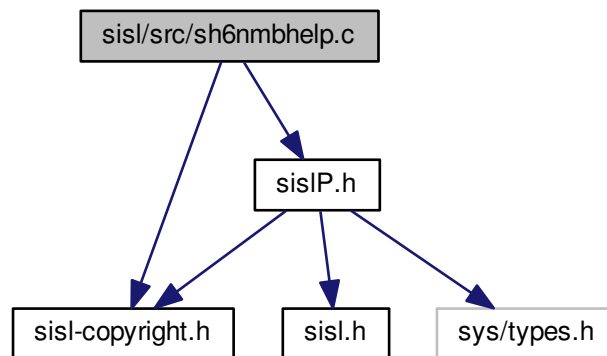
## 30.2864.2 Function Documentation

### 30.2864.2.1 int sh6ismain ( SISLIntpt \* pt )

Definition at line 57 of file sh6ismain.c.

## 30.2865 sisl/src/sh6nmbhelp.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh6nmbhelp.c:
```



## Macros

- `#define SH6NMBHELP`

## Functions

- int `sh6nmbhelp` (SISLIntpt \*pt, int \*jstat)

## 30.2865.1 Macro Definition Documentation

### 30.2865.1.1 #define SH6NMBHELP

Definition at line 49 of file sh6nmbhelp.c.

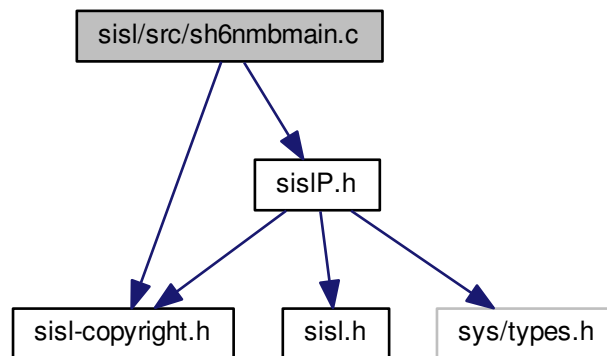
## 30.2865.2 Function Documentation

### 30.2865.2.1 `int sh6nmbhelp ( SISLIntpt * pt, int * jstat )`

Definition at line 57 of file sh6nmbhelp.c.

## 30.2866 `sisl/src/sh6nmbmain.c` File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh6nmbmain.c:
```



### Macros

- `#define SH6NMBMAIN`

### Functions

- `int sh6nmbmain (SISLIntpt *pt, int *jstat)`

## 30.2866.1 Macro Definition Documentation

### 30.2866.1.1 `#define SH6NMBMAIN`

Definition at line 49 of file sh6nmbmain.c.

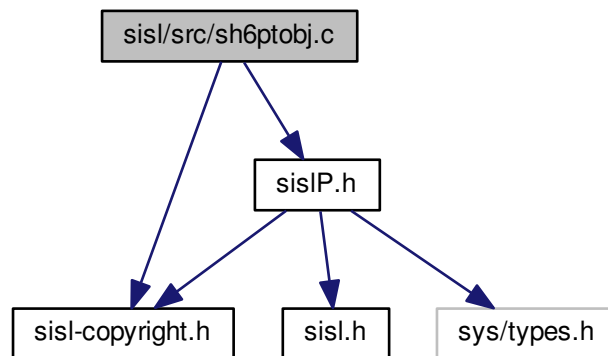
## 30.2866.2 Function Documentation

### 30.2866.2.1 int sh6nmbmain ( SISLIntpt \* pt, int \* jstat )

Definition at line 57 of file sh6nmbmain.c.

## 30.2867 sisl/src/sh6ptobj.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh6ptobj.c:
```



### Macros

- `#define SH6PTOBJ`

### Functions

- void `sh6ptobj` (double \*point, SISLObject \*obj, double aepsge, start, result, int \*jstat)

## 30.2867.1 Macro Definition Documentation

### 30.2867.1.1 #define SH6PTOBJ

Definition at line 49 of file sh6ptobj.c.

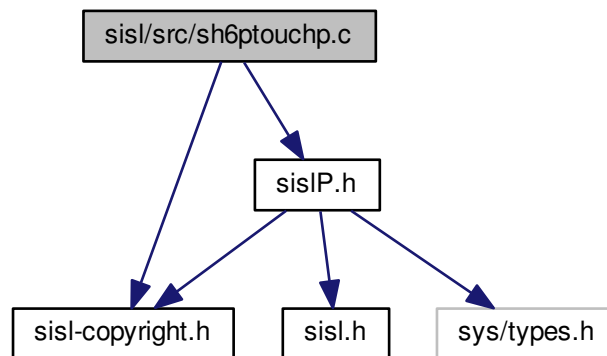
## 30.2867.2 Function Documentation

30.2867.2.1 void sh6ptobj( double \* point, SISLObject \* obj, double aeptsge, start, result, int \* jstat )

Definition at line 59 of file sh6ptobj.c.

## 30.2868 sisl/src/sh6ptouchp.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh6ptouchp.c:
```



### Macros

- #define [SH6PUTTOUCH](#)

### Functions

- void [sh6puttouch](#) (SISLIntpt \*psource, SISLIntpt \*pdest, int seq)

## 30.2868.1 Macro Definition Documentation

30.2868.1.1 #define SH6PUTTOUCH

Definition at line 49 of file sh6ptouchp.c.



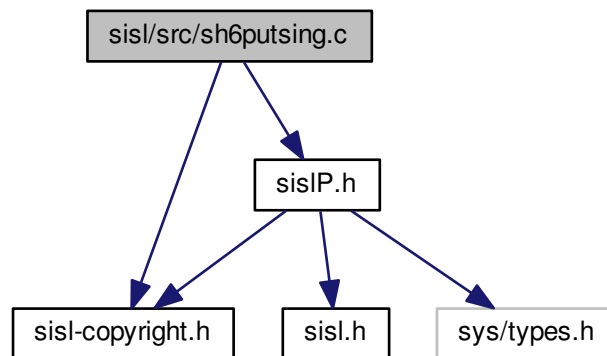
## 30.2868.2 Function Documentation

30.2868.2.1 void sh6puttouch ( SISLIntpt \* psource, SISLIntpt\* pdest, int seq )

Definition at line 57 of file sh6ptouchp.c.

## 30.2869 sisl/src/sh6putsing.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh6putsing.c:
```



### Macros

- #define `SH6PUTSING`

### Functions

- void `sh6putsing` (SISLIntpt \*psource, SISLIntpt \*pdest)

## 30.2869.1 Macro Definition Documentation

30.2869.1.1 #define SH6PUTSING

Definition at line 49 of file sh6putsing.c.

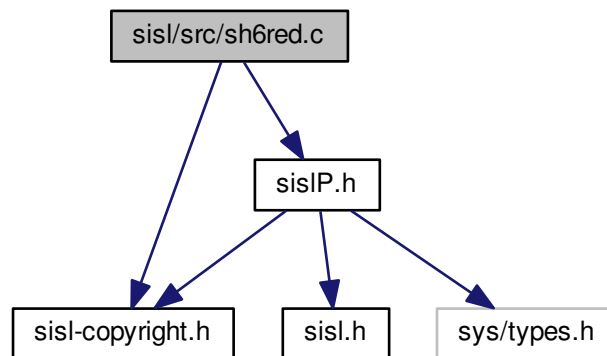
## 30.2869.2 Function Documentation

### 30.2869.2.1 void sh6putsing ( SISLIntpt \* psource, SISLIntpt\* pdest )

Definition at line 57 of file sh6putsing.c.

## 30.2870 sisl/src/sh6red.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh6red.c:
```



### Macros

- `#define SH6RED`

### Functions

- void `sh6red` (SISLObject \*po1, SISLObject \*po2, SISLIntdat \*pintdat, int \*jstat)

## 30.2870.1 Macro Definition Documentation

### 30.2870.1.1 #define SH6RED

Definition at line 49 of file sh6red.c.

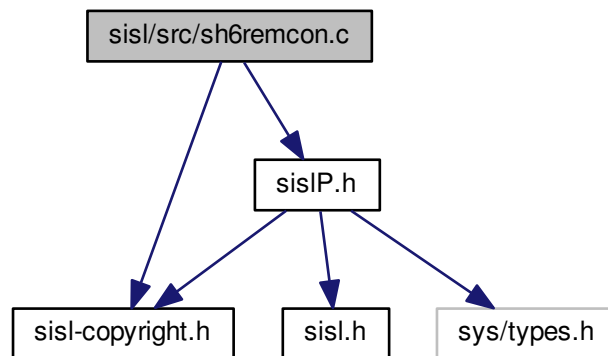
## 30.2870.2 Function Documentation

30.2870.2.1 void sh6red ( SISLObject \* po1, SISLObject \* po2, SISLIntdat \* pintdat, int \* jstat )

Definition at line 59 of file sh6red.c.

## 30.2871 sisl/src/sh6remcon.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh6remcon.c:
```



### Macros

- `#define SH6REMCON`

### Functions

- void `sh6remcon` (SISLIntdat \*\*pintdat, SISLIntpt \*pt1, SISLIntpt \*pt2, int \*jstat)

## 30.2871.1 Macro Definition Documentation

30.2871.1.1 `#define SH6REMCON`

Definition at line 49 of file sh6remcon.c.

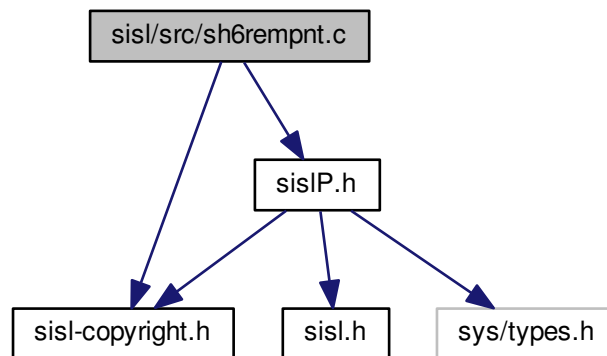
## 30.2871.2 Function Documentation

30.2871.2.1 void sh6remcon ( SISLIntdat \*\* pintdat, SISLIntpt \* pt1, SISLIntpt \* pt2, int \* jstat )

Definition at line 57 of file sh6remcon.c.

## 30.2872 sisl/src/sh6rempnt.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh6rempnt.c:
```



### Macros

- #define [SH6REMOVEPT](#)

### Functions

- void [sh6removept](#) (SISLIntpt \*pt1, SISLIntpt \*pt2, SISLIntpt \*ptold, int \*jstat)

## 30.2872.1 Macro Definition Documentation

30.2872.1.1 #define SH6REMOVEPT

Definition at line 49 of file sh6rempnt.c.

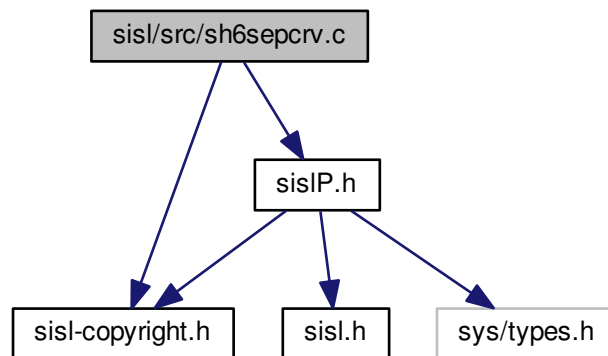
## 30.2872.2 Function Documentation

30.2872.2.1 void sh6removept ( SISLIntpt \* pt1, SISLIntpt \* pt2, SISLIntpt \* ptold, int \* jstat )

Definition at line 57 of file sh6rempnt.c.

## 30.2873 sisl/src/sh6sepcrv.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh6sepcrv.c:
```



### Macros

- #define [SH6SEPCRV](#)

### Functions

- void [sh6sepcrv](#) (SISLCurve \*pc1, SISLCurve \*pc2, double aepsge, ecentre, double \*crad, int \*jstat)

## 30.2873.1 Macro Definition Documentation

30.2873.1.1 #define SH6SEPCRV

Definition at line 49 of file sh6sepcrv.c.

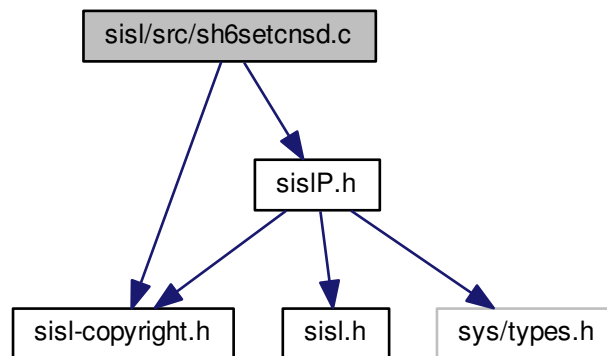
### 30.2873.2 Function Documentation

30.2873.2.1 void sh6sepcrv ( SISLCurve \* pc1, SISLCurve \* pc2, double aepsge, ecentre , double \* crad, int \* jstat )

Definition at line 66 of file sh6sepcrv.c.

### 30.2874 sisl/src/sh6setcnsd.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh6setcnsd.c:
```



#### Macros

- #define [SH6SETCNSDIR](#)

#### Functions

- void [sh6setcnsdir](#) (SISLIntpt \*pt1, SISLIntpt \*pt2, int ipar, int \*jstat)

### 30.2874.1 Macro Definition Documentation

30.2874.1.1 #define SH6SETCNSDIR

Definition at line 49 of file sh6setcnsd.c.

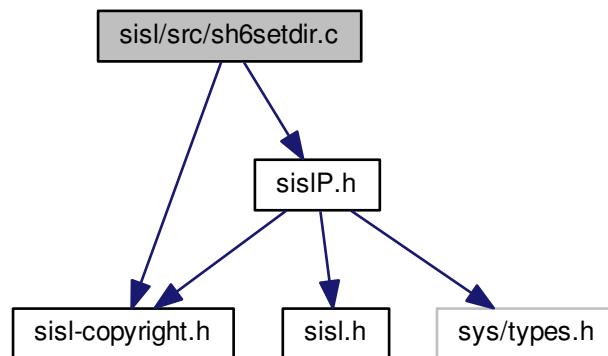
## 30.2874.2 Function Documentation

30.2874.2.1 void sh6setcnsdir ( SISLIntpt \* *pt1*, SISLIntpt \* *pt2*, int *ipar*, int \* *jstat* )

Definition at line 57 of file sh6setcnsd.c.

## 30.2875 sisl/src/sh6setdir.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh6setdir.c:
```



### Macros

- #define `SH6SETDIR`

### Functions

- void `sh6setdir` (SISLIntpt \**pt1*, SISLIntpt \**pt2*, int \**jstat*)

## 30.2875.1 Macro Definition Documentation

30.2875.1.1 #define SH6SETDIR

Definition at line 49 of file sh6setdir.c.

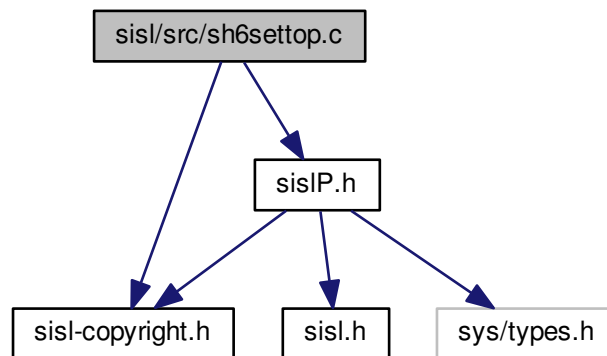
## 30.2875.2 Function Documentation

### 30.2875.2.1 void sh6setdir ( SISLIntpt \* pt1, SISLIntpt \* pt2, int \* jstat )

Definition at line 57 of file sh6setdir.c.

## 30.2876 sisl/src/sh6settop.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh6settop.c:
```



### Macros

- `#define SH6SETTOP`

### Functions

- void `sh6settop` (SISLIntpt \*pt, int ilist, int left1, int right1, int left2, int right2, int \*jstat)

## 30.2876.1 Macro Definition Documentation

### 30.2876.1.1 #define SH6SETTOP

Definition at line 49 of file sh6settop.c.



## 30.2876.2 Function Documentation

30.2876.2.1 void sh6settop ( SISLIntpt \* pt, int ilist, int left1, int right1, int left2, int right2, int \* jstat )

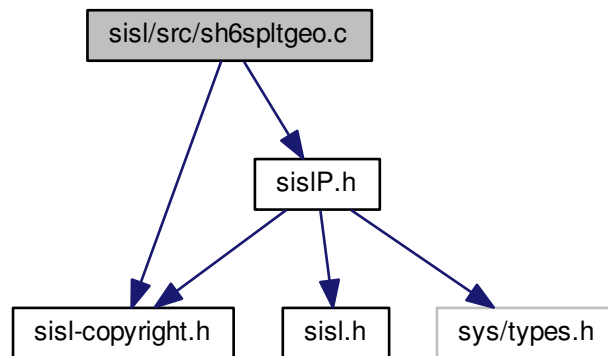
Definition at line 57 of file sh6settop.c.

## 30.2877 sisl/src/sh6spltgeo.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for sh6spltgeo.c:



### Macros

- #define [SH6SPLITGEOM](#)

### Functions

- void [sh6splitgeom](#) (SISLSurf \*ps1, SISLSurf \*ps2, double aepsge, ecentre, eaxis, double \*cdist, double \*crad, int \*jstat)

## 30.2877.1 Macro Definition Documentation

30.2877.1.1 #define SH6SPLITGEOM

Definition at line 49 of file sh6spltgeo.c.

## 30.2877.2 Function Documentation

30.2877.2.1 void sh6splitgeom ( SISLSurf \* *ps1*, SISLSurf \* *ps2*, double *aepsge*, ecentre , eaxis , double \* *cdist*, double \* *crad*, int \* *jstat* )

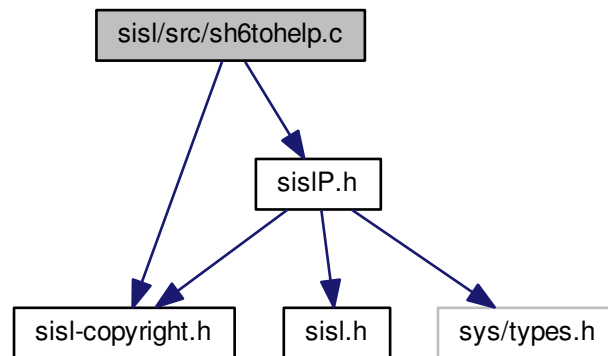
Definition at line 66 of file sh6spltgeo.c.

## 30.2878 sisl/src/sh6tohelp.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for sh6tohelp.c:



### Macros

- #define [SH6TOHELP](#)

### Functions

- void [sh6tohelp](#) (SISLIntpt \*pt, int \*jstat)

## 30.2878.1 Macro Definition Documentation

30.2878.1.1 #define SH6TOHELP

Definition at line 49 of file sh6tohelp.c.

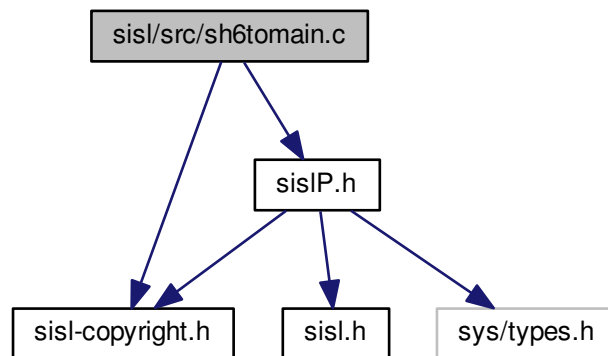
## 30.2878.2 Function Documentation

### 30.2878.2.1 void sh6tohelp ( SISLIntpt \* pt, int \* jstat )

Definition at line 58 of file sh6tohelp.c.

## 30.2879 sisl/src/sh6tomain.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh6tomain.c:
```



## Macros

- `#define SH6TOMAIN`

## Functions

- void `sh6tomain` (SISLIntpt \*pt, int \*jstat)

## 30.2879.1 Macro Definition Documentation

### 30.2879.1.1 #define SH6TOMAIN

Definition at line 49 of file sh6tomain.c.

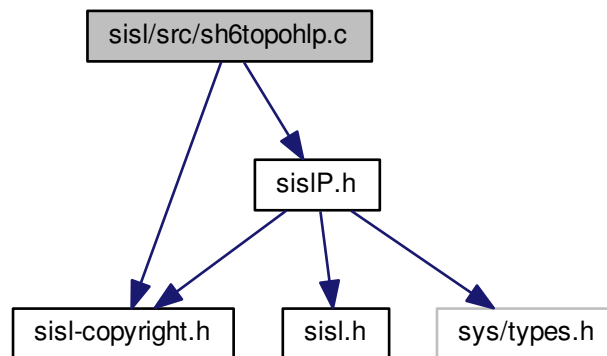
## 30.2879.2 Function Documentation

### 30.2879.2.1 void sh6tomain ( SISLIntpt \* pt, int \* jstat )

Definition at line 58 of file sh6tomain.c.

## 30.2880 sisl/src/sh6topohlp.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh6topohlp.c:
```



### Macros

- #define [SH6GETTOPHLP](#)

### Functions

- void [sh6gettophlp](#) (SISLIntpt \*pt, pretop, int case\_2d, int \*jstat)

## 30.2880.1 Macro Definition Documentation

### 30.2880.1.1 #define SH6GETTOPHLP

Definition at line 49 of file sh6topohlp.c.

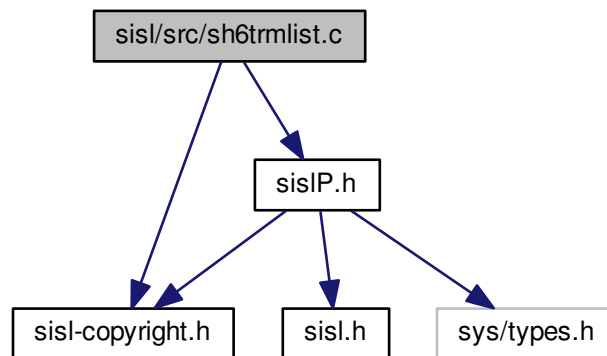
## 30.2880.2 Function Documentation

30.2880.2.1 void sh6gettophlp ( SISLIntpt \* pt, pretop , int case\_2d, int \* jstat )

Definition at line 59 of file sh6topohlp.c.

## 30.2881 sisl/src/sh6trmlist.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh6trmlist.c:
```



### Macros

- #define [SH6TRIMLIST](#)

### Functions

- void [sh6trimlist](#) (SISLIntpt \*pt, SISLIntpt \*\*\*ptlist, int \*no\_of\_points, int \*no\_alloc)

## 30.2881.1 Macro Definition Documentation

30.2881.1.1 #define SH6TRIMLIST

Definition at line 49 of file sh6trmlist.c.

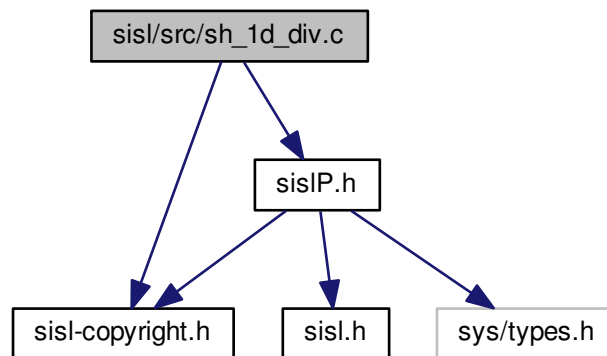
## 30.2881.2 Function Documentation

30.2881.2.1 void sh6trimlist ( SISLIntpt \* pt, SISLIntpt \*\*\* ptlist, int \* no\_of\_points, int \* no\_alloc )

Definition at line 60 of file sh6trimlist.c.

## 30.2882 sisl/src/sh\_1d\_div.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh_1d_div.c:
```



### Macros

- #define [SH\\_1D\\_DIV](#)

### Functions

- void [sh\\_1d\\_div](#) (SISLObject \*po1, SISLObject \*po2, double aepsge, SISLIntdat \*\*pintdat, vedge, int \*jstat)

## 30.2882.1 Macro Definition Documentation

30.2882.1.1 #define SH\_1D\_DIV

Definition at line 49 of file sh\_1d\_div.c.

## 30.2882.2 Function Documentation

30.2882.2.1 void `sh_1d_div` ( `SISLObject * po1`, `SISLObject * po2`, `double aepsge`, `SISLIntdat ** pintdat`, `vedge`, `int * jstat` )

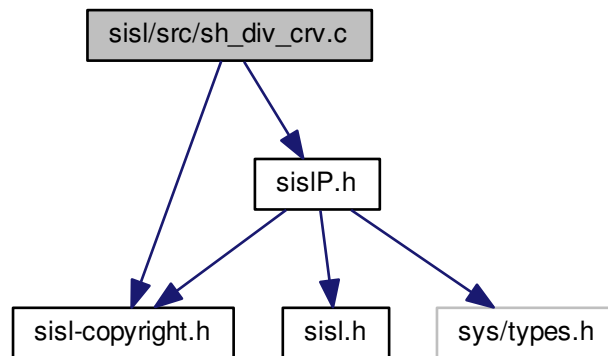
Definition at line 73 of file `sh_1d_div.c`.

## 30.2883 sisl/src/sh\_div\_crv.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for `sh_div_crv.c`:



### Macros

- `#define SH_DIV_CRV`

### Functions

- void `sh_div_crv` (`SISLCurve *pc`, `int which_end`, `double aepsge`, `SISLCurve **rcnew`, `int *jstat`)

## 30.2883.1 Macro Definition Documentation

30.2883.1.1 `#define SH_DIV_CRV`

Definition at line 49 of file `sh_div_crv.c`.

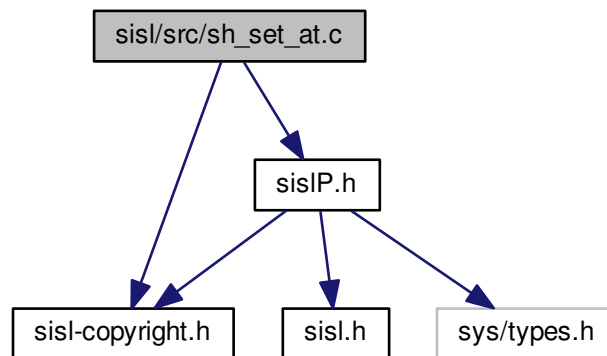
## 30.2883.2 Function Documentation

30.2883.2.1 void sh\_div\_crv ( SISLCurve \* pc, int which\_end, double aepsge, SISLCurve \*\* rcnew, int \* jstat )

Definition at line 58 of file sh\_div\_crv.c.

## 30.2884 sisl/src/sh\_set\_at.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for sh_set_at.c:
```



### Macros

- #define `SH_SET_AT`

### Functions

- void `sh_set_at` (SISLObject \*po1, SISLObject \*po2, SISLIntdat \*pintdat, int \*jstat)

## 30.2884.1 Macro Definition Documentation

30.2884.1.1 #define `SH_SET_AT`

Definition at line 49 of file sh\_set\_at.c.



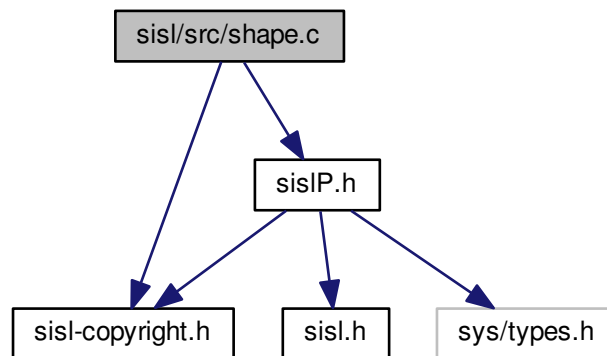
## 30.2884.2 Function Documentation

30.2884.2.1 void sh\_set\_at ( SISLObject \* *po1*, SISLObject \* *po2*, SISLIntdat \* *pintdat*, int \* *jstat* )

Definition at line 62 of file sh\_set\_at.c.

## 30.2885 sisl/src/shape.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for shape.c:
```



### Macros

- `#define` [SHAPE](#)

### Functions

- void [shape](#) (emid, etang, int idim, int iedge, int \*jstat)

## 30.2885.1 Macro Definition Documentation

30.2885.1.1 `#define` [SHAPE](#)

Definition at line 49 of file shape.c.

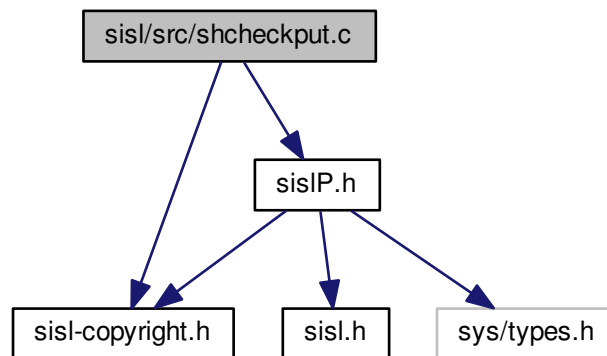
## 30.2885.2 Function Documentation

30.2885.2.1 void shape ( emid , etang , int *idim*, int *iedge*, int\* *jstat* )

Definition at line 57 of file shape.c.

## 30.2886 sisl/src/shcheckpoint.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for shcheckpoint.c:
```



### Macros

- #define `SHCHECKPUT`

### Functions

- void `shcheckpoint` (`SISLObject` \*po1, `SISLIntdat` \*\*rintdat, `SISLIntdat` \*pintdat, int inr, double apar, int \*jstat)

## 30.2886.1 Macro Definition Documentation

30.2886.1.1 #define `SHCHECKPUT`

Definition at line 49 of file shcheckpoint.c.

## 30.2886.2 Function Documentation

30.2886.2.1 void shcheckput ( SISLObject \* *po1*, SISLIntdat \*\* *rintdat*, SISLIntdat \* *pintdat*, int *inr*, double *apar*, int \* *jstat* )

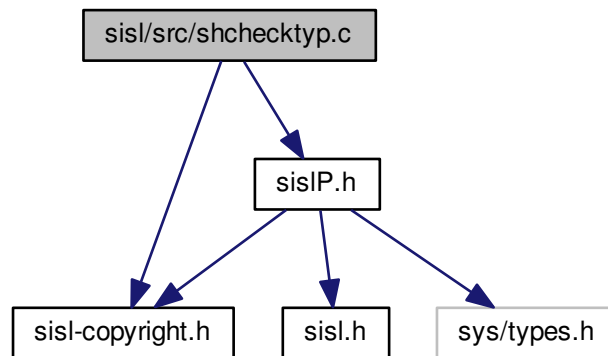
Definition at line 59 of file shcheckput.c.

## 30.2887 sisl/src/shchecktyp.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for shchecktyp.c:



### Macros

- #define `SHCHECKTYPE`

### Functions

- int `shchecktype` (SISLObject \**pobj*, double \**parval*)

## 30.2887.1 Macro Definition Documentation

30.2887.1.1 #define `SHCHECKTYPE`

Definition at line 49 of file shchecktyp.c.

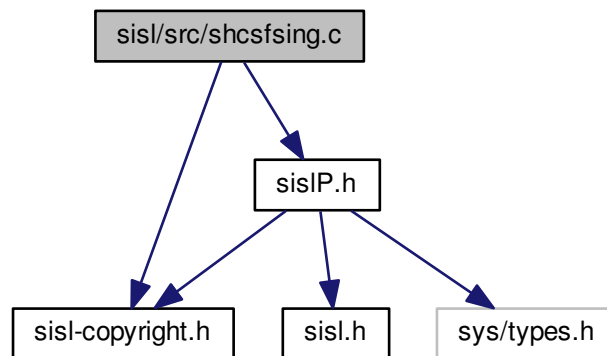
## 30.2887.2 Function Documentation

30.2887.2.1 `int shchecktype ( SISLObject * pobj, double * parval )`

Definition at line 58 of file shchecktyp.c.

## 30.2888 sisl/src/shcsfsing.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for shcsfsing.c:
```



### Macros

- `#define SHCSFSING`

### Functions

- `void shcsfsing ( SISLCurve *pcurve, SISLSurf *psurf, limit, enext, gpos, int *jstat)`

## 30.2888.1 Macro Definition Documentation

30.2888.1.1 `#define SHCSFSING`

Definition at line 48 of file shcsfsing.c.

## 30.2888.2 Function Documentation

30.2888.2.1 void shcsfsing ( SISLCurve \* *pcurve*, SISLSurf \* *psurf*, limit , enext , gpos , int \* *jstat* )

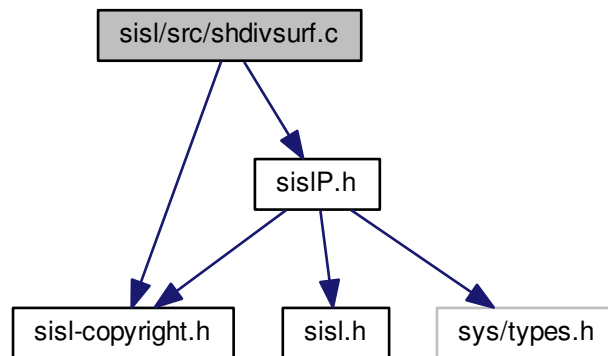
Definition at line 69 of file shcsfsing.c.

## 30.2889 sisl/src/shdivsurf.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for shdivsurf.c:



### Macros

- #define `SH_DIV_SURF`

### Functions

- void `sh_div_surf` (SISLSurf \**ps*, int *which\_end\_1*, int *which\_end\_2*, double *aepsge*, SISLSurf \*\**rsnew*, int \**jstat*)

## 30.2889.1 Macro Definition Documentation

30.2889.1.1 #define `SH_DIV_SURF`

Definition at line 49 of file shdivsurf.c.

## 30.2889.2 Function Documentation

30.2889.2.1 void `sh_div_surf` ( `SISLSurf * ps`, int `which_end_1`, int `which_end_2`, double `aepsge`, `SISLSurf ** rsnew`, int `* jstat` )

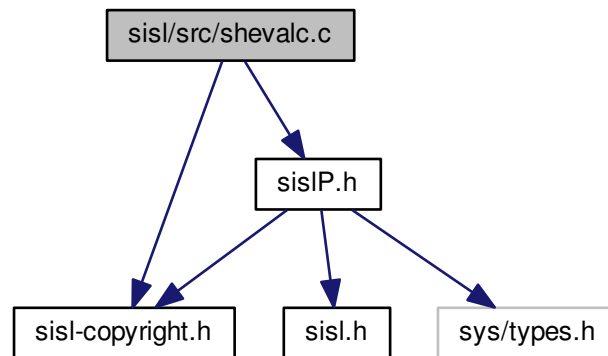
Definition at line 59 of file `shdivsurf.c`.

## 30.2890 sisl/src/shevalc.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for `shevalc.c`:



### Macros

- `#define SHEVALC`

### Functions

- void `shevalc` (`SISLCurve *pc1`, int `ider`, double `ax`, double `aepsge`, int `*ileft`, `eder`, int `*jstat`)

## 30.2890.1 Macro Definition Documentation

30.2890.1.1 `#define SHEVALC`

Definition at line 49 of file `shevalc.c`.

## 30.2890.2 Function Documentation

30.2890.2.1 void shevalc ( SISLCurve \* pc1, int iver, double ax, double aepsge, int \* ileft, eder , int \* jstat )

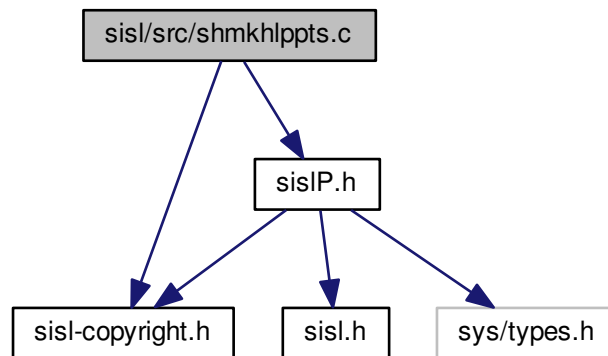
Definition at line 58 of file shevalc.c.

## 30.2891 sisl/src/shmklppts.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for shmklppts.c:



## Macros

- #define [SHMKHLPPTS](#)

## Functions

- void [shmklppts](#) (SISLObject \*po1, SISLObject \*po2, double aepsge, SISLIntdat \*\*rintdat, vedge, int \*jnewpt, int \*jstat)

## 30.2891.1 Macro Definition Documentation

30.2891.1.1 #define SHMKHLPPTS

Definition at line 49 of file shmklppts.c.

### 30.2891.2 Function Documentation

30.2891.2.1 void shmkhppts ( SISLObject \* po1, SISLObject \* po2, double aepsge, SISLIntdat \*\* rintdat, vedge , int \* jnewpt, int \* jstat )

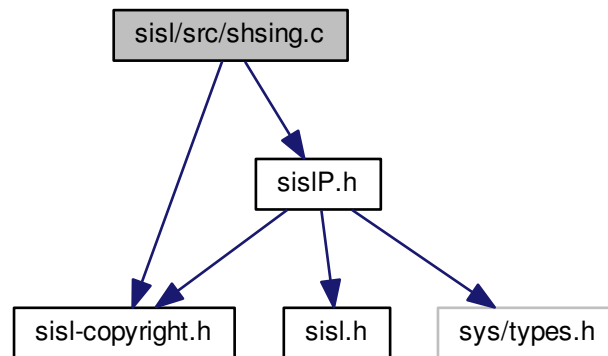
Definition at line 58 of file shmkhppts.c.

### 30.2892 sisl/src/shsing.c File Reference

```
#include "sisl-copyright.h"
```

```
#include "sislP.h"
```

Include dependency graph for shsing.c:



#### Macros

- #define SHSING

#### Functions

- void shsing (SISLSurf \*psurf1, SISLSurf \*psurf2, limit, enext, gpos, int \*jstat)

### 30.2892.1 Macro Definition Documentation

30.2892.1.1 #define SHSING

Definition at line 48 of file shsing.c.



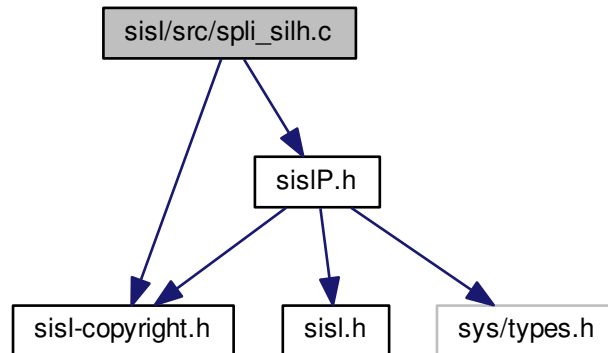
## 30.2892.2 Function Documentation

30.2892.2.1 void shsing ( SISLSurf \* *psurf1*, SISLSurf \* *psurf2*, limit , enext , gpos , int \* *jstat* )

Definition at line 69 of file shsing.c.

## 30.2893 sisl/src/spli\_silh.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for spli_silh.c:
```



### Macros

- #define `SPLI_SILH`

### Functions

- void `spli_silh` (SISLIntdat \*\*pintdat, SISLObject \*po1, int \*jstat)

## 30.2893.1 Macro Definition Documentation

30.2893.1.1 #define `SPLI_SILH`

Definition at line 49 of file `spli_silh.c`.

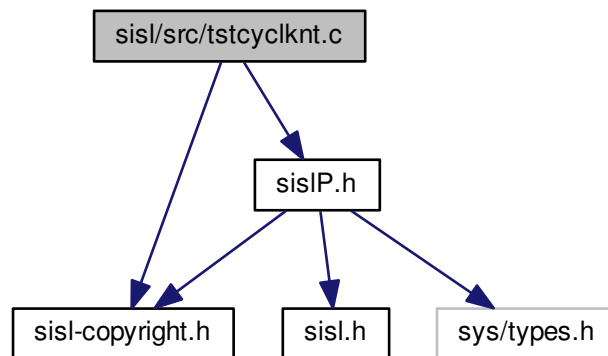
### 30.2893.2 Function Documentation

30.2893.2.1 void `spli_silh` ( SISLIntdat \*\* *pin*tdat, SISLObject \* *po1*, int \* *jstat* )

Definition at line 61 of file `spli_silh.c`.

### 30.2894 sisl/src/tstcyclknt.c File Reference

```
#include "sisl-copyright.h"
#include "sislP.h"
Include dependency graph for tstcyclknt.c:
```



#### Macros

- `#define TEST_CYCLIC_KNOTS`

#### Functions

- void `test_cyclic_knots` (et, int in, int ik, int \*jstat)

### 30.2894.1 Macro Definition Documentation

30.2894.1.1 `#define TEST_CYCLIC_KNOTS`

Definition at line 49 of file `tstcyclknt.c`.

## 30.2894.2 Function Documentation

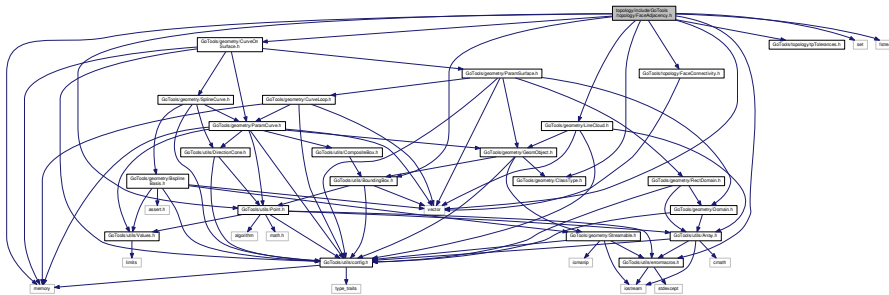
30.2894.2.1 void test\_cyclic\_knots ( et , int in, int ik, int \* jstat )

Definition at line 57 of file tstcyclknt.c.

## 30.2895 topology/include/GoTools/topology/FaceAdjacency.h File Reference

```
#include "GoTools/utils/Point.h"
#include "GoTools/utils/BoundingBox.h"
#include "GoTools/utils/errormacros.h"
#include "GoTools/geometry/ClassType.h"
#include "GoTools/geometry/CurveOnSurface.h"
#include "GoTools/geometry/LineCloud.h"
#include "GoTools/topology/FaceConnectivity.h"
#include "GoTools/topology/tpTolerances.h"
#include <vector>
#include <set>
#include <memory>
#include <fstream>
```

Include dependency graph for FaceAdjacency.h:



## Classes

- class [Go::MarchPoint](#)  
*Helper class for marching.*
- class [Go::FaceAdjacency](#)< [edgeType](#), [faceType](#) >  
*Compute adjacency information for a set of surfaces.*

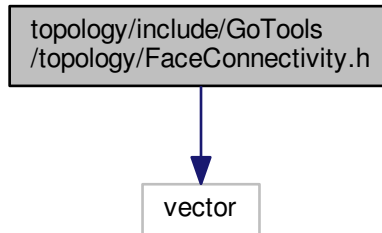
## Namespaces

- [Go](#)

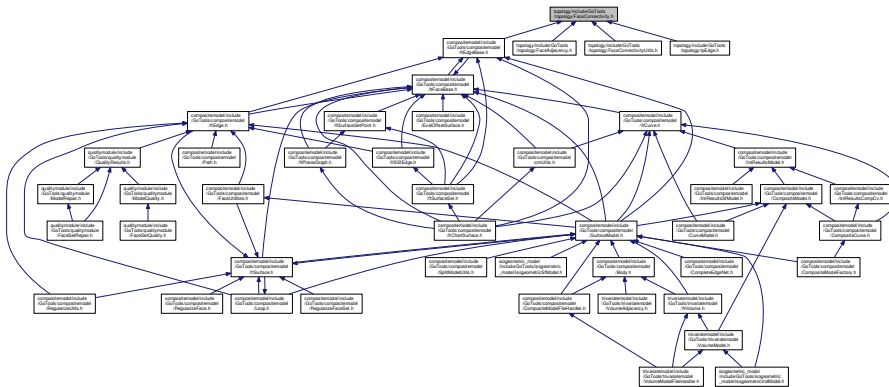
## 30.2896 topology/include/GoTools/topology/FaceConnectivity.h File Reference

```
#include <vector>
```

Include dependency graph for FaceConnectivity.h:



This graph shows which files directly or indirectly include this file:



### Classes

- struct [Go::FaceConnectivity< edgeType >](#)

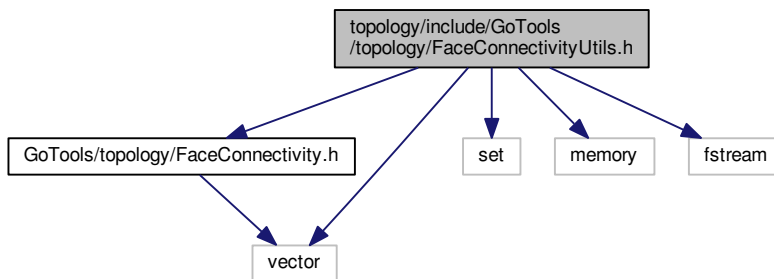
### Namespaces

- [Go](#)

## 30.2897 topology/include/GoTools/topology/FaceConnectivityUtils.h File Reference

```
#include "GoTools/topology/FaceConnectivity.h"
#include <vector>
#include <set>
#include <memory>
#include <fstream>
```

Include dependency graph for FaceConnectivityUtils.h:



### Classes

- class `Go::FaceConnectivityUtils< edgeType, faceType >`

*Utilities used in adjacency computations of face sets.*

### Namespaces

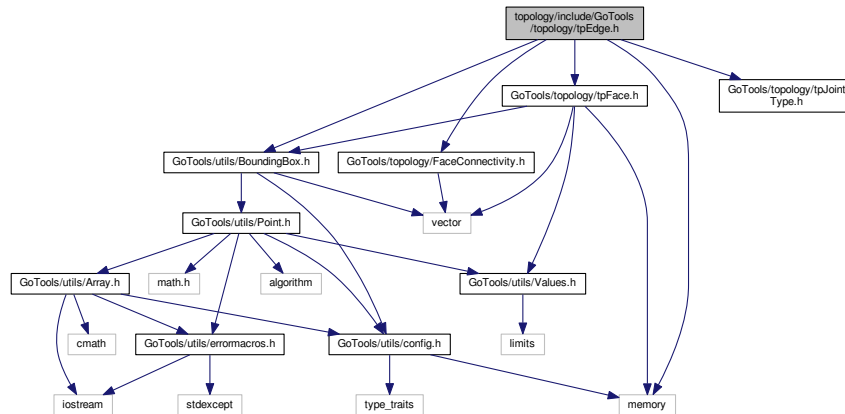
- `Go`

## 30.2898 topology/include/GoTools/topology/topology\_doxymain.h File Reference

## 30.2899 topology/include/GoTools/topology/tpEdge.h File Reference

```
#include "GoTools/topology/tpFace.h"
#include "GoTools/utils/BoundingBox.h"
#include "GoTools/topology/tpJointType.h"
#include "GoTools/topology/FaceConnectivity.h"
#include <memory>
```

Include dependency graph for `tpEdge.h`:



## Classes

- class [Go::tpEdge](#)

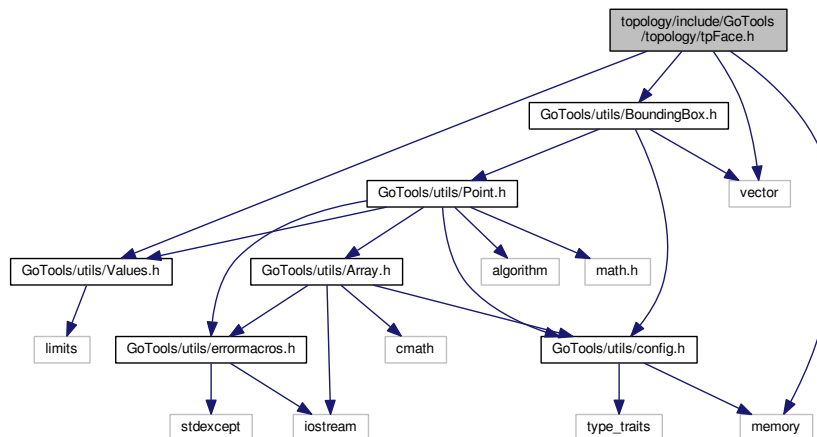
## Namespaces

- [Go](#)

## 30.2900 topology/include/GoTools/topology\_traits/tpFace.h File Reference

```
#include "GoTools/Utils/Values.h"
#include "GoTools/Utils/BoundingBox.h"
#include <memory>
#include <vector>
```

Include dependency graph for `tpFace.h`:



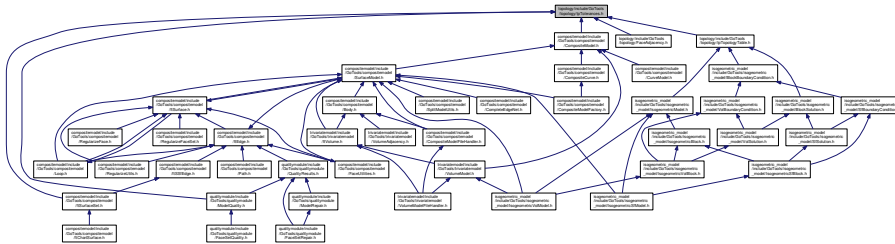


Enumerations

- enum Go::tpJointType {  
Go::JOINT\_G1 = 0, Go::JOINT\_KINK = 1, Go::JOINT\_G0 = 2, Go::JOINT\_GAP = 3,  
Go::JOINT\_DISC = 4, Go::JOINT\_NONE = 5 }

### 30.2902 topology/include/GoTools/topology/tpTolerances.h File Reference

This graph shows which files directly or indirectly include this file:



Classes

- struct Go::tpTolerances

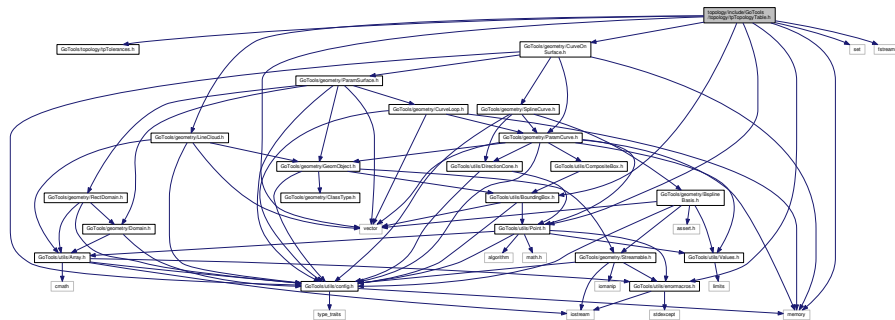
Namespaces

- Go

### 30.2903 topology/include/GoTools/topology/tpTopologyTable.h File Reference

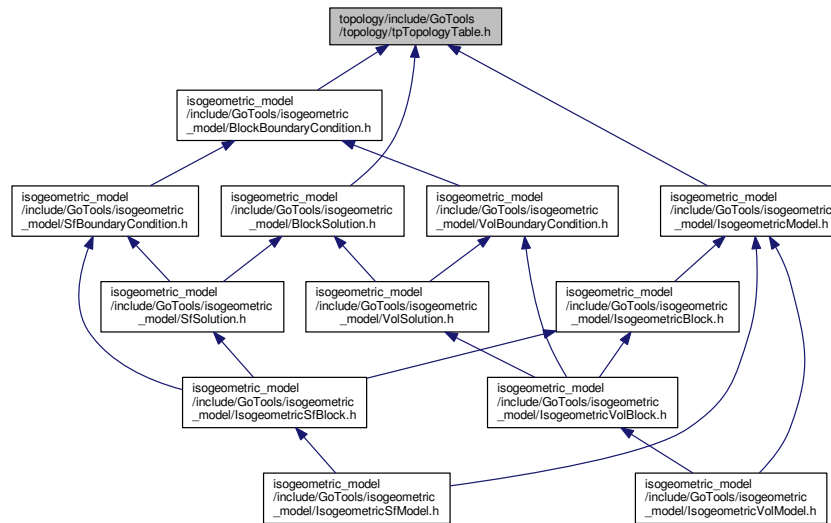
```
#include "GoTools/topology/tpTolerances.h"
#include "GoTools/utils/Point.h"
#include "GoTools/utils/BoundingBox.h"
#include "GoTools/utils/errormacros.h"
#include "GoTools/geometry/LineCloud.h"
#include "GoTools/geometry/CurveOnSurface.h"
#include <vector>
#include <set>
#include <memory>
#include <fstream>
```

Include dependency graph for `tpTopologyTable.h`:





This graph shows which files directly or indirectly include this file:



## Classes

- struct [Go::tpTopologicalInfo](#)
- class [Go::tpTableEntry< edgeType >](#)
- class [Go::tpMarchPoint](#)  
*Helper class for marching.*
- class [Go::tpTopologyTable< edgeType, faceType >](#)

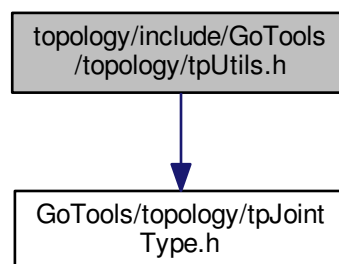
## Namespaces

- [Go](#)

## 30.2904 topology/include/GoTools/topology/tpUtils.h File Reference

```
#include "GoTools/topology/tpJointType.h"
```

Include dependency graph for tpUtils.h:



## Namespaces

- [Go](#)
- [Go::tpUtils](#)

## Functions

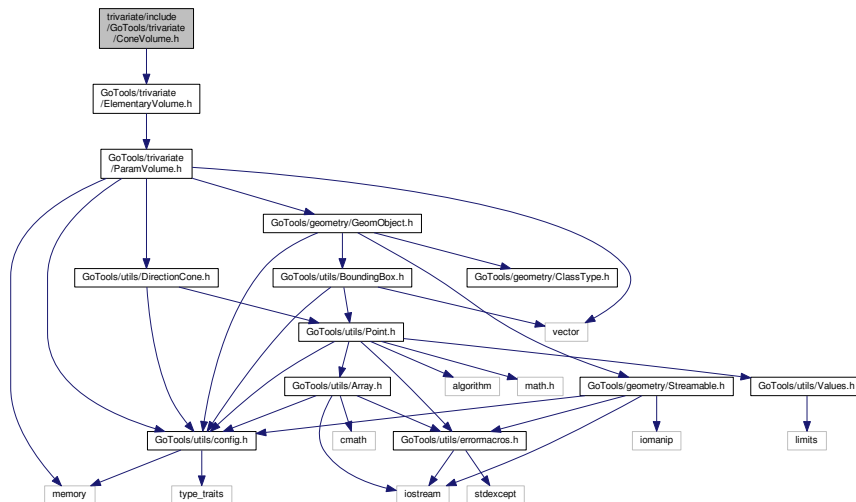
- `template<class edgeType >`  
`void Go::tpUtils::adjacentEdges (edgeType *edge, bool at_start_of_edge, std::vector< edgeType * > &adjacent, std::vector< bool > &at_start)`
- `template<class edgeType >`  
`tpJointType Go::tpUtils::checkContinuity (const edgeType *edge1, const edgeType *edge2, double neighbour, double gap, double bend, double kink)`

*Return continuity between end of edge1 and start of edge2.*

## 30.2905 trivariate/include/GoTools/trivariate/ConeVolume.h File Reference

```
#include "GoTools/trivariate/ElementaryVolume.h"
```

Include dependency graph for ConeVolume.h:



## Classes

- class [Go::ConeVolume](#)

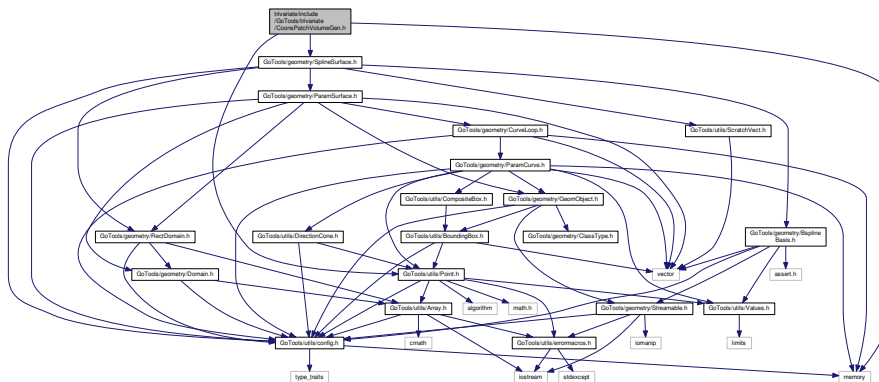
*Class that represents a solid cone. It is a subclass of [ElementaryVolume](#), and has a natural parametrization in terms of a radius  $u$ , an angle  $v$ , and distance  $w$ :  $\mathbf{p}(u, v, w) = \mathbf{C} + u (R + w \tan \alpha)((\cos v) \mathbf{x} + (\sin v) \mathbf{y}) + w \mathbf{z}$ , where  $\mathbf{C}$  is the cone apex,  $R$  is the radius when  $w = 0$ ,  $\alpha$  is the cone angle, and  $\mathbf{x}$ ,  $\mathbf{y}$  and  $\mathbf{z}$  are the (local) axes. The parametrization is bounded by:  $0 \leq u \leq 1$ ,  $0 \leq v \leq 2\pi$ ,  $-\infty < w < \infty$ . The dimension is 3.*

## Namespaces

- [Go](#)

### 30.2906 trivariate/include/GoTools/trivariate/CoonsPatchVolumeGen.h File Reference

```
#include <memory>
#include "GoTools/utils/Point.h"
#include "GoTools/geometry/SplineSurface.h"
Include dependency graph for CoonsPatchVolumeGen.h:
```



#### Namespaces

- [Go](#)
- [Go::CoonsPatchVolumeGen](#)

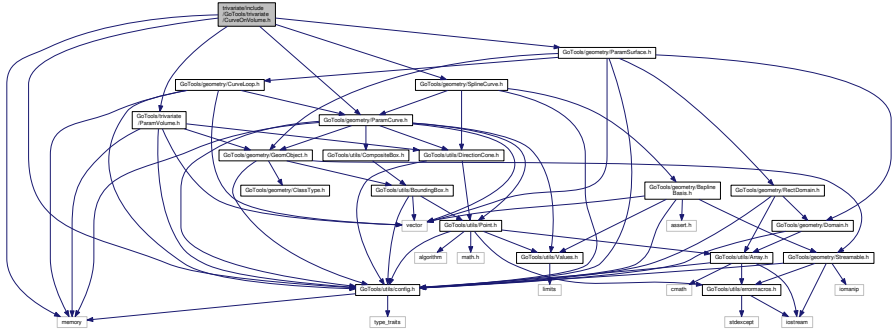
*This namespace contains functions used to create a Coons Patch volume.*

#### Functions

- `void Go::CoonsPatchVolumeGen::get\_corners (shared_ptr< Go::SplineSurface > surf, std::vector< Go::Point > &pts)`  
*Used internally in the geometry construction.*
- `void Go::CoonsPatchVolumeGen::push\_corners (std::vector< Go::Point > &pts_to, const std::vector< Go::Point > pts_from, int pos0, int pos1, int pos2, int pos3)`  
*Used internally in the geometry construction.*
- `bool Go::CoonsPatchVolumeGen::edge\_curves\_equal (shared_ptr< Go::SplineSurface > sf1, double par1, bool is_u_dir1, shared_ptr< Go::SplineSurface > sf2, double par2, bool is_u_dir2, double tol)`  
*Used internally in the geometry construction.*
- `SplineVolume * Go::CoonsPatchVolumeGen::createCoonsPatchDirectly (const Go::SplineSurface *surf_u_min, const Go::SplineSurface *surf_u_max, const Go::SplineSurface *surf_v_min, const Go::SplineSurface *surf_v_max, const Go::SplineSurface *surf_w_min, const Go::SplineSurface *surf_w_max)`
- `SplineVolume * Go::CoonsPatchVolumeGen::createCoonsPatch (const Go::SplineSurface *surf_u_min, const Go::SplineSurface *surf_u_max, const Go::SplineSurface *surf_v_min, const Go::SplineSurface *surf_v_max, const Go::SplineSurface *surf_w_min, const Go::SplineSurface *surf_w_max, double tol=1e-05)`

# 30.2907 trivariate/include/GoTools/trivariate/CurveOnVolume.h File Reference

```
#include "GoTools/geometry/ParamSurface.h"
#include <memory>
#include "GoTools/geometry/ParamCurve.h"
#include "GoTools/geometry/SplineCurve.h"
#include "GoTools/trivariate/ParamVolume.h"
#include "GoTools/Utils/config.h"
Include dependency graph for CurveOnVolume.h:
```



## Classes

- class [Go::CurveOnVolume](#)  
 A curve living on a parametric volume. It either has got information about the curve in geometry space and in the parameter domain of the volume or the ability to compute the other representation given one.

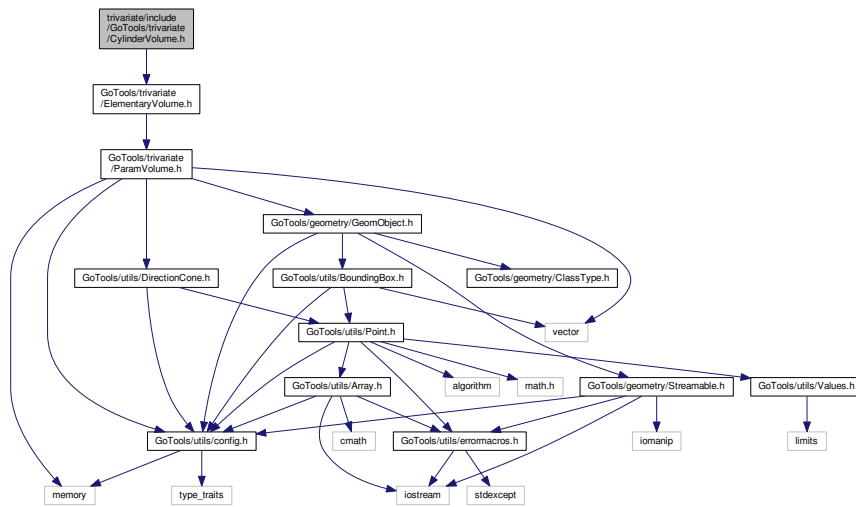
## Namespaces

- [Go](#)

# 30.2908 trivariate/include/GoTools/trivariate/CylinderVolume.h File Reference

```
#include "GoTools/trivariate/ElementaryVolume.h"
```

Include dependency graph for CylinderVolume.h:



## Classes

- class `Go::CylinderVolume`

Class that represents a solid cylinder, maybe with empty interior like a straight tube. It is a subclass of [ElementaryVolume](#), and has a natural parametrization in terms of a radius  $u$ , an angle  $v$ , and distance  $w$ :  $\mathbf{p}(u, v, w) = \mathbf{C} + u(\cos(v)\mathbf{x} + \sin(v)\mathbf{y}) + w\mathbf{z}$ , where  $\mathbf{C}$  is a position vector, and  $\mathbf{x}$ ,  $\mathbf{y}$  and  $\mathbf{z}$  are the (local) axes. The parametrization is bounded by:  $R \leq u \leq S$ ,  $0 \leq v \leq 2\pi$ ,  $-\infty < w < \infty$ , where  $R$  and  $S$  are the inner and outer radius. The dimension is 3.

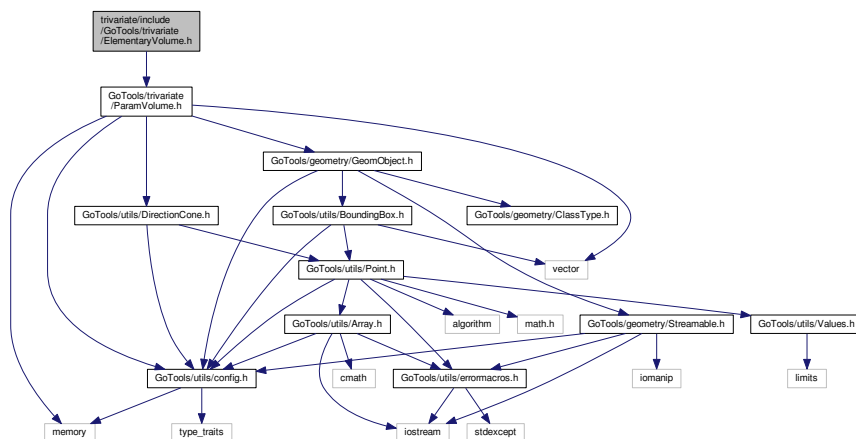
## Namespaces

- `Go`

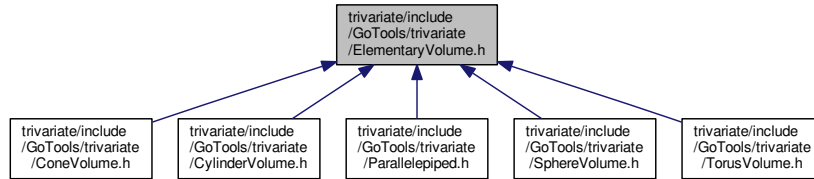
## 30.2909 trivariate/include/GoTools/trivariate/ElementaryVolume.h File Reference

```
#include "GoTools/trivariate/ParamVolume.h"
```

Include dependency graph for ElementaryVolume.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [Go::ElementaryVolume](#)

*ElementaryVolume* is a base class for elementary volumes like boxes and solid cylinders. Such volumes have natural parametrizations and *ElementaryVolume* is therefore a subclass of *ParamVolume*. These volumes are non-self-intersecting.

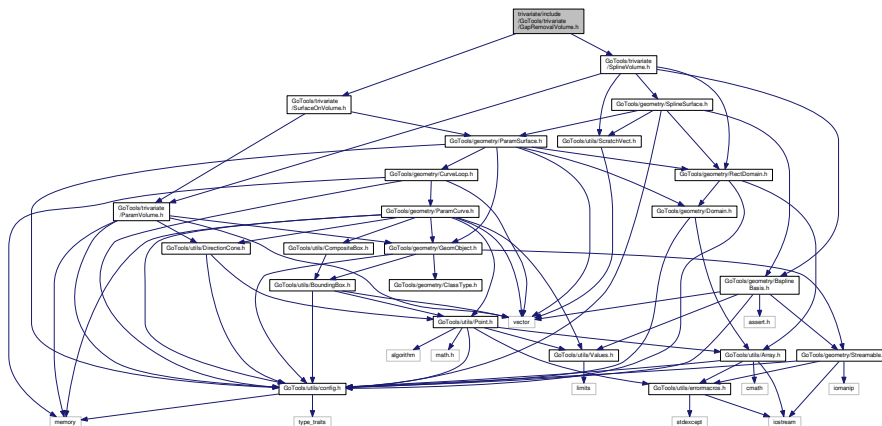
## Namespaces

- [Go](#)

### 30.2910 trivariate/include/GoTools/trivariate/examples\_trivariate\_doxyman.h File Reference

### 30.2911 trivariate/include/GoTools/trivariate/GapRemovalVolume.h File Reference

```
#include "GoTools/trivariate/SplineVolume.h"
#include "GoTools/trivariate/SurfaceOnVolume.h"
Include dependency graph for GapRemovalVolume.h:
```



## Namespaces

- [Go](#)
- [Go::GapRemoval](#)

Functionality for removal of gaps between two adjacent surfaces of various types.

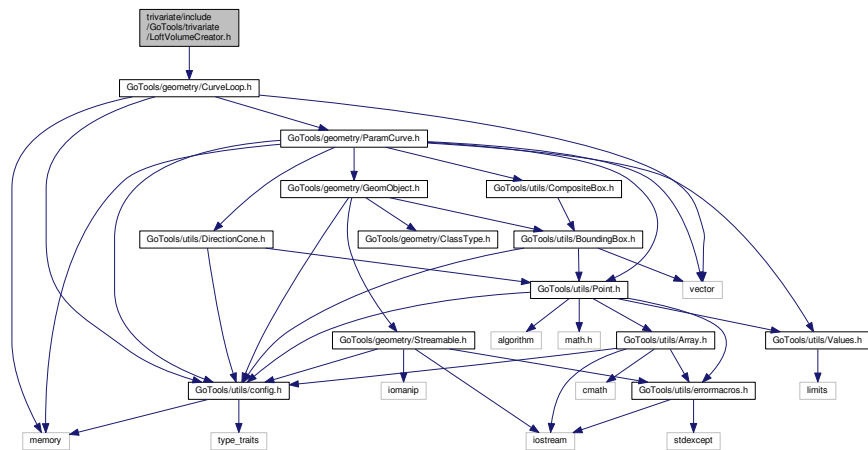
## Functions

- void [Go::GapRemoval::removeGapSpline](#) (shared\_ptr< SplineVolume > &vol1, shared\_ptr< SurfaceOnVolume > &bd\_sf1, double sf1\_start1, double sf1\_end1, double sf1\_start2, double sf1\_end2, shared\_ptr< SplineVolume > &vol2, shared\_ptr< SurfaceOnVolume > &bd\_sf2, double sf2\_start1, double sf2\_end1, double sf2\_start2, double sf2\_end2, Point vertex\_ll, Point vertex\_ur, double epsge, int orientation)

## 30.2912 trivariate/include/GoTools/trivariate/LoftVolumeCreator.h File Reference

```
#include "GoTools/geometry/CurveLoop.h"
```

Include dependency graph for LoftVolumeCreator.h:



## Namespaces

- [Go](#)
- [Go::LoftVolumeCreator](#)

This namespace contains functions used to create lofted volumes.

## Functions

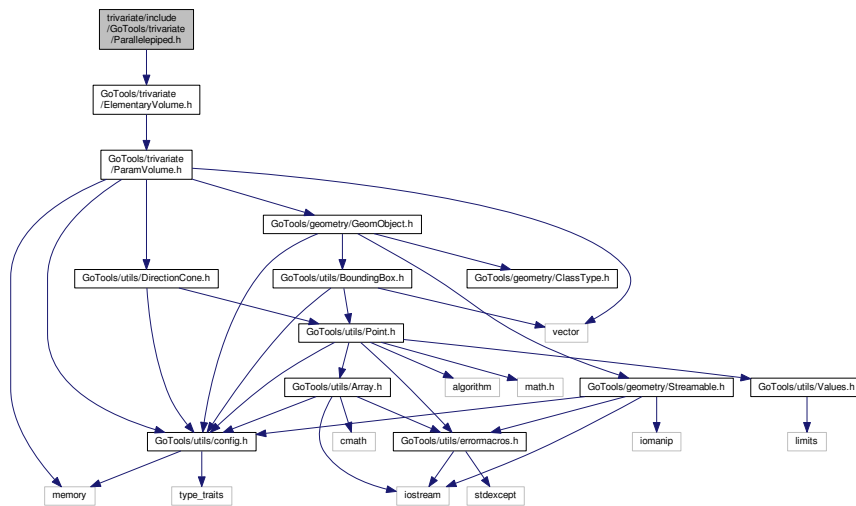
- SplineVolume \* [Go::LoftVolumeCreator::loftVolume](#) (std::vector< shared\_ptr< SplineSurface > >::iterator first\_surface, int nmb\_srf)
- SplineVolume \* [Go::LoftVolumeCreator::loftVolume](#) (std::vector< shared\_ptr< SplineSurface > >::iterator first\_surface, std::vector< double >::iterator first\_param, int nmb\_srf)
- std::vector< shared\_ptr< SplineSurface > > [Go::LoftVolumeCreator::unifiedSurfacesCopy](#) (std::vector< shared\_ptr< SplineSurface > >::iterator first\_surface, int nmb\_srf)

- void [Go::LoftVolumeCreator::makeLoftParams](#) (std::vector< shared\_ptr< SplineSurface > >::const\_iterator first\_surface, int nmb\_crvs, double param\_length, std::vector< double > &params)
- SplineVolume \* [Go::LoftVolumeCreator::loftVolumeFromUnifiedSurfaces](#) (std::vector< shared\_ptr< SplineSurface > >::iterator first\_surface, std::vector< double >::iterator first\_param, int nmb\_srf)
- SplineVolume \* [Go::LoftVolumeCreator::loftNonrationalVolume](#) (std::vector< shared\_ptr< SplineSurface > >::iterator first\_surface, std::vector< double >::iterator first\_param, int nmb\_srf)
- SplineVolume \* [Go::LoftVolumeCreator::loftRationalVolume](#) (std::vector< shared\_ptr< SplineSurface > >::iterator first\_surface, std::vector< double >::iterator first\_param, int nmb\_srf)

### 30.2913 trivariate/include/GoTools/trivariate/Parallelepiped.h File Reference

```
#include "GoTools/trivariate/ElementaryVolume.h"
```

Include dependency graph for `Parallelepiped.h`:



### Classes

- class [Go::Parallelepiped](#)

*Class that represents a solid parallelepiped. It is a subclass of [ElementaryVolume](#), and thus has a parametrization.*

### Namespaces

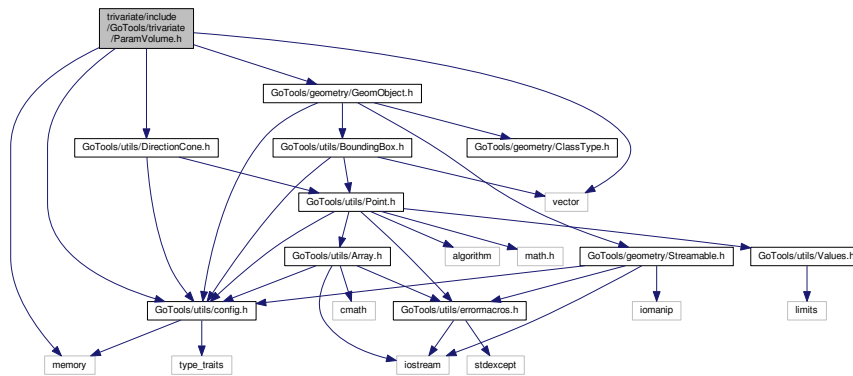
- [Go](#)



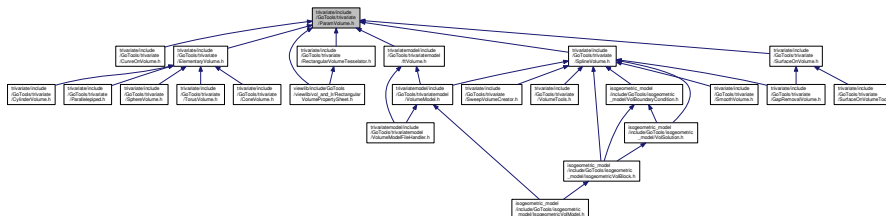
## 30.2914 trivariate/include/GoTools/trivariate/ParamVolume.h File Reference

```
#include "GoTools/Utils/config.h"
#include "GoTools/Utils/DirectionCone.h"
#include "GoTools/geometry/GeomObject.h"
#include <memory>
#include <vector>
```

Include dependency graph for ParamVolume.h:



This graph shows which files directly or indirectly include this file:



### Classes

- class [Go::ParamVolume](#)

### Namespaces

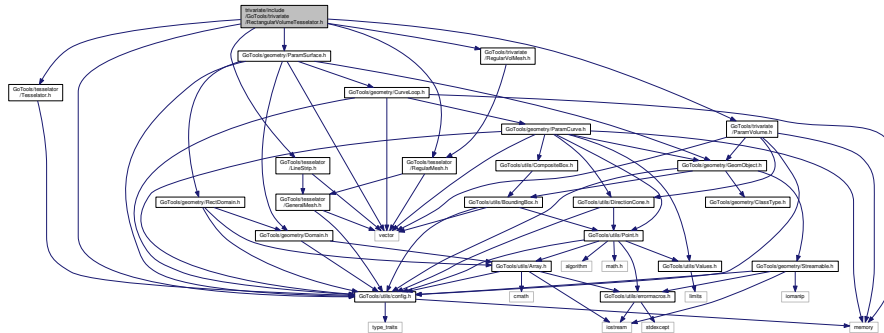
- [Go](#)

## 30.2915 trivariate/include/GoTools/trivariate/RectangularVolumeTesselator.h File Reference

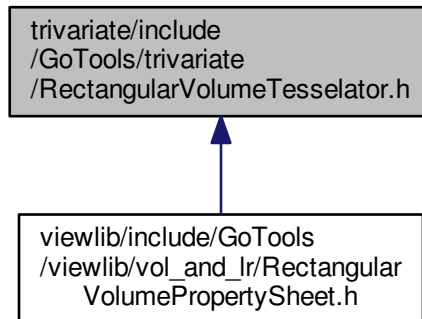
```
#include "GoTools/tesselator/Tesselator.h"
```

```
#include "GoTools/tesselator/RegularMesh.h"
#include "GoTools/tesselator/LineStrip.h"
#include "GoTools/geometry/ParamSurface.h"
#include "GoTools/trivariate/ParamVolume.h"
#include "GoTools/trivariate/RegularVolMesh.h"
#include "GoTools/utils/config.h"
```

Include dependency graph for RectangularVolumeTesselator.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [Go::RectangularVolumeTesselator](#)

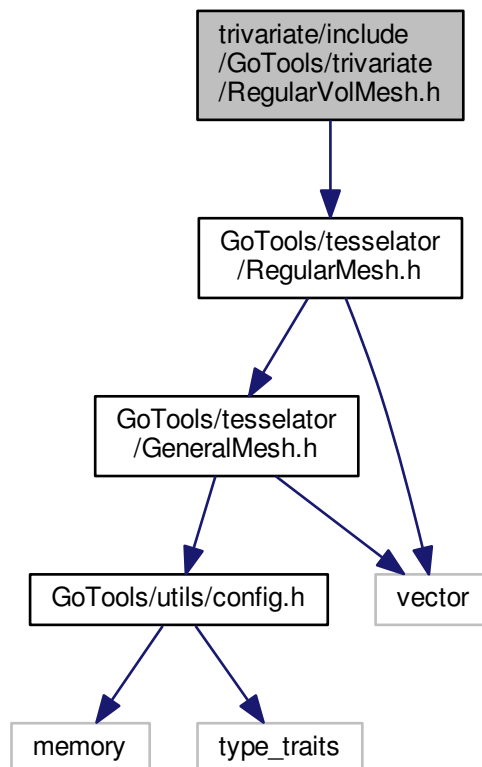
## Namespaces

- [Go](#)

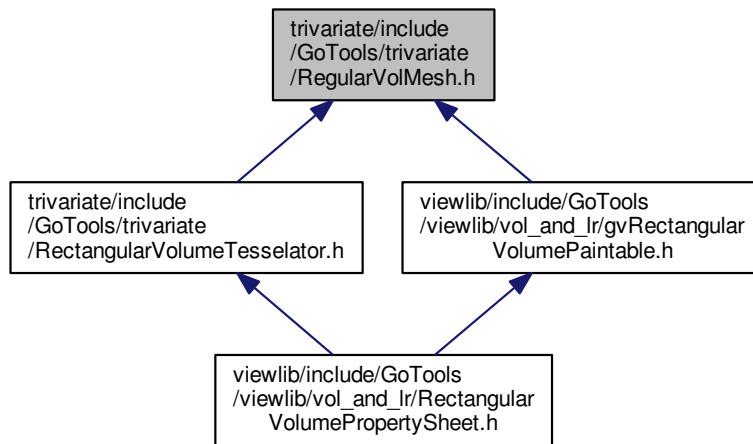
## 30.2916 trivariate/include/GoTools/trivariate/RegularVolMesh.h File Reference

```
#include "GoTools/tesselator/RegularMesh.h"
```

Include dependency graph for RegularVolMesh.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [Go::RegularVolMesh](#)

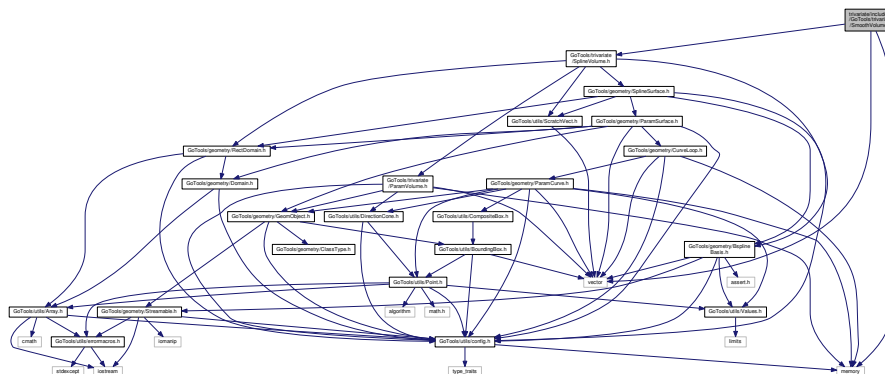
## Namespaces

- [Go](#)

## 30.2917 trivariate/include/GoTools/trivariate/SmoothVolume.h File Reference

```
#include "GoTools/trivariate/SplineVolume.h"
#include <memory>
#include <vector>
```

Include dependency graph for SmoothVolume.h:

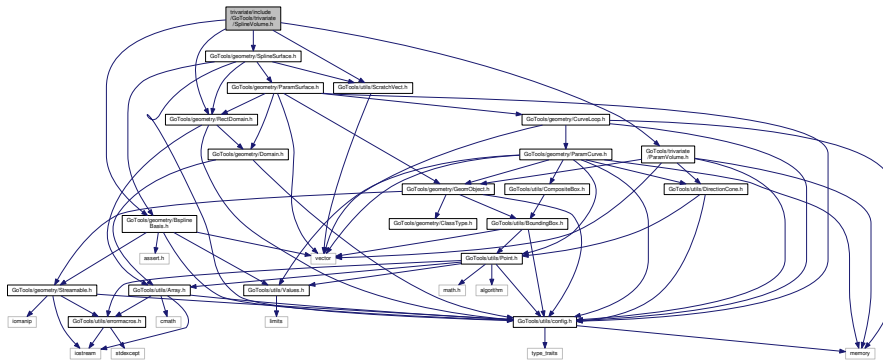




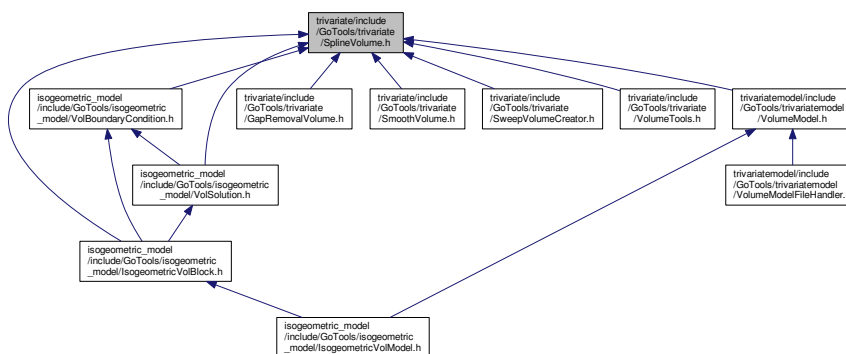
### 30.2919 trivariate/include/GoTools/trivariate/SplineVolume.h File Reference

```
#include "GoTools/trivariate/ParamVolume.h"
#include "GoTools/geometry/BsplineBasis.h"
#include "GoTools/geometry/SplineSurface.h"
#include "GoTools/geometry/RectDomain.h"
#include "GoTools/utils/ScratchVect.h"
```

Include dependency graph for SplineVolume.h:



This graph shows which files directly or indirectly include this file:



#### Classes

- struct [Go::BasisPts](#)
- struct [Go::BasisDerivs](#)
- struct [Go::BasisDerivs2](#)
- class [Go::SplineVolume](#)

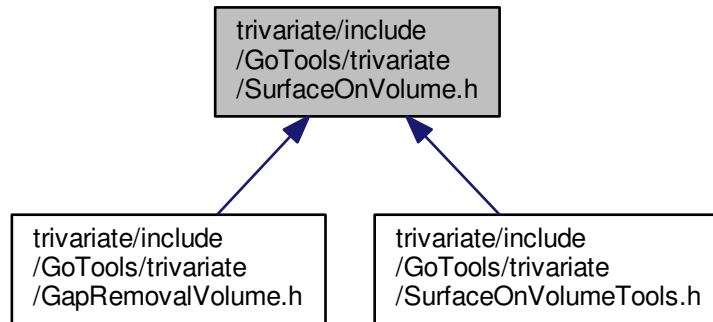
*SplineVolume* provides methodes for storing, reading and manipulating rational and non-rational B-spline volumes.

#### Namespaces

- [Go](#)



This graph shows which files directly or indirectly include this file:



## Classes

- class [Go::SurfaceOnVolume](#)

*A surface living on a parametric volume. It either has got information about the surface in geometry space and in the parameter domain of the volume or both. The surface may have information on whether it is a constant parameter or boundary surface on the volume.*

## Namespaces

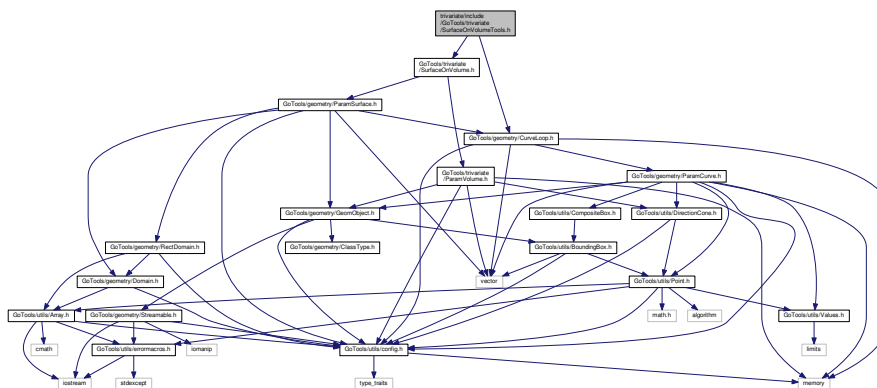
- [Go](#)

## 30.2921 trivariate/include/GoTools/trivariate/SurfaceOnVolumeTools.h File Reference

```
#include "GoTools/trivariate/SurfaceOnVolume.h"
```

```
#include "GoTools/geometry/CurveLoop.h"
```

Include dependency graph for SurfaceOnVolumeTools.h:





Namespaces

- [Go](#)
- [Go::SurfaceOnVolumeTools](#)

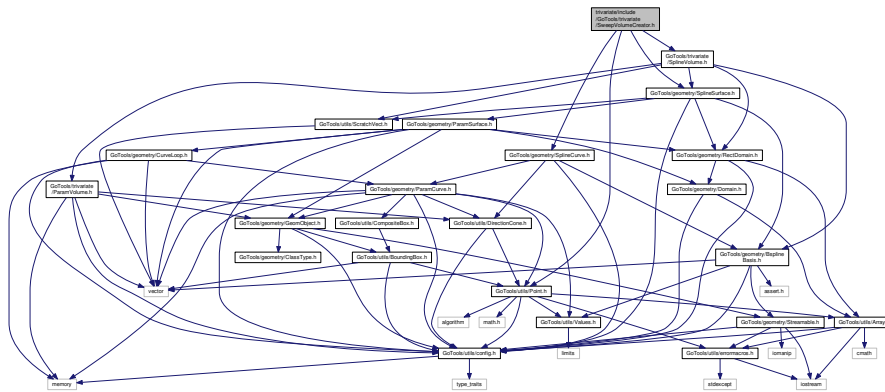
Functions

- [CurveLoop](#) [Go::SurfaceOnVolumeTools::getOuterBoundaryLoop](#) (shared\_ptr< SurfaceOnVolume > sf, double eps)

30.2922 trivariate/include/GoTools/trivariate/SweepVolumeCreator.h File Reference

```
#include "GoTools/trivariate/SplineVolume.h"
#include "GoTools/geometry/SplineSurface.h"
#include "GoTools/geometry/SplineCurve.h"
#include "GoTools/utils/Point.h"
```

Include dependency graph for SweepVolumeCreator.h:



Classes

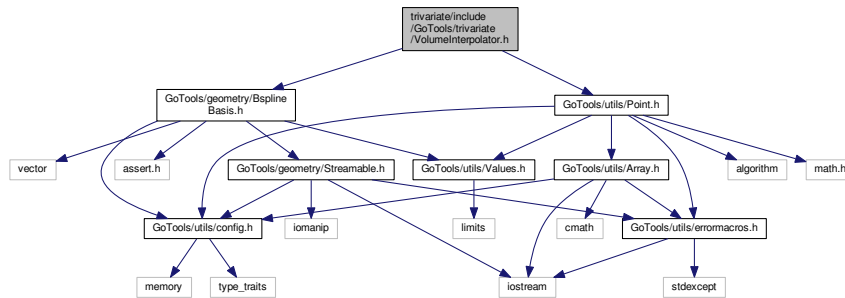
- class [Go::SweepVolumeCreator](#)  
Class with static methods for volume creation by sweep methods.

Namespaces

- [Go](#)



Include dependency graph for VolumeInterpolator.h:



## Namespaces

- [Go](#)
- [Go::VolumeInterpolator](#)

*This namespace contains functions used to interpolate a set of points.*

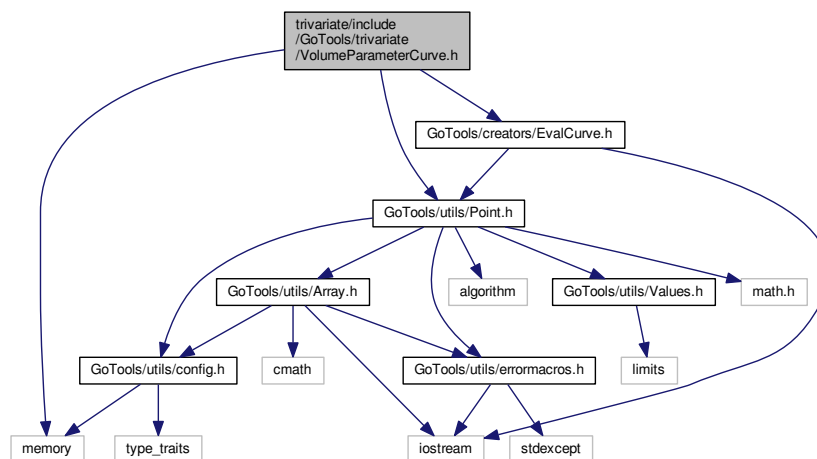
## Functions

- `SplineVolume * Go::VolumeInterpolator::regularInterpolation (const BsplineBasis &basis_u, const BsplineBasis &basis_v, const BsplineBasis &basis_w, std::vector< double > &par_u, std::vector< double > &par_v, std::vector< double > &par_w, std::vector< double > &points, int dimension, bool rational, std::vector< double > &weights)`

## 30.2926 trivariate/include/GoTools/trivariate/VolumeParameterCurve.h File Reference

```
#include <memory>
#include "GoTools/utills/Point.h"
#include "GoTools/creators/EvalCurve.h"
```

Include dependency graph for VolumeParameterCurve.h:



## Classes

- class [Go::VolumeParameterCurve](#)

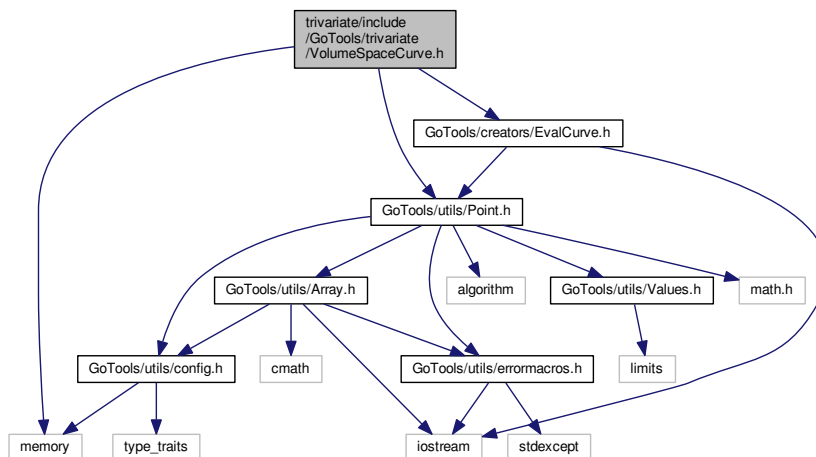
An evaluator based curve representing the parameter domain curve in a given volume which represents the same curve as a given space curve. Project a geometry curve into the parameter domain of a volume.

## Namespaces

- [Go](#)

## 30.2927 trivariate/include/GoTools/trivariate/VolumeSpaceCurve.h File Reference

```
#include <memory>
#include "GoTools/utils/Point.h"
#include "GoTools/creators/EvalCurve.h"
Include dependency graph for VolumeSpaceCurve.h:
```



## Classes

- class [Go::VolumeSpaceCurve](#)

An evaluator based curve representing the space curve corresponding to parameter domain curve in a given volume. Compute the space curve corresponding to a curve in the parameter domain of a volume.

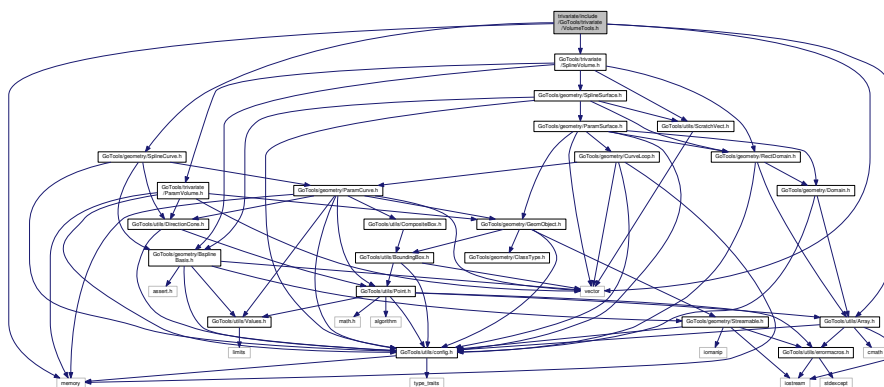
## Namespaces

- [Go](#)

## 30.2928 trivariate/include/GoTools/trivariate/VolumeTools.h File Reference

```
#include "GoTools/geometry/SplineCurve.h"
#include "GoTools/trivariate/SplineVolume.h"
#include "GoTools/utils/Array.h"
#include <memory>
#include <vector>
```

Include dependency graph for VolumeTools.h:



### Namespaces

- [Go](#)
- [Go::VolumeTools](#)

*This namespace contains free functions operating on parametric volumes.*

### Functions

- int [GO\\_API Go::VolumeTools::analyzePeriodicity](#) (const SplineVolume &sf, int direction, double knot\_tol=1e-12)
- shared\_ptr< SplineCurve > [Go::VolumeTools::representVolumeAsCurve](#) (const SplineVolume &volume, int cv\_dir)
- shared\_ptr< SplineVolume > [Go::VolumeTools::representCurveAsVolume](#) (const SplineCurve &curve, int cv\_dir, const BsplineBasis &other\_bas1, const BsplineBasis &other\_bas2, bool rational)
- shared\_ptr< SplineSurface > [Go::VolumeTools::representVolumeAsSurface](#) (const SplineVolume &volume, int sf\_dir1, int sf\_dir2)
- shared\_ptr< SplineVolume > [Go::VolumeTools::representSurfaceAsVolume](#) (const SplineSurface &surface, int sf\_dir1, int sf\_dir2, const BsplineBasis &other\_bas, bool rational)
- bool [Go::VolumeTools::cornerToCornerVols](#) (shared\_ptr< ParamVolume > vol1, shared\_ptr< SurfaceOnVolume > vol\_sf1, shared\_ptr< ParamVolume > vol2, shared\_ptr< SurfaceOnVolume > vol\_sf2, double tol)
- bool [Go::VolumeTools::getVolAdjacencyInfo](#) (shared\_ptr< ParamVolume > vol1, shared\_ptr< SurfaceOnVolume > vol\_sf1, shared\_ptr< ParamVolume > vol2, shared\_ptr< SurfaceOnVolume > vol\_sf2, double tol, int &bd1, int &bd2, int &orientation, bool &same\_seq)
- bool [Go::VolumeTools::getCorrCoefVolEnum](#) (shared\_ptr< SplineVolume > vol1, shared\_ptr< SplineVolume > vol2, int bd1, int bd2, int orientation, bool same\_seq, std::vector< std::pair< int, int > > &enumeration)
- bool [Go::VolumeTools::getVolCoefEnumeration](#) (shared\_ptr< SplineVolume > vol, int bd, std::vector< int > &enumeration)
- bool [Go::VolumeTools::getVolCoefEnumeration](#) (shared\_ptr< SplineVolume > vol, int bd, std::vector< int > &enumeration\_bd, std::vector< int > &enumeration\_bd2)

- [bool Go::VolumeTools::getVolBdCoefEnumeration](#) (shared\_ptr< SplineVolume > vol, int bd, int bd\_cv, std::vector< int > &enumeration)
- [std::vector< shared\\_ptr< ParamSurface > > Go::VolumeTools::getBoundarySurfaces](#) (shared\_ptr< ParamVolume > vol)
 

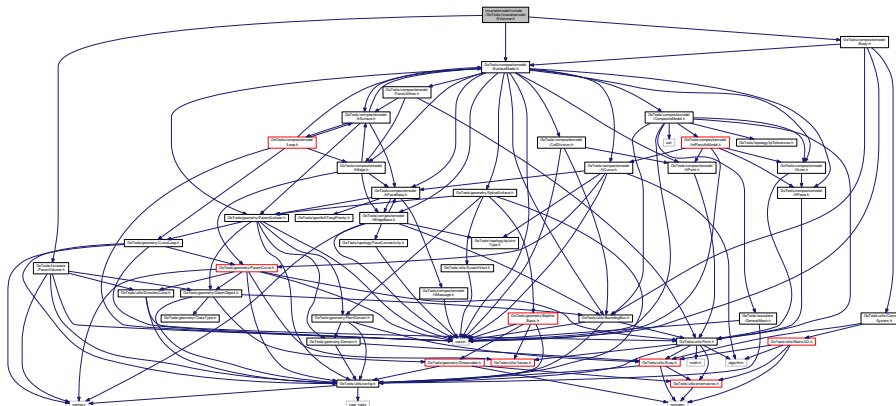
*Given a parametric volume, fetch all boundary surfaces.*
- [std::vector< shared\\_ptr< ParamSurface > > Go::VolumeTools::getOrientedBoundarySurfaces](#) (shared\_ptr< ParamVolume > vol)
- [shared\\_ptr< SurfaceOnVolume > Go::VolumeTools::getBoundarySurface](#) (shared\_ptr< SplineVolume > vol, int idx)
 

*Given a spline volume, fetch all boundary surfaces.*
- [shared\\_ptr< SurfaceOnVolume > Go::VolumeTools::getOrientedBoundarySurface](#) (shared\_ptr< SplineVolume > vol, int idx)
- [void Go::VolumeTools::volCommonSplineSpace](#) (shared\_ptr< SplineVolume > vol1, int bd1, shared\_ptr< SplineVolume > vol2, int bd2, int orientation, bool same\_seq)
- [shared\\_ptr< SplineCurve > Go::VolumeTools::liftVolParamCurve](#) (shared\_ptr< ParamCurve > pcurve, shared\_ptr< ParamVolume > vol, double tol)
- [shared\\_ptr< SplineCurve > Go::VolumeTools::projectVolParamCurve](#) (shared\_ptr< ParamCurve > spacecurve, shared\_ptr< ParamVolume > vol, double tol)
- [shared\\_ptr< SplineCurve > Go::VolumeTools::approxVolParamCurve](#) (shared\_ptr< ParamCurve > spacecurve, shared\_ptr< ParamVolume > vol, double tol, int max\_iter, double &maxdist)

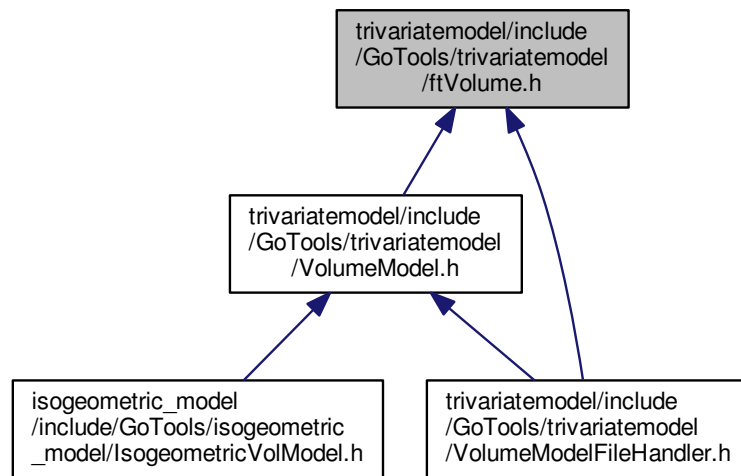
### 30.2929 trivariatemodel/include/GoTools/trivariatemodel/examples\_trivariatemodel\_doxygen.h File Reference

### 30.2930 trivariatemodel/include/GoTools/trivariatemodel/ftVolume.h File Reference

```
#include "GoTools/trivariate/ParamVolume.h"
#include "GoTools/compositemodel/SurfaceModel.h"
#include "GoTools/compositemodel/Body.h"
Include dependency graph for ftVolume.h:
```



This graph shows which files directly or indirectly include this file:



## Classes

- struct [Go::VolumeAdjacencyInfo](#)

*Struct to store information about adjacency relations between two bodies.*

- class [Go::ftVolume](#)

*A topological solid with a trivariate geometry description.*

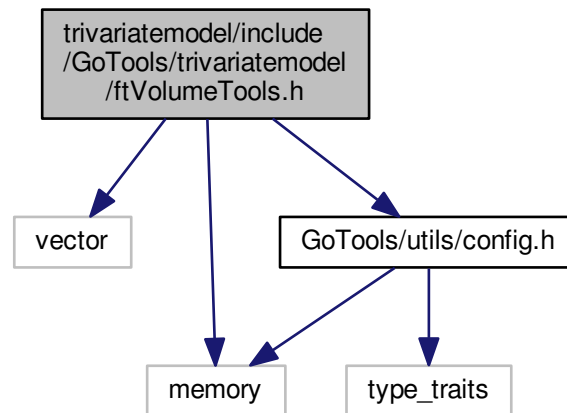
## Namespaces

- [Go](#)

## 30.2931 trivariatemodel/include/GoTools/trivariatemodel/ftVolumeTools.h File Reference

```
#include <vector>
#include <memory>
#include "GoTools/utils/config.h"
```

Include dependency graph for `ftVolumeTools.h`:



## Namespaces

- [Go](#)
- [Go::ftVolumeTools](#)

*This namespace contains service functions related to `ftVolume`.*

## Functions

- `std::vector< shared_ptr< ftVolume > >` [Go::ftVolumeTools::splitVolumes](#) (`shared_ptr< ftVolume > &vol1`, `shared_ptr< ftVolume > &vol2`, `double eps`, `std::vector< int > &config`)
- `std::vector< shared_ptr< ftVolume > >` [Go::ftVolumeTools::splitVolumes](#) (`shared_ptr< ftVolume > &vol`, `shared_ptr< ftSurface > &face`, `double eps`)

*Split one volume according to intersections with a given face.*

- `void` [Go::ftVolumeTools::updateWithSplitFaces](#) (`shared_ptr< SurfaceModel > shell`, `shared_ptr< ftSurface > &face1`, `shared_ptr< ftSurface > &face2`, `std::vector< std::pair< ftEdge *, ftEdge * > > &replaced_wires`)

*Specific functionality. Used from `ftVolume::generateMissingBdSurf`.*

- `int` [Go::ftVolumeTools::boundaryStatus](#) (`ftVolume *vol`, `shared_ptr< ftSurface > &bd_face`, `double tol`)

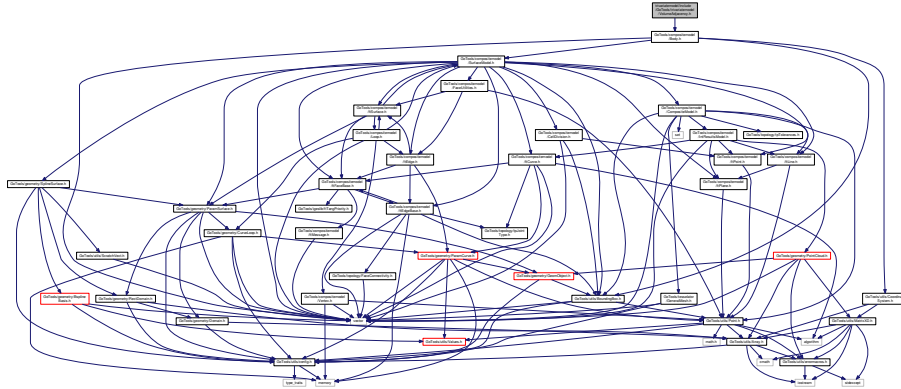
## 30.2932 trivariatemodel/include/GoTools/trivariatemodel/trivariatemodel\_doxygen.h

### File Reference



## 30.2933 trivariatemodel/include/GoTools/trivariatemodel/VolumeAdjacency.h File Reference

```
#include "GoTools/compositemodel/Body.h"
Include dependency graph for VolumeAdjacency.h:
```



### Classes

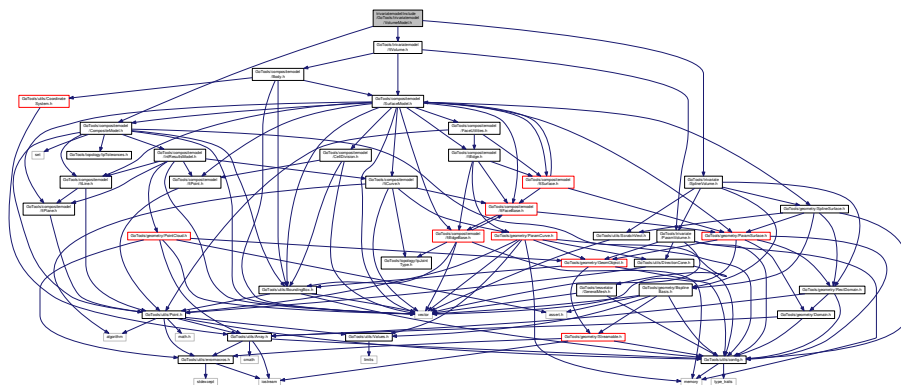
- class [Go::VolumeAdjacency](#)  
*Adjacency analysis of volume models.*

### Namespaces

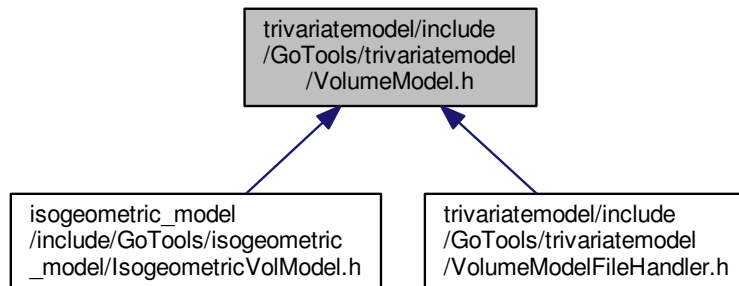
- [Go](#)

## 30.2934 trivariatemodel/include/GoTools/trivariatemodel/VolumeModel.h File Reference

```
#include "GoTools/compositemodel/CompositeModel.h"
#include "GoTools/trivariatemodel/ftVolume.h"
#include "GoTools/trivariate/SplineVolume.h"
Include dependency graph for VolumeModel.h:
```



This graph shows which files directly or indirectly include this file:



## Classes

- class [Go::VolumeModel](#)  
*A set of volumes including topology information.*

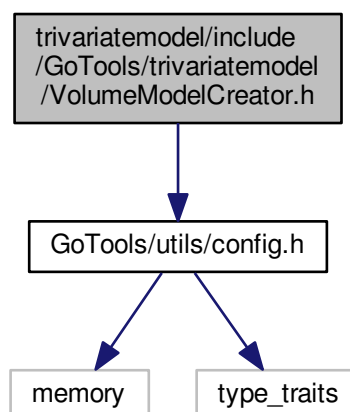
## Namespaces

- [Go](#)

## 30.2935 trivariatemodel/include/GoTools/trivariatemodel/VolumeModelCreator.h File Reference

```
#include "GoTools/utils/config.h"
```

Include dependency graph for VolumeModelCreator.h:



## Namespaces

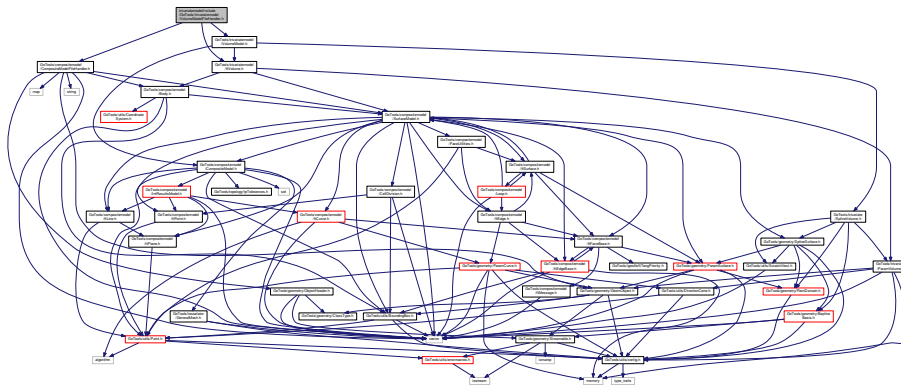
- [Go](#)
- [Go::VolumeModelCreator](#)

## Functions

- [bool Go::VolumeModelCreator::createRotationalModel](#) (shared\_ptr< SurfaceModel > &sfmodel, shared\_ptr< VolumeModel > &volmodel)
- [bool Go::VolumeModelCreator::linearSweptModel](#) (shared\_ptr< SurfaceModel > &sfmodel, shared\_ptr< VolumeModel > &volmodel)

## 30.2936 trivariatemodel/include/GoTools/trivariatemodel/VolumeModelFileHandler.h File Reference

```
#include "GoTools/compositemodel/CompositeModelFileHandler.h"
#include "GoTools/trivariatemodel/ftVolume.h"
#include "GoTools/trivariatemodel/VolumeModel.h"
Include dependency graph for VolumeModelFileHandler.h:
```



## Classes

- class [Go::VolumeModelFileHandler](#)

## Namespaces

- [Go](#)

## Macros

- [#define \\_VOLUMEFILEHANDLER\\_H](#)

### 30.2936.1 Macro Definition Documentation

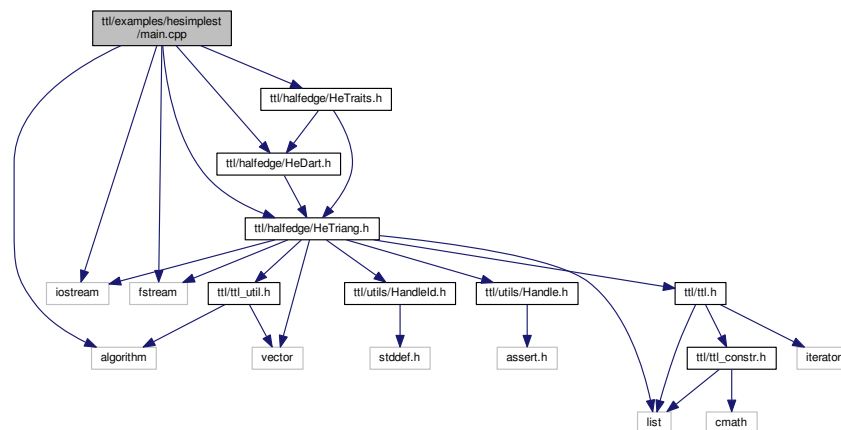
#### 30.2936.1.1 #define \_VOLUMEFILEHANDLER\_H

Definition at line 41 of file VolumeModelFileHandler.h.

### 30.2937 ttl/examples/hesimplest/main.cpp File Reference

```
#include <ttl/halfedge/HeTriang.h>
#include <ttl/halfedge/HeDart.h>
#include <ttl/halfedge/HeTraits.h>
#include <fstream>
#include <iostream>
#include <algorithm>
```

Include dependency graph for main.cpp:



### Functions

- `bool eqPoints (hed::Node *&p1, hed::Node *&p2)`
- `bool ltLexPoint (const hed::Node *p1, const hed::Node *p2)`
- `int main ()`

#### 30.2937.1 Function Documentation

##### 30.2937.1.1 `bool eqPoints ( hed::Node *& p1, hed::Node *& p2 )` [inline]

Definition at line 53 of file main.cpp.

##### 30.2937.1.2 `bool ltLexPoint ( const hed::Node * p1, const hed::Node * p2 )` [inline]

Definition at line 67 of file main.cpp.

30.2937.1.3 `int main ( )`

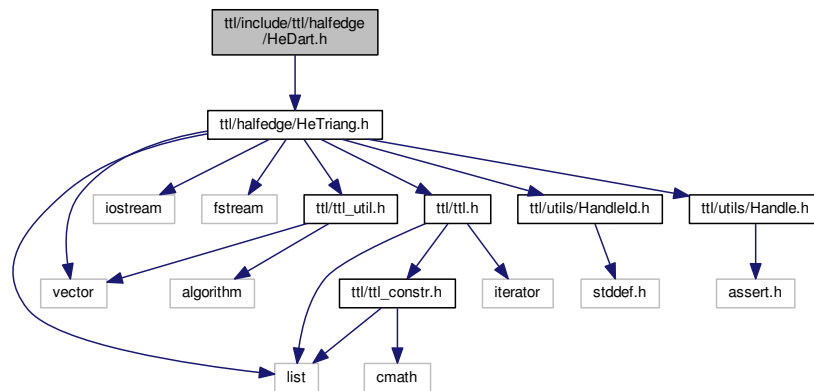
Definition at line 78 of file `main.cpp`.

### 30.2938 `ttl/include/ttl/api.h` File Reference

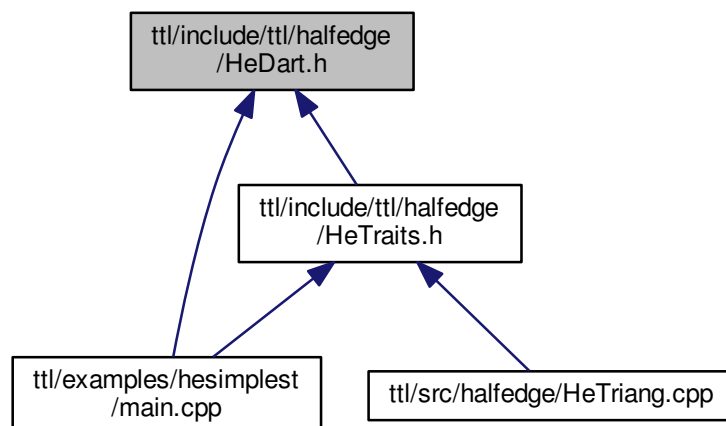
### 30.2939 `ttl/include/ttl/halfedge/HeDart.h` File Reference

```
#include <ttl/halfedge/HeTriang.h>
```

Include dependency graph for `HeDart.h`:



This graph shows which files directly or indirectly include this file:



## Classes

- class [hed::Dart](#)

*Dart* class for the half-edge data structure.

## Namespaces

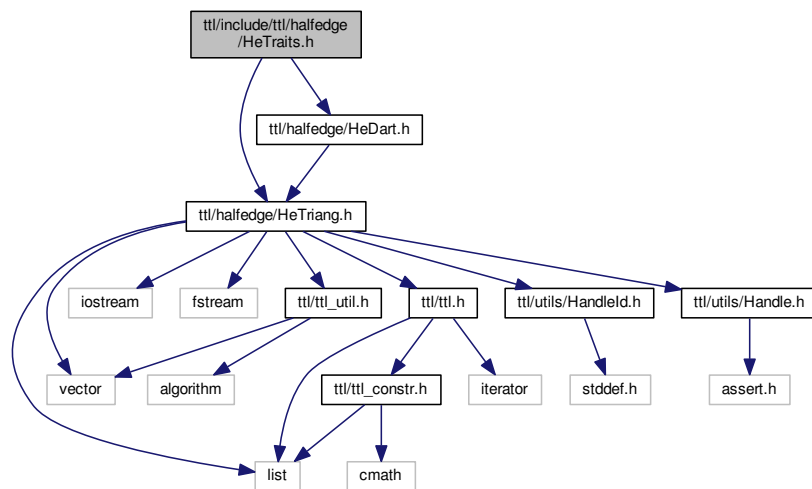
- [hed](#)

30.2940 `ttl/include/ttl/halfedge/HeTraits.h` File Reference

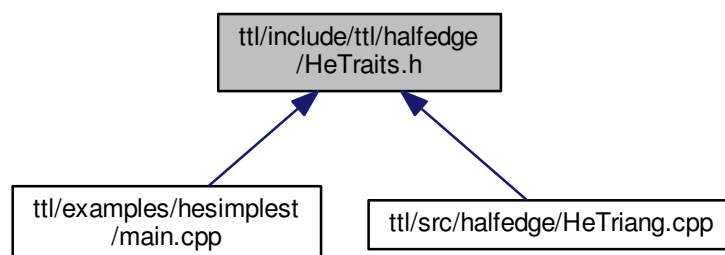
```
#include <ttl/halfedge/HeTriang.h>
```

```
#include <ttl/halfedge/HeDart.h>
```

Include dependency graph for HeTraits.h:



This graph shows which files directly or indirectly include this file:



## Classes

- struct [hed::TTLtraits](#)

*Traits* class (static struct) for the half-edge data structure.

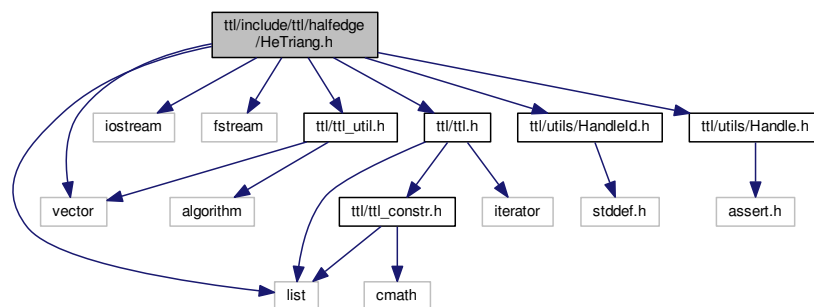
## Namespaces

- [hed](#)

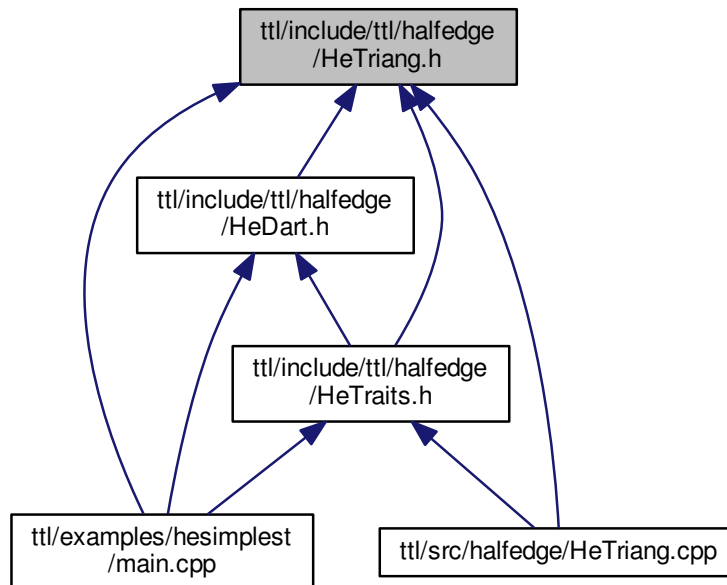
## 30.2941 ttl/include/ttl/halfedge/HeTriang.h File Reference

```
#include <list>
#include <vector>
#include <iostream>
#include <fstream>
#include <ttl/ttl.h>
#include <ttl/ttl_util.h>
#include <ttl/utils/HandleId.h>
#include <ttl/utils/Handle.h>
```

Include dependency graph for HeTriang.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [hed::Node](#)  
*Node* class for data structures (Inherits from *HandleId*)
- class [hed::Edge](#)  
*Edge* class in the in the half-edge data structure.
- class [hed::Triangulation](#)  
*Triangulation* class for the half-edge data structure with adaption to *TTL*.

## Namespaces

- [hed](#)

## Macros

- `#define TTL_USE_NODE_ID`
- `#define TTL_USE_NODE_FLAG`

### 30.2941.1 Macro Definition Documentation

#### 30.2941.1.1 `#define TTL_USE_NODE_FLAG`

Definition at line 45 of file `HeTriang.h`.



30.2941.1.2 `#define TTL_USE_NODE_ID`

Definition at line 44 of file HeTriang.h.

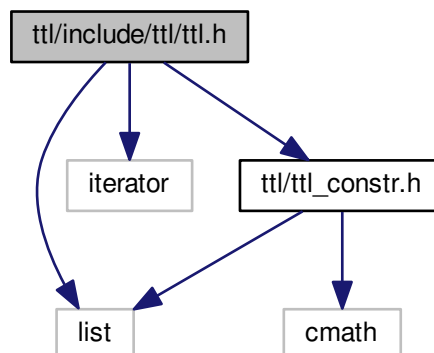
**30.2942** [ttl/include/ttl/halfedge/main\\_he\\_ref.h File Reference](#)

**30.2943** [ttl/include/ttl/halfedge/mainpage\\_hed.h File Reference](#)

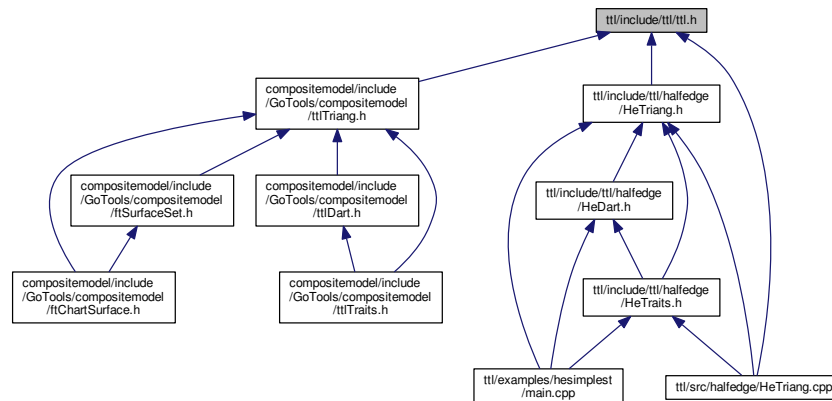
**30.2944** [ttl/include/ttl/mainpage\\_ttl.h File Reference](#)

**30.2945** [ttl/include/ttl/ttl.h File Reference](#)

```
#include <list>
#include <iterator>
#include <ttl/ttl_constr.h>
Include dependency graph for ttl.h:
```



This graph shows which files directly or indirectly include this file:



## Namespaces

- [ttl](#)

*Main interface to TTL.*

## Functions

### Delaunay Triangulation

- `template<class TraitsType , class DartType , class PointType >`  
`bool ttl::insertNode (DartType &dart, PointType &point)`
- `template<class TraitsType , class DartType >`  
`void ttl::removeRectangularBoundary (DartType &dart)`
- `template<class TraitsType , class DartType >`  
`void ttl::removeNode (DartType &dart)`
- `template<class TraitsType , class DartType >`  
`void ttl::removeBoundaryNode (DartType &dart)`
- `template<class TraitsType , class DartType >`  
`void ttl::removeInteriorNode (DartType &dart)`
- `template<class TraitsType , class ForwardIterator , class DartType >`  
`void ttl::insertNodes (ForwardIterator first, ForwardIterator last, DartType &dart)`

### Topological and Geometric Queries

- `template<class TraitsType , class PointType , class DartType >`  
`bool ttl::locateFaceSimplest (const PointType &point, DartType &dart)`
- `template<class TraitsType , class PointType , class DartType >`  
`bool ttl::locateTriangle (const PointType &point, DartType &dart)`
- `template<class TraitsType , class PointType , class DartType >`  
`bool ttl::inTriangleSimplest (const PointType &point, const DartType &dart)`
- `template<class TraitsType , class PointType , class DartType >`  
`bool ttl::inTriangle (const PointType &point, const DartType &dart)`
- `template<class DartType , class DartListType >`  
`void ttl::getBoundary (const DartType &dart, DartListType &boundary)`
- `template<class DartType >`  
`bool ttl::isBoundaryEdge (const DartType &dart)`

- `template<class DartType >`  
`bool ttl::isBoundaryFace (const DartType &dart)`
- `template<class DartType >`  
`bool ttl::isBoundaryNode (const DartType &dart)`
- `template<class DartType >`  
`int ttl::getDegreeOfNode (const DartType &dart)`
- `template<class DartType , class DartListType >`  
`void ttl::get_0_orbit_interior (const DartType &dart, DartListType &orbit)`
- `template<class DartType , class DartListType >`  
`void ttl::get_0_orbit_boundary (const DartType &dart, DartListType &orbit)`
- `template<class DartType >`  
`bool ttl::same_0_orbit (const DartType &d1, const DartType &d2)`
- `template<class DartType >`  
`bool ttl::same_1_orbit (const DartType &d1, const DartType &d2)`
- `template<class DartType >`  
`bool ttl::same_2_orbit (const DartType &d1, const DartType &d2)`
- `template<class TraitsType , class DartType >`  
`bool ttl::swappableEdge (const DartType &dart, bool allowDegeneracy=false)`
- `template<class DartType >`  
`void ttl::positionAtNextBoundaryEdge (DartType &dart)`
- `template<class TraitsType , class DartType >`  
`bool ttl::convexBoundary (const DartType &dart)`
- `template<class TopologyElementType , class DartType >`  
`bool ttl::isMemberOfFace (const TopologyElementType &topologyElement, const DartType &dart)`
- `template<class TraitsType , class NodeType , class DartType >`  
`bool ttl::locateFaceWithNode (const NodeType &node, DartType &dart_iter)`
- `template<class DartType >`  
`void ttl::getAdjacentTriangles (const DartType &dart, DartType &t1, DartType &t2, DartType &t3)`
- `template<class DartType >`  
`void ttl::getNeighborNodes (const DartType &dart, std::list< DartType > &node_list, bool &boundary)`
- `template<class TraitsType , class DartType >`  
`bool ttl::degenerateTriangle (const DartType &dart)`

### Utilities for Delaunay Triangulation

- `template<class TraitsType , class DartType , class DartListType >`  
`void ttl::optimizeDelaunay (DartListType &elist)`
- `template<class TraitsType , class DartType , class DartListType >`  
`void ttl::optimizeDelaunay (DartListType &elist, const typename DartListType::iterator end)`
- `template<class TraitsType , class DartType >`  
`bool ttl::swapTestDelaunay (const DartType &dart, bool cycling_check=false)`
- `template<class TraitsType , class DartType >`  
`void ttl::recSwapDelaunay (DartType &diagonal)`
- `template<class TraitsType , class DartType , class ListType >`  
`void ttl::swapEdgesAwayFromInteriorNode (DartType &dart, ListType &swapped_edges)`
- `template<class TraitsType , class DartType , class ListType >`  
`void ttl::swapEdgesAwayFromBoundaryNode (DartType &dart, ListType &swapped_edges)`
- `template<class TraitsType , class DartType , class DartListType >`  
`void ttl::swapEdgeInList (const typename DartListType::iterator &it, DartListType &elist)`

### Constrained (Delaunay) Triangulation

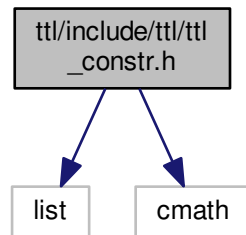
- `template<class TraitsType , class DartType >`  
`DartType ttl::insertConstraint (DartType &dstart, DartType &dend, bool optimize_delaunay)`

## 30.2946 ttl/include/ttl/ttl\_constr.h File Reference

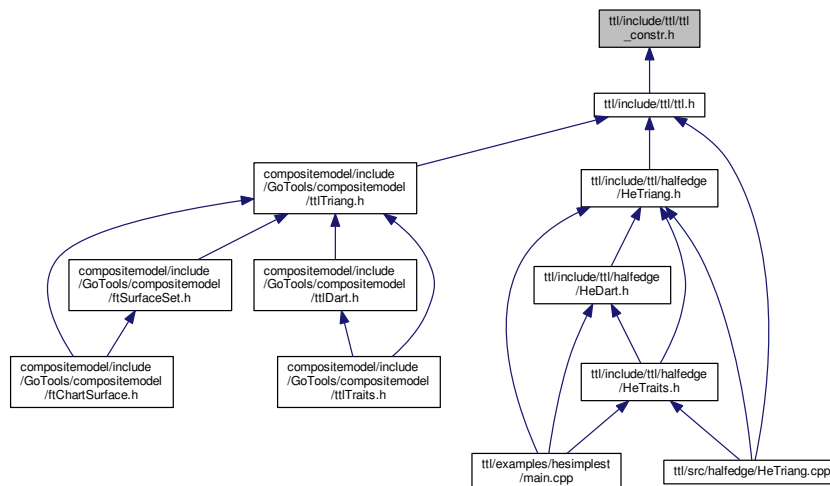
```
#include <list>
```

```
#include <cmath>
```

Include dependency graph for `ttl_constr.h`:



This graph shows which files directly or indirectly include this file:



## Namespaces

- [ttl\\_constr](#)

*Constrained Delaunay triangulation.*

- [ttl](#)

*Main interface to TTL.*

## Functions

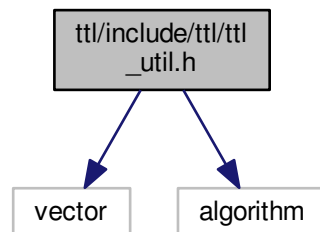
- template<class DartType >  
bool [ttl\\_constr::isTheConstraint](#) (const DartType &dart, const DartType &dstart, const DartType &dend)
- template<class TraitsType , class DartType >  
bool [ttl\\_constr::crossesConstraint](#) (DartType &dstart, DartType &dend, DartType &d1, DartType &d2)
- template<class TraitsType , class DartType >  
DartType [ttl\\_constr::getAtSmallestAngle](#) (const DartType &dstart, const DartType &dend)
- template<class TraitsType , class DartType , class ListType >  
DartType [ttl\\_constr::findCrossingEdges](#) (const DartType &dstart, const DartType &dend, ListType &elist)
- template<class TraitsType , class DartType >  
void [ttl\\_constr::transformToConstraint](#) (DartType &dstart, DartType &dend, std::list< DartType > &elist)

## Constrained (Delaunay) Triangulation

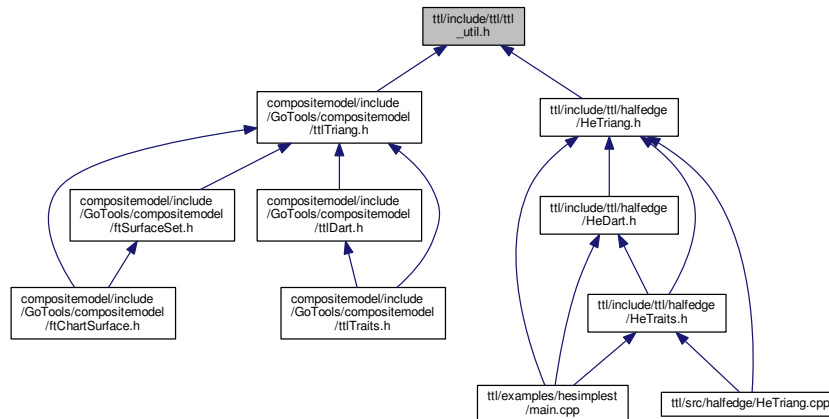
- template<class TraitsType , class DartType >  
DartType [ttl::insertConstraint](#) (DartType &dstart, DartType &dend, bool optimize\_delaunay)

## 30.2947 ttl/include/ttl/ttl\_util.h File Reference

```
#include <vector>
#include <algorithm>
Include dependency graph for ttl_util.h:
```



This graph shows which files directly or indirectly include this file:



## Namespaces

- [ttl\\_util](#)

*Utilities.*

## Functions

### Computational geometry

- `template<class real_type >`  
`real_type ttl_util::scalarProduct2d (real_type dx1, real_type dy1, real_type dx2, real_type dy2)`
- `template<class real_type >`  
`real_type ttl_util::crossProduct2d (real_type dx1, real_type dy1, real_type dx2, real_type dy2)`
- `template<class real_type >`  
`real_type ttl_util::orient2dfast (real_type pa[2], real_type pb[2], real_type pc[2])`

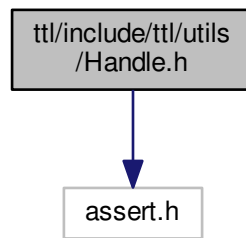
### Utilities involving points

- `template<class PointType >`  
`std::vector< PointType * > * ttl_util::createRandomData (int noPoints, int seed=1)`

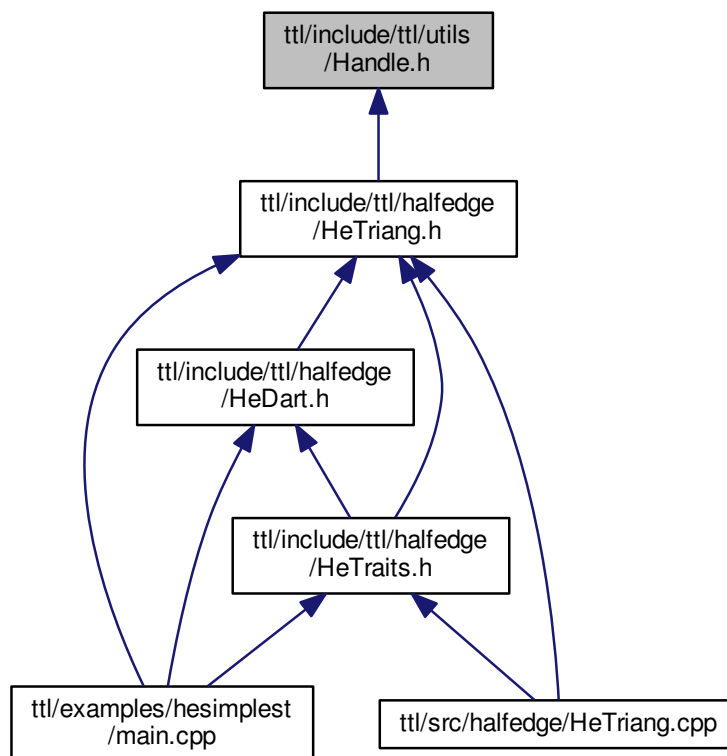
## 30.2948 ttl/include/ttl/utlis/Handle.h File Reference

```
#include <assert.h>
```

Include dependency graph for Handle.h:



This graph shows which files directly or indirectly include this file:



## Classes

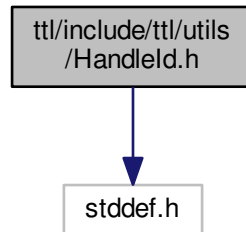
- class [Handle< T >](#)

*Template class for smart pointers. The actual class must inherit from [HandleId](#).*

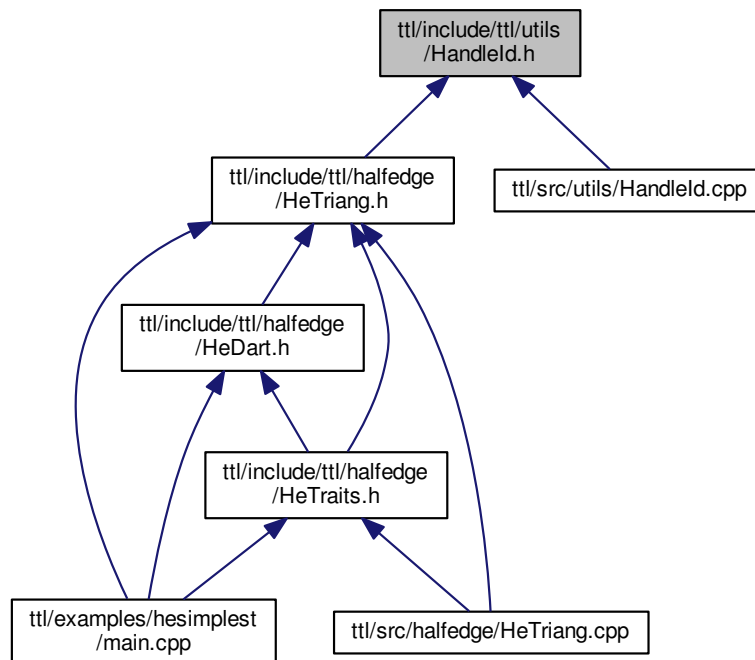
### 30.2949 ttl/include/ttl/utils/HandleId.h File Reference

```
#include <stddef.h>
```

Include dependency graph for HandleId.h:



This graph shows which files directly or indirectly include this file:



#### Classes

- class [HandleId](#)

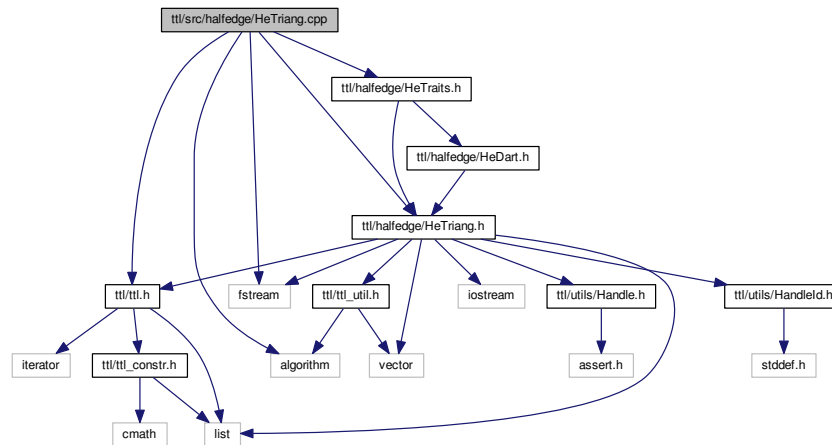
*Base class with reference counting for smart pointers.*



## 30.2950 ttl/src/halfedge/HeTriang.cpp File Reference

```
#include <ttl/halfedge/HeTriang.h>
#include <ttl/halfedge/HeTraits.h>
#include <ttl/ttl.h>
#include <algorithm>
#include <fstream>
```

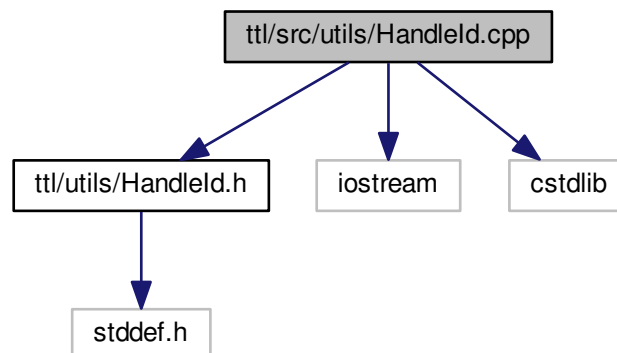
Include dependency graph for HeTriang.cpp:



## 30.2951 ttl/src/utlis/HandleId.cpp File Reference

```
#include <ttl/utlis/HandleId.h>
#include <iostream>
#include <cstdlib>
```

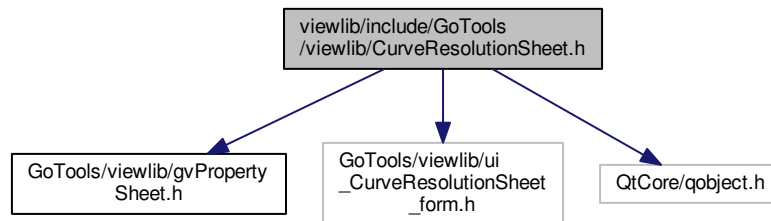
Include dependency graph for HandleId.cpp:



### 30.2952 viewlib/include/GoTools/viewlib/CurveResolutionSheet.h File Reference

```
#include "GoTools/viewlib/gvPropertySheet.h"
#include "GoTools/viewlib/ui_CurveResolutionSheet_form.h"
#include <QtCore/qobject.h>
```

Include dependency graph for CurveResolutionSheet.h:



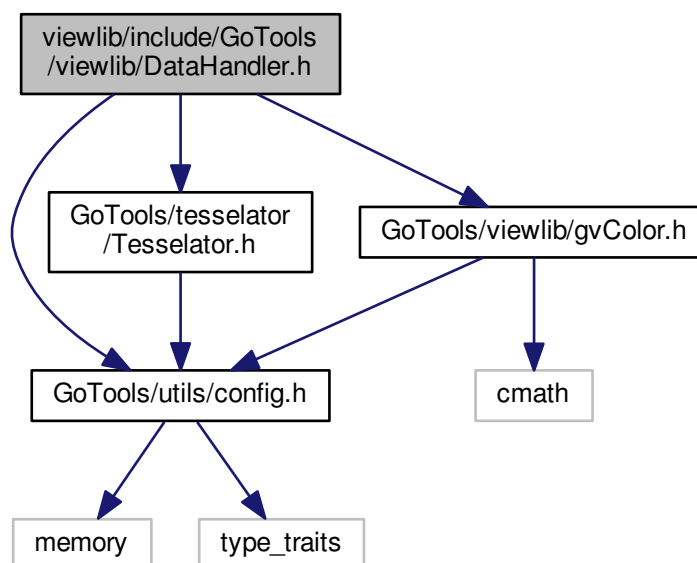
#### Classes

- class [CurveResolutionSheet](#)

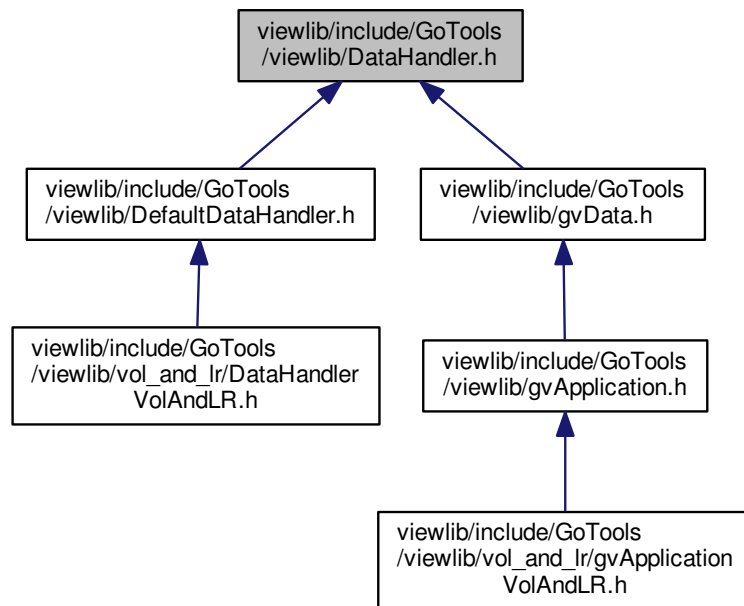
### 30.2953 viewlib/include/GoTools/viewlib/DataHandler.h File Reference

```
#include "GoTools/viewlib/gvColor.h"
#include "GoTools/tesselator/Tesselator.h"
#include "GoTools/utils/config.h"
```

Include dependency graph for DataHandler.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [DataHandler](#)

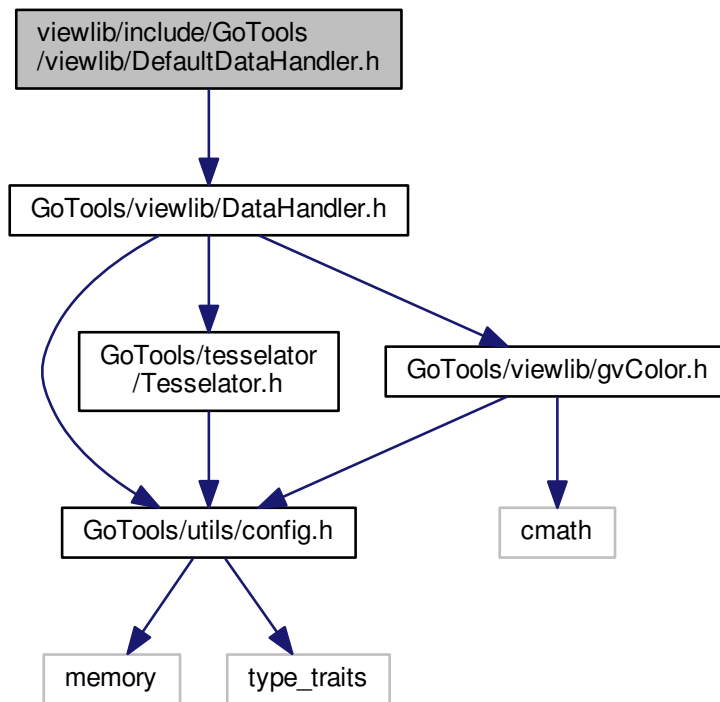
## Namespaces

- [Go](#)

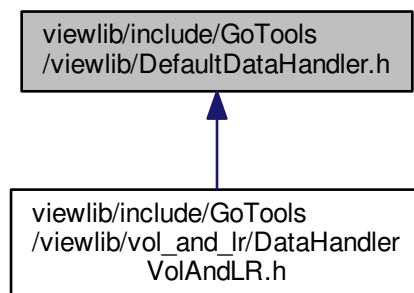
## 30.2954 viewlib/include/GoTools/viewlib/DefaultDataHandler.h File Reference

```
#include "GoTools/viewlib/DataHandler.h"
```

Include dependency graph for DefaultDataHandler.h:



This graph shows which files directly or indirectly include this file:



## Classes

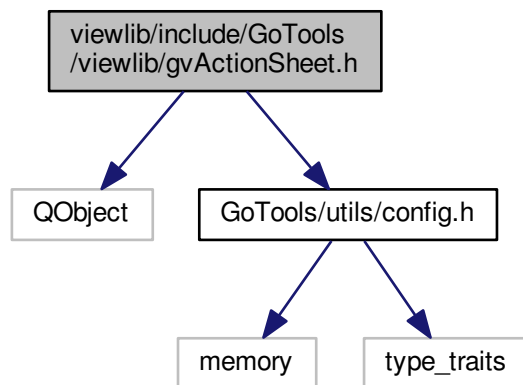
- class [DefaultDataHandler](#)

### 30.2955 viewlib/include/GoTools/viewlib/gvActionSheet.h File Reference

```

#include <QObject>
#include "GoTools/utils/config.h"
Include dependency graph for gvActionSheet.h:

```



#### Classes

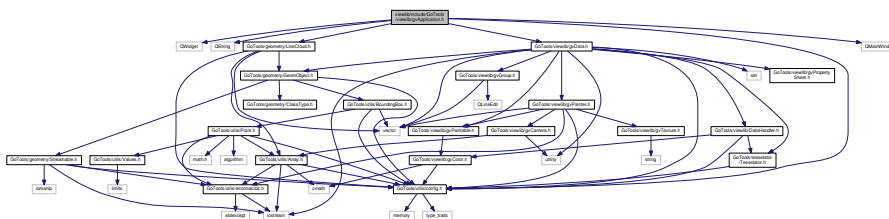
- class [gvActionSheet](#)

### 30.2956 viewlib/include/GoTools/viewlib/gvApplication.h File Reference

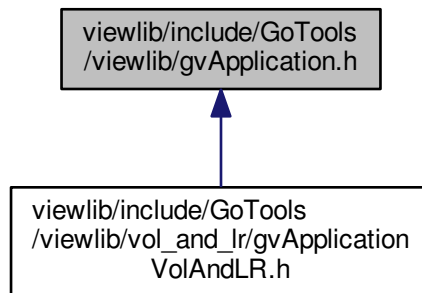
```

#include <QWidget>
#include <QString>
#include "GoTools/viewlib/gvData.h"
#include "GoTools/geometry/LineCloud.h"
#include <QMainWindow>
#include "GoTools/utils/config.h"
Include dependency graph for gvApplication.h:

```



This graph shows which files directly or indirectly include this file:



## Classes

- class [gvApplication](#)

## Typedefs

- typedef `std::vector< shared_ptr< Go::GeomObject > >` [ObjContainer](#)
- typedef `std::vector< shared_ptr< gvColor > >` [ColContainer](#)

### 30.2956.1 Typedef Documentation

#### 30.2956.1.1 typedef `std::vector<shared_ptr<gvColor> >` [ColContainer](#)

Definition at line 66 of file `gvApplication.h`.

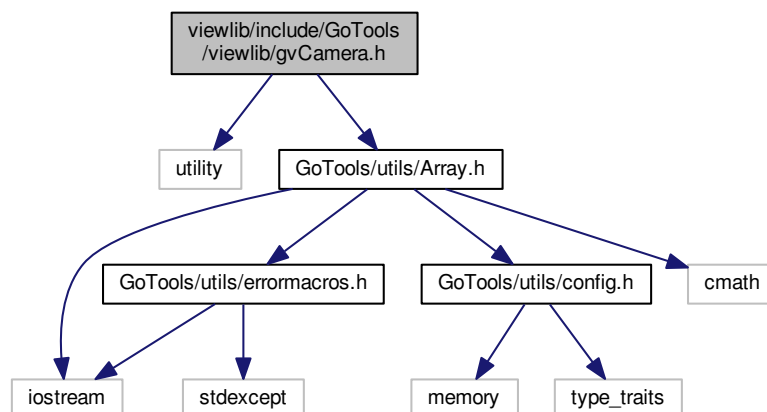
#### 30.2956.1.2 typedef `std::vector<shared_ptr<Go::GeomObject> >` [ObjContainer](#)

[gvApplication](#): etc

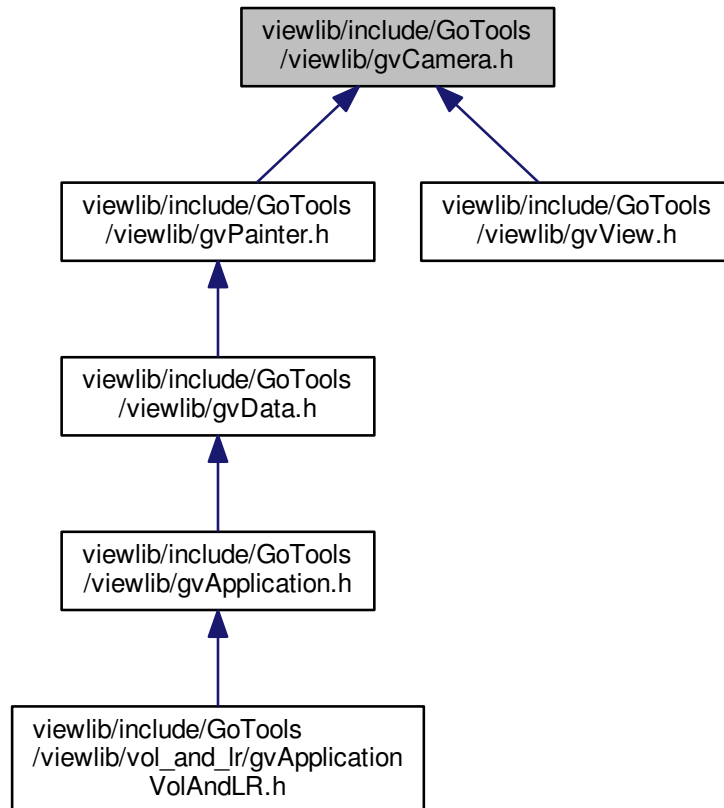
Definition at line 57 of file `gvApplication.h`.

## 30.2957 viewlib/include/GoTools/viewlib/gvCamera.h File Reference

```
#include <utility>
#include "GoTools/utils/Array.h"
Include dependency graph for gvCamera.h:
```



This graph shows which files directly or indirectly include this file:



## Classes

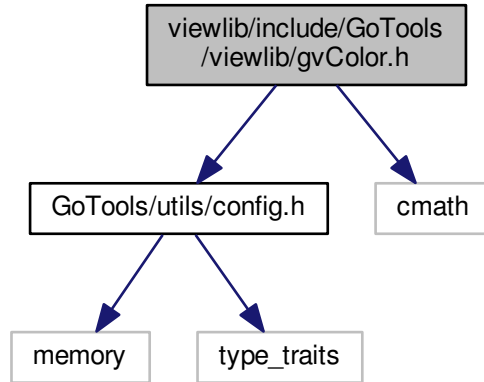
- class [gvCamera](#)

## 30.2958 viewlib/include/GoTools/viewlib/gvColor.h File Reference

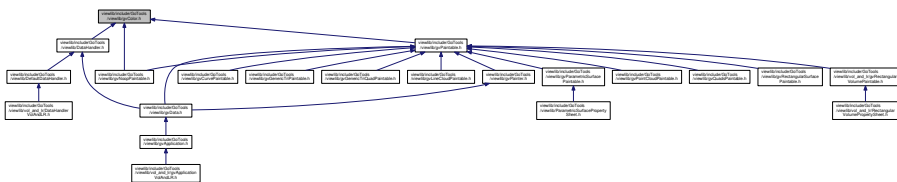
```
#include "GoTools/utils/config.h"
#include <cmath>
```



Include dependency graph for gvColor.h:



This graph shows which files directly or indirectly include this file:



### Classes

- struct [gvColor](#)

*Represents a color by four floats, like in OpenGL.*

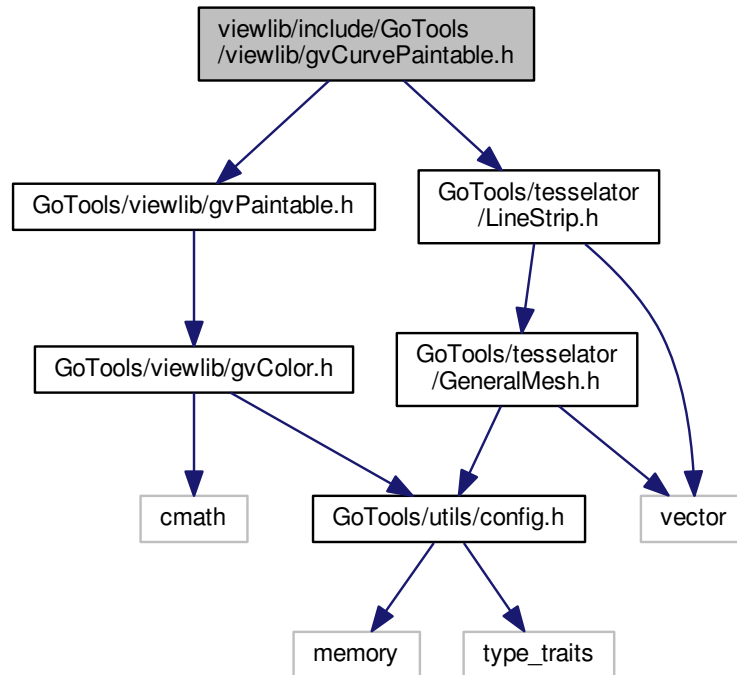
### 30.2959 viewlib/include/GoTools/viewlib/gvCurvePaintable.h File Reference

```

#include "GoTools/viewlib/gvPaintable.h"
#include "GoTools/tesselator/LineStrip.h"

```

Include dependency graph for gvCurvePaintable.h:



## Classes

- class [gvCurvePaintable](#)

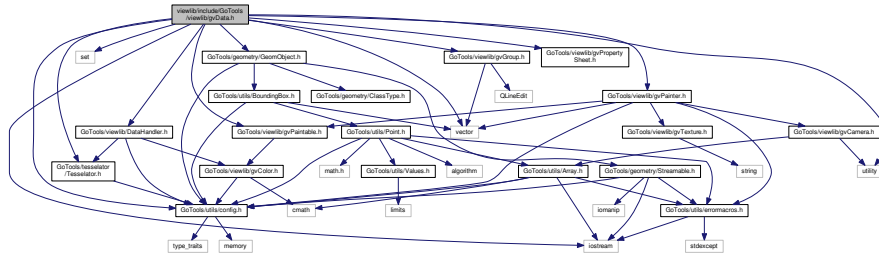
## 30.2960 viewlib/include/GoTools/viewlib/gvData.h File Reference

```

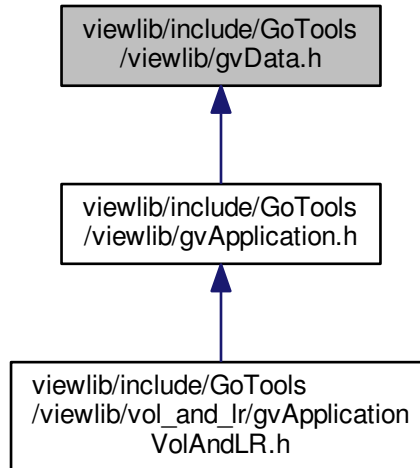
#include <iostream>
#include <vector>
#include <set>
#include <utility>
#include "GoTools/geometry/GeomObject.h"
#include "GoTools/viewlib/DataHandler.h"
#include "GoTools/tesselator/Tesselator.h"
#include "GoTools/viewlib/gvPaintable.h"
#include "GoTools/viewlib/gvPropertySheet.h"
#include "GoTools/viewlib/gvPainter.h"
#include "GoTools/viewlib/gvGroup.h"
#include "GoTools/utills/config.h"

```

Include dependency graph for gvData.h:



This graph shows which files directly or indirectly include this file:



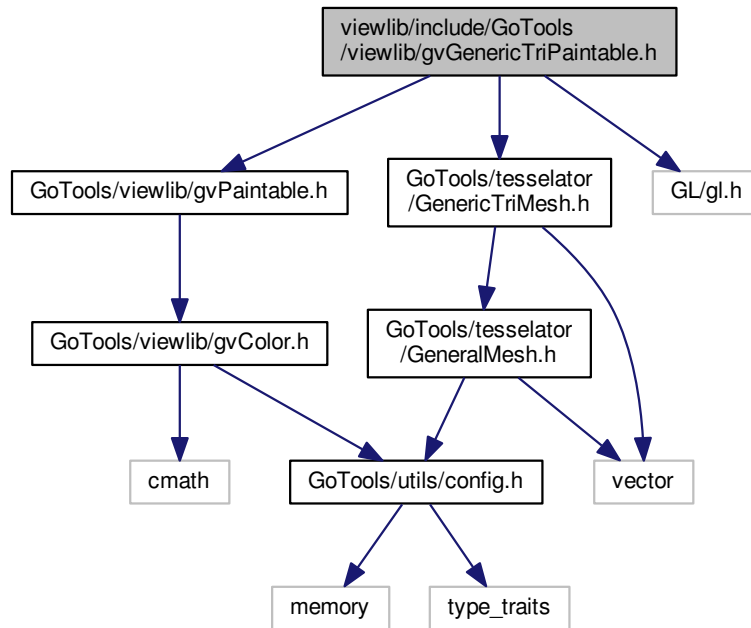
## Classes

- class [gvData](#)

## 30.2961 viewlib/include/GoTools/viewlib/gvGenericTriPaintable.h File Reference

```
#include "GoTools/viewlib/gvPaintable.h"
#include "GoTools/tessellator/GenericTriMesh.h"
#include <GL/gl.h>
```

Include dependency graph for gvGenericTriPaintable.h:



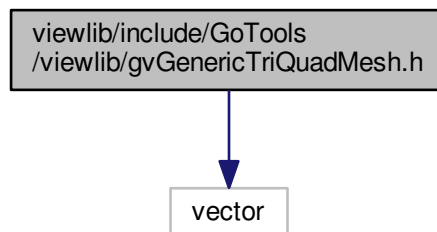
## Classes

- class `gvGenericTriPaintable< FloatType >`

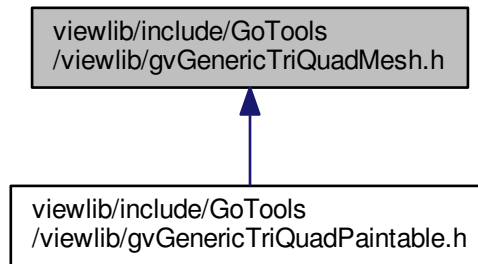
## 30.2962 viewlib/include/GoTools/viewlib/gvGenericTriQuadMesh.h File Reference

```
#include <vector>
```

Include dependency graph for gvGenericTriQuadMesh.h:



This graph shows which files directly or indirectly include this file:



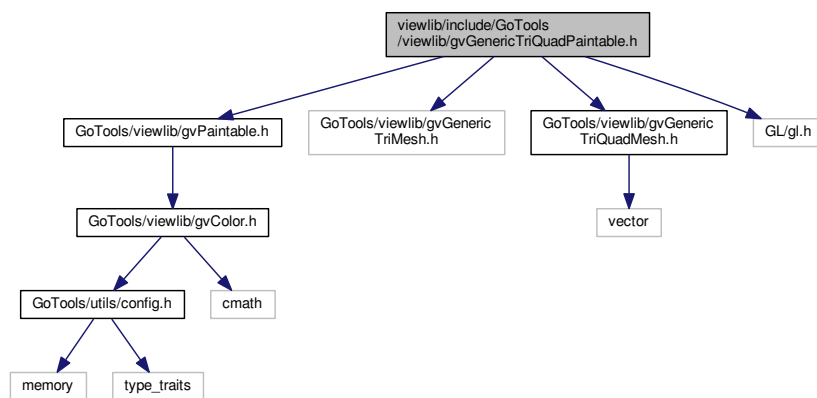
## Classes

- class [gvGenericTriQuadMesh< FloatType >](#)

## 30.2963 viewlib/include/GoTools/viewlib/gvGenericTriQuadPaintable.h File Reference

```
#include "GoTools/viewlib/gvPaintable.h"
#include "GoTools/viewlib/gvGenericTriMesh.h"
#include "GoTools/viewlib/gvGenericTriQuadMesh.h"
#include <GL/gl.h>
```

Include dependency graph for gvGenericTriQuadPaintable.h:

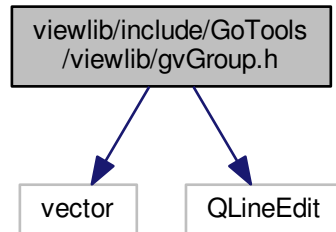


## Classes

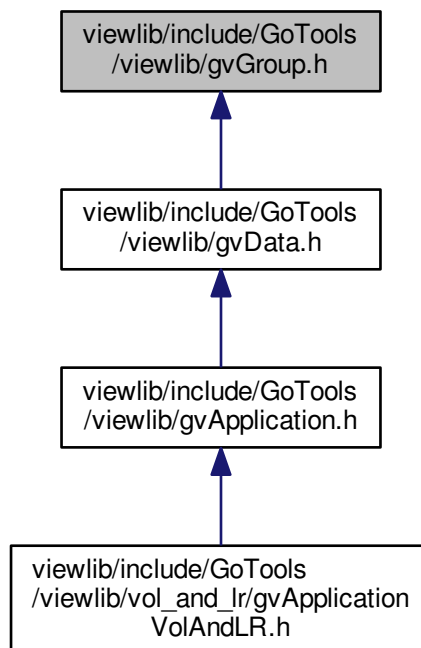
- class [gvGenericTriQuadPaintable< FloatType >](#)

## 30.2964 viewlib/include/GoTools/viewlib/gvGroup.h File Reference

```
#include <vector>
#include <QLineEdit>
Include dependency graph for gvGroup.h:
```



This graph shows which files directly or indirectly include this file:



### Classes

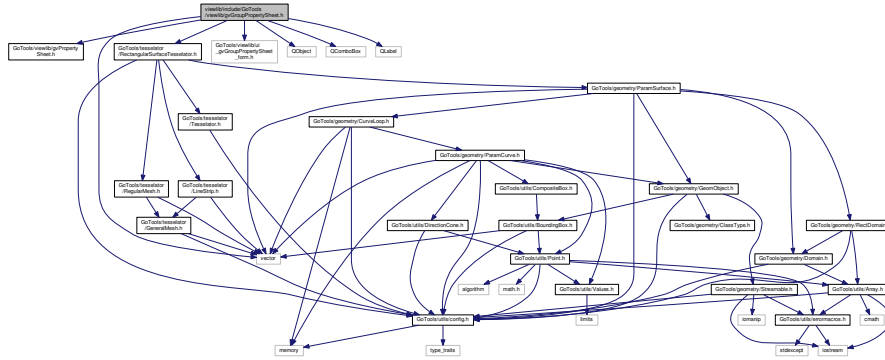
- class [gvGroup](#)

*Structure used to store name and index of members of a group of objects.*

### 30.2965 viewlib/include/GoTools/viewlib/gvGroupPropertySheet.h File Reference

```
#include "GoTools/viewlib/gvPropertySheet.h"
#include "GoTools/tesselator/RectangularSurfaceTesselator.h"
#include "GoTools/viewlib/ui_gvGroupPropertySheet_form.h"
#include <QObject>
#include <QComboBox>
#include <vector>
#include <QLabel>
```

Include dependency graph for gvGroupPropertySheet.h:



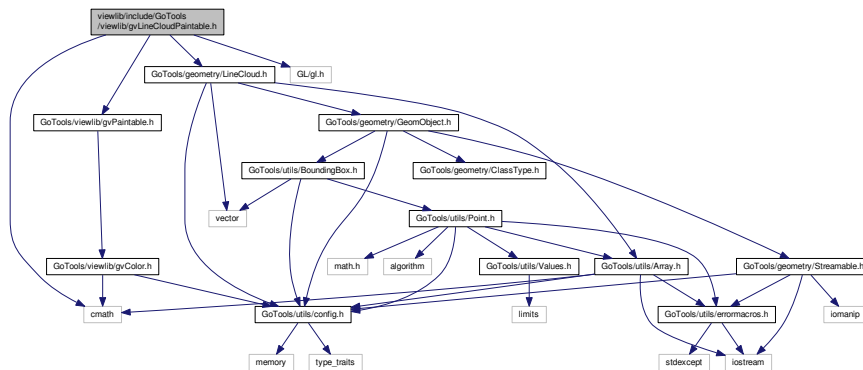
#### Classes

- class gvGroupPropertySheet

### 30.2966 viewlib/include/GoTools/viewlib/gvLineCloudPaintable.h File Reference

```
#include "GoTools/viewlib/gvPaintable.h"
#include "GoTools/geometry/LineCloud.h"
#include <GL/gl.h>
#include <cmath>
```

Include dependency graph for gvLineCloudPaintable.h:

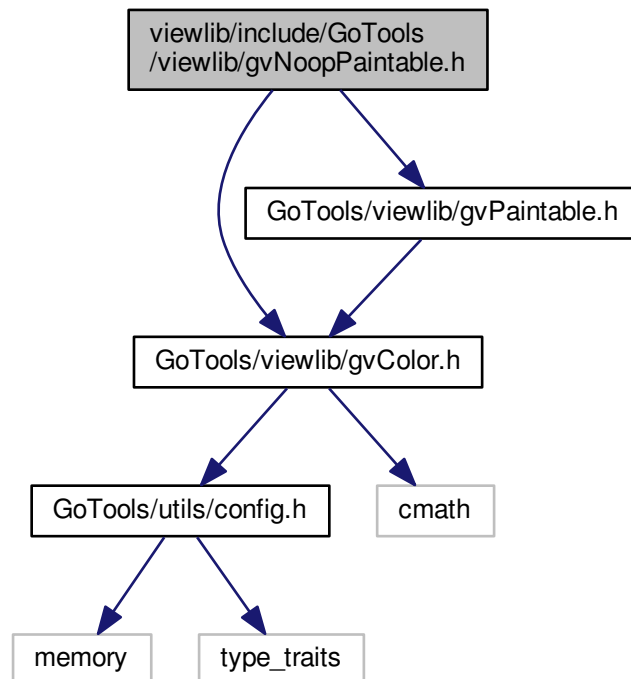


## Classes

- class [gvLineCloudPaintable](#)

## 30.2967 viewlib/include/GoTools/viewlib/gvNoopPaintable.h File Reference

```
#include "GoTools/viewlib/gvColor.h"
#include "GoTools/viewlib/gvPaintable.h"
Include dependency graph for gvNoopPaintable.h:
```



## Classes

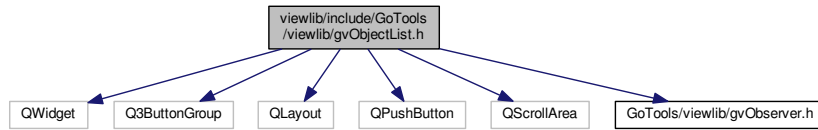
- class [gvNoopPaintable](#)

## 30.2968 viewlib/include/GoTools/viewlib/gvObjectList.h File Reference

```
#include <QWidget>
#include <Q3ButtonGroup>
#include <QLayout>
#include <QPushButton>
#include <QScrollArea>
#include "GoTools/viewlib/gvObserver.h"
```



Include dependency graph for gvObjectList.h:

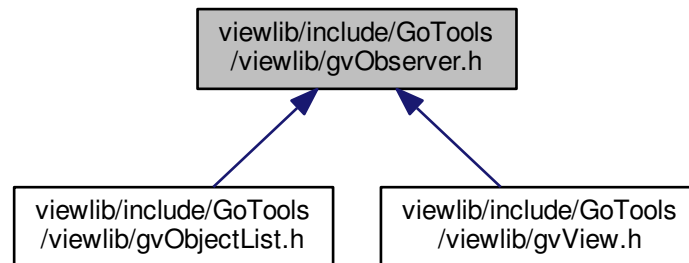


## Classes

- class [gvObjectList](#)

## 30.2969 viewlib/include/GoTools/viewlib/gvObserver.h File Reference

This graph shows which files directly or indirectly include this file:



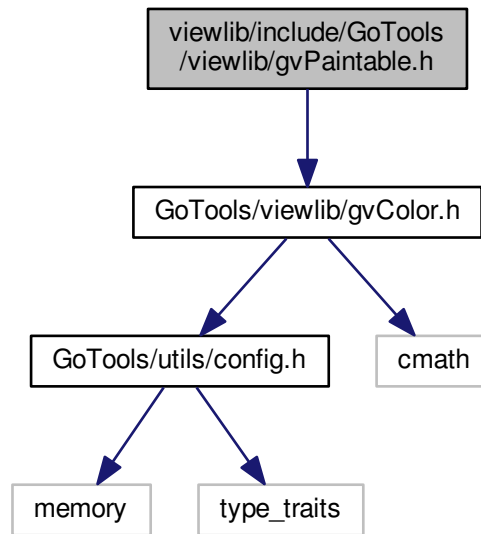
## Classes

- class [gvObserver](#)

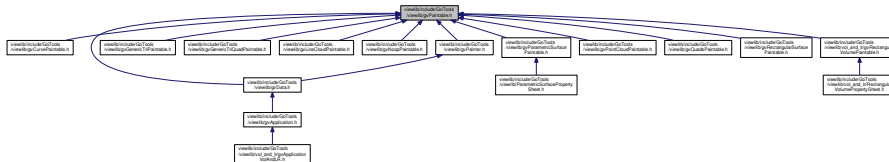
## 30.2970 viewlib/include/GoTools/viewlib/gvPaintable.h File Reference

```
#include "GoTools/viewlib/gvColor.h"
```

Include dependency graph for gvPaintable.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [gvPaintable](#)

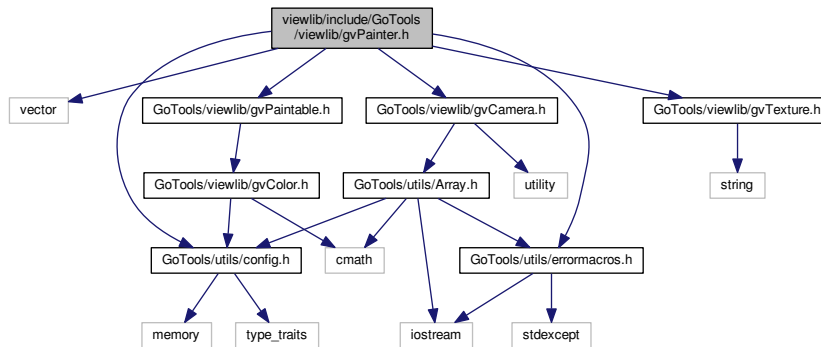
## 30.2971 viewlib/include/GoTools/viewlib/gvPainter.h File Reference

```

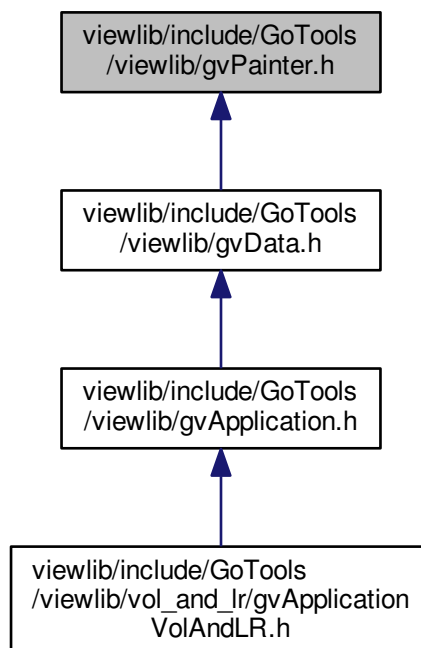
#include <vector>
#include "GoTools/utils/config.h"
#include "GoTools/viewlib/gvPaintable.h"
#include "GoTools/viewlib/gvCamera.h"
#include "GoTools/viewlib/gvTexture.h"
#include "GoTools/utils/errormacros.h"

```

Include dependency graph for gvPainter.h:



This graph shows which files directly or indirectly include this file:



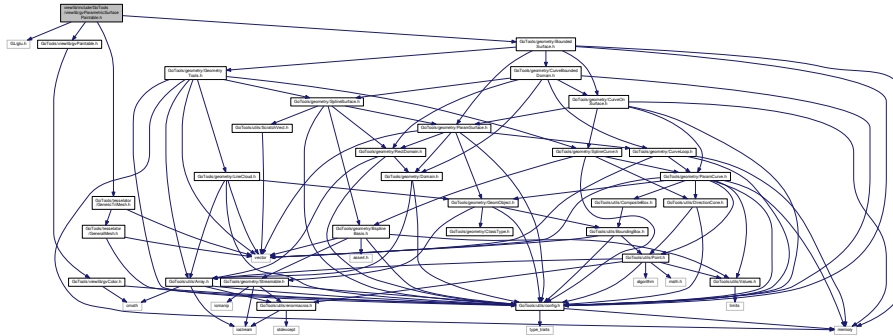
## Classes

- class [gvPainter](#)

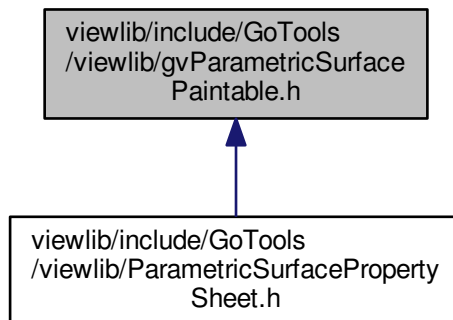
*The [gvPainter](#) is responsible for painting the 3D scene.*

## 30.2972 viewlib/include/GoTools/viewlib/gvParametricSurfacePaintable.h File Reference

```
#include <GL/glu.h>
#include "GoTools/viewlib/gvPaintable.h"
#include "GoTools/tessellator/GenericTriMesh.h"
#include "GoTools/geometry/BoundedSurface.h"
Include dependency graph for gvParametricSurfacePaintable.h:
```



This graph shows which files directly or indirectly include this file:



### Classes

- class [gvParametricSurfacePaintable](#)

### Typedefs

- typedef [Go::GenericTriMesh](#) [genMesh](#)

### 30.2972.1 Typedef Documentation

#### 30.2972.1.1 typedef [Go::GenericTriMesh](#) [genMesh](#)

Definition at line 58 of file [gvParametricSurfacePaintable.h](#).

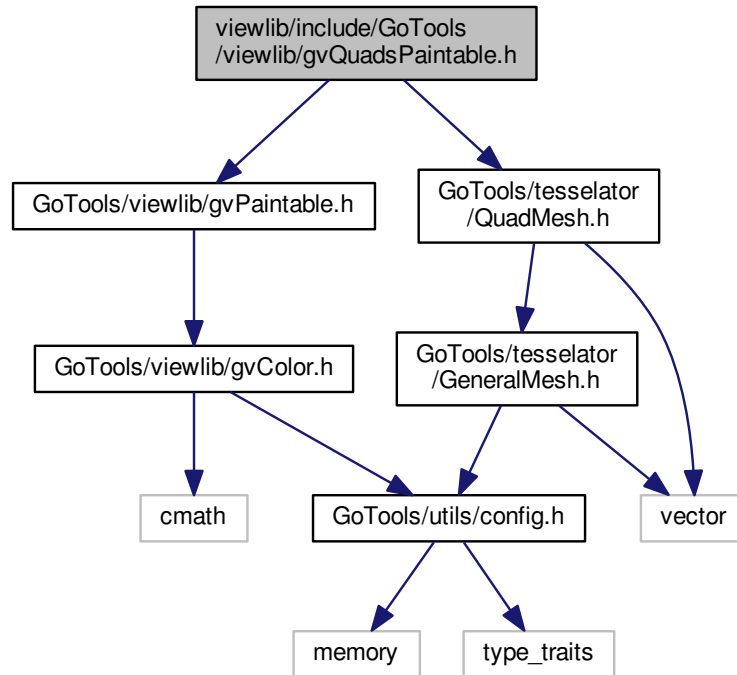


### 30.2975 viewlib/include/GoTools/viewlib/gvQuadsPaintable.h File Reference

```
#include "GoTools/viewlib/gvPaintable.h"
```

```
#include "GoTools/tessellator/QuadMesh.h"
```

Include dependency graph for gvQuadsPaintable.h:



#### Classes

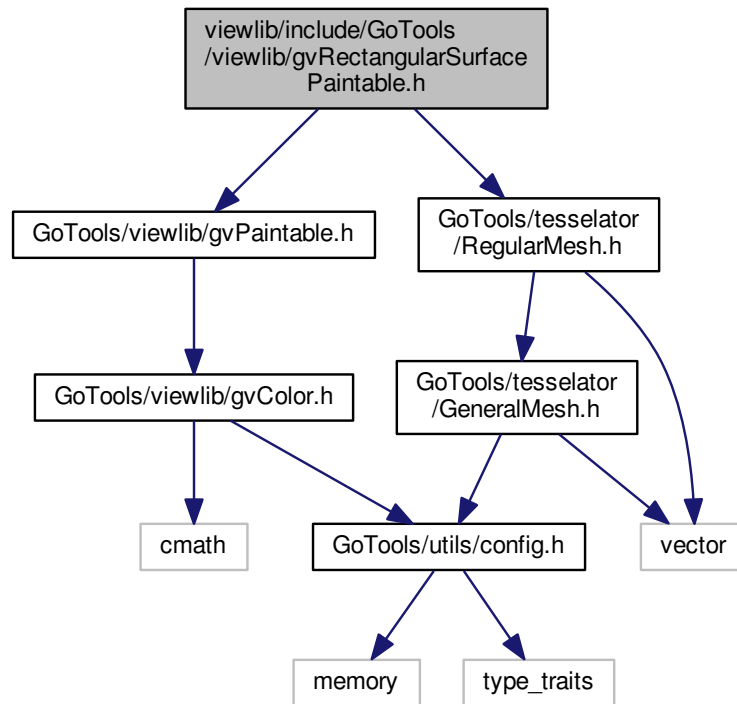
- class [gvQuadsPaintable](#)

### 30.2976 viewlib/include/GoTools/viewlib/gvRectangularSurfacePaintable.h File Reference

```
#include "GoTools/viewlib/gvPaintable.h"
```

```
#include "GoTools/tessellator/RegularMesh.h"
```

Include dependency graph for gvRectangularSurfacePaintable.h:



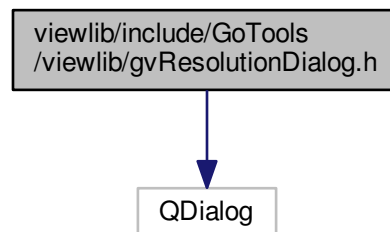
## Classes

- class [gvRectangularSurfacePaintable](#)

## 30.2977 viewlib/include/GoTools/viewlib/gvResolutionDialog.h File Reference

```
#include <QDialog>
```

Include dependency graph for gvResolutionDialog.h:



## Classes

- class [gvResolutionDialog](#)

## 30.2978 viewlib/include/GoTools/viewlib/gvStandardMouseHandler.h File Reference

### Classes

- class [gvStandardMouseHandler](#)

### Macros

- #define [GV\\_STANDARD\\_MOUSEHANDLER\\_H\\_INCLUDED](#)

### 30.2978.1 Macro Definition Documentation

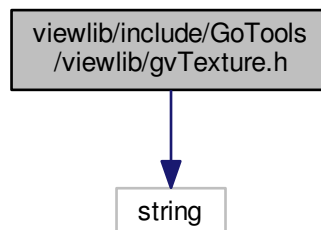
#### 30.2978.1.1 #define GV\_STANDARD\_MOUSEHANDLER\_H\_INCLUDED

Definition at line 86 of file gvStandardMouseHandler.h.

## 30.2979 viewlib/include/GoTools/viewlib/gvTexture.h File Reference

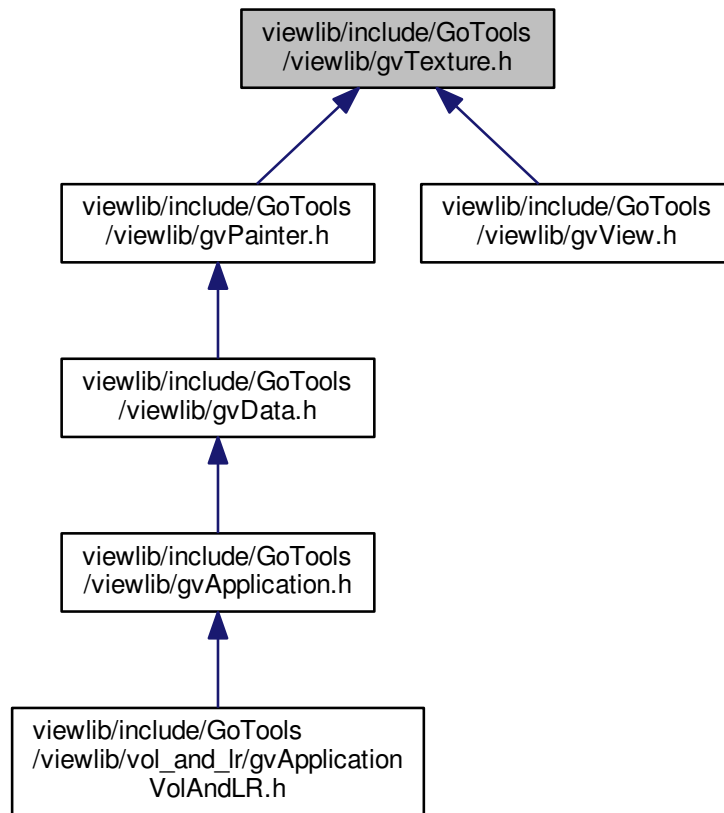
```
#include <string>
```

Include dependency graph for gvTexture.h:





This graph shows which files directly or indirectly include this file:



## Classes

- class [gvTexture](#)

## Enumerations

- enum [MinFilterSet](#) { [minNearest](#), [minLinear](#), [minNearestMipmapNearest](#), [minNearestMipmapLinear](#), [minLinearMipmapNearest](#), [minLinearMipmapLinear](#) }
- enum [MagFilterSet](#) { [magNearest](#), [magLinear](#) }
- enum [EnvModeSet](#) { [envDecal](#), [envReplace](#), [envModulate](#), [envBlend](#) }
- enum [WrapModeSet](#) { [wrapClamp](#), [wrapRepeat](#) }

## 30.2979.1 Enumeration Type Documentation

### 30.2979.1.1 enum EnvModeSet

Enumerator

***envDecal***

*envReplace*  
*envModulate*  
*envBlend*

Definition at line 53 of file gvTexture.h.

#### 30.2979.1.2 enum MagFilterSet

Enumerator

*magNearest*  
*magLinear*

Definition at line 50 of file gvTexture.h.

#### 30.2979.1.3 enum MinFilterSet

Enumerator

*minNearest*  
*minLinear*  
*minNearestMipmapNearest*  
*minNearestMipmapLinear*  
*minLinearMipmapNearest*  
*minLinearMipmapLinear*

Definition at line 45 of file gvTexture.h.

#### 30.2979.1.4 enum WrapModeSet

Enumerator

*wrapClamp*  
*wrapRepeat*

Definition at line 56 of file gvTexture.h.

## 30.2980 viewlib/include/GoTools/viewlib/gvUtilities.h File Reference

### Typedefs

- typedef [double](#) [FLOAT\\_TYPE](#)
- typedef [FLOAT\\_TYPE](#) [MATRIX3](#)[9]
- typedef [FLOAT\\_TYPE](#) [MATRIX4](#)[16]

## Functions

- void `draw_cylinder` (`double x0`, `double y0`, `double z0`, `double x1`, `double y1`, `double z1`, `double radius`, `double radius2`, `int n`)  
*Draw a cylinder, or possibly a cone.*
- void `draw_gl_axes` (`int n`, `double r`, `double radius`, `double rim`, `double l`)  
*Draw a set of axes.*
- void `draw_gl_axes` (`double relscale`)
- `FLOAT_TYPE m3_det` (`MATRIX3 mat`)
- void `m3_identity` (`MATRIX3 mat`)
- void `m3_inverse` (`MATRIX3 mr`, `MATRIX3 ma`)
- void `m4_submat` (`MATRIX4 mr`, `MATRIX3 mb`, `int i`, `int j`)
- `FLOAT_TYPE m4_det` (`MATRIX4 mr`)
- `int m4_inverse` (`MATRIX4 mr`, `MATRIX4 ma`)

### 30.2980.1 Typedef Documentation

#### 30.2980.1.1 typedef double FLOAT\_TYPE

Definition at line 51 of file gvUtilities.h.

#### 30.2980.1.2 typedef FLOAT\_TYPE MATRIX3[9]

Definition at line 52 of file gvUtilities.h.

#### 30.2980.1.3 typedef FLOAT\_TYPE MATRIX4[16]

Definition at line 53 of file gvUtilities.h.

### 30.2980.2 Function Documentation

#### 30.2980.2.1 void draw\_cylinder ( double x0, double y0, double z0, double x1, double y1, double z1, double radius, double radius2, int n )

Draw a cylinder, or possibly a cone.

Definition at line 91 of file gl\_aux.cpp.

#### 30.2980.2.2 void draw\_gl\_axes ( int n, double r, double radius, double rim, double l )

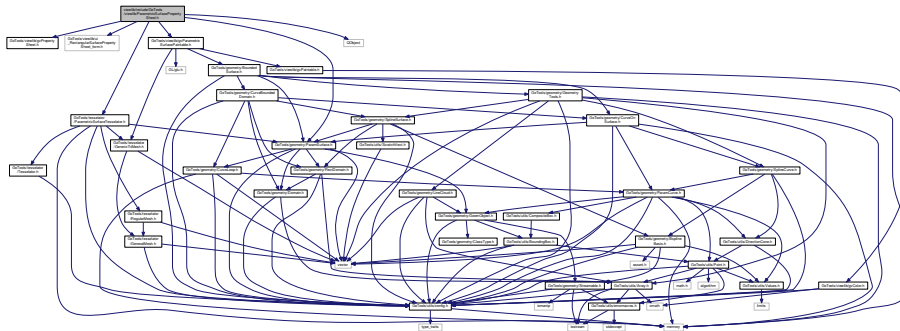
Draw a set of axes.



## 30.2982 viewlib/include/GoTools/viewlib/ParametricSurfacePropertySheet.h File Reference

```
#include "GoTools/viewlib/gvPropertySheet.h"
#include "GoTools/viewlib/ui_RectangularSurfacePropertySheet_form.h"
#include "GoTools/geometry/ParamSurface.h"
#include "GoTools/viewlib/gvParametricSurfacePaintable.h"
#include "GoTools/tesselator/ParametricSurfaceTesselator.h"
#include <QObject>
```

Include dependency graph for ParametricSurfacePropertySheet.h:



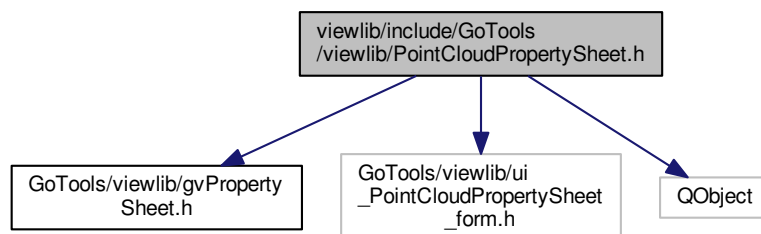
### Classes

- class [ParametricSurfacePropertySheet](#)

## 30.2983 viewlib/include/GoTools/viewlib/PointCloudPropertySheet.h File Reference

```
#include "GoTools/viewlib/gvPropertySheet.h"
#include "GoTools/viewlib/ui_PointCloudPropertySheet_form.h"
#include <QObject>
```

Include dependency graph for PointCloudPropertySheet.h:



### Classes

- class [PointCloudPropertySheet](#)

## 30.2984 viewlib/include/GoTools/viewlib/raster.h File Reference

### Macros

- `#define RASTERUTILS_H_INCLUDED`

### Functions

- unsigned char \* `read_ppm_file` (const char \*const name, int \*const xs, int \*const ys)  
*Returns RGB data in a new'ed array.*
- void `write_ppm_file` (const char \*const name, const unsigned char \*const d, const int xs, const int ys, bool rgb=true)  
*Writes out a ppm file.*

### 30.2984.1 Macro Definition Documentation

#### 30.2984.1.1 `#define RASTERUTILS_H_INCLUDED`

Definition at line 66 of file raster.h.

### 30.2984.2 Function Documentation

#### 30.2984.2.1 unsigned char\* `read_ppm_file` ( const char \*const name, int \*const xs, int \*const ys )

Returns RGB data in a new'ed array.

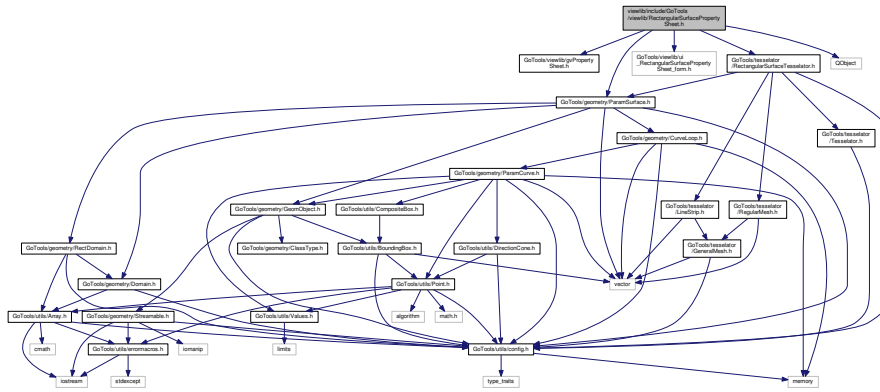
#### 30.2984.2.2 void `write_ppm_file` ( const char \*const name, const unsigned char \*const d, const int xs, const int ys, bool rgb = true )

Writes out a ppm file.

## 30.2985 viewlib/include/GoTools/viewlib/RectangularSurfacePropertySheet.h File Reference

```
#include "GoTools/viewlib/gvPropertySheet.h"
#include "GoTools/geometry/ParamSurface.h"
#include "GoTools/viewlib/ui_RectangularSurfacePropertySheet_form.h"
#include "GoTools/tessellator/RectangularSurfaceTessellator.h"
#include <QObject>
```

Include dependency graph for RectangularSurfacePropertySheet.h:



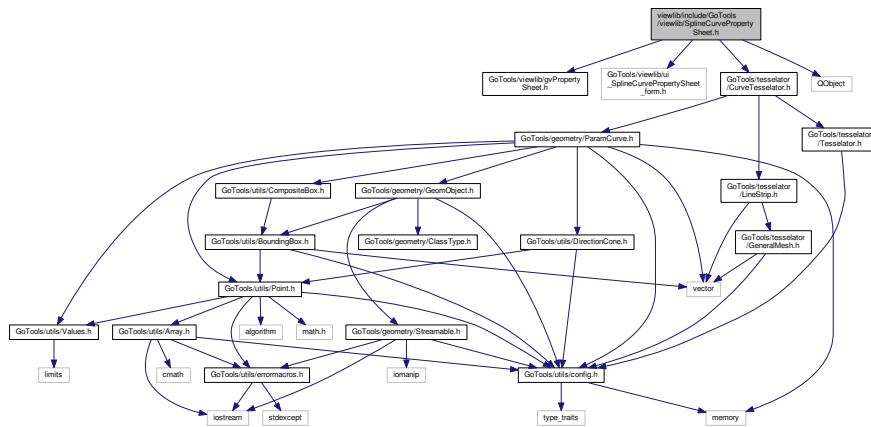
### Classes

- class [RectangularSurfacePropertySheet](#)

## 30.2986 viewlib/include/GoTools/viewlib/SplineCurvePropertySheet.h File Reference

```
#include "GoTools/viewlib/gvPropertySheet.h"
#include "GoTools/viewlib/ui_SplineCurvePropertySheet_form.h"
#include "GoTools/tessellator/CurveTessellator.h"
#include <QObject>
```

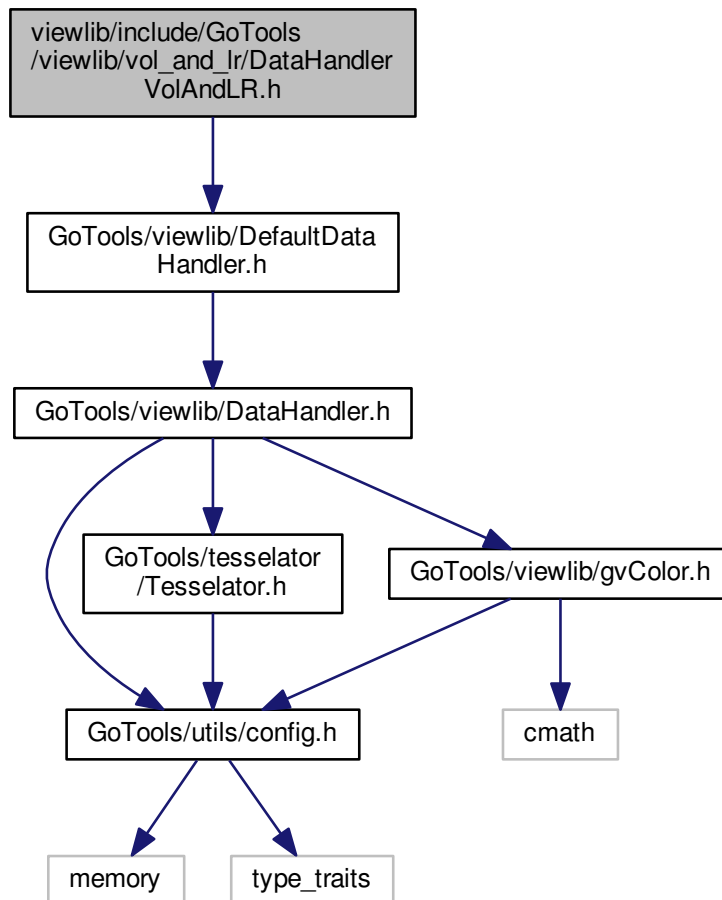
Include dependency graph for SplineCurvePropertySheet.h:







Include dependency graph for DataHandlerVolAndLR.h:



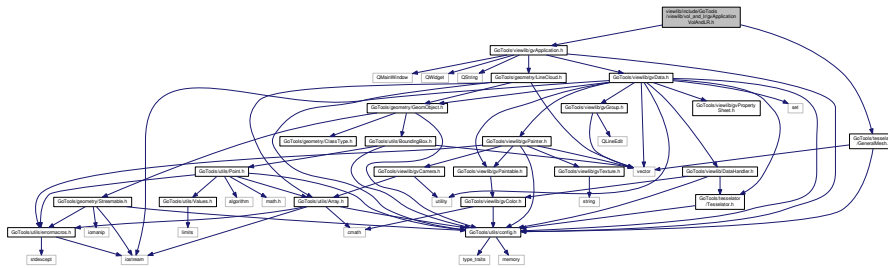
## Classes

- class [DataHandlerVolAndLR](#)

## 30.2990 viewlib/include/GoTools/viewlib/vol\_and\_lr/gvApplicationVolAndLR.h File Reference

```
#include "GoTools/viewlib/gvApplication.h"
#include "GoTools/tesselator/GeneralMesh.h"
```

Include dependency graph for gvApplicationVolAndLR.h:



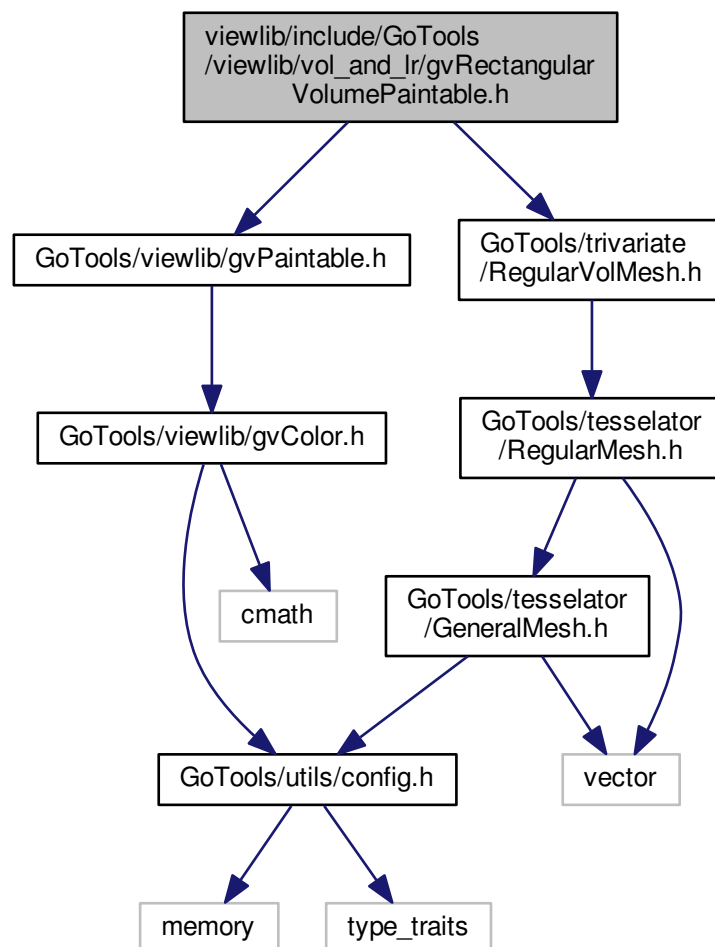
## Classes

- class [gvApplicationVolAndLR](#)

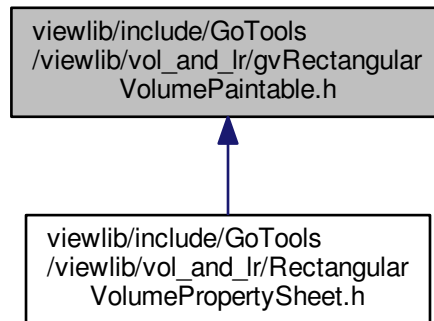
## 30.2991 viewlib/include/GoTools/viewlib/vol\_and\_lr/gvRectangularVolumePaintable.h File Reference

```
#include "GoTools/viewlib/gvPaintable.h"
#include "GoTools/trivariate/RegularVolMesh.h"
```

Include dependency graph for gvRectangularVolumePaintable.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [gvRectangularVolumePaintable](#)

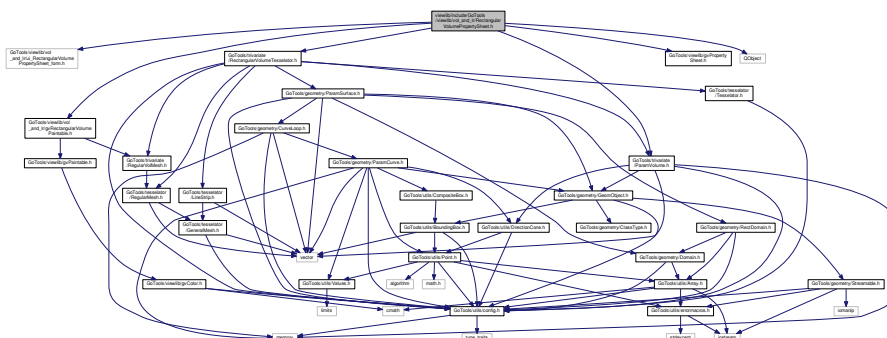
## 30.2992 viewlib/include/GoTools/viewlib/vol\_and\_lr/RectangularVolumePropertySheet.h File Reference

```

#include "GoTools/viewlib/vol_and_lr/ui_RectangularVolumePropertySheet_↵
form.h"
#include "GoTools/viewlib/vol_and_lr/gvRectangularVolumePaintable.h"
#include "GoTools/viewlib/gvPropertySheet.h"
#include "GoTools/trivariate/ParamVolume.h"
#include "GoTools/trivariate/RectangularVolumeTesselator.h"
#include <QObject>

```

Include dependency graph for RectangularVolumePropertySheet.h:



## Classes

- class [RectangularVolumePropertySheet](#)

## Chapter 31

# Example Documentation

### 31.1 adapt\_curve

adapt\_curve.C

This program demonstrates the use of the class `AdaptCurve`. The class can generate a B-spline curve that approximates an evaluator based curve to satisfy a given accuracy.

The program reads a 2D parameter curve and a surface from file, and lifts the curve onto the surface to make an evaluator based curve. This evaluator based curve are then used as input to the `AdaptCurve` constructor.

### 31.2 append\_curve

append\_curve.C

This program demonstrates the use of the function `SplineCurve::append_curve` declared in `SplineCurve.h`. The function joins two `SplineCurves` by appending the start of the second curve to the end of the first curve. The two curves must be of the same type

### 31.3 approx\_curve

approx\_curve.C

This program demonstrates the use of the class `ApproxCurve`. The class can generate a B-spline curve that approximates a set of parametrized points for a given accuracy by inserting new knots in the curve until the required accuracy is reached.

## 31.4 approx\_surface

approx\_surface.C

This program demonstrates the use of the class ApproxSurf. The class can generate a new tensor product B-spline surface with four boundary curves that approximates a set of parametrized points for a given accuracy, or modify an old surface by a set of parametrized points.

## 31.5 BrepToTrivariate

BrepToTrivariate.C

The idea of this program is to read a Brep model in g2-format and create trivariate spline model. Note that the functionality works only for some classes of Brep models.

## 31.6 circle

circle.C

This program demonstrates the use of the class Circle. It is a subclass of ElementaryCurve. The space dimension of a circle is either 2 or 3. The default parametrization is an angle from 0 to  $2\pi$ .

## 31.7 closestpoint\_curve

closestpoint\_curve.C

This program demonstrates the use of the two functions `ParamCurve::closestPoint(const Point& pt, double& clo_t, Point& clo_pt, double& clo_dist)` and `SplineCurve::closestPoint(const Point& pt, double tmin, double tmax, double& clo_t, Point& clo_pt, double& clo_dist, double const *seed = 0)`. They compute the closest point on a curve from a specified point.

## 31.8 closestpoint\_degenerate\_sf

closestpoint\_degenerate\_sf.C

This program demonstrates the use of the function `SplineSurface::closestPoint(const Point& pt, double& clo_u, double& clo_v, Point& clo_pt, double& clo_dist, double epsilon, const RectDomain* domain_of_interest = NULL, double *seed = 0)`. The declaration of the function is in 'ParamSurface.h'. The function compute the closest point on a surface from a specified point. It reads a spline surface file in Go-format and a file in plain ASCII format with the xyz-coordinates of the points we want to find the closest point on the surface to.

## 31.9 closestpoint\_surface

closestpoint\_surface.C

This program demonstrates the use of the function `'SplineSurface::closestPoint(const Point& pt, double& clo_u, double& clo_v, Point& clo_pt, double& clo_dist, double epsilon, const RectDomain* domain_of_interest = NULL, double *seed = 0)'`. The declaration of the function is in 'ParamSurface.h'. The function compute the closest point on a surface from a specified point. It reads a spline surface file in Go-format and a file in plain ASCII format with the xyz-coordinates of the points we want to find the closest point on the surface to.

## 31.10 cone

cone.C

This program demonstrates the use of the class `Cone`. It is a subclass of `ElementarySurface`. The space dimension of a cone is 3. The default parametrization is the angle  $u$  from 0 to  $2\pi$  and the distance  $v$  from minus infinity to plus infinity.

## 31.11 const\_param\_curves

const\_param\_curves.C

The program demonstrates the use of the function `SplineSurface::constParamCurve(double parameter, bool padir_is_u)`. The declaration of the function is in `SplineSurface.h`.

## 31.12 coons\_patch\_gen

coons\_patch\_gen.C

This program demonstrates the use of some of the functions in namespace `CoonsPatchGen`. The functions can be used to create a Coons Patch or a Gordon Surface. The functions returns a `SplineSurface` pointer to the created surface.

## 31.13 coons\_patch\_volume\_gen

coons\_patch\_volume\_gen.C

This program demonstrates the use of the static function `createCoonsPatch` in namespace `CoonsPatchVolumeGen`. The function can create a new `SplineVolume` representing the coons patch of six `SplineSurfaces`, the six faces of the volume.

## 31.14 createBlockStructuredDisc

createBlockStructuredDisc.C

This example file creates a block structured set of spline surfaces from a face set consisting of possibly trimmed surfaces with arbitrary topology (no corner-to-corner conditions).

## 31.15 createCoonsVolume

createCoonsVolume.C

The program creates a Coons volume and performs smoothing of this volume keeping the boundary surfaces fixed.



## 31.16 createLinSweptVol

createLinSweptVol.C

Create a block structured volume model from a face set describing a boundary represented solid that may be created by sweeping a planar face set along a stright line

## 31.17 createMidShip

createMidShip.C

Build a volume model representing a simplified mid ship, hull with stiffeneres in both directions and deck. The model is represented as a block structured volume model.

## 31.18 createRotationalVol

createRotationalVol.C

Create a block structured volume model from a face set describing a boundary represented solid representing a rotational object.

## 31.19 createSplitDisc

createSplitDisc.C

This programs creates a face set representing a disc as two trimmed surfaces: The trimmed disc and a rectangular surface lying inside this disc. This is a starting point for the creation of a disc as a multi patch model with spline surfaces and no degeneracies. The construction uses planar, rectangular surfaces and a truncated sylinder, but the operations performed using these surfaces, do not depend on that level of regularity.

## 31.20 createVolumeBlocks

createVolumeBlocks.C

Create a block structured volume model from a face set consisting of possibly trimmed surfaces with arbitrary topology (no corner-to-corner conditions). The face set represents a boundary represented solid and is described in a g2-file.

## 31.21 createVolumeBoundaries

createVolumeBoundaries.C

This programs creates a set of B-spline surfaces intended as the boundary surfaces for a spline volume. A number of different methods are used in the surface construction. Thus, this example program illustrates some of the possibilities for surface construction.

## 31.22 cylinder

cylinder.C

This program demonstrates the use of the class Cylinder. It is a subclass of ElementarySurface. The space dimension of a cylinder is 3. The default parametrization is the angle  $u$  from 0 to  $2\pi$  and the distance  $v$  from minus infinity to plus infinity.

## 31.23 ellipse

ellipse.C

This program demonstrates the use of the class Ellipse. It is a subclass of ElementaryCurve. The space dimension of an ellipse is either 2 or 3. The default parametrization is an angle from 0 to  $2\pi$ .

## 31.24 face2splineset

face2splineSet.C

This program demonstrates how to create a set of spline surfaces, meeting in a corner-to-corner configuration and with corresponding coefficients at common boundaries, from one possibly trimmed face / The program reads a bounded surface from a file, splits this surface into several bounded surfaces where each surface has (at most) 4 boundary curves. Finally, each bounded surface is approximated by a spline surface within a given tolerance and C0 continuities at common boundaries is ensured.

## 31.25 interpol\_curve\_free

interpol\_curve\_free.C

This program reads a point data set from a file, interpolates a spline curve through the points and writes an output file with the spline curve and the input points.

## 31.26 interpol\_curve\_hermite

interpol\_curve\_hermite.C

This program reads a point data set from a file, interpolates a spline curve through the points and write two output files: One file with spline curve data and one file with tangent vectors from the input points.

## 31.27 linear\_swept\_surface

linear\_swept\_surface.C

This program demonstrates the use of the static function 'linearSweptSurface' in the class 'SweepSurfaceCreator'. The function can generate a B-spline surface by sweeping one curve along another. A given point on the sweeping curve will be swept along the other curve.

### 31.28 linear\_swept\_volume

linear\_swept\_volume.C

This program demonstrates the use of the static function *linearSweptVolume* in the class *SweepVolumeCreator*. The function can generate a *SplineVolume* by sweeping a surface along a curve or sweeping a curve along a surface. A sweeping point on the curve or the surface must be specified. If the point lies on the the surface, the surface will be swept along the curve. If the point lies on the the curve, the curve will be swept along the surface. The curve and the surface must be such that it doesn't lead to self-intersection.

### 31.29 linearBrepToTrivariate

linearBrepToTrivariate.C

The idea of this program is to read a Brep model in g2-format and check if it can be regenerated as a linear sweep. In that case a multi block trivariate spline model will be created.

### 31.30 loft\_volume\_creator

loft\_volume\_creator.C

This program demonstrates the use of a static function *loftVolume* in namespace *LoftVolumeCreator*. The function use lofting to create a new *SplineVolume* based on a set of surfaces. The surfaces are not changed during the lofting process. The surfaces must lie in the same space.

### 31.31 mirrorAndLoft

example\_mirrorAndLoft.C

This program demonstrates how to create a volume model from a set of spline surfaces. The surfaces must be connected. The surface set is mirrored around a given plane, and lofting between corresponding surfaces is performed.

## 31.32 multiPatchSweep

multiPatchSweep.C

The idea of this program is to sweep a set surfaces to create a multi patch volume model. The surface set will be created by the example programs in compositemodel: createSplitDisc and createBlockStructuredDisc

## 31.33 project\_curve

project\_curve.C

This program demonstrates the use of the function `void CurveCreators::projectCurve(shared_ptr<ParamCurve>& space_cv, shared_ptr<ParamSurface>& surf, double epsge, shared_ptr<SplineCurve>& proj_cv, shared_ptr<SplineCurve>& par_cv)`

The function generates a cubic spline curve(order four) which lies on a given SplineSurface and is the projection of a given space curve (in 3D space) onto that surface, within a given tolerance. The given space curve should be close to the surface.

## 31.34 rotational\_swept\_surface

rotational\_swept\_surface.C

This program demonstrates the use of the static function *rotationalSweptSurface* in the class *SweepSurfaceCreator*. The function can generate a B-spline surface by rotating a curve around an axis.

## 31.35 rotational\_swept\_volume

rotational\_swept\_volume.C

This program demonstrates the use of the static function *rotationalSweptVolume* in the class *SweepVolumeCreator*. The function can generate a SplineVolume by rotating a surface around an axis. The surface must be such that it doesn't lead to self-intersection.

### 31.36 sphere

sphere.C

This program demonstrates the use of the class Sphere. It is a subclass of ElementarySurface. The space dimension of a sphere is 3. The default parametrization is the angles  $u$  from 0 to  $2\pi$  and  $v$  from  $-\pi/2$  to  $+\pi/2$ .

### 31.37 surface\_of\_revolution

surface\_of\_revolution.C

This program demonstrates the use of the class SurfaceOfRevolution. SurfaceOfRevolution is swept out by a SplineCurve that is rotated around an axis with a complete revolution, and is thereby a parametric surface. The space dimension is 3. The curve must be such that it doesn't lead to a self-intersecting surface.

### 31.38 torus

torus.C

This program demonstrates the use of the class Torus. It is a subclass of ElementarySurface. The space dimension of a torus is 3. The default parametrization is the angles  $u$  along the major circle and  $v$  along the minor circle from 0 to  $2\pi$ .